Fibered Structures and Applications to Automated Theorem Proving in Certain Classes of Finitely-Valued Logics and to Modeling Interacting Systems

Dissertation

zur Erlangung des akademischen Grades
"Doktor der technischen Wissenschaften"

Eingereicht von

Viorica Sofronie-Stokkermans

März 1997

Erster Begutachter:	Univ.-Prof. Dr. Jochen Pfalzgraf
Zweiter Begutachter:	Univ.-Prof. Dr. Günter Pilz

Angefertigt am: Forschungsinstitut für symbolisches Rechnen
Technisch-Naturwissenschaftliche Fakultät
Johannes Kepler Universität Linz

Eidesstattliche Erklärung

Ich versichere, daß ich die Dissertation selbständig verfaßt habe, andere als die angegebenen Quellen und Hilfsmittel nicht verwendet und mich auch sonst keiner unerlaubten Hilfe bedient habe.

Linz, am 19. März 1997 Viorica Sofronie-Stokkermans

Support

This research was partially supported by:

- ÖAD doctoral fellowship (Österreichischer Akademischer Austauschdienst),
- Doctoral fellowship offered by RISC Linz for collaboration in the following research projects:

  PARAGRAPH (Parallel Computer Graphics), BMWF,

  MEDLAR II (Mechanising Deduction in the Logics of Practical Reasoning), ESPRIT Basic Research Project, nr. 6471, FWF,

  Cooperation with ProFactor in the domain of multi-agent systems with applications in production (ProFactor),
- TEMPUS JEP-2692-92/2 (support for attending some of the courses and seminars of the TEMPUS Semester in "Categorical and Algebraic Methods of Computer Science" held in Prague WS 1993).
- COST action 15 (MVL) (grant for a short-term mission at the University Claude Bernard Lyon 1).

Kurzfassung

Das Ziel dieser Dissertation ist die Untersuchung von Anwendungen von gefaserten Strukturen in der Informatik, genauer im automatischen Beweisen sowie in der Modellierung kooperierender Systeme. Wir präsentieren und untersuchen Situationen, wo gefaserte Strukturen und Garben (möglicherweise bezüglich Grothendieck Topologien auf gewissen Kategorien) auftreten.

Die Dissertation umfaßt zwei Hauptarbeitszweige, die stark miteinander zusammenhängen.

Die erste Richtung der Arbeit beschäftigt sich mit dem *Zerlegen* von vorgegebenen Strukturen in einfachere Strukturen und zwar so, daß gewisse Klassen von Eigenschaften der gegebenen Struktur reduzierbar sind auf Eigenschaften der einfacheren Strukturen. Der Hauptbeitrag in dieser Richtung befaßt sich mit einer Priestley-artigen Darstellung für distributive Verbände mit Operatoren und der Anwendung zur Redizierung der Komplexität beim automatischen Beweisen in einigen Klassen von mehrwertigen Logiken. Diese Methoden werden zuerst für den Fall von *SHn*-Logiken diskutiert und dann auf allgemeinere Klassen von Logiken erweitert. Eine Implementierung in Prolog wird präsentiert und Vergleiche mit verwandten Ansätzen werden gezogen.

Die zweite Richtung der Arbeit beschäftigt sich mit dem *Zusammenbringen* von verschiedenen Strukturen und der Analyse der Eigenschaften solcher Kompositionen; insbesondere mit dem Studium des Zusammenhangs zwischen den Eigenschaften der einzelnen Komponenten und ihrer Komposition. Wir präsentieren einen garbentheoretischen Ansatz des Begriffs "Concurrency". Beim Studium von komplexen Systemen, die aus mehreren kooperierenden "Agenten" zusammengesetzt sind, ist es von grundlegendem Interesse, das gesamte System durch die Eigenschaften seiner Teile auszudrücken. Wir schlagen einen Begriff von System vor, sowie verschiedene Varianten von entsprechenden Morphismen zwischen Systemen. Wir definieren dann Grothendieck Topologien auf den damit definierten Kategorien, welche "Überdeckungsbeziehungen" zwischen Systemen ausdrücken. Es stellt sich heraus, daß ein Großteil der Information zur Beschreibung von Systemeigenschaften ausgedrückt werden kann durch Garben bezüglich dieser Grothendieck Topologien: zum Beispiel Zustände ("states") und parallele Aktionen werden von Garben St bzw. Act modelliert; Übergänge ("transitions") von einer Untergarbe von $Act \times St \times St$; das Verhalten über einer fixen Zeitskala ($\{0, 1, \ldots, n\}, n \in \mathbb{N}$ oder $\mathbb{N}$) wird von einer anderen Garbe modelliert. Wir benützen geometrische Logik zur Erklärung des Zusammenhangs zwischen gewissen Eigenschaften einer gegebenen Familie miteinander verknüpfter Systeme und der Eigenschaften des aus der Verknüpfung resultierenden Systems.

Abstract

The goal of this thesis is to study the applications of *fibered structures* in computer science, more precisely in automated theorem proving in many-valued logics, and in modeling cooperating systems. We present and study situations in which fibered structures and sheaves (possibly with respect to Grothendieck topologies on certain categories) arise.

The thesis contains two main directions of work, strongly interrelated:

The first direction of work is concerned with finding *decompositions* of given structures in terms of simpler structures, in such a way that certain classes of properties of the given structure can be reduced to properties of the simpler structures. The main contribution in this direction of work concerns Priestley-type representation of distributive lattices with operators, and its application for reducing the complexity of automated theorem proving in classes of finitely-valued logics. These methods are first discussed for the case of SHn-logics and then extended to more general classes of logics. An implementation in Prolog is given and comparisons with related approaches are made.

The second direction of work is concerned with *putting together* (interconnecting) different structures and studying the properties of the result of this interconnection; in particular with studying the link between the properties of the component parts and the result of their interconnection. We give a sheaf-theoretic approach to the study of concurrency. In studying complex systems consisting of several interconnected "agents", given a class of agents (a description of every agent, and a description of the way they interact) it is often necessary to study the properties of the system obtained by the interconnection of the agents in this class. We propose a notion of *system* and several variants of a corresponding notion of morphism, depending on the extent of the relationship between systems that we want to express. We define Grothendieck topologies on the categories defined this way, that express "covering relationships" between systems. It turns out that much of the information relevant for expressing properties about systems can be expressed by sheaves with respect to these Grothendieck topologies: for instance states and parallel actions are modeled by sheaves St, Act; transitions are expressed by a subsheaf of $Act \times St \times St$; and behavior over a fixed range of time (of the form $\{0, 1, \ldots, n\}, n \in \mathbb{N}$ or $\mathbb{N}$) can be modeled as a sheaf too. We use geometric logic in order to explain the link between certain properties of a given family of interconnected systems and the properties of the system that results from their interconnection.

Acknowledgements

There are many people who helped and encouraged me in various ways during the preparation of this thesis, and influenced the work on my thesis either directly or indirectly.

First of all I would like to thank Professor Bruno Buchberger who, as chairman and founder of the RISC institute, was directly responsible for creating a very stimulating and pleasant atmosphere at this institute. I would like to thank him for the useful comments he made concerning my work, but also for organizing the classes on "Thinking, Speaking, Writing", that proved to be very useful in the preparation of this thesis, and also for his initiative of organizing an English class for the RISC students who needed it (as I did at that time). Especially, I would like to thank him for all the efforts he invested to impose and maintain a high standard at the institute, and also for all the time and energy he had to spend raising funds in order to allow a better development of the institute.

My sincerest thanks go to Professor Jochen Pfalzgraf, my advisor, who directed my interest to the field of modeling cooperating agents with the help of "fibered structures", for his enthusiasm, constant encouragement and interest and for his permanent preoccupation in maintaining contacts with various other groups working in category theory or in logic, and for organizing the visit to Linz of several researchers working in these areas, as for instance (in chronological order) Dr. Barry Jay, Professor Jiři Adámek, Professor Jiři Rosicky, Professor William Lawvere, Professor Jim Cunningham, Dr. Reiner Hähnle, Professor Ewa Orlowska. I also would like to mention the cooperation we had in MEDLAR, for which he was the coordinator at RISC (and which was also my source of income during the years 1994-1996), as well as his fruitful efforts in establishing links with other groups working in the field of many-valued logics and applications, which concretized when our group joined the COST action 15 on many-valued logics. Moreover, Jochen Pfalzgraf set up the working group on applied and computational category theory at RISC and organized a very interesting seminar with this group.

I would also like to thank Professor Dana Scott who, during his one-year stay at RISC, organized the seminar of the category theory group together with Jochen Pfalzgraf. It was a great privilege to attend this seminar; both because of the highly interesting topics Professor Dana Scott disclosed to us – in his talks and by the questions during the talks of the other participants – (including details and links between various areas that are hard to find in books), and because he showed us what research and teaching can mean.

I want to thank Professor Günter Pilz for his willingness to be the second "Gutachter" of my thesis and for giving me the opportunity to attend the seminar of his group and the possibility to present my own work in this seminar, where I found a very friendly atmosphere.

I would also like to thank Professor Matthias Baaz for his many interesting courses on various topics such as semantics of programming languages, temporal logic, analogical reasoning, and automated theorem proving in non-classical logics that he gave at the University of Linz. The course in automated theorem proving in non-classical logics – in which among other very interesting topics he also presented his own research on automated theorem proving by resolution in finitely-valued logics – was an important source of inspiration for the part of my thesis where I present a method for automated theorem proving in some classes of finitely-valued logics. I also would like to thank him for giving our group the possibility of presenting its work in a special session of the Kurt Gödel Society in Vienna in June 1996.

I would also like to thank Professor Adámek and Professor Rosicky for inviting our RISC group to a seminar at the Charles University in Prague in May 1993, where I had the opportunity of attending very interesting talks of Professor Adámek, Professor Pultr, Professor Rosicky, and Professor Trnková, and for inviting the RISC group to become part of the TEMPUS project, so enabling our group to visit the seminars held during the TEMPUS semester in 1993 in Prague and to attend the highly interesting talks of Professor Adámek and Professor Pultr.

I thank Dr. Reiner Hähnle for the very interesting discussions on many-valued logics and automated theorem proving we had during his visit at RISC.

I thank Professor Ricardo Caferra and Professor Ewa Orlowska for the contribution they had in strenghtening the relationships between RISC and other groups in the COST 15 action on many-valued logics. I would especially like to thank Professor Ewa Orlowska for sending me one of her papers (writen together with Professor Iturrioz) on a Kripke semantics for SHn-logics, that actually gave me the idea of the method for automated theorem proving that I describe in Chapter 5 of the thesis. I thank her also for the very inspiring discussions we had during her visit at RISC, as well as for her generosity to provide me with copies of papers and chapters of books which were difficult for me to access.

Sincere thanks also to Professor Luisa Iturrioz, for accepting my application for a short-term mission at the University Claude Bernard in Lyon in the frame of the COST 15 action on many-valued logics in February 1997, and for giving me the possibility of presenting my work in the seminar of the Laboratory of Discrete Mathematics at the University Claude Bernard in Lyon. I thank her for giving me numerous bibliographical references concerning representation theorems for Łukasiewicz, Post, and Heyting algebras, for her very interesting remarks on the development of Łukasiewicz and Post logics, and on the relationships between Łukasiewicz logics and the so-called Łukasiewicz-Moisil algebras as well as with the proper Łukasiewicz algebras. I also thank her for reading parts of my thesis and making several valuable suggestions, which helped me to improve Chapter 5.

I would also like to thank my professors and colleagues at RISC who, in one way or another helped me in my scientific work. It would be hard to single out everyone of them for thanking them for their friendship and help. However, I would like to thank Dr. Hoon Hong for his suggestions concerning the way of presenting my ideas. I would also like to thank all former and present members of the category theory group at RISC: Eugen Ardeleanu, Olga Caprotti, Drew Dean, Wolfgang Gehrke, Manfred Minimair, Victor Pollara, Josef Schicho, Karel Stokkermans, Frederik van den Plancke, Kim Ritter Wagner, Yasushige Watase, and Todd Wilson.

I would like to also thank my professors during my stay at the University of Lisbon, Portugal in 1992, especially to Professor Gabriela Hauser Bordalo for her course on universal algebra and lattice theory, and for advising me in the preparation of a seminar talk on Priestley duality for Ockham algebras. I used this in Section 5.4.1 as an example for the approach to automated theorem proving presented in Chapter 5. Many thanks also to Professor Gracinda Gomes for her very interesting course in semigroup theory and for the kindness with which she helped me to get financial support in order to attend the Summer school on semigroup theory held in York in 1993.

Last, but not least I would like to express many thanks to all my professors at the University of Bucharest in Romania. Among them I would especially like to thank Professor Mircea Maliţa, my diploma thesis advisor, for teaching me the first notions on automated theorem proving, for suggesting me critical-pair/completion algorithms such as the Knuth-Bendix algorithm and Buchberger's algorithm for computing Gröbner bases in polynomial ideal theory as a topic for the diploma thesis, and for later introducing me to non-classical logics, and encouraging my idea of using the Knuth-Bendix algorithm for automated theorem proving in some classes of modal logics by rewriting in the associated equational theories. Warm thanks also to Professor Solomon Marcus for his courses in which he put a very strong emphasis on the real understanding of mathematics and its beauty, for the way he encouraged and led his students to research, for his constant encouragement during the time when I was a student, and for the support in the times when I applied for grants; Professor Sergiu Rudeanu for his excellent lectures in universal algebra and for encouraging me to attend the seminar on sheaf theory at the University of Bucharest in 1991; Professor Virgil Emil Cazanescu for his highly rigorous courses and seminars in logic, category theory and denotational semantics.

And, most importantly, I would like to thank my parents and my husband Karel for all their love, understanding, and support over the years.

Contents

Chapter 1

Introduction

The goal of this thesis is to study the applications of *fibered structures* in computer science, more precisely in automated theorem proving in many-valued logics, and in modeling cooperating systems.

The notions of fiber bundle and sheaf were originally developed in geometry and topology. Sheaf theory is a particularly effective tool in those areas which ask for global solutions to problems whose hypotheses are local: it was developed in mathematics because of the necessity of studying such relationships between local and global phenomena. The concept of sheaf was formally introduced by Jean Leray and Henry Cartan in 1950. Originally, the theory of sheaves was conceived as a tool in topology and algebraic geometry, for axiomatizing notions such as "local coefficient system". The influence of sheaf theory has spread since then in many areas of mathematics: besides the fields where its origins are, such as analysis, topology (the study of germs of holomorphic functions, see e.g. [Ler45, Car50]) and algebraic geometry ([Hir56, Zar56, God58]), it is now also used in algebra (representations of algebras by continuous sections; global subdirect products see e.g. [Hof72], [Dav73], [Wer75], [KC79]), and logic (for details see also [FS79], [MLM92]).

In algebraic geometry it was soon discovered that the topological notion of a sheaf was not entirely adequate: the Zariski topology on an abstract algebraic variety turned out to not have "enough open sets" to provide a geometric notion of localization, and furthermore, it turned out that it was important to replace monomorphisms between neighborhoods of points by more general mappings (not necessarily injective). For this reason A. Grothendieck introduced the notion of "Grothendieck topology" on an arbitrary category, and a generalized notion of sheaf for such a topology. For more information on the history and development of sheaf theory see [Gra79, Joh82, Gol84, MLM92].

In the thesis we present and study situations in which sheaves (possibly with respect to Grothendieck topologies on certain categories) or fibered structures arise. The thesis contains two main directions of work, strongly interrelated: We use the Priestley representation theorem in order to reduce the complexity of automated theorem proving in finitely-valued logics, and then we give a sheaf-theoretic approach to the study of concurrency.

The main topics of the thesis are the following:

- Fibered structures in universal algebra.
  - Sheaf representation theorems in universal algebra and applications to unification in discriminator varieties,
  - Priestley-type representations for distributive lattices with operators, and applications to automated theorem proving in many-valued logics,
- A sheaf-theoretical approach to modeling cooperating agents.

The first direction of work is concerned with finding *decompositions* of given structures in terms of simpler structures, in such a way that certain classes of properties of the given structure can be reduced to properties of the simpler structures.

The second direction of work is concerned with *putting together* (interconnecting) different structures and studying the properties of the result of this interconnection; in particular with studying the link between the properties of the component parts and the result of their interconnection.

There is a strong link between these two main directions of work, as will be shown in what follows.

1.1 Presentation of the Main Results

In what follows I give a succinct presentation of the main results and contributions contained in the thesis.

1. Fibered representations in algebra with applications in automated theorem proving

Representation theorems in universal algebra – such as sheaf representation and Priestley type representation theorems – are presented, and their applications are investigated, as explained in what follows.

1a. Sheaf representation in Universal Algebra. Concerning sheaf representation theorems my contribution is only tangential: I prove that a theorem in [Bur92] – that shows that in discriminator varieties a most general unifier of two terms can be constructed from a particular unifier – can be extended to systems of equations in discriminator varieties.

A sheaf representation theorem in universal algebra due to [Dav73] is used in the study of cooperating agents.

1b. Priestley representation for distributive lattices with operators and applications in automated theorem proving. My main contribution in this direction of work concerns Priestley-type representation for distributive lattices with operators, and its application in automated theorem proving in classes of finitely-valued logics. The explanation of the notions used in what follows can be found in Chapter 5.

I start with a study of the propositional SHn-logics (first introduced by L. Iturrioz in [Itu82]), as a motivating example, since the idea of using the Priestley dual of the algebra of truth values for automated theorem proving occured to me when studying SHn-logics. An algebraic as well as a Kripke semantics for these logics are known [Itu83, IO96]. Additionally a topological representation theorem induced by the Priestley representation theorem is proved in [Itu83]. I extended the topological representation in [Itu83] to a dual equivalence between the category of SHn-algebras and a suitable category SHnSp of Priestley spaces with operators. I also made the properties of the additional operators on these Priestley spaces explicit. It turned out that the objects of SHnSp are SHn-frames in the sense of [IO96]. The study also gave hints about the way of defining morphisms between SHn-frames, and thus defining a category of SHn-frames.

In [Itu83] it is proved that for every $n \in \mathbb{N}$, the SHn-logic is sound and complete with respect to the variety of SHn-algebras, which is generated by the finite algebra S_{n^2}. As a consequence, it turned out that a formula is a SHn-theorem if and only if it is valid in the finite SHn-frame defined by the dual space $D(S_{n^2})$ of S_{n^2}.

Since the dual space $D(S_{n^2})$ has $2(n-1)$ elements whereas S_{n^2} has n^2 elements, the idea occurred to me that one might define a more efficient proof procedure using the dual space. I begin by considering the propositional case: in this case I define positive and negative literals as being of the form $\boxed{x}\ p^t$, resp. $\boxed{x}\ p^f$, where $x \in D(S_{n^2})$ and p an atomic formula; clauses (sets of literals), and a notion of satisfiability. I give a procedure that for every formula ϕ in the language of SHn-logic constructs a set Φ of clauses such that ϕ is a theorem if and only if Φ is unsatisfiable. The unsatisfiability of Φ can be checked with a procedure called signed negative hyperresolution: it turns out that the proof of [AB70] can be adapted to this case without major modifications. This proves the correctness of the procedure.

Next, I consider a more general case, namely the case of logics $\mathcal{L}$ that are sound and complete with respect to a variety $\mathcal{V}$ of algebras that satisfies the following properties:

(i) $\mathcal{V} = HSP(A)$, where A is a finite algebra;

(ii) The algebras in $\mathcal{V}$ are distributive lattices with operators, and the Priestley duality induces a dual equivalence between $\mathcal{V}$ and a suitable category $\mathcal{V}\mathsf{Sp}$ of Priestley spaces.

The form of operations on the dual category in the case when the additional operators considered are morphisms, antimorphisms, join-hemimorphisms and meet-hemimorphisms are analyzed.

I show that the dual category $\mathcal{V}\mathsf{Sp}$ provides a class of topological Kripke models for the logic $\mathcal{L}$; a way of defining notions such as satisfiability and validity in these models is discussed.

The isomorphism between the finite algebra A and the set of order-filters of its dual $D(A)$ induces a notion of satisfiability (resp. validity) of a formula in the dual space $D(A)$.

I prove a similar result, namely that a formula in the logic $\mathcal{L}$ is a theorem in $\mathcal{L}$ if and only if it is valid in $D(A)$. As in the case of the SHn-logics, I give a procedure for automated theorem proving (consisting of two steps, namely transformation to clause form and negative hyperresolution). I also consider the first-order logic having A as a set if truth values, and show that both the translation to clause form and the signed hyperresolution procedure can be extended to first-order formulas.

2. A sheaf-theoretical approach to modeling cooperating agents scenarios

In studying complex systems consisting of several interconnected "agents", the problems that arise can be described as follows:

Given: A family $\{S_i \mid i \in I\}$ of interconnected agents, i.e.
- a description of every agent, and
- a description of the way they interact.

Task: Study the properties of the system obtained by their interconnection.

The basic idea of my formalism is that even relatively simple agents, such as a robot that provides an assembly bench with pieces, is in fact a complex system composed of interacting subsystems, like joints and wrists, a locomotion module, etc. Complex systems as well as their component parts can be essentially described in the same way; the level at which we "stop" the refinement process, and consider a subsystem as being "atomic" depends on the given application and on the degree of accuracy needed[1].

Therefore instead of "individual" agents, a category of "systems" is considered. I study in detail a subcategory of this category, where the objects are systems and the morphisms describe the relation "is a subsystem of", and show that under certain (non-restrictive) hypotheses a Grothendieck topology (describing a "covering" relation between systems) can be defined on this category. In addition, within this framework, the states of systems as well as their admissible parallel actions can be modeled by sheaves. Then, I restrict to a category $\mathsf{SYS}_{\mathsf{il}}$ having systems as objects and transition-connected inclusions as morphisms (in order to impose the condition that transitions in a system restrict to valid transitions in its subsystems), and show that this category has pullbacks and colimits of families of systems that are all transition-connected subsystems of a given system. In concrete applications we usually are interested only in some subcategory of $\mathsf{SYS}_{\mathsf{il}}$, having as objects those systems relevant for the given application. Therefore I continue by considering the category of those systems

[1] This situation is somehow similar to the situation that arises in geometry, when defining the notion of a *point*. Intuitively, a point can be defined as the "limit" of a family $\{U_i \mid i \in I\}$ of "spots" that (informally said) get smaller and smaller (for instance, such that $I = \mathbb{N}$ and for every $i \leq j$, $U_j \subset U_i$).

obtained by interconnecting a given family InSys of interacting systems, all contained in a given system S_U (to enforce compatibility of models) and which is assumed to be closed under all subsystems by means of which communication can be done. A system obtained by interconnecting the elements of the family InSys can be regarded either as a system on its own, or as the set of all elements of InSys by whose interaction it arises (a downwards-closed subset of InSys).

I show that on both categories defined this way, Sys(InSys) resp. Ω(InSys), suitable Grothendieck topologies can be defined, expressing the way systems arise from smaller subsystems. In both these approaches one can define notions as *states* and *parallel actions*, and show that these define sheaves St_i, resp. Act_i $(i = 1, 2)$ with respect to the corresponding Grothendieck topologies. Moreover, transitions can be expressed in both cases by natural transformations $Tr_i : Act_i \to \Omega^{St_i \times St_i}$ $(i = 1, 2)$ (or, alternatively, by subsheaves of $Act_i \times St_i \times St_i$). The link between the categories Sys(InSys) and Ω(InSys) is also investigated: I show that an adjunction between these two categories exists; additionally the right adjoint preserves covers, which implies that a geometric morphism between the category of sheaves over Sys(InSys) and the category of sheaves over Ω(InSys) (with the corresponding Grothendieck topologies) can be established.

I continue by studying the behavior in time of systems. The starting point of my approach is the formalism developed in [Gog92]. Assume that time is discrete and the execution of every action needs one unit of time. Let $\mathcal{T}$ be the category of all subsets $\{1, \dots, n\}, n \in \mathbb{N}$ (and includes $\mathbb{N}$ itself), with inclusions as arrows. A result in [Gog92] states that the behavior of a system can be modeled as a sheaf over $\mathcal{T}$. I show that actually two gluing properties hold: one with respect to covers on the category $\mathcal{T}$, and one with respect to covers on the category Sys of systems (one of the two categories considered above). This is expressed by introducing two functors, $B : \mathsf{Sys}^{op} \to \mathsf{Sh}(\mathcal{T})$, and $B' : \mathcal{T}^{op} \to \mathsf{Sh}(\mathsf{Sys}, \mathsf{J})$. A possible model for the behavior of interconnected systems by sheaves of partially commutative monoids is also considered.

All these results are used in the last section of Chapter 8, where I use classical results in sheaf theory and geometric logic to investigate the links between the properties of the elements of InSys and the system obtained by interconnecting them. These theoretical considerations are illustrated on three examples: deadlock freedom, determinism, and fairness of execution.

These two directions of work are strongly interrelated, as will be shown in what follows.

On the one hand, the Priestley representation for distributive lattices can be regarded as follows: Let L be a set of truth values that has an underlying distributive lattice structure. In practical situations, e.g. in robotics, the different truth values may be checked by sensors; it may also happen that the sensors cannot distinguish between all the values of L.

Assume that the sensors can only return the values 0 (false) and 1 (true); and additionally, that they respect the order of truth values (i.e., if x is perceived as "true" and $x \leq y$ in L then y will also be perceived as "true") as well as the lattice operations $\vee$ and $\wedge$. (For example, if $L = \{0, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}, 1\}$, the truth value (in L) of a parameter v can be completely reconstructed if we assume given an ordered set of $n-1$ 2-valued sensors, $S_1 \leq \ldots \leq S_{n-1}$, such that for every i, the sensor S_i perceives as "true" a parameter if its value (in L) is greater than or equal to $\frac{i}{n-1}$ and as false otherwise.)

The set of 2-valued sensors necessary in order to recover the information about the values of parameters in L is then exactly the (ordered) set of all $\{0,1\}$-lattice homomorphisms from L into $\{0,1\}$, i.e. exactly the Priestley dual of the algebra L.

On the other hand, in the study of agents (e.g. in robotics), the description of *states* plays a fundamental rôle. The actions induce then *transitions* between states. In a multi-modal logic approach to the study of agents, these states can be seen for instance as *possible worlds*, and the transitions between states can be expressed by corresponding relations between these worlds. Thus, Kripke models arise in a natural way.

This is the idea *dynamic logic* (also known as *the logic of programs*) relies upon (for an introduction to dynamic logic we refer to [Har84]). In the propositional case the algebraic models for dynamic logic are dynamic algebras (Boolean algebras endowed with operators that satisfy certain properties) and the Kripke models are sets endowed with families of relations that satisfy suitable properties.

However, it seems that the same ideas can be applied in more general contexts, for example in situations when the underlying logic is not classical (propositional) logic.

Finally, we point out one more link between the theoretical study of sheaves of algebras mentioned before and the theory of cooperating agents. The behavior of a given agent can be expressed for instance as the set of all sequences of actions the agent can execute. Sometimes the order in which two actions are executed is not relevant. If we specify an independence relation on the set of actions, this relation induces a congruence on the set of all finite sequences of actions of the given agent. When putting together a family $\mathcal{S} = \{S_i \mid i \in I\}$ of interacting agents, one of the problems that may occur is to decide if "local plans" can be *glued together* to a "global plan". That is, given a family of sequences of actions $\{w_i\}_{i\in I}$, where each w_i is a finite sequence of actions for the agent S_i, such that the elements of this family are compatible on common subsystems (in a sense that will be explained in Section 7.2.4), the problem is to decide if there exists a finite sequence of actions for the system S (obtained by interconnecting the agents in $\mathcal{S}$) that contains all the sequences $\{w_i\}_{i\in I}$ as substrings. It turns out that this problem can be formulated in terms of sheaf theory; namely one has to decide whether the (partially commutative) monoids expressing the finite sequences of actions of the systems in the family $\{S_i \mid i \in I\}$ define a sheaf of monoids, and whether the set of global sections of this sheaf is isomorphic to the (partially commutative) monoid expressing the finite sequences of actions of S.

1.2 A Short Summary of Results

The thesis includes the following contributions:

- A comparative study of the sheaf representation theorems and Priestley representation theorems: we show that both induce a notion of fibering. The similarities and the differences are analyzed, and some directions of future work are indicated.

- Sheaf representation theorems in universal algebras

  1. An extension of the construction of a most general unifier in discriminator varieties to systems of equations.

- Priestley representation for SHn-algebras.

  - Theoretical Considerations:
    1. An extension of the topological representation for SHn-algebras given in [Itu83] to a dual equivalence between the categories of SHn-algebras and SHn-spaces,
       A restriction of the dual equivalence between the categories of SHn-algebras and SHn-spaces to a dual equivalence between the category of $SHKn$-algebras (which is the category of Lukasiewicz-Moisil algebras) and the category of $SHKn$-spaces.
    2. The proof of the fact that SHn-spaces are in particular SHn-frames, according to the definition in [IO96],
    3. A possible way of defining morphisms between SHn-frames (and thus of defining a category of SHn-frames),
    4. The link between valuations in SHn-algebras and SHn-spaces,
    5. The link between provability in SHn-logics and validity in the SHn-frame $D(S_{n^2})$.
  - A procedure for automated theorem proving by resolution that uses the dual space of a SHn-algebra:
    1. A procedure for transformation to clause form (also definitions for signed literals, clauses, satisfiability),
    2. A signed hyperresolution procedure (proof of correctness).

- Extension to larger classes of logics

  - We study logics $\mathcal{L}$ that are sound and complete with respect to varieties $\mathcal{V}$ of algebras with an underlying distributive lattice structure, that are generated by one finite algebra, and with the property that the Priestley duality extends to a dual equivalence between $\mathcal{V}$ and a category $\mathcal{V}\mathsf{Sp}$ of Priestley spaces with operators.
    1. We analyze the way the satisfiability relation $\overset{a}{\models}$ with respect to algebras in $\mathcal{V}$ induces a satisfiability relation $\underset{rc}{\models}$ with respect to Priestley spaces in $\mathcal{V}\mathsf{Sp}$.

2. We show that showing $\vdash_{\mathcal{L}} \phi$ reduces to testing whether $D(A) \overset{rc}{\models} \phi$.

– We study the properties of distributive lattices with operators, in order to obtain a better understanding of the link between algebraic and relational models.
  1. The starting point is the theory presented in [Gol89], where distributive lattices endowed with join- and meet-hemimorphisms are considered. We additionally consider homomorphisms and antimorphisms.
     The corresponding operations (resp. relations) induced on the dual space are analyzed.
  2. We define accordingly a notion of frames (partially-ordered sets endowed with operations and relations).
  3. We discuss a notion of satisfiability (resp. validity) in such frames. The link between satisfiability (validity) in these frames and satisfiability (validity) with respect to algebraic models is also discussed.

– A procedure for automated theorem proving by resolution for logics $\mathcal{L}$ (sound and complete with respect to varieties $\mathcal{V}$ of algebras with an underlying distributive lattice structure, such that $\mathcal{V}$ is generated by one finite algebra A and the Priestley duality extends to a dual equivalence between $\mathcal{V}$ and a category $\mathcal{V}\mathsf{Sp}$ of Priestley spaces with operators). This automated theorem proving procedure uses the dual space $D(A)$ of A.
  1. Definitions for signed literals, clauses, satisfiability,
  2. A procedure for transformation to clause form,
  3. A signed hyperresolution procedure (proof of correctness),
  4. Extension to first-order logics.

– An implementation in Prolog.

Note that it turned out that the condition we imposed on the logic $\mathcal{L}$, namely that it should be sound and complete with respect to a variety $\mathcal{V}$ of algebras with an underlying distributive lattice structure, such that $\mathcal{V}$ is generated by one finite algebra A and the Priestley duality extends to a dual equivalence between $\mathcal{V}$ and a category $\mathcal{V}\mathsf{Sp}$ of Priestley spaces with operators, can be relaxed, as we now explain.

In the description of the procedure for automated theorem proving (and in its proof of correctness) we only use the fact that the logic $\mathcal{L}$ is sound and complete with respect to a finite Kripke frame endowed with an order relation and with additional relations associated to the operations in the logic. We would like to investigate the degree of generality of this approach. We kept the initially imposed set of conditions on the logic $\mathcal{L}$ because it furnishes an intuitive description of how such a Kripke frame can be constructed.

- A sheaf-theoretic approach to cooperating robotics scenarios:

    1. definition of a system,
    2. definition of a morphism between systems; category of systems SYS; expressing states, parallel actions, transitions in this category,
    3. consider other types of morphisms depending on the extent of the relationship between systems to be expressed; we focus on categories having inclusions as morphisms,
    4. definition of Grothendieck topologies on different categories of systems; study of states, transitions and behavior in these categories (gluing properties are satisfied, which suggest that a sheaf-theoretical approach is appropriate in order to express the link local-global),
    5. the links between these categories are analyzed,
    6. classical results from sheaf theory (geometric logic) are used in order to study properties of systems that are preserved by interconnection.

1.3 Structure of the Thesis

The thesis is structured as follows:

Chapter 2 begins with a look at the background and motivation of fibered structures and their use in computer science. It continues with a brief presentation of related results, and the way these influenced our work.

In Chapter 3 we give a brief review of the main concepts from universal algebra, logic, category theory and sheaf theory that will be used in our work. This is done in order to make the thesis self-contained.

In Chapter 4 we review concepts and results that are directly linked to our own results that will be presented in the thesis, as well as related work. This includes sheaf representation theorems in universal algebra and the Priestley representation theorem for distributive lattices, as well as various models for concurrency.

In Chapter 5 we present an approach to automated theorem proving for certain finitely-valued logics, based on the Priestley dual of the algebra of truth values.

In Chapter 6 we begin a study of distributed agents, having as goal a better modeling of the link between local and global properties in complex systems, composed by interconnecting intercommunicating agents. As a motivation for this theoretical study, we illustrate the problems that appear on a simple example, adapted from [Pfa93]. This example leads to a formal definition of a system. We then define morphisms between systems, and introduce a category SYS of systems, and study states, parallel actions and transitions in this category.

In Chapter 7 we study the category SYS_i (that has systems as objects and inclusions of systems as morphisms). In a first approximation we pay no attention to the transitions between states induced by the actions. We show that states and parallel actions can nevertheless be modeled by sheaves with respect to a suitable Grothendieck topology. In order to capture the dynamical aspect of systems we then take into account also transitions induced by actions. We consider therefore $\mathsf{SYS}_{\mathsf{il}}$, the subcategory of SYS having as objects systems and as morphisms so-called transition-connected inclusions (which ensure that valid transitions in a system restrict to valid transitions in its subsystems). The properties of states, actions, transitions and behavior are studied also for this category.

In Chapter 8 we analyze the situation arising from interconnecting a given family InSys of communicating systems. We can regard the system obtained by interconnecting the elements of InSys either as a system on its own, or as the set of all elements of InSys by whose interaction they arise (i.e. as a downwards-closed subset of InSys). We analyze both these approaches, and then the relationship between them. We use these results for expressing properties of systems and reasoning about them. We show that results from sheaf theory, in particular geometric logic help us in deciding which properties are inherited by the system obtained by interconnecting a family of given systems. Several examples are provided.

Chapter 9 contains applications of the theoretical results above (presented as algorithms).

In Chapter 10 we summarize the main results and indicate the directions for future research.

Chapter 2

Motivation

This chapter begins with a look at the background and motivation of fibered structures and their use in computer science. It continues with a brief presentation of related results, and the way these influenced our work.

2.1 Background and Motivation

The goal of the thesis is to study some of the applications of so-called *fibered structures* in computer science.

Informally, a *fiber bundle* consists of a set B (called base space) and a family of mutually disjoint sets $\mathcal{E} = \{E_b \mid b \in B\}$, together with a projection map $p : \coprod_{b \in B} E_b \to B$ (where $\coprod_{b \in B} E_b$ is the disjoint union – i.e. the coproduct in Sets – of the family $\{E_b \mid b \in B\}$) that for every $x \in E = \coprod_{b \in B} E_b$ associates $b \in B$ if $x \in E_b$; such a fiber bundle is denoted by (B, E, p). The sets E_b, $b \in B$ are called *fibers* or *stalks*. These notions originally were developed in geometry and topology. The construction presented above is very general. The sets E and B can be endowed for example with a topology, with relations, or with an algebraic structure. For instance, we could just work in the category of topological spaces instead of sets, and impose that the projection map p respects the corresponding structure of the spaces, for example it has to be a continuous map in the category of topological spaces (we can think in this case of the family $\{E_b \mid b \in B\}$ as being "continuously indexed" by B). A fibre bundle (B, E, p) where B and E are topological spaces is called a sheaf space if $p : E \to B$ is a local homeomorphism, (i.e. for every point $x \in E$ there exists a neighborhood U of x in E such that $p(U)$ is open and $p : U \to p(U)$ is a homeomorphism).

Many of the representation theorems in universal algebra lead to the occurence of fiber bundles or even sheaves with the property that the stalks have a certain algebraic structure.

The basic idea of representation theorems is to provide *decompositions* of certain structures in terms of simpler structures. In what follows we will refer to two kinds of representation theorems, namely to sheaf representation theorems for discriminator varieties and to the Priestley duality for distributive lattices.

Also the inverse process, namely that of *combining* structures, that arises for example in modeling concurrency, can be modeled using fiber bundles, and – as shown in this thesis – the link between local and global properties can be expressed in certain situations using sheaf theory.

Since in what follows we will make use of sheaf theory (including sheaves of algebras), we would like to point out the importance of sheaf theory (and geometric logic) in universal algebra. The following considerations are inspired by the ideas stated in [KC79], and turned out to be also a source of inspiration for our approach to modeling concurrency.

The basic idea behind representation theorems in universal algebra is to "decompose" a given structure into "simpler" structures in such a way that the properties of the given structure can be "reduced" to properties of the simpler structures. The theorem of Birkhoff, one of the best known theorems in universal algebra, asserts that every algebra is isomorphic to a subdirect product of subdirectly irreducible algebras. This theorem is however not a satisfactory representation theorem in universal algebra for two reasons: first, usually it is very difficult to determine the subdirectly irreducible factors; second, even when these factors are known, subdirect products are so "loose" that very little can be inferred about an algebra from the properties of the factors. An example of "good" representation is the *direct sum representation* for abelian groups. That is because direct sums are special subdirect products which are "tight" enough such that significant information can be obtained from the properties of the factors. Unfortunately, for important classes of rings interesting direct sum representations are not known. This situation provided the motivation for the development of sheaf representations for algebras, i.e. representations of algebras as sheaves of global sections over a certain topological space.

One of the properties that make structures of global sections "tight" subdirect products is the "patchwork property": structures of global sections patch over the so-called equalizer topology. (However, unrestricted patching is usually hard to verify in applications. In some cases, unrestricted patching can usually be reduced to finite patching via a compactness argument.)

Another reason is the fact that the global section functor (an example of a direct image functor) preserves limits (but in general it does not preserve unions and images); therefore it preserves the validity of *cartesian formulae* relative to a given theory $\mathbb{T}$, i.e. the formulae constructed from atomic formulae using only conjunction and existential quantification over "$\mathbb{T}$-provably unique" variables.

Sheaf representation theorems also have applications in solving algorithmical problems, as for example unification. In 1966 Dauns and Hoffmann study algebras of global sections of sheaves of algebras over Boolean spaces, named *Boolean products* in 1979 by Burris and Werner. For such products one can analyze not only equations but also positive primitive sentences (sentences of the form $\exists x \bigwedge(p_i(x) = q_i(x)))$ in terms of the behavior of the stalks [BW79], i.e. we can solve a system of equations in a Boolean product if and only if we can solve it in every stalk of the Boolean product, and a given sequence of elements is a solution if and only if it provides a solution in each stalk. Based on these results, [Bur92] proves that discriminator varieties have unitary unification and gives a method for constructing most general unifiers starting from particular unifiers.

The Priestley representation theorem for distributive lattices states that every distributive lattice L is isomorphic to the set of continuous, order-preserving functions from the Priestley space $D(L)$ of all prime filters of L to the 2-element chain $\{0, 1\}$. Thus, "fibered structures" appear also in this context; here the base space is endowed – besides the topology – also with an order relation, and, in the case of extensions of the Priestley representation theorem to distributive lattices with operators, with additional operations and relations that correspond to these operators, whereas the "fibers" are all isomorphic to the 2-element lattice. In the thesis we point out the relation between Kripke models and this type of spaces and use the result for giving an automated theorem proving procedure in non-classical logics.

The inverse process, namely that of *combining* structures, that arises for example in modeling concurrency, can be also modeled using fiber bundles. Fibered models for cooperating agents scenarios have already been used by [Pfa91], see also [Pfa96], and developed in [PS92, PSS95, PSS96a, PSSS95, PSS96c, PSS96b]. The notion this model is based on is that of fiber bundle. The main idea is that the general concept of fiberings allows to mix different structures (spaces of different types) by taking them as fibers over a certain index system (base space). This is important when looking for a unifying mathematical framework for modeling complex and heterogenous interacting structures. We also refer here to the extensive research of Gabbay on Labelled Deductive Systems and to his method for combining logics based on the notion of "fibred semantics" [Gab92, Gab94, Gab96].

In [Sof96], when studying states and admissible parallel actions of systems by interconnecting communicating systems, we noticed that they satisfy a gluing property similar to the property of a sheaf. Thus, the idea occurred that sheaf theory can be an appropriate tool for modeling cooperating agents scenarios. This is not surprising. In what follows we would like to explain why we think that sheaf theory can be a useful framework for modeling cooperating agents scenarios.

As pointed out before, sheaf theory was developed in mathematics because of the necessity of studying the relationship between "local" and "global" phenomena. The same situation arises in the study of interacting systems: when modeling states or behavior it is often necessary to make a link between "local" properties (characteristic for given subsystems) and "global" properties (relevant for the whole system). The goal of our study is an analysis of subsystem interaction, taking into account the contribution of subsystems to the behavior of the whole system. The interaction between systems can be described through common behavior (or states) at shared "locations".

The alternance "local - global" that occurs in this case suggests that it would be natural to use sheaf theory when studying systems of cooperating agents (and in the study of concurrency in general).

Moreover, the tools of sheaf theory (and of topos theory in general, in particular geometric logic cf. [MLM92]), should explain why some properties of systems are preserved when restricting to subsystems, and why there are cases when properties of subsystems are not transferred to the system obtained by their interconnection (we will illustrate this on several examples, among which the properties of deadlock freedom, determinism, and fairness of execution).

2.2 Brief Overview of Related Results

We give a brief overview of previous work in the fields considered in the thesis, namely representation theorems in universal algebra (sheaf representations in universal algebra and Priestley representation for distributive lattices with operators); automated theorem proving in non-classical logics; and models for cooperating agents and concurrency. Details about those results and methods that are relevant to our work will be given in Chapter 4.

Sheaf representation theorems in universal algebra as well as the Priestley duality theorems can both be seen as "fiberings". This link is discussed and illustrated in Section 4.1.5. We briefly give here some historical milestones in the development of the corresponding theories.

Sheaf representation of algebras. The use of sheaf representations in various parts of algebras have become popular in the late 60's and early 70's. There exist sheaf representation theorems for rings [AK48, DH66, Hof72], semigroups [Kei70], l-groups (lattice-ordered groups) [Kei71], distributive lattices [Dav72] and universal algebras [Com71, Dav73, Wer75, KC79]. In 1953 Foster [Fos53a, Fos53b] proved that every n-valued Post algebra is isomorphic to the algebra of global sections of a sheaf having as fibers the Post algebra P_n with n elements (so-called Boolean power of P_n). [Cig72] showed that Łukasiewicz-Moisil algebras embed in algebras of global sections of sheaves (and that in the case of Post algebras this embedding is an isomorphism). A sheaf representation (up to isomorphism) for Łukasiewicz-Moisil algebras follows as a particular case of a theorem by Werner [Wer75]. In [BW79], elementary properties of sheaf constructions are investigated, and in [Bur92] these results are used in order to study the unification problem in discriminator varieties. These results generalize methods already known for the variety of Boolean algebras. (For the description of a program that uses these methods for solving systems of Boolean equations we refer to [Sof89].)

In the thesis we show that one of the results in [Bur92] – concerning the construction of most general unifiers for discriminator varieties from a given unifier of two given terms – can be extended to systems of equations.

Priestley duality for distributive lattices with operators. The Priestley duality theorem for distributive lattices is due to Priestley [Pri70, Pri72]. It has been extended to duality theorems between various classes of distributive lattices with operators and corresponding categories of Priestley spaces endowed with additional operations: we refer e.g. to [CF77] that establishes a Priestley duality for de Morgan algebras; to [Urq79] and [Gol81] that establish such a dual equivalence for varieties of Ockham algebras; to [BP90] where relative Ockham lattices are studied and their order-theoretic and algebraic characterization is given (see also [BP94] for further results); to [Tra77] that establishes a Priestley duality for Post algebras; to [Fil80] that gives a duality theorem for θ-valued Lukasiewicz algebras without negation; to [Ior84] that gives a duality theorem for θ-valued Lukasiewicz algebras with negation; and to [Mar90] that gives a Priestley duality theorem for Wajsberg algebras. We especially refer to [Itu83] where a topological (Priestley-type) representation theorem for SHn-algebras is given. Note that in [Itu83] only the objects are considered; in this thesis we extend the representation theorem given in [Itu83] to a dual equivalence theorem between suitable categories, and show that the dual equivalence between the category of SHn-algebras and the category of SHn-spaces restricts to a dual equivalence between the category of so-called $SHKn$-algebras (Lukasiewicz-Moisil algebras of order n) and a suitable category of Priestley spaces with operators. We also refer to [CLP91] for some further remarks on the Priestley duality for distributive lattices with unary operators that are join- respectively meet-hemimorphisms, and to [Cig91], where distributive lattices with quantifiers are analyzed and a Priestley duality theorem for this type of structures is given. Goldblatt [Gol89] studies such representation theorems in a more general framework, i.e. for distributive lattices endowed with operators that are join- and meet-hemimorphisms (i.e. maps with arbitrary (finite) arity that preserve joins (resp. meets) in all the components). His research is mainly motivated by the study of algebraic and set-theoretical (Kripke) semantics for propositional modal logics. A modal algebra is a single unary operation on a Boolean algebra, while a Kripke model is a particularly simple kind of relational structure: a single binary relation. Jónsson and Tarski [JT51, JT52] studied varieties of Boolean algebras with operators and the link between these varieties and classes of relational structures. Inspired by Jónsson and Tarski's work, [Gol89] considers distributive lattices endowed with a family of join- and meet-hemimorphisms, and establishes a Priestley duality theorem between the category of distributive lattices with operators and a suitable category of Priestley spaces endowed with relations. Then, relational structures (similar to the Kripke frames in modal logic) are introduced. They are in this case spaces endowed with a family of binary relations with certain properties. The goal of [Gol89] is to explain the extent to which the "modal case" can be seen as the simplest illustration of a general theory that forms a chapter of universal algebra.

Since the research of [Gol89] makes reference to Kripke models, we present the main idea behind this type of models.

Kripke models. Kripke frames were introduced by Kripke for the study of modal logic [Kri63]. A Kripke frame is a set (of "possible worlds") endowed with one relation (called also "accessibility relation") between the possible worlds. Kripke models are Kripke frames endowed additionally with valuations (or meaning functions). It turned out that Kripke-style semantics can be given also for other types of logics – like for instance intuitionistic logic and temporal logic. Generalizations of Kripke frames and models (in the sense that several relations are defined on the set of possible worlds) are used for giving a semantics for the dynamic logic of programs: the relations can be seen as "accessibility relations" between possible worlds, induced by corresponding "programs".

Kripke models can be seen as fibered structures. Usually, one assumes that, given a meaning function, the "local logic" at every possible world (used for evaluating formulae) is classical. There exist attempts of mixing logics by using their Kripke models, in the following sense: given two logics say $\mathcal{L}_1$ and $\mathcal{L}_2$, one considers a Kripke model K_1 of $\mathcal{L}_1$, and at every possible world in K_1, a Kripke model for $\mathcal{L}_2$. This principle is presented in [Gab92, Gab94] and further used for combining logics. A more general approach to combining logics (in a more general framework) appears in [Gab96].

Automated theorem proving in non-classical logic We begin by saying some words about automated theorem proving in classical logic. For propositional logic this reduces to testing all the combinations of truth values 1 (true) and 0 (false) for the variables. In the case of classical first-order logic a classical method for automated theorem proving is the *resolution principle*, due to Robinson [Rob65]. For some refinements of the resolution principle that help in reducing the number of clauses that are generated we refer to [CL73]. Also rewriting techniques have been applied for automated theorem proving. We refer here to the well-known algorithm due to Buchberger [Buc65] for computing Gröbner bases in polynomial ideal theory. This algorithm provides algorithmic solutions to a whole class of problems in polynomial ideal theory and also in various other fields where polynomial ideal theory can be used; the algorithm has been extended also to more general reduction rings [Buc83] and non-associative reduction structures [Sti93].

Also based on the notion of rewriting is the algorithm due to Knuth and Bendix, for solving word problems in equational theories [KB67]. For some of its extensions and refinements we refer to [Hue80, PS81, JK86].

Ideas of rewriting have been applied in automated theorem proving in classical logic. A good overview of the different methods developed can be found in [HKLR92]. The fundamental idea behind the term rewriting approach to equational theorem proving based on resolution is to treat boolean formulae as rewriting rules, and then to apply suitable superposition inferences in order to produce new rules. By means of certain reduction inferences the boolean terms are then simplified using the boolean rules discovered and the process is continued until the contradictory rule $1 \to 0$ is generated.

A first approach based on this idea, due to Hsiang and Dershowitz, appeared in [HD83], and it was followed by the approaches of Kapur and Narendran [KN85], and Hsiang [Hsi87]. In [Buc85], Buchberger presents for the first time the resolution procedure in the framework of general "critical-pair/completion" algorithms, together with the algorithm for computing Gröbner bases and with the Knuth-Bendix completion algorithm. We refer also to the work of Winkler [Win84], where the connection between Buchberger's algorithm for computing Gröbner bases and the Peterson-Stickel algorithm for first-order terms modulo a equational theory is analyzed, and to the work of Stokkermans [Sto95], where the "critical-pair/completion" algorithms are analyzed in a unifying way with the tools of category theory, and where a generic (categorical) "critical-pair/completion" algorithm is proposed.

There exist various attempts to automated theorem proving in non-classical logic. Since non-classical logics are so different in nature, it is natural that these methods are adapted to the specific characteristics of the respective logics. For finitely-valued propositional logics the problem is simple: test all combinations of truth values for the variables that appear. There are however some attempts to improve the efficiency of these methods. [CRAB91] for instance presents an approach where polynomials are associated to formulae in many-valued logics and the proof of validity of a formula reduces to proving that 1 belongs to the ideal generated by the corresoponding set of polynomials.

Concerning non-classical first-order logics, there are various approaches to automated theorem proving. There is no uniform method, and it is not possible to say that there is a "best approach", due to the diversity of non-classical logics.

We distinguish *clausal* and *non-clausal* methods of automated theorem proving. (The methods based on *tableaux systems* are non-clausal, whereas the methods based on *resolution* are clausal.)

The formal proof system called semantic (or analytic) tableaux was introduced by Beth [Bet55, Bet59] (see also [Bet86]) and Hintikka [Hin55], its ancestors being Gentzen systems. Smullyan [Smu68] defined a particularly elegant notion of tableaux which largely increased their popularity. We refer also to [Häh93] for more details on analytic tableaux and tableaux-based provers.

From the approaches based on tableaux systems we would like to mention here the approach of Surma [Sur84], further developed by Carnielli [Car87] (see also [Car91]), the method due to Suchoń [Suc74] for the special case of n-valued logics (which has the advantage of yielding much shorter proofs than the ones obtained by Surma's approach) and the method of Hähnle who defined a generalized notion of signs in tableaux that makes it possible to speak concisely about the truth values a formula can take at a certain stage during the construction of the tableau [Häh90, Häh91, Häh93].

A very active research group in the field of automated theorem proving in non-classical logics (and a former partner of RISC in the MEDLAR project) is the group led by Ricardo Caferra at Leibnitz IMAG in Grenoble. An implementation of a theorem prover based on Carnielli and Surma's approach for propositional logic was developed in [CZ90a]; the propositional part of Carnielli's work is used to implement a theorem prover for some modal logics. From the work of this group we would also like to mention [CZ90b, CHZ91, CZ92, CDH93]. (The diversity of the methods for automated theorem proving in non-classical logics is reflected very well in the work of this group, which has as goal to combine efficiency with generality and user-friendliness, in order to build a large set of user-oriented inference tools.)

Automated theorem proving by resolution in non-classical logic consists of two steps: the first step, *clause generation* takes into account the properties of the given logis, and the second step, *proof by resolution* is a "logic-free" approach, that reduces to a simple manipulation of symbols.

There are two general directions in theorem proving by resolution in non-classical logics, depending on the way the given logic is described: the logic can for instance be described by giving the semantics, or by providing a (Gentzen-type) calculus.

The simplest example of logics described by their semantics are the many-valued logics. These logics are supposed to be sound and complete with respect to a given model (seen as a set of truth values). A method for automated theorem proving in first-order finitely-valued logics, due to Baaz and Fermüller [BF92, BF95] uses (many-valued) resolution. This method is very general, being suitable for all finitely-valued logics; for clause generation only the truth-tables of the operators and the definition of the quantifiers are used. Results along similar lines appear in the work of Hähnle, [Häh94, Häh96b]. Other approaches to automated theorem proving based on resolution (in which a different view is taken) appear in [Mor76] and [Orl78]. Orlowska was the first (to our knowledge) to introduce the notion of "resolution-interpretability of a logic in another logic" and applied it for constructing theorem proving systems for algorithmic and m-valued Post logics [Orl79, Orl80].

There are however situations when a logic is not characterized by a single model, but by a class of models. A method for clause-generation for Gentzen-type logical systems, as well as for the generation of resolution rules is due to Maslov and Mints [Mas64, Min90].

Similar problems also arise in modal logics. Modal logics have two types of models: modal algebras and Kripke models. Among methods that use the Kripke models in automated theorem proving we refer to a method due to Ohlbach [Ohl93], who uses possible worlds semantics to express the validity of formulas, and then encodes the semantics into (classical) first-order logic.

A method for automated theorem proving in several systems of propositional modal logic by rewriting in the equational theories corresponding to the classes of modal algebras that characterize those logics, as well as an implementation, can be found in [Sof88]. However, there are systems of propositional modal logic for which this method cannot be applied because in the process of completion infinitely many rules are generated.

In this thesis we define an automated proof procedure by resolution for logics that are sound and complete with respect to a variety $\mathcal{V}$ of algebras (all with an underlying distributive lattice structure) that has the property that

(i) $\mathcal{V}$ is generated by a single finite algebra A,

(ii) the Priestley duality induces a dual equivalence between the variety $\mathcal{V}$ and a suitable category of ordered topological spaces endowed with functions and possibly also relations.

This means that the logic is sound and complete with respect to the algebra A, which can be seen as a set of truth values. We show that instead of the algebraic model A its dual, which usually has less elements, can be used. Thus, in this case, less clauses will be generated than with the very general procedure described in [BF95].

Models for Cooperating Agents and Concurrency There exist several models for concurency and distributed computation that are used and studied within theoretical computer science. Among these we mention transition systems (which provide the basic operational semantics for Milner's Calculus of Communicating Systems (CSS) [Mil80, Mil89]; for a presentation cf. [WN93]), Petri nets (cf. [Pet62a, Pet62b], [WN93]), trace languages (notion due to Mazurkiewicz [Maz77], cf. also [Die90]), and event structures. Common to all these models is the fact that they rest on the central idea of atomic actions, over which the behavior of the system is defined. The models differ mainly with respect to what behavioral features of the systems are represented. Some models are more abstract than others, and this fact is often used in informal classifications of the models with respect to expressibility.

In [WN93], category theory is used as a common language for the study of the relationships between the models for concurrency mentioned above. By using adjunctions, it is possible to study the links between these models and transfer techniques specific to one model to other models.

We briefly mention some newer approaches: higher dimensional automata [Pra91], partially ordered multisets [Pra82, Pra86, Gis88, Cre91, Cas91], geometric automata (for a brief presentation see [Gup94]), and Chu spaces [Pra94, Gup94].

There also exist approaches to multi-agent systems based on modal logic. We refer for example to [Cos90].

In addition, there already exist a number of approaches based on "fiberings", presheaf and sheaf theory. Among them, [Pfa91], [Gog92], [Mal94], [Lil93], [MP86], [JNW94], [Win96], [CW96]. We will now briefly point out the main ideas of the approaches based on fiber bundles and (pre-)sheaf theory.

Since the starting point, as well as the main motivation, of our work in this direction came from the idea of logical fiberings due to Jochen Pfalzgraf, we start by presenting his approach to modeling cooperating agents scenarios based on logical fiberings.

Approaches to Concurency Based on Fibered Structures. The idea of "logical fiberings" originates from J. Pfalzgraf's work on polycontextural logics, cf. also [Pfa91], [PS95], [Pfa96]. In [PS92], Pfalzgraf sketches a novel approach characterized by a so-called "logical controller" for robotics scenarios cf. also [DPSS91], [PS92]. That approach is based on so-called *logical fiberings* introduced in [Pfa91] as a concept for mathematical modeling of a system of (possibly different) logical spaces (the fibers) residing over a base manifold (index system), forming as a whole the logical fibering. In a series of papers, Pfalzgraf develops the idea of "logical fiberings", having as goal the development of a (non-classical) "fibered logical calculus", by means of which one could construct logical controllers for multi-tasking scenarios in a formal way; in later papers space- and time-depencency of formulas is taken into account (for details see e.g. [Pfa93], [Pfa96], [PS95], [PSS95], [PSS96a], [PSSS95], [PSS96c], [PSS96b]). In [Pfa96], he points out that the notion of a fibering is closely related to indexed systems (indexed categories).

These methods and concepts arose from concrete scenario modeling problems (see [Pfa91]) and have been illustrated, in the frame of the MEDLAR project, on concrete toy examples (see for example [PSS95], [PSS96a], [PS95]).

Presheaf and Sheaf-Theoretical Approaches to Concurrency. Concerning the existing approaches to concurrency based on sheaf semantics, we would like to point out the approach due to Goguen [Gog92], further developed by Lilius [Lil93], Malcolm [Mal94] and Cîrstea [Cîr95]; the approach of Monteiro and Pereira [MP86]; and the approach of Winskel and Cattani [Win96, CW96].

In [Gog92], a sheaf semantics aimed at modeling the behavior of concurrent interacting objects is presented. The approach is based on an earlier paper [Gog75]. These ideas have been applied to Petri nets by Lilius [Lil93]. The ideas from [Gog92] have been developed further by Malcolm in [Mal94], where a formalization of object classes and systems of objects is given, in order to study basic properties of ways in which systems of objects may be interconnected. He defines an adjunction between PO-systems (functors $S : \mathcal{C}^{op} \to \mathsf{Obj}$, where $\mathcal{C}$ is a partially ordered set) and sheaves of objects (PO-systems $S : \mathcal{C}^{op} \to \mathsf{Obj}$ where $\mathcal{C}$ is a complete Heyting algebra), and expresses the hope that, by using a more general notion of sheaf as a functor on a category with a Grothendieck topology, an adjunction between system specifications and sheaves of objects can be obtained. In this thesis we show that (for our definition of a system) a similar adjunction can be defined; moreover we show that this adjunction preserves the covering relation and thus induces a geometric morphism between the corresponding topoi of sheaves over the respective sites. In [Cîr95] Cîrstea shows how transition systems and sheaves can be related by means of an adjunction between the corresponding categories and uses this in giving a sheaf-theoretic formalization of the distributed semantics for the programming language FOOPS developed in [Cîr95].

In [MP86] the authors aim at developing a structural theory of concurrency in which the locality of interaction between subsystems is described with the mathematical tools of sheaf theory. They show that the behavior of a given family of interconnected systems can be modeled by so-called behavior monoids (which form sheaves of monoids). Also possible behaviors are analyzed (prefix-closed languages contained in these free monoids).

[Win96] investigates presheaf models for process calculi with value passing; denotational semantics in presheaf models are shown to correspond to operational semantics in the sense that bisimulation obtained from the so-called "open maps" is proved to coincide with bisimulation as defined traditionally from the operational semantics. A presheaf model and denotational semantics are proposed for a language allowing process-passing. [CW96] is concerned with modeling process constructions on presheaves, showing that these preserve open maps, and with transferring such results to traditional models for processes.

Approaches to logic and the study of modularity. There are several approaches to modularity in logic and automated theorem proving. Among them we mention [BHK90, DGS91, Fia96]. These approaches are intrinsically linked to concurrency, although they are not always presented as approaches to concurrency; the motivation for these approaches came from programming languages and specifications for parallelism. However we were influenced (and helped) in our work by the results presented there. In [BHK90] an axiomatic algebraic calculus of modules is given, based on operators "combination/union", "export", "renaming" and "taking visible signature". Reusability of modules is discussed. In [DGS91] properties of logical systems that support the definition, combination, parametrization and reuse of modules are investigated. Links between the preservation of various kinds of conservative extensions under pushouts, various distributive laws for information hiding over sums, and Craig-style interpolation properties are established. The logical systems are represented by institutions. In [Fia96] a categorical semantics of parallel program design is given.

The starting point and the main motivation of our own work comes from the idea of logical fiberings due to Jochen Pfalzgraf. The notion of logical fiberings introduced by [Pfa91], as well as the general modeling principle based on logical fiberings are very general: to each point of the basis set (corresponding to an agent) the logical system of that agent is associated; communication between different fibers is modeled by so-called transjunctions.

In our approach, we specialize this very general notion, by pointing out a possibility of describing the way *information* and *actions* are represented for every agent, as well as a possible way in which interaction between different agents can be modeled. This offers us the possibility of obtaining a general framework in which several of the existing sheaf-theoretic approaches to concurrency fit in in a natural way.

Due to the fact that we allow the existence of some relationships between the control variables (described by a set of formulae) it turnes out that some of the categories of systems we define (such as $\mathsf{SYS}_\mathsf{i}, \mathsf{SYS}_\mathsf{il}, \mathsf{Sys}(\mathsf{InSys})$; all partially-ordered sets) do not satisfy a distributivity of meets over joins (when these exist). Therefore, we have to introduce a more general notion of Grothendieck topology on these categories (they cannot be seen as locales).

It turns out that much of the information relevant when expressing properties about systems can be expressed by sheaves with respect to this Grothendieck topology:

- states and parallel actions are modeled by sheaves St, Act,
- transitions are expressed by a subsheaf of $Act \times St \times St$,
- time (e.g. $\mathbb{N}$) can be expressed internally as a sheaf (allowing also to express the fact that independent systems may have independent time cycles),
- behavior in a fixed interval of time (e.g. $\mathbb{N}$) can be modeled as a sheaf.

Behavior of systems in time can be expressed either by observations over a "basis of observations" over time (as done in [Gog92]) or as sheaves of monoids [MP86] or partially-commutative monoids [Die90] (we give a sheaf-theoretic formulation and a new proof to the results from [Die90] concerning the study of the partially-commutative monoids and interacting systems).

The possibility of applying general results from topos theory to the study of concurrency is pointed out in [Gog92] as a topic for future research. Since many properties of systems involve statements about their states, actions, transitions, we decided to express these properties in a many-sorted language $\mathcal{L}$ having among its sorts St (for states) and Act (for actions), relations like $=_X \subseteq X \times X$, $Tr \subseteq Act \times St \times St$ etc. We give interpretations of $\mathcal{L}$ in the topoi discussed in the previous chapters, $\mathcal{E} = \mathsf{Sh}(\mathsf{InSys})$ and $\mathcal{F} = \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$, and use geometric logic in order to explain the link between certain properties of a given family of interconnected systems and the properties of the system that results from their interconnection.

Chapter 3

Background

We will now briefly review the main concepts that will be used in our work. In the beginning we present basic notions on universal algebra and (many-sorted) first-order logic. We also present the basic ideas concerning the resolution principle. We continue by giving an overview of the basic definitions in category theory and sheaf theory, and then give the more general definitions for a Grothendieck topology on a category and for sheaves on a site. Finally, we recall some basic properties of topoi and geometric logic. Further details can be found in [ML71], [Joh82], and [MLM92] among others.

3.1 Universal Algebra — Basic Notions

We begin by a brief presentation of lattices. Then we continue by a brief presentation of the basic results in universal algebra: the isomorphism theorem, basic constructions in universal algebras, definitions for varieties and equational classes. We continue then with some brief remarks concerning polynomial and algebraic functions, and then we define discriminator varieties and give some examples. For more details we refer to [BS81] and [MMT87].

3.1.1 Lattices

There are two ways of defining lattices: the first is algebraic, namely as sets endowed with a family of operations that satisfy certain identities, and the other as ordered sets.

We first present the algebraic definition.

Definition 3.1 (Lattice) *A non-empty set L together with two binary operationns $\vee$ and $\wedge$ on L is called* lattice *if it satisfies the following identities:*

(L1)	$x \vee y = y \vee x$	*(L1′)*	$x \wedge y = y \wedge x$
(L2)	$x \vee (y \vee z) = (x \vee y) \vee z$	*(L2′)*	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$
(L3)	$x = x \vee (x \wedge y)$	*(L3′)*	$x = x \wedge (x \vee y)$
(L4)	$x \vee x = x$	*(L4′)*	$x \wedge x = x$

Note that *(L4)* and *(L4′)* are consequences of *(L1)*, *(L1′)*, *(L2)*, *(L2′)*, *(L3)*, *(L3′)*.

The second definition is based on the notion of *partial order*.

Definition 3.2 (Partial Order) *A binary relation $\leq$ on a set P is a* partial order *on P if the following conditions hold for every $x, y, z \in P$:*

(i) $x \leq x$,

(ii) $x \leq y$ *and* $y \leq x$ *implies* $x = y$,

(iii) $x \leq y$ *and* $y \leq z$ *implies* $x \leq z$.

A set P endowed with a partial order is called partially ordered set, or for short poset.

Let P be a poset and $S \subseteq P$. An element $p \in P$ is an *upper bound* for S if $x \leq p$ for every $x \in S$. An element $p \in P$ is the *least upper bound* of S (or *supremum* of S) if p is an upper bound, and $x \leq y$ for every $x \in S$ implies $p \leq y$ (i.e. p is the smallest among the upper bounds of S). In this case we denote $p = \sup S$ or $p = l.u.b(S)$. Similarly we can define the notion of *lower bound* of S and *greatest lower bound* (or *infimum*) of S. If p is the greatest lower bound of S we write $p = \inf S$ or $p = g.l.b(S)$.

Definition 3.3 *A poset L is a* lattice *if and only if for every elements $x, y \in L$ both sup $\{x, y\}$ and inf $\{x, y\}$ exist (in L).*

It is easy to see that the two definitions are equivalent. For more details see also [BS81], pp.3-21.

Definition 3.4 (Distributive Lattice) *A* distributive lattice *is a lattice that satisfies either of the distributive laws (which are equivalent in a lattice):*
(D1) $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$
(D2) $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$

Definition 3.5 (Modular Lattice) *A* modular lattice *is a lattice that satisfies the* modular law*:*
(M) If $x \leq y$ *then* $x \vee (y \wedge z) = y \wedge (x \vee z)$.

Every distributive lattice is modular, but there exist modular lattices that are not distributive. There also exist lattices that are not even modular. Criteria for distributivity and modularity in terms of forbidden substructures, as well as further details concerning complete and algebraic lattices can be found in [BS81], pp.11-21.

Definition 3.6 (Complete Lattice) *A lattice L is* complete *if suprema and infima of arbitrary families of elements exist in L.*

We say that a lattice L has a *first (least) element* if there is an element $0 \in L$ such that $0 \leq x$ for every $x \in L$. A lattice L has a *last (greatest) element* if there is an element $1 \in L$ such that $x \leq 1$ for every $x \in L$.

Definition 3.7 (Pseudocomplement) *Let L be a lattice with first element* 0*. For $x \in L$ an element $x^\wedge$ is called* pseudocomplement of x *if it is the largest element in L such that $x \wedge x^\wedge = 0$.*

Definition 3.8 (Complement with respect to $\vee$) *Let L be a lattice with last element* 1*. The* complement of x with respect to $\vee$*, denoted by $x^\vee$ is the smallest element in L with the property that $x \vee x^\vee = 1$.*

In general it can happen that both the pseudocomplement, $x^\wedge$, and the complement $x^\vee$ of x with respect to $\vee$ exist and are different. If L is distributive, then $x^\wedge \leq x^\vee$. If the two complements are equal, say $x^\wedge = x^\vee = x'$ we say that x has a *complement*, namely x'. In this case,

$$x' \vee x = 1 \text{ and } x' \wedge x = 0.$$

Definition 3.9 (Boolean Algebra) *A* Boolean algebra $B = (B, \vee, \wedge, 0, 1, \neg)$ *is a complemented distributive lattice, i.e. $(B, \vee, \wedge)$ is a distributive lattice with first and last elements* 0 *resp.* 1*, and such that for every $x \in B$, $\neg x$ is the complement of x.*

Definition 3.10 (Relative pseudo-complement) *Let L be a lattice and x, y two elements of L. The pseudo-complement of x relative to y is the largest element $z \in L$ such that $x \wedge z \leq y$. The pseudocomplement of x relative to y is denoted by $x \to y$.*

Definition 3.11 (Relatively pseudo-complemented lattice) *A lattice L is* relatively pseudo-complemented *if for every $x, y \in L$ the pseudo-complement of x relative to y, $x \to y$, exists.*

Every relatively pseudo-complemented lattice includes a greatest element, 1. For every $x \in L$, $1 = x \to x$. Moreover, every relatively pseudocomplemented lattice is distributive. For more details we refer to [Ras74] and the literature quoted there.

Definition 3.12 (Heyting Algebra) *A lattice L is a* Heyting algebra *(also called* pseudo-Boolean algebra *or* pseudo-complemented lattice*) if it is a relatively pseudocomplemented lattice with a smallest and a greatest element* 0 *and* 1*. Then for every $x \in L$ the pseudocomplement $x^\wedge$ of x exists and $x^\wedge = x \to 0$. The definition of the relative pseudocomplement is characterized by*

$$z \leq (x \to y) \textit{ if and only if } z \wedge x \leq y.$$

Definition 3.13 (Complete Heyting Algebra) *A* complete Heyting algebra *is a Heyting algebra which is complete as a lattice, i.e. suprema and infima of arbitrary families of elements exist.*

If A is a complete Heyting algebra and $\{a_i \mid i \in I\}$ is a family of elements in A and $b \in A$, then the *infinite distributive law* holds in A:

$$\bigvee_{i \in I} (b \wedge a_i) = b \wedge \bigvee_{i \in I} a_i.$$

Definition 3.14 (De Morgan Algebra) *A distributive lattice $L = (L, \vee, \wedge)$ endowed with an additional operation $\sim$ is a* de Morgan algebra *if $\sim$ satisfies the following conditions:*

(DM1) $\sim\sim x = x$

(DM2) $\sim (x \vee y) = \sim x \wedge \sim y$

The de Morgan algebras are called also *quasi-boolean algebras* in [Ras74].

We present two more examples of classes of lattices with operators, namely the class of Łukasiewicz-Moisil algebras and the class of Post algebras.

The Łukasiewicz-Moisil algebras (sometimes called Łukasiewicz algebras) were created by Moisil in 1940 ($n = 3$) and 1960 (arbitrary n) [Moi63, Moi65] as an algebraic counterpart for the many-valued logics of Łukasiewicz. However, it turned out that n-valued Łukasiewicz-Moisil algebras are models for the n-valued logics of Łukasiewicz only for $n = 3$ and $n = 4$. Nevertheless, Łukasiewicz-Moisil algebras are an interesting subject of study in themselves, and are models for another class of many-valued logics ($SHKn$-logics) as will be pointed out in Section 5.4.2. In what follows we present an equational definition for Łukasiewicz-Moisil algebras due to Cignoli.

Definition 3.15 (Łukasiewicz-Moisil Algebra) *A Łukasiewicz-Moisil algebra of order n is an algebra $L = (L, \vee, \wedge, \sim, S_1, \dots, S_{n-1}, 0, 1)$ satisfying the following properties:*

(L0) $(L, \vee, \wedge, 0, 1)$ *is a distributive lattice,*

(L1) $\sim\sim x = x; \sim (x \wedge y) = \sim (x) \vee \sim (y)$ *(De Morgan Laws),*

(L2) $S_i(x \wedge y) = S_i(x) \wedge S_i(y); S_i(x \vee y) = S_i(x) \vee S_i(y)$, *for every* $1 \leq i \leq n-1$,

(L3) $S_i(x) \leq S_j(x)$, *for every* $1 \leq i \leq j \leq n-1$,

(L4) $S_i(x) \vee \sim (S_i(x)) = 1, S_i(x) \wedge \sim (S_i(x)) = 0$, *for every* $1 \leq i \leq n-1$,

(L5) $S_i(\sim (x)) = \sim (S_{n-i}(x))$, *for every* $1 \leq i \leq n-1$,

(L6) $S_i(S_j(x)) = S_j(x)$, *for every* $1 \leq i, j \leq n-1$,

(L7) $S_1(x) \leq x \leq S_{n-1}(x)$,

(L8) $\sim (x) \wedge S_1(x) = 0, \sim (x) \vee S_{n-1}(x) = 1$,

(L9) $S_i(0) = 0, S_i(1) = 1$, *for every* $1 \leq i \leq n-1$,

(L10) $y \leq x \wedge \sim (S_i(x)) \wedge S_{i+1}(y)$, *for every* $1 \leq i \leq n-2$.

Example 3.1 (L_n) *The* n-element Łukasiewicz-Moisil algebra *is the algebra*

$$L_n = (\{0, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}, 1\}, \vee, \wedge, \sim, S_1, \dots, S_{n-1}, 0, 1),$$

where $x \vee y = max(x, y), x \wedge y = min(x, y), \sim (x) = 1 - x$, *and for every* $i \in \{1, \dots, n-1\}$, $S_i(\frac{j}{n-1}) = \begin{cases} 1 & if\, i + j \geq n \\ 0 & if\, i + j < n \end{cases}$.

The n-element Post algebras have been introduced by Rosenbloom in 1942 as an algebraic counterpart of Post logics. The initial definition involved a very small number of axioms, and was quite difficult to use. In 1963 Traczyk gave another definition, by means of equations. In what follows we present a definition due to Cignoli, that showed that Post algebras are Lukasiewicz-Moisil algebras endowed with a chain such that some additional conditions are satisfied.

Definition 3.16 (Post Algebra) *A* Post algebra *is an algebra*

$$P = (P, \vee, \wedge, \sim, S_1, \dots, S_{n-1}, 0, 1, e_1, \dots, e_{n-2})$$

such that $(P, \vee, \wedge, \sim, S_1, \dots, S_{n-1}, 0, 1)$ *is a Lukasiewicz algebra and* $e_1 \leq \dots \leq e_{n-2}$ *are n distinguished constants, such that for every* $1 \leq i \leq n-1$ *and every* $0 \leq j \leq n-1$, $S_i(e_j) = \begin{cases} 1 & \text{if } i+j \geq n \\ 0 & \text{if } i+j < n \end{cases}$. *It is convenient to define* $e_0 = 0$ *and* $e_{n-1} = 1$.

Example 3.2 (n-element Post Algebra) *The* n-element Post algebra *is the algebra*

$$P_n = (\{0, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}, 1\}, \vee, \wedge, ', 0, 1),$$

where $x \vee y = max(x, y)$, $x \wedge y = min(x, y)$, *and for every* $x \in \{0, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}, 1\}$, $x' = \begin{cases} x - \frac{1}{n-1} & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases}$.

It can be seen that every finitary map $h : P_n^k \to P_n$ can be expressed in terms of the operations $\{\vee, \wedge, ', 0, 1\}$. In particular, the constants $\{0, \frac{1}{n-1} \dots, \frac{n-2}{n-1}, 1\}$, as well as the operations $\sim x = 1 - x$, $\sim (\frac{j}{n-1}) = 1 - \frac{j}{n-1}$ and for every $i \in \{1, \dots, n-1\}$, $S_i(\frac{j}{n-1}) = \begin{cases} 1 & \text{if } i+j \geq n \\ 0 & \text{if } i+j < n \end{cases}$ can be expressed in terms of the signature $\{\vee, \wedge, ', 0, 1\}$.

It can also be shown that x' can be expressed in function of the operations $\sim$, S_i, $i \in \{1, \dots, n-1\}$, and the constants $\{0, \frac{1}{n-1} \dots, \frac{n-2}{n-1}, 1\}$. Therefore we can regard the n-valued Post algebra as a Post algebra according to definition 3.16

$$P_n = (\{0, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}, 1\}, \vee, \wedge, \sim, S_1, \dots, S_{n-1}, 0, 1, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}).$$

The relationships between different lattices can be expressed by lattice homomorphisms.

Definition 3.17 (Lattice homomorphism) *Let* L_1, L_2 *be two lattices. A map* $h : L_1 \to L_2$ *is a* lattice homomorphism *if* h *is join-preserving and meet-preserving, i.e. if for every* $x, y \in L$, $h(x \vee y) = h(x) \vee h(y)$ *and* $h(x \wedge y) = h(x) \wedge h(y)$. *A bijective homomorphism is a* (lattice-)isomorphism.

If L_1 and L_2 are lattices with both least and greatest element 0 resp. 1, it is often appropriate to consider homomorphisms $h : L_1 \to L_2$ such that $h(0) = 0$ and $h(1) = 1$. Such maps are called $\{0, 1\}$-homomorphisms.

If B_1 and B_2 are Boolean algebras, then a map $h : B_1 \to B_2$ is a morphism of Boolean algebras if h is a $\{0, 1\}$-homomorphism and additionally for every $x \in B_1$, $h(\neg x) = \neg h(x)$.

We continue by briefly presenting some well-known representation theorems for certain classes of lattices. We begin with representation theorems for finite Boolean algebras, continue with a representation theorem for finite distributive lattices, and at the end we mention the Stone representation theorem for Boolean algebras. The Priestley duality theorem for distributive lattices will be presented separately in Section 4.1.4.

Definition 3.18 (Atom) *Let L be a lattice with least element 0. An element $a \in L$ is called an* atom *if $0 < a$ and for every $y \in L$, $0 \leq y < a$ implies $y = 0$. The set of atoms of L is denoted by $\mathcal{A}(L)$.*

It may happen that a lattice has no atoms at all (the chain of non-negative real numbers is such an example). Even a Boolean algebra may have no atoms (e.g. let B be the family of all finite unions of subintervals of $\mathbb{R}$ of the following types: $(-\infty, a), [a, b), [b, \infty)$, where $-\infty < a < b < \infty$, together with $\emptyset$. Then B is a Boolean algebra with no atoms).

Theorem 3.1 (Representation theorem for finite Boolean algebras) *Let B be a finite boolean algebra. Then the map $\eta : B \to \mathcal{P}(\mathcal{A}(B))$, defined by*

$$\eta(a) = \{x \in \mathcal{A}(B) \mid x \leq a\}$$

is a lattice isomorphism between B and $\mathcal{P}(\mathcal{A}(B))$.

Definition 3.19 (Join and meet irreducible elements) *Let L be a lattice. An element $x \in L$ is called* join irreducible *if it cannot be expressed as the join of two other elements in L, i.e. if it has the property that $x = y \vee z$ implies $y = x$ or $z = x$.*

An element $x \in L$ is called meet irreducible *if it has the property that $x = y \wedge z$ implies $y = x$ or $z = x$.*

We denote the set of join-irreducible elements of L by $\mathcal{J}(L)$ and the set of meet-irreducible elements of L by $\mathcal{M}(L)$. Each of these sets inherits the order relation of L and will be regarded as an ordered set.

Theorem 3.2 (Birkhoff's representation of finite distributive lattices) *Let L be a finite distributive lattice. Let $\eta : L \to \mathcal{O}(\mathcal{J}(L))$ be defined by*

$$\eta(a) = \{x \in \mathcal{J}(L) \mid x \leq a\}.$$

Then η is a lattice isomorphism between L and $\mathcal{O}(\mathcal{J}(L))$.

Theorem 3.3 *Let P be a finite ordered set. Then the map $\epsilon : P \to \mathcal{J}(\mathcal{O}(P))$ defined by $\epsilon(x) = \downarrow x$ is an order-isomorphism*[1] *from P onto $\mathcal{J}(\mathcal{O}(P))$.*

Theorems 3.2 and 3.3 assert that

$$L \simeq \mathcal{O}(\mathcal{P}(L)) \text{ and } P \simeq \mathcal{J}(\mathcal{O}(P))$$

for all finite distributive lattices L and all finite partially ordered sets P.

Note that for every distributive lattice L the partially ordered set $\mathcal{J}(L)$ is generally much smaller and less complex than the lattice itself. This means that lattice problems concerning finite distributive lattices are likely to become simpler when translated into problems about finite partially ordered sets. We may regard the maps $L \mapsto \mathcal{J}(L)$ and $P \mapsto \mathcal{O}(P)$ as playing a rôle analogous to that of the logarithm and exponential functions. For more details we refer to [DP90], p.172.

We now briefly present the representation theorem for Boolean algebras due to Stone. The representation theorem for distributive lattices due to Priestley will be presented in Section 4.1.4.

Definition 3.20 (Ideal) *Let L be a lattice. A non-empty subset J of L is called an* ideal *if*

(I1) $x, y \in J$ imply $x \vee y \in J$,

(I2) $x \in L$, $y \in J$ and $x \leq y$ imply $x \in J$.

A non-empty subset F of L is called a filter *if*

(F1) $x, y \in J$ imply $x \wedge y \in J$,

(F2) $x \in L$, $y \in J$ and $y \leq x$ imply $x \in J$.

Thus, an ideal is a non-empty down-set closed under join and a filter is a non-empty upwards-closed set closed under meet. An ideal or filter is called *proper* if it does not coincide with L.

For every $x \in L$ the set $\downarrow x = \{y \in L \mid y \leq x\}$ is an ideal (the *principal ideal generated by x*); dually, the set $\uparrow x = \{y \in L \mid x \leq y\}$ is a filter (the *principal filter generated by x*). Given any non-empty subset A of L there is a smallest ideal containing A, namely

$$(A] = \{x \in L \mid x \leq \bigvee T \text{ for some finite subset } T \text{ of } A\}.$$

Similarly, the smallest filter containing A is

$$[A) = \{x \in L \mid x \geq \bigvee T \text{ for some finite subset } T \text{ of } A\}.$$

Definition 3.21 (Prime Ideal, Prime Filter) *Let L be a lattice and J a proper ideal in L. The ideal J is said to be* prime *if*

(PI) $x, y \in L$ and $x \wedge y \in J$ imply $x \in J$ or $y \in J$.

The set of prime ideals of L is denoted by $\mathcal{I}_p(L)$. It is ordered by set inclusion. A prime filter is defined dually, i.e. it is a filter that satisfies

(PF) $x, y \in L$ and $x \vee y \in J$ imply $x \in J$ or $y \in J$.

The set of prime filters is denoted by $\mathcal{F}_p(L)$.

[1] An order-isomorphism between two partially ordered sets is a bijective map that preserves the order.

Theorem 3.4 (Stone's representation theorem for Boolean algebras) *Let B be a Boolean algebra. Let $X = \mathcal{I}_p(B)$ be the set of prime ideals of B endowed with the topology τ generated by $\mathcal{B} = \{X_a \mid a \in B\}$ as a basis, where for every $a \in B$, $X_a = \{I \in \mathcal{I}_p(B) \mid a \notin I\}$. Then the following hold:*

(1) Each element of $\mathcal{B}$ is clopen in $\mathcal{I}_p(B)$ (because $X \backslash X_a = X_{a'}$).

(2) Every clopen subset of (X, τ) is of the form X_a for some $a \in B$.

(3) (X, τ) is a compact totally disconnected[2] topological space.

(4) The map $\eta : B \to \mathcal{B}$ defined by $\eta(a) = X_a$ is a Boolean algebra isomorphism between B and the Boolean algebra $\mathcal{B} = \{X_a \mid a \in B\}$ of clopen subsets of the space (X, τ).

We briefly note that the set of all prime filters of a lattice L (as well as the set of all prime ideals of L) is in bijective correspondence with the set of all $\{0,1\}$-lattice homomorphisms from L to the lattice with 2 elements $\{0,1\}$.

Namely, for every $\{0,1\}$-lattice homomorphism $h : L \to \{0,1\}$, $h^{-1}(0)$ is a prime ideal and $h^{-1}(1)$ is a prime filter. Conversely, if I is a prime ideal of L, then the map $h : L \to \{0,1\}$ defined by $h(x) = 0$ if and only if $x \in I$ is a $\{0,1\}$-lattice homomorphism. Similarly, if F is a prime filter of L, then the map $h : L \to \{0,1\}$ defined by $h(x) = 1$ if and only if $x \in F$ is a $\{0,1\}$-lattice homomorphism.

These results are used in Section 4.1.4, where Priestley's representation theorem for distributive lattices is presented.

3.1.2 General Notions of Universal Algebra

Let Σ be a signature, i.e. a set of operation symbols endowed with an arity function $a : \Sigma \to \mathbb{N}$. A Σ-algebra is a structure $A = (A, \{\sigma_A\}_{\sigma \in \Sigma})$, where for every $\sigma \in \Sigma$ with $a(\sigma) = n$, $\sigma_A : A^n \to A$. We also say that the algebra A is of *similarity type* Σ.

Definition 3.22 (Subalgebra) *Given two Σ-algebras $A = (A, \{\sigma_A\}_{\sigma \in \Sigma})$ and $B = (B, \{\sigma_B\}_{\sigma \in \Sigma})$, we say that A is a* subalgebra *of B if $A \subseteq B$ and for every $\sigma \in \Sigma$ and every $a_1, \dots, a_{a(\sigma)} \in A$, $\sigma_A(a_1, \dots, a_{a(\sigma)}) \in A$.*

If A, B are two algebras of the same similarity type, then "B is a subalgebra of A" will be symbolised by $B \leq A$.

A *subuniverse* of $A = (A, \{\sigma_A\}_{\sigma \in \Sigma})$ is a subset B of A which is closed under the fundamental operations of A, i.e. if $\sigma \in \Sigma$ with $a(\sigma) = n$ and $a_1, \dots, a_n \in B$ then $\sigma_A(a_1, \dots, a_n) \in B$. The set of all subuniverses of A is denoted $\mathsf{Sub}(A)$.

Definition 3.23 (Subuniverse of A generated by X) *Let A be an algebra and $X \subseteq A$. Let $Sg(X) = \bigcap\{B \mid X \subseteq B$ and B is a subuniverse of $A\}$. $Sg(X)$ is* the subuniverse of A generated by *X.*

[2]A topological space is *totally disconnected* if, given distinct points $x, y \in X$, there exist a clopen subset V of X such that $x \in V$ and $y \notin V$.

$\mathsf{Sg}(X) = (Sg(X), \{\sigma_{Sg(X)}\}_{\sigma\in\Sigma})$ is a subalgebra of A (the subalgebra of A generated by X), where $\sigma_{Sg(X)}$ is the restriction of σ_A to $Sg(X)$. For $X \subseteq A$ we say that X *generates* A (or A is generated by X, or X is a set of generators of A) if $Sg(X) = A$.

Definition 3.24 (Homomorphism) *Let A and B be two Σ-algebras. A mapping $h : A \to B$ is called a* homomorphism *(or shortly* morphism*) from A to B if $h(\sigma_A(a_1, \ldots, a_n)) = \sigma_B(h(a_1), \ldots, h(a_n))$ for every n-ary operation symbol $\sigma \in \Sigma$ and all $a_1, \ldots, a_n \in A$. If, in addition, the mapping h is onto, then B is said to be a* homomorphic image *of A.*

The set of all equivalence relations of a given set A is denoted by $Eq(A)$.

Definition 3.25 (Congruence) *A* congruence *of a Σ-algebra A is an equivalence relation θ with the property that for every n-ary operation symbol $\sigma \in \Sigma$ and every elements $a_i, b_i \in A$, if $a_i\theta b_i$ holds for $1 \le i \le n$ then*

$$\sigma_A(a_1, \ldots, a_n)\theta\sigma_A(b_1, \ldots, b_n).$$

The compatibility property is an obvious condition for introducing an algebraic structure on the set A/θ of equivalence classes of A with respect to θ; an algebraic structure which is inherited from the algebra A: for every n-ary operation symbol $\sigma \in \Sigma$ and every n-tuple of equivalence classes $[a_i]_\theta, 1 \le i \le n$, $\sigma_{A/\theta}([a_1]_\theta, \ldots, [a_n]_\theta) = [\sigma_A(a_1, \ldots, a_n)]_\theta$.

The set of all *congruences* of an algebra A is denoted by $Con(A)$. The congruence lattice of A, denoted by $\mathsf{Con}A$, is the lattice whose universe is $Con(A)$ and meets and joins are calculated the same as when working with equivalence relations.

We will denote by Δ_A the identity congruence on A, $\Delta_A = \{(a, a) \mid a \in A\}$ and by ∇_A the trivial congruence on A, $\nabla_A = A \times A$. It is easy to see that Δ_A is the smallest congruence on A and ∇_A is the largest congruence on A.

If θ is a congruence on A, there is a canonical onto homomorphism $p : A \to A/\theta$ defined by $p(a) = [a]_\theta$ for every $a \in A$.

Let $h : A \to B$ be a homomorphism. Then the *kernel* of h, denoted $ker(h)$, is defined by $ker(h) = \{(a_1, a_2) \mid h(a_1) = h(a_2)\}$. The kernel of a homomorphism h is a congruence on A.

Theorem 3.5 (First Isomorphism Theorem) *Let $h : A \to B$ be a homomorphism onto B. Then there is a isomorphism g from $A/ker(h)$ to B such that $h = g \circ p$, where p is the natural homomorphism from A to $A/ker(h)$.*

The first isomorphism theorem is also called *the homomorphism theorem.*

Assume that A is an algebra and $\phi, \theta \in Con(A)$ with $\theta \subseteq \phi$. Let $\phi/\theta = \{([a]_\theta, [b]_\theta) \in (A/\theta)^2 \mid (a, b) \in \phi\}$. Then ϕ/θ is a congruence on A/θ and the following theorem holds.

Theorem 3.6 (Second Isomorphism Theorem) *If $\phi, \theta \in Con(A)$ and $\theta \subseteq \phi$ then the map $h : (A/\theta)/(\phi/\theta) \to A/\phi$ defined by $h([[a]_\theta]_{\phi/\theta}) = [a]_\phi$ is an isomorphism from $(\mathcal{A}/\theta)/(\phi/\theta)$ to $\mathcal{A}/\phi$.*

Let B be a subset of A and θ a congruence on A. Let $B^\theta = \{a \in A \mid B \cap [a]_\theta \neq \emptyset\}$. Let $\mathbf{B}^\theta$ be the subalgebra of A generated by B^θ. Also, define $\theta_{|B} = \theta \cap B^2$, the restriction of θ to B. If B is a subalgebra of $\mathcal{A}$, then the universe of $\mathbf{B}^\theta$ is B^θ and $\theta_{|B}$ is a congruence on $\mathcal{B}$.

Theorem 3.7 (Third Isomorphism Theorem) *If B is a subalgebra of $\mathcal{A}$ and $\theta \in Con(\mathcal{A})$, then $B/\theta_{|B}$ is isomorphic with $B^\theta/\theta_{B^\theta}$*

Let L be a lattice, and let $a, b \in L$, $a \leq b$. The interval $[a, b] = \{c \in L \mid a \leq c \leq b\}$ is a subuniverse of L.

Theorem 3.8 (Correspondence Theorem) *Let A be an algebra and let $\theta \in Con(A)$. Then the mapping $h : [\theta, \nabla_A] \to \mathsf{Con}(\mathsf{A}/\theta)$ defined by $h(\phi) = \phi/\theta$ is a lattice isomorphism.*

Definition 3.26 (Simple Algebra) *An algebra A is* simple *if $Con(A) = \{\Delta, \nabla\}$.*

Sometimes one requires simple algebras to be non-trivial; following [BS81] we admit trivial simple algebras.

Definition 3.27 (Maximal Congruence) *A congruence θ on an algebra A is* maximal *if the interval $[\theta, \nabla]$ in $Con(A)$ has exactly two elements.*

From Theorem 3.8 the following result follows immediately.

Theorem 3.9 *Let $\theta \in Con(A)$. Then A/θ is a simple algebra if and only if θ is a maximal congruence on A or $\theta = \nabla$.*

Let $\{A_i\}_{i \in I}$ be an indexed family of Σ-algebras.

Definition 3.28 (Direct Product) *The* direct product *$A = \prod_{i \in I} A_i$ of the family $\{A_i\}_{i \in I}$ is an algebra with universe $\prod_{i \in I} A_i$ and such that for every n-ary operation symbol $\sigma \in \Sigma$ and $a_1, \ldots, a_n \in \prod_{i \in I} A_i$, $f_A(a_1, \ldots, a_n)(i) = f_A(a_1(i), \ldots, a_n(i))$, i.e. f_A is defined component-wise.*

We have the projection maps $\pi_j : \prod_{i \in I} A_i \to A_j$ for $j \in I$ defined by $\pi_j(a) = a(j)$ which give surjective homomorphisms $\pi_j : \prod_{i \in I} A_i \to A_j$.

Definition 3.29 (Subdirect Product) *An algebra A is a* subdirect product *of an indexed family $\{A_i\}_{i \in I}$ of Σ-algebras if $A \leq \prod_{i \in I} A_i$ and $\pi_i(A) = A_i$ for every $i \in I$.*

An embedding $h : A \to \prod_{i \in I} A_i$ is *subdirect* if $h(A)$ is a subdirect product of the family $\{A_i\}_{i \in I}$. If $\theta_i \in Con(A)$ for $i \in I$, and $\bigcap_{i \in I} \theta_i = \Delta$, then the natural homomorphism $p : A \to \prod_{i \in I} A/\theta_i$ is a subdirect embedding.

Definition 3.30 (Subdirectly Irreducible Algebra) *An algebra A is* subdirectly irreducible *if for every subdirect embedding $p : A \to \prod_{i \in I} A_i$, there is an $i \in I$ such that $\pi_i \circ p : A \to A_i$ is a isomorphism.*

The following characterization of subdirectly irreducible algebras is very useful in practice:

Theorem 3.10 *An algebra A is subdirectly irreducible if and only if A is trivial or there is a minimum congruence in $Con(A)\backslash\{\Delta\}$. In the latter case the minimum element is $\bigcap(Con(A)\backslash\{\Delta\}$, a principal congruence, and the congruence lattice of A has the form in Figure 3.1.*

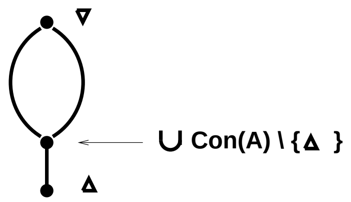


Figure 3.1: The lattice of congruences of a subdirectly irreducible lattice.

Theorem 3.11 (Birkhoff) *Every algebra A is isomorphic to a subdirect product of subdirectly irreducible algebras (which are homomorphic images of A).*

We can define the following operators, which map classes of algebras to classes of algebras (all of the same type):

$A \in I(K)$	iff	A is isomorphic to some member of K,
$A \in S(K)$	iff	A is a subalgebra of some member of K,
$A \in H(K)$	iff	A is a homomorphic image of some member of K,
$A \in P(K)$	iff	A is a direct product of a nonempty family of algebras in K,
$A \in P_s(K)$	iff	A is a subdirect product of a nonempty family of algebras in K.

Definition 3.31 (Variety) *A nonempty class of algebras of the same similarity type is called a* variety *if it is closed under subalgebras, homomorphic images and direct products.*

Since the intersection of a class of varieties of similarity type Σ is again a variety, and as the class of all algebras of similarity type Σ forms a variety, it follows that for every class K of algebras there is a smallest variety containing K. Let $V(K)$ be the smallest variety containing K. A result due to Tarski states that $V = HSP$ (hence, for every class of algebras of the same similarity type, $V(K) = HSP(K)$). It can be also shown that $V = HP_s$.

Given an algebra A there are usually many functions besides the fundamental operations which are compatible with the congruences on A and which "preserve" subalgebras of A. The most obvious such functions of this type are those obtained by composition of the fundamental operations. This leads to the study of terms.

Definition 3.32 (Terms) *Given a set Σ of operation symbols and a set X of (distinct) objects called variables, we denote by $T_\Sigma(X)$ the set of* terms *of type Σ over X, i.e. the smallest set such that:*

(i) $X \cup \Sigma_0 \subseteq T_\Sigma(X)$,

(ii) If $p_1, \dots, p_n \in T_\Sigma(X)$ *and* $\sigma \in \Sigma_n$, *then the "string"* $\sigma(p_1, \dots, p_n) \in T_\Sigma(X)$.

where for every $i \in \mathbb{N}$, Σ_i *is the set of operation symbols in* Σ *of arity* i.

For $p \in T_\Sigma(X)$ we often write $p(x_1, \dots, x_n)$ to indicate that all the variables occurring in p are *among* $x_1, \dots, x_n$. A term p is *n-ary* if the number of variables appearing explicitly in p is $\leq n$.

The *term algebra* of type Σ over X will be denoted $\mathsf{T}_\Sigma(X)$.

The term algebra of type Σ over X, $\mathsf{T}_\Sigma(X)$, satisfies the following universality property: For every Σ-algebra $\mathcal{A}$ and every map $f : X \to A$ there is a unique homomorphism of Σ-algebras $\overline{f} : \mathsf{T}_\Sigma(X) \to A$ with the property that $\overline{f}_{|X} = f$.

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d] & A \\ \mathsf{T}_\Sigma(X) \arrow[ur, "\exists!\overline{f}"'] & \end{tikzcd}$$

Let A be an algebra of similarity type Σ. Let $p(x_1, \dots, x_n) \in T_\Sigma(X)$ be a term. Then p defines a mapping $p^A : A^n \to A$.

An *identity* of type Σ over X is an expression of the form $p_1 = p_2$, where $p_1, p_2 \in T_\Sigma(X)$. Let $Id(X)$ be a set of identities of type Σ over X. An algebra of type Σ *satisfies* an identity

$$p_1(x_1, \dots, x_n) = p_2(x_1, \dots, x_n)$$

(or the identity is *true* in A, or *holds* in A), abbreviated by

$$A \models p_1(x_1, \dots, x_n) = p_2(x_1, \dots, x_n) \text{ (or } A \models p_1 = p_2),$$

if for every choice of $a_1, \dots, a_n \in A$ we have $p_1^A(a_1, \dots, a_n) = p_2^A(a_1, \dots, a_n)$.

A class K of algebras satisfies $p_1 = p_2$ (written $K \models p_1 = p_2$) if each member of K satisfies $p_1 = p_2$. If Id is a set of identities, we say that K satisfies Id (written $K \models Id$) if $K \models p_1 = p_2$ for every $p_1 = p_2 \in Id$.

Let K be a class of algebras, let $q_1 = q_2$ be an identity and Id a set of identities. If from $K \models Id$ it follows that $K \models q_1 = q_2$, we write $Id \models_K q_1 = q_2$.

Given K and X let

$$Id_K(X) = \{p_1 = p_2 \in Id(X) \mid K \models p_1 = p_2\}.$$

Let Id be a set of identities of type Σ. Let $M(Id)$ be the class of algebras that satisfy Id.

A class K of algebras is an *equational class* if there is a set Id of identities such that $K = M(Id)$. In this case we say that K is defined or axiomatized by Id.

Theorem 3.12 (Birkhoff) *K is an equational class if and only if K is a variety.*

Definition 3.33 (Freely generated algebras) *Let K be a class of Σ-algebras and let X be a set of (distinct) objects called variables. Let $F(X)$ be a Σ-algebra generated by X. If for every $A \in K$ and for every map $f : X \to A$ there is a homomorphism $\overline{f} : F(X) \to A$ which extends f (i.e., $\overline{f}(x) = f(x)$ for every $x \in X$), then we say that $F(X)$ has* the universal mapping property for K over X, *X is called a set of* free generators *of $F(X)$, and $F(X)$ is said to be* freely generated *by X.*

Let K be a family of Σ-algebras and let X be a set of variables. Let θ_K be defined by $\theta_K(X) = \bigcap \Phi_K(X)$, where $\Phi_K(X) = \{\phi \in Con(T_\Sigma(X)) \mid T_\Sigma(X)/\phi \in IS(K)\}$.

Let $\overline{X} = X/\theta_K(X)$. Then $F_K(\overline{X}) = T_\Sigma(X)/\theta_K(X)$.

Theorem 3.13 (Birkhoff) *Suppose that $T_\Sigma(X)$ exists*[3]*. Then $F_K(\overline{X})$ has the universal mapping property for K over $\overline{X}$.*

It can be shown (cf. e.g. [BS81] p.68) that $F_K(\overline{X}) \in ISP(K)$. Hence, if K is closed under I, S, and P, in particular if K is a variety, then $F_K(\overline{X}) \in K$.

Given a class K of Σ-algebras and terms $p, q \in T_\Sigma(X)$ it can easily be seen ([BS81] p.73) that $K \models p = q$ if and only if $(p, q) \in \theta_K(X)$. Thus, $Id_X(K) = \{(p, q) \in T_\Sigma(X)^2 \mid K \models p = q\} = \{(p, q) \in T_\Sigma(X)^2 \mid (p, q) \in \theta_K(X)\} = \theta_K(X)$.

In particular, let $\mathcal{V}$ be a variety. The free algebra in $\mathcal{V}$ freely generated by X, $F_\mathcal{V}(\overline{X}) = T_\Sigma(X)/\theta_K(X)$ is then isomorphic to $F_\Sigma(X)/Id_X(V)$.

3.1.3 Polynomial Functions, Algebraic Functions

Let A be a Σ-algebra and X a set. Then A^{A^X} can be made into an Σ-algebra in the canonical way, defining the operations pointwise.

Definition 3.34 *Let x be an element of X. Then the function $p_x : A^X \longrightarrow A$ defined by $p_x((a_y)_{y \in X}) = a_x$ is called* the x-th projection function.

Definition 3.35 *Let $P^X(A)$ be the subalgebra of A^{A^X} generated by the projection functions $(p_x)_{x \in X}$. Then $P^X(A)$ is the algebra of* polynomial functions in the variables X on A.

Definition 3.36 *Let $F^X(A)$ be the subalgebra of A^{A^X} generated by the projection functions and by the constant functions. Then $F^X(A)$ is the algebra of* algebraic functions in the variables X on A.

If $card(X) = n \in \mathbb{N}$, then $A^{A^X} \cong A^{A^n}$. Then P^n(A), the subalgebra of A^{A^n} generated by the projection functions $\{p_i \mid i = 1, \ldots, n\}, p_i \in A^{A^n}$, is *the algebra of n-ary polynomial functions* on A and F^n(A), the subalgebra of A^{A^n} generated by the projection functions and by the constant functions, is *the algebra of n-ary algebraic functions* on A.

[3] $T_\Sigma(X)$ exists iff $X \neq \emptyset$ or $\Sigma_0 \neq \emptyset$.

Remark 3.14 For every $p \in \mathsf{T}_\Sigma(X)$ there exists $n \in \mathbb{N}$ and $x_1, ..., x_n \in X$ such that $p \in \mathsf{T}_\Sigma(\{x_1, ..., x_n\})$. Let $f : \{x_1, ..., x_n\} \longrightarrow P^n(A)$, $f(x_i) = p_i$. Then by the universality property of $\mathsf{T}_\Sigma(\{x_1, ..., x_n\})$ there is a unique morphism $\overline{f} : \mathsf{T}_\Sigma(\{x_1, ..., x_n\}) \longrightarrow P^n(A)$ which extends f. The image by $\overline{f}$ of the term p will be denoted $\overline{f}(p) = p_A \in P^n(A)$ and will be called the *polynomial function associated to* p.

So, to each term corresponds a n-ary polynomial function. Reciprocally, any polynomial function depends in fact only on a finite number of variables.

Let $h : A^n \to A$ be a n-ary function on A, and $p \in T_\Sigma(\{x_1, \ldots, x_n\})$. We say that h is *represented* by the term p if $h = p_A \in P^n(A)$.

h is a *polynomial function* if there exists a term $p \in T_\Sigma(\{x_1, \ldots, x_n\})$ such that h is represented by p (i.e. $h = p_A$).

Let $p \in \mathsf{T}_\Sigma(\{x_1, ..., x_n\})$, and let $a_1, ..., a_n$ be arbitrary elements of A. Let $f : \{x_1, ..., x_n\} \longrightarrow A$ be defined by $f(x_i) = a_i$ for $i = 1, ..., n$. Then there is a unique morphism $\overline{\overline{f}} : \mathsf{T}_\Sigma(\{x_1, ..., x_n\}) \longrightarrow A$ which extends f. Note that $\overline{\overline{f}}(p)$ is in fact $p_A(a_1, ..., a_n)$, where p_A is the polynomial function associated to p (easy proof by structural induction on the structure of p) .

For a given variety $\mathcal{V}$, the free algebra in $\mathcal{V}$ freely generated by X will be denoted $\mathsf{F}_\mathcal{V}(X)$.

Remark 3.15 Let $\mathsf{T}_\Sigma(X)$ be the term algebra of type Σ over X. Let $f : X \longrightarrow P^X(A)$, $f(x) = p_x$. By the universality property of $\mathsf{T}_\Sigma(X)$ there is a unique morphism $\overline{f} : \mathsf{T}_\Sigma(X) \longrightarrow P^X(A)$ which extends f. For any $p \in \mathsf{T}_\Sigma(X)$, $\overline{f}(p)$ is the *X-variate polynomial associated to p.*

$$\begin{aligned} ker\overline{f} &= \{(p_1, p_2) \in \mathsf{T}_\Sigma(X)^2 \mid \overline{f}(p_1) = \overline{f}(p_2)\} = \\ &= \{(p_1, p_2) \in \mathsf{T}_\Sigma(X)^2 \mid \overline{v}(p_1) = \overline{v}(p_2) \text{ for all } v : X \to A\} = \\ &= \{(p_1, p_2) \in \mathsf{T}_\Sigma(X)^2 \mid A \models p_1 = p_2\} = Id_X(\mathrm{A}) \end{aligned}$$

$Im\overline{f} = P^X(A)$ (because $f(X)$ generates $P^X(A)$) and so $P^X(A)$ and $\mathsf{T}_\Sigma(X)/Id_X(A)$ are isomorphic. Note that $\mathsf{T}_\Sigma(X)/Id_X(A)$ is isomorphic with $F_{V(A)}(X)$. So $P^X(A) \cong F_{V(A)}(X)$.

The next subsection gives the basic definitions for the so-called discriminator varieties, which will be used later in the thesis.

3.1.4 Discriminator Varieties

Definition 3.37 (Discriminator) *A* discriminator function *on a set A is a function $t : A^3 \to A$ defined by*

$$t(a, b, c) = \begin{cases} a & \text{if } a \neq b \\ c & \text{if } a = b \end{cases}$$

Definition 3.38 (Switching Function) *A* switching function *on a set A is a function $s : A^4 \to A$ defined by*

$$s(a, b, c, d) = \begin{cases} c & \text{if } a = b \\ d & \text{if } a \neq b \end{cases}$$

If A is an algebra then a ternary term $t(x, y, z)$ representing the discriminator function on A is called a *discriminator term* on A. A term $s(x, y, u, v)$ representing the switching function on A is called a switching term for A.

It is easy to see that from a discriminator term we can construct a switching term and vice-versa:

$$s(x, y, u, v) = t(t(x, y, u), t(x, y, v), v)$$

$$t(x, y, z) = s(x, y, z, x)$$

Therefore an algebra has a discriminator term if and only if it has a switching term. We also know (cf. [BS81], p.165) that an algebra with a discriminator term is simple.

Examples of algebras with a discriminator term:

(1) Let $(H, \vee, \wedge, \Rightarrow, 0, 1)$ be a Heyting algebra with an additional unary operation $d : H \to H$, such that $d(x) = \begin{cases} 1 & \text{if } x = 1 \\ 0 & \text{if } x \neq 1 \end{cases}$

A ternary discriminator is $t(x, y, z) = [z \wedge d(x \vee y \Rightarrow x \wedge y)] \vee [x \wedge (d(x \vee y \Rightarrow x \wedge y) \Rightarrow 0)]$.

(2) The 2-element Boolean algebra $(d(x) = x)$.

(3) The n-element Lukasiewicz algebra $(d(x) = S_1(x))$.

(4) The n-element Post algebra $(d(x) = S_1(x))$.

(5) A Heyting algebra with a join-irreducible 1 and with a dual pseudocomplementation $+$ $(d(x) = x^{++})$.

Definition 3.39 (Discriminator Variety) *Let K be a class of algebras with a common discriminator term. Then the variety generated by K is called a* discriminator variety.

In the thesis we will use two important properties of discriminator varieties (cf. e.g. [BS81] p.165).

(1) The subdirectly indecomposable elements of a discriminator variety are simple (discriminator varieties are *semisimple*).

(2) For every algebra in a discriminator variety, the intersection of all its maximal congruences is Δ.

Examples of discriminator varieties:

(1) The equational class of all algebras $H = (H, \vee, \wedge, +, \Rightarrow, 0, 1)$, where $(H, \vee, \wedge, \Rightarrow, 0, 1)$ is a Heyting algebra and $+$ is a dual pseudocomplementation.

(2) The equational class of all Boolean algebras.

(3) The equational class of all Łukasiewicz algebras of order n.

(4) The equational class of all Post algebras of order n.

3.2 Logic — Basic Notions

In this section we present the basic notions of logic that will be needed in what follows. Since in the thesis we will not restrict ourselves to classical logics, but will consider more general logical systems, we will present the facts in a very general framework. For details on classical first-order logic we refer to [Mon76]; for a proof-theoretic approach we refer to [Tak75]; for more information on non-classical logics we refer to [Ras74]. For an introduction to algebraic logic containing many motivational comments see [ANSK94], [AN93], [ANS94] and [Ném94]. Also, a presentation of general logics can be found in [Mes89]; for details about institutions (that are not described here) we refer to [GB85].

We begin with a very general presentation of the notion of "logic". We continue with an overview of some of the main properties of classical first-order logic. We then sketch the links between logic and algebra and make some model-theoretical considerations.

3.2.1 Generalities

Roughly speaking, we can think of a logic $\mathcal{L}$ as a five-tuple

$$\mathcal{L} = (F_{\mathcal{L}}, \vdash_{\mathcal{L}}, M_{\mathcal{L}}, mng_{\mathcal{L}}, \models_{\mathcal{L}})$$

where

- $F_{\mathcal{L}}$ is a set, called the set of all *formulae* of $\mathcal{L}$,
- $\vdash_{\mathcal{L}}$ is a binary relation between sets of formulae and individual formulae, i.e. $\vdash_{\mathcal{L}} \subseteq \mathcal{P}(F_{\mathcal{L}}) \times F_{\mathcal{L}}$ (for every set X, $\mathcal{P}(X)$ denotes the powerset of X). $\vdash_{\mathcal{L}}$ is called the *provability relation* of $\mathcal{L}$,
- $M_{\mathcal{L}}$ is a class, called the class of all *models* of $\mathcal{L}$,
- $mng_{\mathcal{L}}$ is a function with domain $F_{\mathcal{L}} \times M_{\mathcal{L}}$, called the meaning function of $\mathcal{L}$,
- $\models_{\mathcal{L}}$ is a binary relation, $\models_{\mathcal{L}} \subseteq M_{\mathcal{L}} \times F_{\mathcal{L}}$, called the *validity relation* of $\mathcal{L}$.

In the existing logics, the set $F_{\mathcal{L}}$ of formulae is defined by specifying a language and rules for constructing "well-formed" formulae, and $\vdash_{\mathcal{L}}$ is defined for example by a set of axioms and inference rules, or by a sequent calculus. The rules that define $F_{\mathcal{L}}$ and $\vdash_{\mathcal{L}}$ can be seen as "grammatical rules".

The class of models can contain very different types of models (we also allow the possibility that $M_{\mathcal{L}}$ is empty, in which case the logic is not endowed with a semantics), and the meaning function associates with every formula ϕ and model M the meaning $mng_{\mathcal{L}}(\phi, M)$ of ϕ in M. We did not explicitly specify the codomain of $mng_{\mathcal{L}}$, in order not to impose restrictions on the notion of "meaning". For instance, in the particular case of the logics considered in Chapter 5 two types of models are considered: algebraic models (of the form (A, v), where A is an algebra over a suitable signature and v a function that assigns

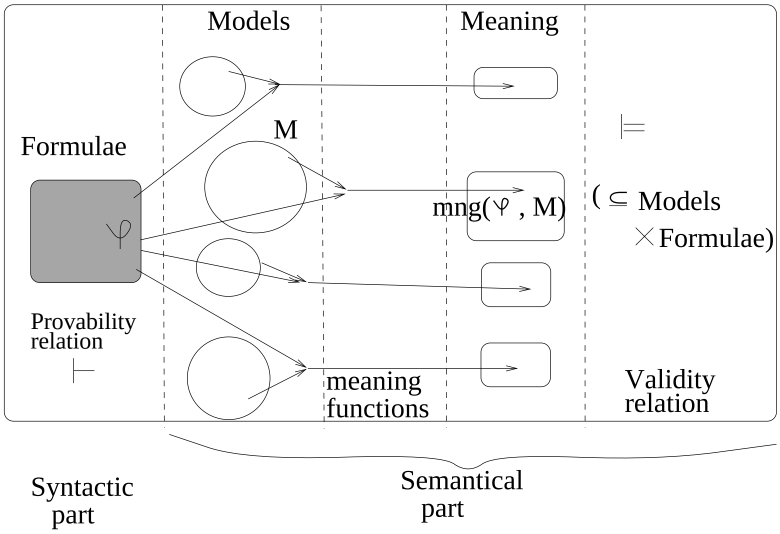


Figure 3.2: The general pattern of a logic.

values in A to the propositional variables) and relational (or Kripke) models (of the form (K, m), where K is a set of "possible worlds" or "states" endowed with certain operations and relations and m a function that assigns subsets of states to the propositional variables). The meaning of a formula in an algebraic model is an element of the corresponding algebra, whereas the meaning of a formula in a relational model is a set of possible worlds (intuitively, the set of those worlds where the formula is true).

$(F_{\mathcal{L}}, \vdash_{\mathcal{L}})$ together with their defining "grammar" is called the *syntactical part of* $\mathcal{L}$, while $(M_{\mathcal{L}}, mng_{\mathcal{L}}, \models_{\mathcal{L}})$ is the semantical part of $\mathcal{L}$.

In Figure 3.2 we illustrate the general pattern of a logic [ANSK94].

There are two directions in the study of logics:

Proof theory, which seeks to axiomatize the entailment relation $\Gamma \vdash \phi$ between a set Γ of sentences and a sentence ϕ.

Model theory, which focuses on the satisfaction relation of the type $M \models \phi$, where M is a model and ϕ is a sentence.

The proof-theoretic approach has a long tradition, dating back to the work of Tarski [Tar56] on "consequence relations" and of Gentzen on the entailment relation. (Of course, semantic considerations are also included, especially in Tarski's work.) The model-theoretic approach is exemplified by Barwise's axioms for abstract model theory [Bar74, BF85, Ebb85]. We would also like to mention the framework of institutions, due to Goguen and Burstall [GB85], which belongs to the model-theoretic approach, but it achieves much greater generality by using category theory and avoiding a commitment to particular notions such as "language" and "structure".

3.2.2 Basic Properties of Propositional Logics

Each formalized system of a propositional calculus is an ordered pair $\mathcal{S} = (\mathcal{L}, C_{\mathcal{L}})$, where $\mathcal{L}$ is a formalized language and $C_{\mathcal{L}}$ is a consequence operation on $\mathcal{L}$. The formalized language $\mathcal{L}$ of $\mathcal{S}$ is – roughly speaking – a set of certain finite sequences of elements formed starting from a "alphabet" of $\mathcal{L}$, termed *formulae*.

We begin with the notion of *language of zero order*.

Definition 3.40 (Language of zero order) *An (alphabet of a) language of zero order consists of an ordered system $\mathcal{L} = (V, L, U)$, where*
(1) V is the set of propositional variables,
(2) L is the set of propositional connectives,
(3) U is a set of auxiliary signs.
We assume that also an arity function $a : L \to \mathbb{N}$. *is given. Intuitively, the arity function specifies the number of arguments of every propositional connective.*

The set $\mathsf{Fma}(\mathcal{L})$ of *formulae over the alphabet* $\mathcal{L}$ is the least set of finite sequences of signs in $\mathcal{L}$ such that

- all propositional variables (considered as one-element sequences) are in $\mathsf{Fma}(\mathcal{L})$,
- all connectives of arity 0 are in $\mathsf{Fma}(\mathcal{L})$,
- if $f_1, \dots, f_n$ are in $\mathsf{Fma}(\mathcal{L})$ and σ is a propositional connective with arity n, then $\sigma(f_1, \dots, f_n)$ is in $\mathsf{Fma}(\mathcal{L})$.

$\mathsf{Fma}(\mathcal{L})$ can thus be regarded as a L-algebra. It is easy to see that the algebra $\mathsf{Fma}(\mathcal{L})$ is a free L-algebra, the set V of all propositional variables in $\mathcal{L}$ being a set of free generators for $\mathsf{Fma}(\mathcal{L})$.

A map $h : V \to A$, where A is a L-algebra is called a *valuation*. From the universality property of $\mathsf{Fma}(\mathcal{L})$ it follows that for every L-algebra A and every valuation $h : V \to A$ there is a unique homomorphism of L-algebras $\overline{h} : \mathsf{Fma}(\mathcal{L}) \to A$ that extends h.

Thus, every formula $\phi \in \mathsf{Fma}(\mathcal{L})$ induces a mapping $\phi_A : A^V \to A$ by

$$\phi_A(h) = \overline{h}(\phi) \text{ for any valuation } h : V \to A.$$

Definition 3.41 (Consequence operation) *Let $\mathcal{L}$ be a language of zero order. A* consequence operation *in $\mathcal{L}$ is a map $C : \mathcal{P}(\mathsf{Fma}(\mathcal{L})) \to \mathcal{P}(\mathsf{Fma}(\mathcal{L}))$ satisfying the following conditions:*

(Extensivity) $\Gamma \subseteq C(\Gamma)$,

(Monotonicity[4]) $\Gamma_1 \subseteq \Gamma_2$ *implies* $C(\Gamma_1) \subseteq C(\Gamma_2)$,

[4] We briefly note that this condition is imposed in many logical systems; however in the so-called *non-monotonic logics* the consequence operator is not required to have the monotonicity property. We include it among the properties of a consequence operation since in the thesis we do not take the non-monotonic approach into consideration.

(Idempotence) $C(C(\Gamma)) = C(\Gamma)$

A consequence operation C has a *finite character* if the following condition is satisfied:

- If $\phi \in C(\Gamma)$ then there exists a finite subset Γ_0 of Γ such that $\phi \in C(\Gamma_0)$.

A consequence operation in a formalized language of zero order can be introduced by the following method. We choose a set A of logical formulae called a set of *logical axioms* and a finite set $\{(r_1), \ldots, (r_n)\}$ of *rules of inference*. Any rule of inference is a mapping $(r) : P \to \mathsf{Fma}(\mathcal{L})$, where $P \subseteq \mathsf{Fma}(\mathcal{L})^n$ for some $n \in \mathbb{N}$ (n is then called the arity of (r)). Instead of $(r)(\alpha_1, \ldots, \alpha_n) = \beta$ we usually write

$$(r)\ \frac{\alpha_1, \ldots, \alpha_n}{\beta}.$$

Definition 3.42 (Formal Proof) *By a* formal proof *of a formula ϕ from a set Γ of formulae with respect to a set A of logical axioms and rules of inference $\{(r_1), \ldots, (r_n)\}$ we mean any finite sequence $\phi_1, \ldots, \phi_k$ of formulae in $\mathcal{L}$ such that*

- $\phi_1 \in \Gamma \cup A$,
- *for every $1 < i \leq k$, either $\phi_i \in \Gamma \cup A$, or $\phi_i = (r_l)(\phi_{i_1}, \ldots, \phi_{i_{n_l}})$, where $i_1, \ldots, i_{n_l} < i$ and (r_l) is a rule of inference in $\{(r_1), \ldots, (r_n)\}$ with arity n_l,*
- $\phi_k = \phi$.

If there exists a formal proof of a formula ϕ from a set Γ with respect to the logical axioms[5] A and inference rules $\{(r_1), \ldots, (r_n)\}$, then we write $\Gamma \vdash \phi$. In particular, if $\Gamma = \emptyset$ we write $\vdash \phi$.

A *deductive system* S over $\mathcal{L}$ is defined by a set of axioms A and a set of inference rules $\{(r_1), \ldots, (r_n)\}$; it consists of a pair $(\mathcal{L}, \vdash_S)$, where $\vdash_S$ is the relation between sets of formulae and individual formulae defined above. The relation $\vdash_S$ is called the consequence relation of S.

Let $S = (\mathcal{L}, \vdash_S)$ be a deductive system, and let $C : \mathcal{P}(\mathsf{Fma}(\mathcal{L})) \to \mathcal{P}(\mathsf{Fma}(\mathcal{L}))$ be defined by $C(\Gamma) = \{\phi \mid \Gamma \vdash_S \phi\}$. The operator C is the *consequence operation in $\mathcal{L}$ determined by the set A of logical axioms and the set $\{(r_1), \ldots, (r_n)\}$ of inference rules*. It is easy to prove that C is a consequence relation and has a finite character.

Let S be a deductive system. If $\phi \in \mathsf{Fma}(\mathcal{L})$ then ϕ is called a *theorem* of S if $\vdash_S \phi$. A set $T \subseteq \mathsf{Fma}(\mathcal{L})$ is called a S*-theory* if $T \vdash_S \phi$ implies $\phi \in T$, for all $\phi \in \mathsf{Fma}(\mathcal{L})$. Observe that the theorems of S belong to every S-theory.

[5] The axioms are usually given by so-called *axiom schemes*, where every variable that occurs can be instantiated with an arbitrary formula.

Example 3.3 We briefly discuss the axiomatizations of an important deductive system, namely **Classical Propositional Calculus** (CPC).

The set of propositional connectives of CPC is $\mathcal{L} = (\Rightarrow, \wedge, \vee, \neg, \perp, \top)$. The axiom schemes are:

(A1) $a \Rightarrow (b \Rightarrow a)$,
(A2) $(a \Rightarrow (b \Rightarrow c)) \Rightarrow ((a \Rightarrow b) \Rightarrow (a \Rightarrow c))$,
(A3) $(\neg b \Rightarrow \neg a) \Rightarrow (a \Rightarrow b)$.

There is one rule of inference:

$(r)\ \dfrac{a, a \Rightarrow b}{b}$ (modus ponens).

$\neg(a)$ stands for $a \Rightarrow \perp$. The operations $\vee$ and $\wedge$ can be defined in terms of the operations $\Rightarrow$ and $\neg$ by:

$$a \vee b = (\neg a) \Rightarrow b$$

$$a \wedge b = \neg((\neg a) \vee (\neg b)).$$

One of the basic results in CPC is the deduction theorem. In what follows Fma denotes the set of formulae.

Deduction Theorem *Let* $\Gamma \cup \{\phi, \psi\} \subseteq \mathsf{Fma}$. *Then*

$$\Gamma \cup \{\phi\} \vdash_{CPC} \psi \textit{ iff } \Gamma \vdash_{CPC} \phi \Rightarrow \psi.$$

The class of Boolean algebras forms an algebraic semantics for CPC. For the sake of simplicity we will regard a Boolean algebra B as an algebra $(B, \vee_B, \wedge_B, \Rightarrow_B, \neg_B, \perp_B, T_B)$, where $(B, \vee_B, \wedge_B, \neg_B, \perp_B, \top_B)$ is a Boolean algebra in the sense of Definition 3.9 and $\Rightarrow_B$ is a *relative complementation*, namely $a \Rightarrow_B b = \neg_B a \vee_B b$.

For every formula ϕ let $\phi_B : B^V \to B$ be the function associated to ϕ. The following completeness theorem holds.

Weak Completeness Theorem *For every formula* ϕ,

$$\vdash_{CPC} \phi \textit{ iff } B \models \phi = 1 \textit{ for every Boolean algebra } B \textit{ iff } 2 \models \phi = 1,$$

where 2 is the 2-element Boolean algebra.

Completeness Theorem *For every set of formulae* Γ *and every formula* ϕ,

$$\Gamma \vdash_{CPC} \phi \textit{ iff } \{\psi = 1 \mid \psi \in \Gamma\} \models_{\mathsf{Bool}} \phi = 1 \textit{ iff } \{\psi = 1 \mid \psi \in \Gamma\} \models_2 \phi = 1.$$

The second equivalence reflects the fact that the variety Bool of Boolean algebras is generated by the 2-element Boolean algebra 2.

3.2.3 Basic Properties of First-Order Logic

The first notion in first-order logic we consider is the notion of *language*.

Definition 3.43 (Language) *A first order (formal) language $\mathcal{L} = (\Sigma, V, L)$ consists of:*

(1) *A* signature Σ, *consisting of a set of function (or operation) symbols O and a set of predicate (or relation) symbols P (with arity functions $a_O : O \to \mathbb{N}, a_P : P \to \mathbb{N}$). For every $n \in \mathbb{N}$ we will denote the set of all operations of arity n by O_n, and the the set of all relation symbols of arity n by P_n,*

(2) *A set of* variables $V = (V_f, V_b)$, *where V_f is a set of* free variables, *and V_b is a set of* bound variables,

(3) *A set of* logical connectors *(L):*
The set L may contain operators such as $\neg$ (not), $\wedge$ (and), $\vee$ (or), $\to$ (implies), $\forall$ (for all), $\exists$ (exists). (In non-classical logics also other operators may occur.)

The algebra $\mathsf{Term}_{\mathcal{L}}$ of *terms* in a given language $\mathcal{L}$ is the free O-algebra freely generated by the set of free variables. The set of *atomic formulae* in the language $\mathcal{L}$ is the set

$$\mathsf{At}_{\mathcal{L}} = \{R(t_1, \ldots, t_n) \mid t_1, \ldots, t_n \in \mathsf{Term}_{\mathcal{L}}, R \in P, \text{ with arity } n\}.$$

Formulae are inductively defined as follows:

(1) Every atomic formula is a formula,

(2) If ϕ and ψ are formulae, then $\neg\phi$, $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \to \psi$ are formulae.

(3) If ϕ is a formula, a a free variable in ϕ, and x a bound variable not occurring in ϕ, then $\forall x\phi'$ and $\exists x\phi'$ are formulae, where ϕ' is the expression obtained from ϕ by replacing every occurrence of a by x.

(4) Formulae are exactly those expressions obtained by the rules (1)–(3).

Definition 3.44 *A variable x is* free *in ϕ if some occurrence of x in ϕ is not in the domain of a quantifier. A variable x is* bound *in ϕ if all occurrences of x are in the domain of some quantifier.*

A sentence *is a formula with no free variables.*

In first-order logic a notion of *provability* or *logical deduction* can be defined. The notion is based on a set of *axioms*, a set of *inference rules*, and a notion of *proof*. For details on the form of the axioms and inference rules, and about the notion of proof we refer to any standard text on logic, e.g. [Tak75] or [Mon76].

Definition 3.45 *A* theory *in the first-order logic, $T = (\mathcal{L}, Th)$ consists of a language $\mathcal{L}$ and a set Th of first-order sentences over $\mathcal{L}$ (the free variables in the sentences are assumed to be universally quantified) which contains all the axioms of classical first-order logic and is closed under first-order logical deduction.*

More generally, let $\mathcal{L}$ be a language and C a consequence relation on the set of formulae in the language $\mathcal{L}$. A theory $(\mathcal{L}, C, Th)$ consists of the language $\mathcal{L}$, the consequence relation C and a set Th of first-order sentences over $\mathcal{L}$ (the free variables in the sentences are assumed to be universally quantified) which is closed under the consequence relation C.

In what follows if not otherwise stated we refer to classical first-order logic. In classical first-order logic the so-called Craig interpolation property holds.

Theorem 3.16 (Craig Interpolation Property) *Let ϕ and ψ be two formulae such that $\vdash \phi \Rightarrow \psi$. If ϕ and ψ have at least one predicate constant in common, then there exists a formula ρ called an* interpolant *of $\phi \Rightarrow \psi$ such that ρ contains only those atomic formulae that occur in both ϕ and ψ, and such that $\vdash \phi \Rightarrow \rho$ and $\vdash \rho \Rightarrow \psi$.*

If ϕ and ψ have no predicate constant in common, then either $\vdash \phi$ or $\vdash \neg\psi$ is provable.

The Craig Interpolation Property is basic for the proof of Theorem 3.17 below cf. [BHK90] p.359.

Definition 3.46 *Let $T = (\mathcal{L}, Th)$ be a theory and $\mathcal{L}'$ a language. The* restriction *of the theory T to the language $\mathcal{L}'$, $\mathcal{L}'\Box T$ is defined as $(\mathcal{L} \cap \mathcal{L}', Th')$ where Th' is the intersection of Th with the set of sentences in the language $\mathcal{L}'$.*

Definition 3.47 *Let $T_1 = (\mathcal{L}_1, Th_1)$ and $T_2 = (\mathcal{L}_2, Th_2)$ be two theories in the first-order logic. The* union *of T_1 with T_2 is the theory $T_1 + T_2 = (\mathcal{L}_1 \cup \mathcal{L}_2, Th)$, where Th is the closure under first-order logical deduction of $Th_1 \cup Th_2$.*

Theorem 3.17 *Let $T_1 = (\mathcal{L}_1, Th_1)$ and $T_2 = (\mathcal{L}_2, Th_2)$ be two theories in the first-order logic, and let $\mathcal{L}$ be a language. If $\mathcal{L}_1 \cap \mathcal{L}_2 \subseteq \mathcal{L}$ then $\mathcal{L}\Box(T_1 + T_2) = \mathcal{L}\Box T_1 + \mathcal{L}\Box T_2$.*

This distributivity property does not hold in general if $\mathcal{L}_1 \cap \mathcal{L}_2$ is not contained in $\mathcal{L}$.

More considerations concerning the equivalence between such distributivity properties and different variants of the Craig Interpolation Property can be found in [DGS91].

The relation $\models$ is based on a satisfiability relation with respect to Σ-structures. We now give the main definitions.

Definition 3.48 *If Σ is a signature with set of function symbols O and set of predicate symbols P, then a* Σ-structure *is an ordered pair (A, L), where A is a nonempty set and L consists of a family $\{R_A\}_{R \in P}$ of fundamental relations (with the arity of R_A equal to the arity of R if $R \in P$), and a family $\{f_A\}_{f \in O}$ of fundamental operations on A (with the arity of f_A equal to the arity of f, for $f \in O$).*

A is called the universe *of A. If $P = \emptyset$ then A is an algebra; if $O = \emptyset$ then A is a relational structure.*

A notion of satisfiability $A \models \phi$ is defined first for sentences ϕ (taking into account the structure of ϕ), then on formulae by $A \models \phi$ if and only if $A \models \forall x_1 \ldots \forall x_n \phi$. For details see for example [BS81], p.195.

This notion of satisfiability can be extended to classes of structures and sets of formulae: If K is a class of Σ-structures and ϕ is a formula we say

$$K \models \phi \quad \text{iff} \quad A \models \phi \text{ for every } A \in K.$$

If E is a set of formulae then

$$A \models E \quad \text{iff} \quad A \models \phi \text{ for every } \phi \in E,$$

$$K \models E \quad \text{iff} \quad K \models \phi \text{ for every } \phi \in E.$$

(If $A \models E$ we also say that A is a *model* for E.) Then we say

$$E \models \phi \quad \text{iff} \quad \text{for every } A, A \models E \text{ implies } A \models \phi,$$

$$E \models E' \quad \text{iff} \quad E \models \phi \text{ for every } \phi \in E'.$$

In classical first-order logic the following holds:

Theorem 3.18 (Soundness and Completeness) *For every formula ϕ in the language $\mathcal{L}$, $\vdash \phi$ if and only if $\models \phi$.*

Therefore in what follows the symbol $\models$ can be used instead of $\vdash$.

Definition 3.49 (cf. [Mon76]) *Let $\mathcal{L}$ be a language.*

(1) A theory *is a pair $(\mathcal{L}, \Gamma)$ such that Γ is a set of sentences in $\mathcal{L}$ and $\phi \in \Gamma$ whenever $\Gamma \models \phi$,*

(2) If A is a $\mathcal{L}$-structure, the $\mathcal{L}$-theory *of A is the pair $(\mathcal{L}, \Gamma)$ where $\Gamma = \{\phi \mid A \models \phi\}$.*

(3) A theory $(\mathcal{L}, \Gamma)$ is an extension *of a theory $(\mathcal{L}', \Gamma')$ provided that $\mathcal{L} \subseteq \mathcal{L}'$ and $\Gamma \subseteq \Gamma'$.*

(4) We say that $(\mathcal{L}, \Gamma)$ is a conservative extension *of $(\mathcal{L}', \Gamma')$ provided that, in addition, $\Gamma = \Gamma' \cap \mathsf{Fma}(\mathcal{L})$.*

(5) If $(\mathcal{L}, \Gamma)$ is a theory, a set $\Delta \subseteq \mathsf{Fma}(\mathcal{L})$ is a set of axioms *for Γ provided that $\Gamma = \{\phi \in \mathsf{Fma}(\mathcal{L}) \mid \Delta \models \phi\}$.*

Definition 3.50 (cf. [Mon76]) *Let $\mathcal{L} \subseteq \mathcal{L}'$ be a language extension and Γ, Γ' be theories over $\mathcal{L}, \mathcal{L}'$ respectively.*

(1) If R is a relation symbol of $\mathcal{L}'$ but not of $\mathcal{L}$, then a possible definition of R over Γ is any formula ϕ in the language $\mathcal{L}$ with free variables $\{v_0, \ldots, v_{m-1}\}$, where m is the arity of R.

(2) *If σ is an operation symbol of $\mathcal{L}'$ but not of $\mathcal{L}$, then a possible definition of σ over Γ is a formula ϕ in the language $\mathcal{L}$ with free variables $\{v_0, \dots, v_m\}$, where m is the arity of σ, such that the following existence and uniqueness conditions are in Γ:*

$$\forall v_0, \dots, v_{m-1} \exists v_m \phi;$$

$$\forall v_0, \dots, v_m, v_{m+1}[\phi(v_1, \dots, v_m) \wedge \phi(v_1, \dots, v_{m+1}) \Rightarrow v_m = v_{m+1}].$$

(3) *We say that $(\mathcal{L}', \Gamma')$ is a* definitional extension *of $(\mathcal{L}, \Gamma)$ provided that for every non-logical constant C of $\mathcal{L}'$ but not in $\mathcal{L}$ there is a possible definition ϕ_C of C over Γ such that*

$$\Gamma' = \{\phi \mid \phi \in \mathsf{Fma}(\mathcal{L}'), \text{ and } \Gamma \cup \{\phi'_C \mid C \text{ a non-logical constant of } \mathcal{L}' \text{ but not in } \mathcal{L}\} \models \phi\},$$

where ϕ'_C is the sentence $\forall v_0, \dots, v_m (C(v_0, \dots, v_m) \Leftrightarrow \phi_C)$ if C is a relation symbol of arity m, while ϕ'_C is $\forall v_0, \dots, v_m (C(v_0, \dots, v_{m-1}) = v_m \Leftrightarrow \phi_C)$ if C is an operation symbol of arity m.

3.2.4 Link Between Logic and Algebra

The idea of solving problems in logic by first translating them to algebra, solving them by using the powerful methodology of algebra, and then translating the solution back to logic is quite old. Papers on the history of logic point out that this method was fruitfully applied in the 19th century not only to propositional logics, but also to quantifier logics (cf. the works of De Morgan, Peirce). The number of applications has grown ever since. The main reason for this is that, when working with a problem, it is often useful to "transform" the problem into a well-understood and streamlined area of mathematics, solve the problem there, and translate the result back. In this case, the advantage of this approach is that universal algebra is not only a unifying framework but also contains powerful theoretical results. Another reason is that, with the rapidly growing variety of applications of logic (in diverse areas like computer science, linguistics, AI, law, etc.) there is a growing number of new logics to be investigated. In this situation translating these problems into algebraic terms proves often useful: it offers a tool for economy and unification in various ways. Several logical properties can be translated to properties of the class of their algebraic models, and vice-versa.

Among the special classes of algebras in which powerful theoretical results have been established are the discriminator varieties and, more generally, the arithmetical varieties[6]. It turns out that in most cases, algebras originating from logic fall into one of these two categories. The varieties of algebras investigated in Chapter 5, for example those corresponding to the SHn-logics or to the $SHKn$ logics, are discriminator varieties.

[6] A variety is called *arithmetical* if it is both congruence-distributive and congruence-permutable (i.e. the lattices of congruences of all the algebras in the variety are distributive, and, moreover, for every algebra in the variety its congruences commute). Alternatively, a variety $\mathcal{V}$ is arithmetical if there is a term $m(x, y, z)$ such that $\mathcal{V} \models m(x, y, x) = m(x, y, y) = m(y, y, x) = x$, cf. [BS81]. It has been shown that every discriminator variety is an arithmetical variety; there exist arithmetical varieties which are not discriminator varieties.

(T1)	$a \Rightarrow a$
(T2)	$(a \Rightarrow b) \Rightarrow ((b \Rightarrow c) \Rightarrow (a \Rightarrow c))$
(T3)	$a \Rightarrow (a \vee b)$
(T4)	$b \Rightarrow (a \vee b)$
(T5)	$(a \Rightarrow c) \Rightarrow ((b \Rightarrow c) \Rightarrow ((a \vee b) \Rightarrow c))$
(T6)	$(a \wedge b) \Rightarrow a$
(T7)	$(a \wedge b) \Rightarrow b$
(T8)	$(a \Rightarrow b) \Rightarrow ((a \Rightarrow c) \Rightarrow (a \Rightarrow (b \wedge c)))$
(T9)	$(a \Rightarrow (b \Rightarrow c)) \Rightarrow ((a \wedge b) \Rightarrow c)$
(T10)	$((a \wedge b) \Rightarrow c) \Rightarrow (a \Rightarrow (b \Rightarrow c))$
(T11)	$(a \wedge \neg a) \Rightarrow b$
(T12)	$(a \Rightarrow (a \wedge \neg a)) \Rightarrow \neg a$
(T13)	$a \vee \neg a$

Figure 3.3: Axiom Schemes

Many properties of the logics can be translated to properties of classes of algebraic models for such logics. To give only a few examples, the Beth definability property reduces (for certain classes of logics) to the property that all the epimorphisms in a corresponding category of algebraic models are surjective; and certain versions of the Craig interpolation property reduce to amalgamation properties on the category of algebraic models. For details in these directions we refer to [Cze82], [Sai88]; the main results are also presented without proofs in [ANSK94], p.64.

We illustrate here – as a very simple example – the link between logic and algebra for the case of propositional logic. We chose this example since it explains the link between the axioms of the SHn-logics and the properties of the algebraic models discussed in Section 5.1.

Let $\mathcal{L}$ be a propositional language, and $\mathcal{T} = (\mathcal{L}, C, \Gamma)$ a 0-order theory based on $\mathcal{L}$.

In what follows we will consider some classes of logics based on the connectors $\wedge, \vee, \Rightarrow, \neg$, that are axiomatized by subsets of set of axioms for classical propositional logic.

Consider the axiom schemes presented in Figure 3.3.

The *Lindenbaum-Tarski*-algebra associated with the theory $\mathcal{T}$ is the quotient of the algebra of formulae $\mathsf{Fma}(\mathcal{L})$ to the equivalence relation $\simeq$ defined by $\phi \simeq \psi$ iff $\phi \Rightarrow \psi \in C(\Gamma)$ and $\psi \Rightarrow \phi \in C(\Gamma)$.

The links between the axiom schemes in Figure 3.3 and the properties of the Lindenbaum-Tarski algebras are given in Figure 3.4

Similar theorems hold for systems of the modal logic, and for wider classes of logics.

The theory contains theorems	The Lindenbaum-Tarski algebra of the theory is
$(T1) - (T2)$	Partially-ordered set
$(T1) - (T8)$	Lattice
$(T2) - (T10)$	Relatively pseudocomplemented lattice
$(T2) - (T12)$	Heyting algebra
$(T1) - (T13)$	Boolean algebra

Figure 3.4: Properties of the Lindenbaum-Tarski Algebras

3.3 Brief Overview on Many-Sorted Structures and Many-Sorted Logic

Since in Chapters 6–8 we consider *many-sorted* structures and logics (the control variables are allowed to have different sorts, the functions and relations can have arguments of different sorts, etc.) in what follows we briefly present the relevant basic notions.

Many-sorted algebras and structures often appear in theoretical computer science, since – for example when modeling programs – "entities" of different types have to be put together. Obviously up to a certain extent many-sorted structures also appear in classical algebra: we would like to mention modules over a ring. These can be seen as many-sorted structures, having two sorts: the "module-element" sort and the "scalar" sort. The main results in universal algebra extend to many-sorted algebras.

In what follows we give some basic notions of (many-sorted) logic that will be used in Section 6. We define many-sorted signatures and structures, morphisms of structures, terms, formulae and interpretations of many-sorted languages in many-sorted structures. Morphisms of signatures are also considered, and the way formulae can be translated along morphisms of signatures is discussed. For more details we refer to [Dia96] and [Gog96].

Definition 3.51 (Signature) *A* signature Σ *consists of a set of sorts* Sort*, a set of function (or operation) symbols* O*, and a set of predicate (or relation) symbols* P*, with arities* $a_O : O \rightarrow (\mathsf{Sort}^* \times \mathsf{Sort})$*,* $a_P : P \rightarrow \mathsf{Sort}^*$*. For every* $s_1 \dots s_n \in \mathsf{Sort}^*$ *and every* $s \in \mathsf{Sort}$*, we will denote the set of all operations of arity* $(s_1 \dots s_n, s)$ *by* $O_{s_1 \dots s_n, s}$*, and the the set of all relation symbols of arity* $s_1 \dots s_n$ *by* $P_{s_1 \dots s_n}$*.*

Definition 3.52 (Σ-Structure) *Let* $\Sigma = (\mathsf{Sort}, O, P)$ *be a signature consisting of a set of sorts* Sort*, a set of operation symbols* O*, and a set of relation symbols* P*, with arities* $a_O : O \rightarrow (\mathsf{Sort}^* \times \mathsf{Sort})$*,* $a_P : P \rightarrow \mathsf{Sort}^*$*. A* Σ-structure *is a structure* $M = ((M_s)_{s \in \mathsf{Sort}}, \{f_M\}_{f \in O}, \{R_M\}_{R \in P})$ *where if* $f \in O$ *and* $a_O(f) = (s_1 \dots s_n, s)$ *then* $f_M : M_{s_1} \times \dots \times M_{s_n} \rightarrow M_s$ *and if* $R \in P$ *and* $a_P(R) = s_1 \dots s_n$ *then* $R_M \subseteq M_{s_1} \times \dots \times M_{s_n}$*.*

If P is empty we obtain the notion of (many-sorted) algebra; if O is empty we obtain the notion of relational structure.

Definition 3.53 (Morphism of Σ-structures) *Let M_1 and M_2 be two Σ-structures. A* morphism *of Σ-structures from M_1 to M_2 is a Sort-indexed family $\{h_s\}_{s \in \mathsf{Sort}}$ of maps $h_s : M_{1_s} \to M_{2_s}$ with the following properties:*

(1) For every $f \in O_{s_1 \dots s_n, s}$, and every $(m_1, \dots, m_n) \in M_{1_{s_1}} \times \dots \times M_{1_{s_n}}$,

$$h_s(f_{M_1}(m_1, \dots, m_n)) = f_{M_2}(h_{s_1}(m_1), \dots, h_{s_n}(m_n)),$$

(2) For every $R \in P_{s_1 \dots s_n}$,

$$(m_1, \dots, m_n) \in R_{M_1} \textit{ implies } (h_{s_1}(m_1), \dots, h_{s_n}(m_n)) \in R_{M_2}$$

for all $(m_1, \dots, m_n) \in M_{1_{s_1}} \times \dots \times M_{1_{s_n}}$.

The composition of two Σ-morphisms is their composition as functions. The identity Σ-morphism on M, denoted 1_M, is the identity of M. A Σ-morphism $h : M \to M'$ is a Σ-isomorphism if there is a Σ-morphism $h' : M' \to M$ such that $h' \circ h = 1_M$ and $h \circ h' = 1_{M'}$. Such a morphism h' is called an *inverse* of h.

For a given signature $\Sigma = (\mathsf{Sort}, O, P)$ we will denote by Str_Σ the category of Σ-structures, with Σ-morphisms as arrows.

Definition 3.54 (Terms, Formulae) *Let $\Sigma = (\mathsf{Sort}, O, P)$ be a signature and $X = (X_s)_{s \in \mathsf{Sort}}$ be a many-sorted set.*

- *The algebra of* terms *in the signature Σ and variables X is the many-sorted algebra $\mathsf{T_O(X)}$ of terms over the signature (Sort, O) (the free (Sort, O)-algebra freely generated by X).*
- *The set $\mathsf{At_\Sigma(X)}$ of* atomic formulae *over the signature Σ is the set of all expressions of the form $R(t_1, \dots, t_n)$ where $t_1, \dots, t_n$ are terms of sort $s_1, \dots, s_n$ and R is a predicate symbol of arity $s_1 \dots s_n$.*
- *The set of* formulae *over the signature Σ freely generated by X is the free $\{\vee, \wedge, \neg, 0, 1, \{\forall x\}_{x \in X}, \{\exists x\}_{x \in X}\}$-algebra generated by $\mathsf{At_\Sigma(X)}$, where for every $x \in X$, $\forall x$ and $\exists x$ are regarded as unary operators.*

Definition 3.55 (Interpretation) *An* interpretation *of a language $\mathcal{L}$ (consisting of a signature Σ and a set of variables X) is a Σ-structure M together with a (sort-preserving) mapping $v : X \to M$. The mapping v is called an assignment of values from M to the variables in X.*

Definition 3.56 (Satisfiability in a given interpretation) *Let $v : X \to M$ be an interpretation and let $\overline{v}$ be the unique extension of v to a morphism from the set $T_O(X)$ of terms over Σ in variables X to M. The interpretation v* satisfies *a formula ϕ (denoted $v \models \phi$) if this follows from the following inductive definition:*

(0) *The unique extension* $\overline{v}$ *of* v *to a morphism from* $T_O(X)$ *to* M *can be inductively defined for every term* t *in the language* $\mathcal{L}$*. We define* $\overline{v}(x) = v(x)$ *for every variable* $x \in X$*; if* f *is an operation symbol of arity* $(s_1 \dots s_n, s)$ *and* $t_1, \dots, t_n$ *terms of sorts respectively* $s_1, \dots, s_n$*, then* $\overline{v}(f(t_1, \dots, t_n)) = f_M(\overline{v}(t_1), \dots, \overline{v}(t_n))$*.*

(1) *If* R *is a predicate symbol of arity* $s_1 \dots s_n$ *and* $t_1, \dots, t_n$ *are terms of sorts respectively* $s_1, \dots, s_n$*, then* v *satisfies* $R(t_1, \dots, t_n)$ *if and only if* $(\overline{v}(t_1), \dots, \overline{v}(t_n)) \in R_M$*.*

(2) *The assignment* v *satisfies* $\neg\phi$ *if and only if it does not satisfy* ϕ*;* v *satisfies* $\phi \wedge \psi$ *if and only if it satisfies both* ϕ *and* ψ*;* v *satisfies* $\phi \vee \psi$ *if and only if it satisfies* ϕ *or* ψ*;*

(3) *The assignment* v *satisfies* $\forall x\phi$ *if and only if for every* $v' : X \to M$ *such that* v *and* v' *agree, except possibly on* x*,* v' *satisfies* ϕ*;* v *satisfies* $\exists x\phi$ *if and only if for some* $v' : X \to M$ *such that* v *and* v' *agree, except possibly on* x*,* v' *satisfies* ϕ*.*

Let $v : X \to M$ be an interpretation and let $\overline{v}$ be the unique extension of v to a morphism from the set $T_O(X)$ of terms over Σ in variables X to M. We can define a function $v_{\mathsf{Fma}} : \mathsf{Fma}_\Sigma(X) \to \{0, 1\}$ as follows:

1. $v_{\mathsf{Fma}}(R(t_1, \dots, t_n)) = \begin{cases} 1 & \text{if } (\overline{v}(t_1), \dots, \overline{v}(t_n)) \in R_M \\ 0 & \text{otherwise} \end{cases}$,
2. $v_{\mathsf{Fma}}(\phi_1 \wedge \phi_2) = v_{\mathsf{Fma}}(\phi_1) \wedge v_{\mathsf{Fma}}(\phi_2)$, $v_{\mathsf{Fma}}(\phi_1 \vee \phi_2) = v_{\mathsf{Fma}}(\phi_1) \vee v_{\mathsf{Fma}}(\phi_2)$, $v_{\mathsf{Fma}}(\neg\phi) = 1$ iff $v_{\mathsf{Fma}}(\phi) = 0$,
3. $v_{\mathsf{Fma}}(\forall x\phi(x)) = min\{v^{x/m}_{\mathsf{Fma}}(\phi(x)) \mid m \in M\}$, where $v^{x/m}(y) = v(y)$ for every $y \neq x$ and $v^{x/m}(x) = m$,
4. $v_{\mathsf{Fma}}(\exists x\phi(x)) = max\{v^{x/m}_{\mathsf{Fma}}(\phi(x)) \mid m \in M\}$, where $v^{x/m}(y) = v(y)$ for every $y \neq x$ and $v^{x/m}(x) = m$.

Thus, v satisfies ϕ if and only if $v_{\mathsf{Fma}}(\phi) = 1$.

Definition 3.57 (Subsignature) *Let* Σ_1 *and* Σ_2 *be two signatures,* $\Sigma_1 = (\mathsf{Sort}_1, O_1, P_1)$ *and* $\Sigma_2 = (\mathsf{Sort}_2, O_2, P_2)$*.* Σ_1 *is a* subsignature *of* Σ_2 *(denoted by* $\Sigma_1 \subseteq \Sigma_2$*) if* $\mathsf{Sort}_1 \subseteq \mathsf{Sort}_2, O_1 \subseteq O_2, P_1 \subseteq P_2$*.*

Definition 3.58 (Restriction) *Let* $\Sigma = (\mathsf{Sort}, O, P)$ *and* $\Sigma' = (\mathsf{Sort}', O', P')$ *be two signatures such that* $\Sigma' \subseteq \Sigma$*. Let* $M = (\{M_s\}_{s \in \mathsf{Sort}}, \{f_M\}_{f \in O}, \{R_M\}_{R \in P})$ *be a* Σ*-structure. The* restriction *of* M *to the signature* Σ' *is the structure* $M' = (\{M'_s\}_{s \in \mathsf{Sort}'}, \{f_{M'}\}_{f \in O'}, \{R_{M'}\}_{R \in P'})$*, where*

(1) *For every* $s \in \mathsf{Sort}'$*,* $M'_s = M_s$*,*

(2) For every $f \in O'$ with arity $a_O(f) = (s_1 \dots s_n, s)$ (with $s_1, \dots, s_n, s \in$ Sort'), $f_{M'} : {M'}_{s_1} \times \dots \times {M'}_{s_n} \to {M'}_s$ coincides with $f_M : M_{s_1} \times \dots \times M_{s_n} \to M_s$,

(3) For every $R \in P'$ with arity $a_P(R) = s_1 \dots s_n$ (with $s_1, \dots, s_n \in \mathsf{Sort}'$), $R_{M'} \subseteq {M'}_{s_1} \times \dots \times {M'}_{s_n}$ coincides with $R_M \subseteq M_{s_1} \times \dots \times M_{s_n}$.

The restriction of M to Σ' will be denoted $U^{\Sigma}_{\Sigma'} M$. We also say that M' is the Σ'-reduct *of M, or that M is the* Σ-expansion *of M'. The application $U^{\Sigma}_{\Sigma'}$ that associates with every Σ-structure its restriction to Σ' is a functor from the category Str_Σ of Σ-structures to the category $\mathsf{Str}_{\Sigma'}$ of Σ'-structures called* the forgetful functor*.*

The following lemma will be useful in Section 7.1.2.

Lemma 3.19 *Let $\mathcal{L}_1 = (\Sigma_1, X_1)$ and $\mathcal{L}_2 = (\Sigma_2, X_2)$ be two languages such that $\mathcal{L}_1$ is a sublanguage of $\mathcal{L}_2$ (i.e. $\Sigma_1 \subseteq \Sigma_2$ and $X_1 \subseteq X_2$). Let M_1 be a Σ_1-structure and M_2 a Σ_2-structure such that M_1 is the restriction of M_2 to the signature Σ_1.*

Let ϕ be a formula in the language $\mathcal{L}_1$ (i.e. containing only sorts, operation symbols, predicate symbols and variables from $\mathcal{L}_1$). Then for every (sort-preserving) assignment $s : X_2 \to M_2$, s satisfies ϕ if and only if its restriction to X_1, $s_{|X_1} : X_1 \to M_1$, satisfies ϕ.

Proof: Note first that it is easy to see that if $s : X_2 \to M_2$ then its restriction to X_1 has indeed values in M_1, i.e. $s_{|X_1} : X_1 \to M_1$. Therefore it follows immediately by structural induction that for every term t in the language $\mathcal{L}_1$, $\overline{s}(t) = \overline{s}_{|X_1}(t) \in M_1$.

Moreover, if R is a predicate of arity $s_1 \dots s_n$ in P_1 and $t_1, \dots, t_n$ terms in the language $\mathcal{L}_1$, then $(\overline{s}(t_1), \dots, \overline{s}(t_n)) \in R_{M_2}$ if and only if $(\overline{s}(t_1), \dots, \overline{s}(t_n)) \in R_{M_1}$.

We prove by structural induction that for every assignment $s : X_2 \to M_2$ and every formula ϕ in the language $\mathcal{L}_1$, s satisfies ϕ if and only if its restriction to X_1 satisfies ϕ.

Induction basis: Assume that ϕ is an atomic formula, i.e. $\phi = R(t_1, \dots, t_n)$ for some terms $t_1, \dots, t_n$, and let $s : X_2 \to M_2$ be an arbitrary assignment. Then s satisfies ϕ if and only if $(\overline{s}(t_1), \dots, \overline{s}(t_n)) \in R_{M_2}$. But since M_1 is the restriction of M_2 to Σ_1, $(\overline{s}(t_1), \dots, \overline{s}(t_n)) \in R_{M_2}$ if and only if $(\overline{s}(t_1), \dots, \overline{s}(t_n)) \in R_{M_1}$. Since for every term t in the language $\mathcal{L}_1$ $\overline{s}(t) = \overline{s}_{|X_1}(t)$ it follows that s satisfies ϕ if and only if $(\overline{s}_{|X_1}(t_1), \dots, \overline{s}_{|X_1}(t_n)) \in R_{M_1}$, i.e. if and only if $s_{|X_1}$ satisfies ϕ.

Induction step: We distinguish several cases:

Case 1: Let $\phi = \neg\psi$ and $s : X_2 \to M_2$ be an arbitrary assignment. Assume known that for every assignment $s' : X_2 \to M_2$, s' satisfies ψ if and only if its restriction to X_1 satisfies ψ. In particular it follows that s does not satisfy ψ if and only if its restriction to X_1 does not satisfy ψ, i.e. that s satisfies ϕ if and only if its restriction to X_1 satisfies ϕ.

Case 2: Let $\phi = \phi_1 \wedge \phi_2$ and $s : X_2 \to M_2$ be an arbitrary assignment. Assume that for $i = 1, 2$ and every assignment $s' : X_2 \to M_2$, s' satisfies ϕ_i if and only if $s'_{|X_1}$ satisfies ϕ_i. In particular s satisfies ϕ_i if and only if $s_{|X_1}$ satisfies ϕ_i.

We know that s (resp. $s_{|X_1}$) satisfies ϕ if and only if it satisfies both ϕ_1 and ϕ_2. Therefore it follows immediately that s satisfies ϕ if and only if $s_{|X_1}$ satisfies ϕ.

The case $\phi = \phi_1 \vee \phi_2$ can be proved similarly.

Case 3: Let $\phi = \forall x \psi$ and $s : X_2 \to M_2$ be an arbitrary assignment. Assume that for every assignment $s' : X_2 \to M_2$, s' satisfies ψ if and only if $s'_{|X_1}$ satisfies ψ.

We know that s satisfies $\forall x \psi$ if and only if for every $s' : X_2 \to M_2$ such that s and s' agree, except possibly on x, s' satisfies ϕ, and that $s_{|X_1}$ satisfies $\forall x \psi$ if and only if for every $s'' : X_1 \to M_1$ such that $s_{|X_1}$ and s'' agree, except possibly on x, s'' satisfies ϕ.

We have to prove that the following statements are equivalent:

(1) For every $s' : X_2 \to M_2$ such that s and s' agree, except possibly on x, s' satisfies ψ.

(2) For every $s'' : X_1 \to M_1$ such that $s_{|X_1}$ and s'' agree, except possibly on x, s'' satisfies ψ.

(1) $\Rightarrow$ (2): Assume that (1) holds. Let $s'' : X_1 \to M_1$ be such that s'' agrees with $s_{|X_1}$ except possibly on x. We can define an assignment $s' : X_2 \to M_2$ by
$$s'(x) = \begin{cases} s''(x) & \text{if } x \in X_1 \\ s(x) & \text{if } x \in X_2 \backslash X_1 \end{cases}$$

It is easy to see that s' agrees with s except possibly on x, and $s'_{|X_1} = s''$. By (1) it follows that s' satisfies ψ, and by the induction hypothesis we conclude that $s'_{|X_1} = s''$ satisfies ψ.

(2) $\Rightarrow$ (1): Assume that (2) holds. Let $s' : X_2 \to M_2$ be such that s and s' agree, except possibly on x. Then $s'_{|X_1}$ and $s_{|X_1}$ agree except possibly on x, therefore $s'_{|X_1}$ satisfies ψ and by the induction hypothesis s' satisfies ψ.

It follows therefore that s satisfies $\forall x \psi$ if and only if $s_{|X_1}$ satisfies $\forall x \psi$.

The case $\phi = \exists x \psi$ can be proved similarly. □

More general relationships between different signatures can be expressed by morphisms.

Definition 3.59 (Signature Morphism) *Let Σ, Σ' be two signatures, $\Sigma = (\mathsf{Sort}, O, P)$ and $\Sigma' = (\mathsf{Sort}', O', P')$. A* signature morphism *$\phi : \Sigma \to \Sigma'$ consists of a triple (ϕ_S, ϕ_O, ϕ_P) where*

(1) $\phi_S : \mathsf{Sort} \to \mathsf{Sort}'$ is a map on sorts,

(2) ϕ_O *is a* $\mathsf{Sort}^* \times \mathsf{Sort}$*-indexed family of maps on operation symbols,* $\phi_O^{s_1\dots s_n,s} : O_{s_1\dots s_n,s} \to O'_{\phi_S(s_1)\dots\phi_S(s_n),\phi_S(s)}$,

(3) ϕ_P *is a* Sort^**-indexed family of maps on predicate symbols,* $\phi_P^{s_1\dots s_n} : P_{s_1\dots s_n} \to P'_{\phi_S(s_1)\dots\phi_S(s_n)}$.

Example 3.4 *An example of a morphism of signatures is the inclusion. Let* $\Sigma_1 = (\mathsf{Sort}_1, O_1, P_1)$ *and* $\Sigma_2 = (\mathsf{Sort}_2, O_2, P_2)$ *be such that* $\mathsf{Sort}_1 \subseteq \mathsf{Sort}_2$, $O_1 \subseteq O_2$ *and* $P_1 \subseteq P_2$. *The morphism* $i = (i_S, i_O, i_P) : \Sigma_1 \hookrightarrow \Sigma_2$, *where* i_S, i_O *and* i_P *are the corresponding inclusions is a morphism of signatures.*

Proposition 3.20 *Any morphism of signatures* $\phi : \Sigma_1 \to \Sigma_2$ *induces a functor* $\mathsf{Str}(\phi) : \mathsf{Str}_{\Sigma_2} \to \mathsf{Str}_{\Sigma_1}$.

Proof: Let $\phi : \Sigma_1 \to \Sigma_2$ be a signature morphism, and let M be a Σ_2-structure. Assume $M = ((M_s)_{s \in \mathsf{Sort}_2}, \{f_M\}_{f \in O_2}, \{R_M\}_{R \in P_2})$. Then $Str(\phi)(M) = \overline{M} = ((\overline{M}_s)_{s \in \mathsf{Sort}_1}, \{f_{\overline{M}}\}_{f \in O_1}, \{R_{\overline{M}}\}_{R \in P_1})$, where for every $s \in \mathsf{Sort}$, $\overline{M}_s = M_{\phi(s)}$; for every $f \in O$ with arity $(s_1 \dots s_n, s)$, $f_{\overline{M}} : \overline{M}_{s_1} \times \dots \times \overline{M}_{s_n} \to \overline{M}_s$ is the operation $f'_M : M_{\phi(s_1)} \times \dots \times M_{\phi(s_n)} \to M_{\phi(s)}$ of M, where $f' = \phi_O^{s_1\dots s_n,s}(f)$; and for every $R \in P$ with arity $s_1 \dots s_n$, $R_{\overline{M}} \subseteq \overline{M}_{s_1} \times \dots \times \overline{M}_{s_n}$ is the predicate $R'_M \subseteq M_{\phi(s_1)} \times \dots \times M_{\phi(s_n)}$ of M, where $R' = \phi_R^{s_1\dots s_n}(R)$. □

For every morphism of signatures $\phi : \Sigma_1 \to \Sigma_2$ and every Σ'-structure M we will also denote $\mathsf{Str}(\phi)(M)$ by $M_{|\phi}$.

Let M, M' be two Σ_2-structures and let $h : M \to M'$ be a morphism of Σ_2-structures. Then $\mathsf{Str}(h) : \mathsf{Str}(M) \to \mathsf{Str}(M')$ is defined as follows: For every $s \in \mathsf{Sort}_1$, let $\mathsf{Str}(h)_s = h_{\phi_s} : \mathsf{Str}(M)_s = M_{\phi(s)} \to \mathsf{Str}(M')_s = M'_{\phi(s)}$.

It is easy to see that $\mathsf{Str}(h)$ is a morphism of Σ_1-structures: Let $\sigma \in O_1$ be an operation of arity $(s_1 \dots s_n, s)$. Let $(m_1, \dots, m_n) \in \mathsf{Str}(M)_{s_1} \times \dots \times \mathsf{Str}(M)_{s_n}$. Then
$$\begin{aligned}\mathsf{Str}(h)_s(\sigma_{\mathsf{Str}(M)}(m_1, \dots, m_n)) &= h_{\phi(s)}(\phi_O(\sigma)_M)(m_1, \dots, m_n)) = \\ &= \phi_O(\sigma)_{M'})(h_{\phi(s)}(m_1), \dots h_{\phi(s)}(m_n)) = \\ &= \sigma_{\mathsf{Str}(M')}(\mathsf{Str}(h)_{s_1}(m_1), \dots, \mathsf{Str}(h)_{s_n}(m_n)).\end{aligned}$$

Let $R \in P_1$ be a predicate symbol of arity $s_1 \dots s_n$. Let $(m_1, \dots, m_n) \in \mathsf{Str}(M)_{s_1} \times \dots \times \mathsf{Str}(M)_{s_n}$ be such that $R_{\mathsf{Str}(M)}(m_1, \dots, m_n)$.

Then $\phi(R)_M(m_1, \dots, m_n)$ and hence, $\phi(R)_{M'}(h_{\phi(s_1)}(m_1), \dots, h_{\phi(s_n)}(m_n))$.

Therefore, it follows that $R_{\mathsf{Str}(M')}(\mathsf{Str}(h)_{s_1}(m_1), \dots, \mathsf{Str}(h)_{s_n}(m_n))$. □

Example 3.5 *Let* $\Sigma_1 = (\mathsf{Sort}_1, O_1, P_1)$ *and* $\Sigma_2 = (\mathsf{Sort}_2, O_2, P_2)$ *be such that* $\mathsf{Sort}_1 \subseteq \mathsf{Sort}_2$, $O_1 \subseteq O_2$ *and* $P_1 \subseteq P_2$. *Let* $i = (i_S, i_O, i_P) : \Sigma_1 \hookrightarrow \Sigma_2$ *be the inclusion morphism between the signatures* Σ_1 *and* Σ_2. *The functor* $\mathsf{Str}(i) : \mathsf{Str}_{\Sigma_2} \to \mathsf{Str}_{\Sigma_1}$ *induced by the inclusion* i *is the forgetful functor.*

We now analyze the way terms and formulae can be translated along morphisms between signatures.

Let $\phi = (\phi_S, \phi_O, \phi_P)$ be a morphism of signatures, where $\phi_S : \mathsf{Sort} \to \mathsf{Sort}'$ be a map between sorts, and $\phi_O : O \to O'$ a map between operation symbols, and $\phi_P : P \to P'$ a map between predicate symbols.

(1) The map between sorts, ϕ_S : $\mathsf{Sort} \to \mathsf{Sort}'$ translates a Sort-sorted set $X = (X_s)_{s \in \mathsf{Sort}}$ to a Sort'-sorted set $\tilde{X}$ with $\tilde{X}_{s'} = \coprod_{\phi_S(s)=s'} X_s$ for every $s \in \mathsf{Sort}$. (In [Dia96] it is noted that $\tilde{X}$ is the pointwise left Kan extension of ϕ^S along X.) Let h be the identity map, $h : X \to \tilde{X}$ defined for every $s \in \mathsf{Sort}$ and every $x \in X_s$ by $h_s(x) = x \in \tilde{X}_{\phi_S(s)}$.

(2) (ϕ_S, ϕ_O) define a Sort-sorted map $\phi^\natural_{T(X)} : T_O(X) \to T_{O'}(\tilde{X})_\phi$ as follows:

For every $x \in X_s$, $x \in \tilde{X}_{\phi_S(s)} \subseteq T_{O'}(\tilde{X})_{\phi_S(s)}$. We know that $T_{O'}(\tilde{X})_{\phi_S(s)} = (T_{O'}(\tilde{X})_{|\phi})_s$. Hence, $x \in (T_{O'}(\tilde{X})_{|\phi})_s$. Therefore, for every $s \in \mathsf{Sort}$, $X_s \subseteq (T_{O'}(\tilde{X})_{|\phi})_s$. Let $j : X \to T_{O'}(\tilde{X})_{|\phi}$ be the inclusion. Then $\phi^\natural_{T(X)} = j^\natural : T_O(X) \to T_{O'}(\tilde{X})_{|\phi}$ is the unique extension of j to a homomorphism of O-algebras.

(3) The translation from $\mathsf{Fma}_\Sigma(X)$ to $\mathsf{Fma}_{\Sigma'}(\tilde{X})$ can be defined in a similar way. Let $\phi^\natural_{\mathsf{At(X)}} : \mathsf{At}_\Sigma(\mathsf{X}) \to \mathsf{At}_\Sigma(\tilde{\mathsf{X}})$, be defined by $\phi^\natural_{\mathsf{At(X)}}(R(t_1, \dots, t_n)) = \phi_P(R)(\phi^\natural_{T(X)}(t_1), \dots, \phi^\natural_{T(X)}(t_n))$.

Then $\phi^\natural_{\mathsf{Fma}} : \mathsf{Fma}_\Sigma(X) \to \mathsf{Fma}_{\Sigma'}(\tilde{X})$ is the unique morphism w.r.t. the operations $\{\vee, \wedge, \neg, 0, 1, \{\forall x\}_{x \in X}, \{\exists_x\}_{x \in X}\}$ that extends $\phi^\natural_{\mathsf{At(X)}}$.

Remark 3.21 *Assume now that the variables have a rôle of* generators *rather than of* variables*. We allow quantified formulae, with the mention that $(\forall x)\phi(x)$ holds in a given model iff $\phi(d)$ holds for any* value *d attributed to x.*

Similarly, any signature morphism $\phi = (\phi_S, \phi_O, \phi_P) : \Sigma \to \Sigma'$ together with an arbitrary map $\phi_X : X \to X'$ such that if $x \in X_s$ then $\phi_X(x) \in X'_{\phi_S(s)}$ (renaming of the generators), uniquely extends to morphisms:

$$\phi^\natural_{\mathsf{Term}} : \mathsf{T_O}(X) \to \mathsf{T_{O'}}(X')_{|\phi},$$

$$\phi^\natural_{\mathsf{At}} : \mathsf{At}_\Sigma(\mathsf{X}) \to \mathsf{At}_{\Sigma'}(\mathsf{X}'),$$

$$\phi^\natural_{\mathsf{Fma}} : \mathsf{Fma}_\Sigma(\mathsf{X}) \to \mathsf{Fma}_{\Sigma'}(\mathsf{X}').$$

Proof: The existence of $\phi^\natural_{\mathsf{Term}}$ follows from the fact that if $x \in X_s$ then $\phi_X(x) \in X'_{\phi_S(s)}$, from the definition of $\mathsf{T_{O'}}(X')_{|\phi}$, and from the universality property of $\mathsf{T_O}(X)$.

$\phi^\natural_{\mathsf{At}} : \mathsf{At}_\Sigma(\mathsf{X}) \to \mathsf{At}_{\Sigma'}(\mathsf{X}')$, is defined by

$$\phi^\natural_{\mathsf{At}}(R(t_1, \dots, t_n)) = \phi_P(R)(\phi^\natural_{\mathsf{Term}}(t_1), \dots, \phi^\natural_{\mathsf{Term}}(t_n)).$$

The map $\phi_X : X \to X'$ induces a morphism of signatures

$$\phi_L : \{\vee, \wedge, \neg, 0, 1, \{\forall x\}_{x \in X}, \{\exists x\}_{x \in X}\} \to \{\vee, \wedge, \neg, 0, 1, \{\forall x\}_{x \in X'}, \{\exists x\}_{x \in X'}\}$$

as follows:

- ϕ_L preserves the sorts,

- $\phi_L(\vee) = \vee, \phi_L(\wedge) = \wedge, \phi(\neg) = \neg, \phi(0) = 0, \phi(1) = 1,$
- $\phi_L(\forall x) = \forall\phi(x), \phi_L(\exists x) = \exists\phi(x).$

The existence of $\phi^{\natural}_{\mathsf{Fma}} : \mathsf{Fma}_{\Sigma}(\mathsf{X}) \to \mathsf{Fma}_{\Sigma'}(\mathsf{X}')_{|\phi_L}$ follows from the universality property of $\mathsf{Fma}_{\Sigma}(\mathsf{X})$ and from the definition of $\mathsf{Fma}_{\Sigma'}(\mathsf{X}')_{|\phi_L}$ □

3.4 Automated Theorem Proving: The Resolution Principle

We give a short introduction to resolution in classical first order logic, based mainly on [CL73] and Chapter 12 of [Rob79]. A short introduction to resolution in many-valued logic based on [BF92] will be given in Section 4.2. An extensive overview of resolution-based theorem proving in many-valued logics can be found in [BF95], [Häh94], [Häh96b].

3.4.1 The Resolution Principle

The basic idea behind the resolution procedure is to prove statements by *refutation*, i.e. the strategy of resolution is to take the negation of the statement that one wants to prove, and then to show that this negation produces a contradiction with the known statements. One also says that the negation of the given statement is *unsatisfiable*.

We begin by giving the main definitions.

Definition 3.60 (Literal, Clause, Clausal Form) *A* literal *is an atomic formula or a negation thereof. A* clause *is a set of literals. A clause $C = \{L_1, \ldots, L_n\}$ is understood to represent the disjunction of its members, $L_1 \vee \ldots \vee L_n$. The symbol □ will be used to denote the* empty clause*. A* ground *clause (term, literal) is a clause (term, literal) without variables. For any clause C, the set of variables occurring in C is denoted by $V(C)$. The* clausal form *$F(C)$ of a clause C is the universally quantified formula $\forall x_1 \ldots \forall x_m C$, where $V(C) = \{x_1, \ldots, x_m\}$.*

The standard notions of substitution and unifier play an important role in the resolution method.

Definition 3.61 (Substitution, Variant, Instance, Unifier, M.g.u.) *Let V be the set of all variables, T the set of terms. A mapping $\sigma : V \to T$ is called a* substitution *if $\sigma(v) \neq v$ for only finitely many $v \in V$. The* domain *of a substitution σ is defined as $dom(\sigma) = \{v \mid v \in V, \sigma(v) \neq v\}$.*

Substitutions can operate on terms, literals, and clauses by the usual extensions. A substitution σ such that $\sigma(V) \subseteq V$ and injective on its domain is called a renaming.

If σ is a renaming with $dom(\sigma) = V(C)$ then $\sigma(C)$ is called a variant *of C.*

For any substitution σ, we call $\sigma(C)$ an instance *of C.*

Let M be a set of literals. A substitution σ is called a unifier of M if $\sigma(M)$ contains only one element. A substitution σ is called a most general unifier *(short* m.g.u.*) of M if σ is a unifier such that for every other unifier τ of M there is a ρ such that $\rho\sigma = \tau$.*

Definition 3.62 (Factor) *If two or more positive literals of a clause C have a m.g.u. σ, then $\sigma(C)$ is called a* factor *of C. Likewise, if two or more negative literals of a clause C have a m.g.u. σ, then $\sigma(C)$ is called a* factor *of C.*

Definition 3.63 (Binary Resolvent) *Let C_1 and C_2 be clauses with no variable in common. Let L_1 and L_2 be literals occurring in C_1 and C_2, respectively. If L_1 and $\neg L_2$ have a m.g.u. σ, then the clause $C_3 := (\sigma(C_1) - \sigma(L_1)) \cup (\sigma(C_2) - \sigma(L_2))$ is called a* binary resolvent *of C_1 and C_2.*

Definition 3.64 (Resolvent) *A* resolvent *of two clauses C_1 and C_2 is one of the following binary resolvents:*

1. *a binary resolvent of C_1 and C_2,*
2. *a binary resolvent of C_1 and a factor of C_2,*
3. *a binary resolvent of a factor of C_1 and C_2,*
4. *a binary resolvent of a factor of C_1 and a factor of C_2.*

Robinson's Resolution Algorithm

Input: *a set of clauses F in first-order predicate logic.*

Output: $\Box$ *if F is unsatisfiable.*

Algorithm *:*

> **while** *F has not been proven unsatisfiable and new clauses can be added*
> **do**
> $(c_1, c_2) :=$ *a pair in F*;
> $R :=$ *Resolvents*$((c_1, c_2))$;
> **if** $\Box \in R$
> **then** F *is* unsatisfiable
> **else** $F := F \cup R$
> **od**

Computing the resolvents of two clauses

Input: *Two clauses, C_1, C_2,*

Output: *R the set of all resolvents of clauses C_1, C_2.*

Algorithm *:*

$$\begin{aligned} R := {} & \textit{Binary-resolvents}(C_1, C_2) \cup \\ & \cup\ \textit{Binary-resolvents}(C_1,\ \textit{Factor}(C_2)) \cup \\ & \cup\ \textit{Binary-resolvents}(\ \textit{Factor}(C_1), C_2) \cup \\ & \cup\ \textit{Binary-resolvents}(\ \textit{Factor}(C_1),\ \textit{Factor}(C_2)). \end{aligned}$$

The proof of the completeness of the resolution procedure, as well as those of its refinements (see e.g. section 3.4.2), requires the proof of a lifting lemma stating that a resolvent for two instances of a given clause can be "lifted" to a resolvent for those clauses themselves, and such that the original resolvent (for the two instances) is an instance of the resolvent for the clauses. For this, we refer to the literature, e.g. Chapter 5 of [CL73].

3.4.2 Semantic Resolution

The resolution principle can be seen as a inference rule that can be used to generate new clauses from old ones. However, unlimited application of resolution may generate many irrelevant and redundant clauses besides useful ones. Although a deletion strategy [CL73] can be used in order to delete some of these irrelevant and redundant clauses after they are generated, time has already been wasted by generating them.

Therefore, in order to have efficient theorem proving procedures, we must prevent large numbers of useless clauses from being generated. This leads to refinements of resolution. Below we will discuss semantic resolution.

The main idea of *semantic resolution* is to use an interpretation to divide clauses into two groups, and an ordering to reduce the number of possible resolutions.

Definition 3.65 *Let I be an interpretation and P an ordering of predicate symbols. A finite set of clauses $\{E_1, \ldots, E_q, N\}, q \geq 1$ is called a* semantic clash *with respect to P and I (PI-*clash*, for short) if and only if $E_1, \ldots, E_q$ (called* electrons*) and N (called* nucleus*) satisfy the following conditions:*

1. *$E_1, \ldots, E_q$ are false in I,*
2. *Let $R_1 = N$. For each $i = 1, \ldots, q$, there is a resolvent R_{i+1} of R_i and E_i,*
3. *The literal in E_i, which is resolved upon, contains the largest predicate symbol in $E_i, i = 1, \ldots, q$,*
4. *R_{q+1} is false in I.*

*R_{q+1} is called a PI-*resolvent *of the PI-clash $\{E_1, \ldots, E_q, N\}$*

Definition 3.66 *Let I be an interpretation for a set of clauses S, and P be an ordering of predicate symbols appearing in S. A deduction from S is called a PI-*deduction *if and only if each clause in the deduction is either a clause in S, or a PI-resolvent.*

Theorem 3.22 *If P is an ordering of predicate symbols in a finite and unsatisfiable set S of clauses, and if I is an interpretation of S, then there is a PI-deduction of $\square$ from S.*

The proof of the completeness theorem is based, as in the case of the resolution principle, on a lifting lemma.

In what follows we will present a special kind of interpretation to be used in semantical resolution, that leads to *hyperresolution*.

3.4.3 Hyperresolution

Let us consider an interpretation I in which every literal is the negation of an atom. If this interpretation is used, every electron and every PI-resolvent must contain only atoms. Similarly, if every literal in I is an atom, then every electron and every PI-resolvent must contain only negations of atoms. Hyperresolution is based on these considerations.

Definition 3.67 *A clause is called* positive *if it does not contain any negation sign. A clause is called* negative *if every literal of it contains the negation sign. A clause is called* mixed *if it is neither positive nor negative.*

Definition 3.68 *A* positive hyperresolution *is a special case of PI-resolution in which every literal in the interpretation I contains the negation sign. It is called* positive hyperresolution *because all the electrons and PI-resolvents in this case are positive.*

A negative hyperresolution *is a special case of PI-resolution in which every literal in the interpretation I does not contain any negation sign. It is called* negative hyperresolution *because all the electrons and PI-resolvents in this case are negative.*

From Theorem 3.22 it follows that both positive and negative hyperresolution are complete. For details we refer for instance to [CL73].

3.5 Category Theory — Basic Notions

Definition 3.69 (Category) *A* category $\mathcal{C}$ *consists of:*

(1) a class of objects, Obj($\mathcal{C}$) (denoted also $|\mathcal{C}|$),

(2) for each pair of objects A, B a class of morphisms, $\mathsf{Hom}_{\mathcal{C}}(A, B)$, and

(3) a composition relation on morphisms,

such that

(i) For any two morphisms $f \in \mathsf{Hom}_{\mathcal{C}}(A, B)$ and $g \in \mathsf{Hom}_{\mathcal{C}}(B, C)$ the composition of f and g, $g \circ f \in \mathsf{Hom}_{\mathcal{C}}(A, C)$,

(ii) The composition of morphisms is associative, that is $h \circ (g \circ f) = (h \circ g) \circ f$,

(iii) For every object A there is the identity morphism $id_A \in \mathsf{Hom}_{\mathcal{C}}(A, A)$ with the property $f \circ id_A = f$ and $id_B \circ f = f$ for all $f \in \mathsf{Hom}_{\mathcal{C}}(A, B)$.

Remark: Instead of $\mathsf{Hom}_{\mathcal{C}}(A, B)$ one also writes $\mathcal{C}(A, B)$. The notations $f : A \to B$ and $A \xrightarrow{f} B$ both denote a morphism f in $\mathsf{Hom}_{\mathcal{C}}(A, B)$.

In the following, $\mathcal{C}$ and $\mathcal{D}$ will always denote arbitrary categories.

Typical examples of categories in mathematics are the category of *groups* (objects are groups, morphisms are the group homomorphisms), the category of *monoids* (objects are monoids, morphisms are the monoid homomorphisms), the category of *topological spaces* (objects are topological spaces, morphisms are the continuous functions between them), and, of course, the category of *sets* (objects are sets, morphisms are set mappings). Note also that to every preorder $(P, \leq)$ we can associate a category having as objects the elements of P and a (unique) morphism between p_1 and p_2 if and only if $p_1 \leq p_2$. Summarizing, we can say that category theory extracts the basic features of "every-day's work" when dealing with spaces in a certain discipline and studying structure preserving mappings (the morphisms) between those spaces.

The objects of a category do not necessarily form a set. A category $\mathcal{C}$ is called *locally small* if for any two objects C and D of $\mathcal{C}$ the set of morphisms $\mathcal{C}(C, D)$ is a set, while $\mathcal{C}$ is called *small* if both its collection of objects and its collection of morphisms are sets.

Definition 3.70 (Dual Category) *Let $\mathcal{C}$ be a category. The* dual *of $\mathcal{C}$, is the category $\mathcal{C}^{op}$ having the same objects as $\mathcal{C}$ and an arrow $f^{op} \in \mathcal{C}^{op}(B, A)$ for every $f \in \mathcal{C}(A, B)$.*

A morphism $f \in \mathcal{C}(C, D)$ is called *isomorphism* if there exists a morphism $g \in \mathcal{C}(D, C)$ such that $f \circ g = 1_D$ and $g \circ f = 1_C$.

A morphism $f \in \mathcal{C}(C, D)$ is called *epimorphism* if for any object E and any two parallel morphisms $g, h : D \rightrightarrows E$ in $\mathcal{C}$, $g \circ f = h \circ f$ implies $g = h$.

Dually, $f \in \mathcal{C}(C, D)$ is called *monomorphism* (or *monic*) if for any object B and any two parallel morphisms $g, h : B \rightrightarrows C$ in $\mathcal{C}$, $f \circ g = f \circ h$ implies $g = h$.

3.5.1 Limits and Colimits

We now recall the notion of *(co-)limit* which is one of the basic notions in category theory and in our modeling approach.

Definition 3.71 (Diagram) *A diagram $\mathbb{D} = (\{X_i\}_{i \in I}, \{F_{ij}\}_{i,j \in I})$ in a category $\mathcal{C}$ is defined as an indexed family of objects $\{X_i\}_{i \in I}$ and a family of morphism sets $F_{ij} \subseteq \mathcal{C}(X_i, X_j)$, for $i, j \in I$.*

Definition 3.72 (Co-Cone) *A co-cone of the diagram $\mathbb{D} = (\{X_i\}_{i \in I}, \{F_{ij}\}_{i,j \in I})$ consists of an object $X \in Obj(\mathcal{C})$ and, for every $i \in I$, a morphism $f_i : X_i \to X$,* *such that* $f_i = f_j \circ f_{ij}$ *for all* $j \in I$, *i.e. such that for every* $i, j \in I$ *and for every* $f_{ij} \in F_{ij}$ *the following triangle commutes*

$$\begin{tikzcd} & X & \\ X_i \arrow[ur, "f_i"] \arrow[rr, "f_{ij}"] & & X_j \arrow[ul, "f_j"'] \end{tikzcd}$$

Definition 3.73 (Colimit) *A colimit of the diagram* $\mathbb{D} = (\{X_i\}_{i\in I}, \{F_{ij}\}_{i,j\in I})$ *is a co-cone with the property that for every other co-cone given by morphisms* $f'_i : X_i \to X'$, $i \in I$, *there exists exactly one morphism* $f : X \to X'$, *such that* $f'_i = f \circ f_i$, *for all* $i \in I$ *(universality property).*

The colimit of a diagram $\mathbb{D} = (\{X_i\}_{i\in I}, \{F_{ij}\}_{i,j\in I})$ will be denoted by $\varinjlim(\mathbb{D})$. (In the literature, several other notations may be found, such as $\varinjlim_{i\in I} X_i$ or $\varinjlim_{\mathbb{D}} X_i$ or $\varinjlim X_i$.)

Reversing the arrows in the definition of a colimit of a diagram $\mathbb{D}$ results in the dual notion called *limit* of $\mathbb{D}$, denoted by $\varprojlim(\mathbb{D})$.

Remark: Starting with a diagram $\mathbb{D} = (\{X_i\}_{i\in I}, \{F_{ij}\}_{i,j\in I})$ that consists only of the objects X_i, $i \in I$, as "nodes" but without morphisms, i.e. all $F_{ij} = \emptyset$, we obtain the notion of the categorical *co-product*, $\coprod_{i\in I} X_i$ (as a special colimit) and *product*, $\prod_{i\in I} X_i$ (as a special limit), respectively. The morphisms f_i in the corresponding definition of $\varinjlim(\mathbb{D})$ and $\varprojlim(\mathbb{D})$ are called "canonical injections" of the co-product and "canonical projections" of the product, respectively. This in particular means that we can derive special notions of limits and colimits , corresponding to the shape of the base diagram $\mathbb{D}$.

Other important colimits are:

Initial Object Colimit of the diagram consisting of the empty set.

Coequalizers Colimits of diagrams consisting of two parallel arrows $A \underset{g}{\overset{f}{\rightrightarrows}} B$.

Pushouts Colimits of diagrams of the form: $\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"] & B \\ C & \end{tikzcd}$.

The dual notions are:

Terminal Object Limit of the diagram consisting of the empty set.

Equalizers Limits of diagrams consisting of two parallel arrows $A \underset{g}{\overset{f}{\rightrightarrows}} B$.

Pullbacks Limits of diagrams of the form: $\begin{tikzcd} & B \arrow[d, "f"] \\ C \arrow[r, "g"'] & A \end{tikzcd}$.

3.5.2 Functors and Natural Transformations

Definition 3.74 (Functor) *Let $\mathcal{C}$ and $\mathcal{D}$ be two categories. A* functor *from $\mathcal{C}$ to $\mathcal{D}$ is an operation which assigns to each object C of $\mathcal{C}$ an object $F(C)$ of $\mathcal{D}$, and to each morphism f of $\mathcal{C}$ a morphism $F(f)$ of $\mathcal{D}$ in such a way that F respects the domain and the codomain as well as the identities and the composition:*

- *If $f \in \mathcal{C}(A, B)$ then $F(f) \in \mathcal{D}(F(A), F(B))$,*
- *$F(id_A) = id_{F(A)}$,*
- *$F(f \circ g) = F(f) \circ F(g)$.*

A functor as defined above is also called *covariant functor*. A *contravariant functor* from $\mathcal{C}$ to $\mathcal{D}$ is a functor $F : \mathcal{C}^{op} \to \mathcal{D}$.

Definition 3.75 (Natural Transformation) *Let F, G be functors from a category $\mathcal{C}$ to a category $\mathcal{D}$. A* natural transformation *τ from F to G is a mapping assigning to each object A in $\mathcal{C}$ a morphism τ_A from $F(A)$ to $G(A)$ in $\mathcal{D}$ such that for every arrow $f : A \to B$ in $\mathcal{C}$ the following diagram in $\mathcal{D}$*

$$\begin{array}{ccc} F(A) & \xrightarrow{F(f)} & F(B) \\ {\scriptstyle \tau_A}\downarrow & & \downarrow{\scriptstyle \tau_B} \\ G(A) & \xrightarrow[G(f)]{} & G(B) \end{array}$$

commutes. That is, for every arrow $f : A \to B$ in $\mathcal{C}$ we have: $G(f) \circ \tau_A = \tau_B \circ F(f)$.

The arrows τ_A are called the components *of the natural transformation τ.*

If F and G are functors we will denote the family of all natural transformations from F to G by $[F, G]$.

Definition 3.76 (Natural Isomorphism) *A natural transformation $\tau : F \to G$ is called a* natural isomorphism *if every component τ_A is invertible (i.e., there is a natural transformation $\tau^{-1} : G \to F$ with $\tau_A^{-1} = (\tau_A)^{-1}$).*

It is also important to note that one can compose functors and natural transformations in a straightforward way.

Let a natural transformation $\tau : F \to G$ between two functors $F, G : \mathcal{C} \to \mathcal{D}$ be given. Let additionally functors $H : \mathcal{B} \to \mathcal{C}$ and $K : \mathcal{D} \to \mathcal{E}$ be given. Then $\tau \circ H$ (or τH) denotes the natural transformation from FH to GH given by the components $(\tau H)_A := \tau_{H(A)}$ (A an object in $\mathcal{B}$) and $K \circ \tau$ (or $K\tau$) denotes the natural transformation from KF to KG defined componentwise as $(K\tau)_A := K(\tau_A)$ (A an object in $\mathcal{C}$).

It is straightforward to check that the fact that τ is a natural transformation guarantees that τH and $K\tau$ are natural transformations.

Moreover, given three functors F, G and H, all from $\mathcal{C}$ to $\mathcal{D}$, and two natural transformations $\tau : F \to G$ and $\theta : G \to H$, one easily defines the composition of τ and θ: $(\theta \circ \tau)_A = \theta_A \circ \tau_A$, where A is an object in $\mathcal{C}$ and τ_A and θ_A, by definition, arrows in $\mathcal{D}$. Checking that $\theta \circ \tau$ defines a natural transformation is trivial.

3.5.3 On the Yoneda Lemma

We now introduce one more concept, that of hom-functors, in order to state one of the fundamental theorems in category theory, the Yoneda lemma.

Definition 3.77 (Hom-functors) *Let $\mathcal{C}$ be a locally small category. For an arbitrary object A in $\mathcal{C}$ we define a functor $F = \mathcal{C}(A, \_) : \mathcal{C} \to \mathsf{Sets}$ by $F(B) = \mathcal{C}(A, B)$ and for any arrow $f : B \to C$ in $\mathcal{C}$ we let $F(f) = \mathcal{C}(A, f) : \mathcal{C}(A, B) \to \mathcal{C}(A, C)$ map $h : A \to B$ to $f \circ h : A \to C$.*

We obtain a covariant functor $F : \mathcal{C} \to \mathsf{Sets}$.

Analogously, we define $G = \mathcal{C}(\_, A)$, a contravariant functor from $\mathcal{C}$ to Sets, on objects B of $\mathcal{C}$ and arrows $f : B \to C$ in $\mathcal{C}$ by $G(B) = \mathcal{C}(B, A)$ and $G(f) = \mathcal{C}(f, A) : \mathcal{C}(C, A) \to \mathcal{C}(B, A)$ which maps any $h : C \to A$ to $h \circ f : B \to A$.

Each object C of $\mathcal{C}$ gives rise to a functor $y(C) : \mathcal{C}^{op} \to \mathsf{Sets}$, defined on objects by $y(C)(D) = \mathcal{C}(D, C)$ and on morphisms $\alpha : D' \to D$ by $y(C)(\alpha) : \mathcal{C}(D, C) \to \mathcal{C}(D', C)$, $y(C)(\alpha)(u) = u \circ \alpha$ for every $u : D \to C$.

If $f : C_1 \to C_2$ is a morphism in $\mathcal{C}$, there is a natural transformation $y(C_1) \to y(C_2)$ obtained by composition with f. This makes y into a functor

$$y : \mathcal{C} \to \mathsf{Sets}^{\mathcal{C}^{op}}, y(C) = \mathcal{C}(\_, C).$$

The functor y is called the *Yoneda embedding*.

Theorem 3.23 (Yoneda Lemma) *Let $\mathcal{C}$ be a locally small category, and A an object of $\mathcal{C}$. If $F : \mathcal{C} \to \mathsf{Sets}$ is a covariant functor and if $\alpha : \mathcal{C}(A, \_) \to F$ denotes a natural transformation then there is a 1-1-correspondence between the set of natural transformations $[\mathcal{C}(A, \_), F]$ and the set $F(A)$, given by: $\alpha \mapsto \alpha_A(id_A)$.*

If we choose $F = \mathcal{C}(A', \_)$ we obtain a 1-1-correspondence between the set of morphisms $\mathcal{C}(A', A)$ and the set of natural transformations $[\mathcal{C}(A, \_), \mathcal{C}(A', \_)]$.

3.5.4 Adjoint Functors

We continue with another basic and widely applicable concept, that of adjoint functors.

Definition 3.78 (Adjoint Functors) *Let $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{C}$ be functors. We say that F is* left adjoint *to G if there is a natural isomorphism $\theta : \mathcal{D}(F\_, \_) \xrightarrow{\sim} \mathcal{C}(\_, G\_)$ between these hom-functors from $\mathcal{C}^{op} \times \mathcal{D} \to \mathsf{Sets}$, given by the components $\theta_{AB} : \mathcal{D}(FA, B) \xrightarrow{\sim} \mathcal{C}(A, GB)$.*

Pictorially:

$$\begin{array}{ccc} F(A) & & A \\ \downarrow & & \downarrow \\ B & & G(B) \end{array}$$

where the left part is in $\mathcal{D}$ and the right in $\mathcal{C}$.

The adjointness expresses that there is a 1-1 correspondence between the arrows $F(A) \to B$ in $\mathcal{D}$ and $A \to G(B)$ in $\mathcal{C}$.

The above definition says that there are two natural transformations, θ from the hom-functor $\mathcal{D}(F\_, \_)$ to $\mathcal{C}(\_, G\_)$ and $\tau = \theta^{-1}$ back from $\mathcal{C}(\_, G\_)$ to $\mathcal{D}(F\_, \_)$.

Definition 3.79 (Unit of an Adjunction) *Let θ be the natural isomorphism describing an adjunction between functors $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{C}$, with F left adjoint to G.*

The unit *η of this adjunction is the natural transformation between $id_{\mathcal{C}}$ and GF given by the components $\eta_A := \theta_{AF(A)}(id_{F(A)}) : A \to GF(A)$.*

Proposition 3.24 *The unit η of a transjunction satisfies the following universal property: for every morphism $g : A \to G(B)$ in $\mathcal{C}$ there is exactly one morphism $f : F(A) \to B$ in $\mathcal{D}$ such that $g = G(f) \circ \eta_A$.*

The co-unit of an adjunction can be defined by simply proceeding dually.

Definition 3.80 (Co-unit of an Adjunction) *Let θ be the natural isomorphism describing an adjunction between functors $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{C}$, with F left adjoint to G. Let τ be the inverse of θ. The* co-unit *ε of this adjunction is the natural transformation between $F \circ G$ and $id_{\mathcal{D}}$ given by the components $\varepsilon_B := \tau_{G(B)B}(id_{G(B)}) : FG(B) \to B$.*

The co-unit of course also has a universal property, namely: for all $f : F(A) \to B$ (arrows in $\mathcal{D}$) there is exactly one $g : A \to G(B)$ in $\mathcal{C}$ such that $f = \varepsilon_B \circ F(g)$.

Theorem 3.25 *Let F and G be functors $\mathcal{C} \underset{G}{\overset{F}{\rightleftarrows}} \mathcal{D}$. An adjunction (F, G, θ) such that F is left adjoint to G is completely given by one of the following three equivalent properties.*

1. *Functors F and G and a natural transformation $\eta : id_{\mathcal{C}} \to GF$ with the universal property of the unit η. (Then, θ is defined by $\theta_{AB}(f) = G(f) \circ \eta_A$.)*

2. *Functors F and G and a natural transformation $\varepsilon : FG \to id_{\mathcal{D}}$ with the universal property of the co-unit ε. (Then, θ is defined as the inverse of τ defined as $\tau_{AB} = \varepsilon_B \circ F(g)$.)*

3. *Functors F and G and natural transformations η and ε such that $\varepsilon F \circ F\eta = id_F$ and $G\varepsilon \circ \eta G = id_G$. These two* triangular identities *are represented in Diagram 3.1. (Then, θ is again defined in terms of η as above.)*

$$
\begin{tikzcd}
F \arrow[d, "F\eta"] \arrow[dr, "id_F"] & \\
FGF \arrow[r, "\varepsilon F"'] & F
\end{tikzcd}
\qquad
\begin{tikzcd}
G \arrow[r, "\eta G"] \arrow[dr, "id_G"'] & GFG \arrow[d, "G\varepsilon"] \\
& G
\end{tikzcd}
\tag{3.1}
$$

3.5.5 Other Categorical Notions

Definition 3.81 (Exponentiation) *A category $\mathcal{C}$ has exponentiation if it has all binary products and if for every pair of objects A, B there is an object B^A and a morphism $ev : B^A \times A \to B$ (the* evaluation map*) with the following universal property: for every $f : C \times A \to B$ there exists exactly one $\hat{f} : C \to B^A$ such that $ev \circ (\hat{f} \times id_A) = f$; this can be expressed by the commutativity of the following diagram:*

$$
\begin{tikzcd}
C \times A \arrow[d, "\hat{f}\times id_A"'] \arrow[dr, "f"] & \\
B^A \times A \arrow[r, "ev"'] & B
\end{tikzcd}
$$

The assignment $f \mapsto \hat{f}$ defines a canonical bijection from $\mathcal{C}(C \times A, B)$ to $\mathcal{C}(C, B^A)$. Recalling the definition of an adjunction, which involves a natural isomorphism between $\mathcal{D}(F\_, \_)$ and $\mathcal{C}(\_, G\_)$, this suggests that we may be able to find an adjunction based on this bijection. Indeed, the morphisms f and $\hat{f}$ are called *exponential adjoints*, motivated by the following result.

Proposition 3.26 *Let $\mathcal{C}$ be a category with exponentiation. Define the endofunctors $F := \_ \times A$ (with $F(f) = \hat{f} \times id_A$) and $G = (\_)^A$. Then G is right adjoint to F.*

For further basic categorical notions we refer to the literature on category theory. We refer here only to a few selected titles, namely [AHS90], [ML71], [HS79], [Pie91], [Gol84], and [MLM92].

3.6 Sheaf and Topos Theory — Basic Notions

In this section we present basic notions of sheaf and topos theory. For details we recommend [MLM92] and [Gol84]. For some remarks on sheaf theory and especially remarks on sheaves of algebras we refer to [Joh82], Chapter V.1.

3.6.1 Sheaves over Topological Spaces

In what follows we present well known results, that can be found for example in [Joh82] or [MLM92]. We closely follow the presentation of sheaves given in [Joh82].

An indexed system of sets $(F_i)_{i\in I}$ can alternatively be regarded as a map $f : F = \coprod_{i\in I} F_i \to I$, with the property that for every $x \in F$, $f(x) = i$ if and only if $x \in F_i$. If the index set I has a topology, then the set F can be endowed with a topology such that f is continuous (this then expresses the fact that the sets in the family $(F_i)_{i\in I}$ are continuously indexed).

Definition 3.82 (Bundle) *A* bundle *over I is a triple (F, f, I) where F and I are topological spaces and $f : F \to I$ is a continuous map with codomain I.*

For every $i \in I$, $f^{-1}(i)$ will be denoted by F_i, and $F = \coprod_{i\in I} F_i$.

Definition 3.83 (Morphism of Bundles) *Let (F, f, I) and (G, g, I) be two bundles over I. A morphism between (F, f, I) and (G, g, I) is a continuous map $h : F \to G$ such that $g \circ h = f$, i.e. the following diagram is commutative*

$$\begin{tikzcd} F \arrow[rr, "h"] \arrow[dr, "f"'] & & G \arrow[dl, "g"] \\ & I & \end{tikzcd}$$

The *category of bundles* over I is denoted Sp/I.

Definition 3.84 (Sections) *Let (F, f, I) be a bundle over I. A* partial section *defined on a open subset $U \subseteq I$ is a continuous map $s : U \to F$ with the property that $f \circ s$ is the inclusion $U \subseteq I$. A section defined on I is called* global section. *The set of all partial sections over the open subset U of I will be denoted by $\Gamma(F, f)(U)$.*

In what follows $\Omega(I)$ will denote the topology on I. $\Omega(I)$ is a poset and can be regarded as a category, with inclusions as morphisms.

Definition 3.85 (Presheaf) *A* presheaf *on a topological space I is a functor $P : \Omega(I)^{op} \to \mathsf{Sets}$.*

Let $U \subseteq V$ be open sets in I, and let $i_U^V : U \hookrightarrow V$ be the corresponding morphism in $\Omega(I)$. We will denote the restriction $P(i_U^V) : P(V) \to P(U)$ by ρ_U^V.

Definition 3.86 (Sheaf) *A* sheaf *on a topological space I is a presheaf $F : \Omega(I)^{op} \to \mathsf{Sets}$ that additionally satisfies the following condition:*

- *Given an open cover $(U_i)_{i\in I}$ of U and a family of elements $s_i \in F(U_i)$ such that for every pair (i, j) we have $\rho^{U_i}_{U_i\cap U_j}(s_i) = \rho^{U_j}_{U_i\cap U_j}(s_j)$, there is a unique $s \in F(U)$ with $\rho^U_{U_i}(s) = s_i$ for all $i \in I$.*

The morphisms of (pre)sheaves are natural transformations of functors. The category of presheaves over I will be denoted $\mathsf{PreSh}(I)$, and the category of sheaves over I will be denoted $\mathsf{Sh}(I)$.

For every bundle (F, f, I) let $\Gamma(F) = \{s : I \to F \mid s \text{ continuous and } f \circ s = id_I\}$ be the set of all the global sections of F. Γ defines a functor $\Gamma : \mathsf{Sp}/I \to \mathsf{PreSh}(I)$.

Definition 3.87 (Stalk) *Let F be a presheaf on I. The* stalk *of F at a point $i \in I$ is the colimit $F_i = \varinjlim_{i \in U} F(U)$, where U ranges over all open neighborhoods of i.*

The collection of stalks $(F_i)_{i\in I}$ is an I-indexed family of sets. Let $D(F)$ denote the disjoint union of the stalks, and let $\pi : D(F) \to I$ be the canonical projection on I. For $s \in F(U)$ and $i \in U$, let s_i be the image of s in F_i (the germ of s at i). The map $\overline{s} : U \to D(F)$, $\overline{s}(i) = s_i$ defines a partial section of the projection $D(F) \to I$; we impose on $D(F)$ the coarsest topology for which all such sections are continuous.

Let F be a presheaf. The construction above defines a bundle $D(F) = (D(F), \pi, I)$. A functor $D : \mathsf{PreSh}(I) \to \mathsf{Sp}/I$ can be defined this way.

Theorem 3.27 *The functor $D : \mathsf{PreSh}(I) \to \mathsf{Sp}/I$ preserves finite limits and is left adjoint to $\Gamma : \mathsf{Sp}/I \to \mathsf{PreSh}(I)$.*

Let LH/I be the full subcategory of Sp/I whose objects are of the form (F, f, I) with $f : F \to I$ a local homeomorphism (i.e. for every $a \in F$ there are open neighborhoods U and U' of a respectively $f(a)$ such that $f : U \to U'$ is a homeomorphism). It can be shown that any bundle map between two local homeomorphisms $f : F \to I$, $f' : F' \to I$ is itself a local homeomorphism.

Theorem 3.28 *The functors D and Γ restrict to an equivalence of categories between $\mathsf{Sh}(I)$ and LH/I.*

Note that for every presheaf P, $\Gamma(D(P))$ is a sheaf. We obtain therefore a functor $\Gamma \circ D : \mathsf{PreSh}(X) \to \mathsf{Sh}(X)$.

Theorem 3.29 *The inclusion $\mathsf{Sh}(X) \to \mathsf{PreSh}(X)$ has a left adjoint, namely the composition $\Gamma \circ D : \mathsf{PreSh}(X) \to \mathsf{Sh}(X)$. This functor is known as the* associated sheaf functor *or the* sheafification functor*.*

The associated sheaf functor is left exact, in the sense that it preserves all finite limits.

The category $\mathsf{Sh}(X)$ of sheaves over a topological space X has many good properties: it has limits, colimits and exponentiation; and additionally it has a notion of subobject and a subobject classifier.

Limits: For any space X, Sh(X) has all *small limits* (and they are computed pointwise). The inclusion of Sh(X) in PreSh(X) preserves all these limits.

Colimits: All small colimits exist in $\mathsf{Sh}(I)$. They can be computed by first computing the colimit in the category of presheaves and then taking the associated sheaf of the resulting presheaf, using the principle that left adjoints preserve colimits.

Exponentiation: For every topological space X, the category $\mathsf{Sh}(X)$ has exponentiation. Namely, let P and $F \in \mathsf{Sets}^{\Omega(X)^{op}}$ be presheaves. If F is a sheaf, then so is the (presheaf) exponential F^P. Since this will be needed later, we briefly indicate the way F^P is defined:

On objects:	$F^P(U) = \mathsf{Hom}(P_{\vert U}, F_{\vert U})$ in the category of presheaves over U ($P_{\vert U}, F_{\vert U}$ are the restrictions of P, F to $\mathcal{O}(U)^{op}$),
On morphisms:	for $V \subseteq U$, $F^P(i) : F^P(U) = \mathsf{Hom}(P_{\vert U}, F_{\vert U}) \to F^P(V) = \mathsf{Hom}(P_{\vert V}, F_{\vert V})$ is defined by $F^P(i)(\alpha) = \alpha_{\vert P_{\vert V}}$.

Definition 3.88 (Subfunctor) *Let $F : \mathcal{C}^{op} \to \mathsf{Sets}$. A* subfunctor *of F is a functor $G : \mathcal{C}^{op} \to \mathsf{Sets}$ such that*

(1) $G(C) \subseteq F(C)$ for every $C \in |\mathcal{C}|$.

(2) for every $f : C \to D$, $G(f) : G(D) \to G(C)$ is a restriction of $F(f)$.

Definition 3.89 (Subsheaf) *A* subsheaf *of a sheaf F is a subfunctor of F which is itself a sheaf.*

Definition 3.90 (Equivalence of Morphisms) *Two monomorphisms $f : A \to D$ and $g : B \to D$ are* equivalent *if there is an isomorphism $h : A \to B$ with $f = g \circ h$.*

Definition 3.91 (Subobject) *A* subobject *of D is an equivalence class of monomorphisms into D.*

It follows that a subobject of a sheaf F in the category $\mathsf{Sh}(X)$ is isomorphic to a subsheaf of F.

Definition 3.92 (Subobject Classifier) *A* subobject classifier *is a monic $true : 1 \to \Omega$ (in a category with finite limits) such that for every monic $S \to X$ there is a unique $\phi : X \to \Omega$ which forms the pullback square*

$$\begin{tikzcd} S \arrow[r, "!"] \arrow[d] & 1 \arrow[d, "true"] \\ X \arrow[r, "\phi"] & \Omega \end{tikzcd}$$

Let X be a topological space and $\mathcal{O}(X)$ be the family of open sets of X. Let the presheaf $\Omega : \mathcal{O}(X)^{op} \to \mathsf{Sets}$ be defined:

On objects:	$\Omega(U) = \{W \mid W \subseteq U, W \text{ open in } X\}$,
On morphisms:	If $i : V \subseteq U$ is the inclusion, then $\Omega(i) = \rho^U_V : \Omega(U) \to \Omega(V)$ is defined by $\rho^U_V(W) = W \cap V$.

Theorem 3.30 *For any topological space X, the presheaf Ω is a sheaf and is a subobject classifier for* $\mathsf{Sh}(X)$

3.6.2 Grothendieck Topologies

A Grothendieck topology is a generalization of the concept of a topology on a set. It is based on a notion of "cover" which is a generalization of open covers in a topology. Here, we briefly give the basic definitions – for more details cf. [MLM92].

Definition 3.93 (Sieve) *A* sieve S *on an object* C *in a category* $\mathcal{C}$ *is a collection of morphisms in* $\mathcal{C}$ *with codomain* C *which is closed under right composition (i.e. if* $f : B \to C \in S$*, then for any* $g : A \to B$*, the composition* $g \circ f : A \to C$ *is in* S*).*

Alternatively, a sieve can be seen as a subobject S of $y(C)$, where $y : \mathcal{C} \to \mathsf{Sets}^{\mathcal{C}^{op}}$ is the Yoneda embedding (cf. Section 3.5.3), $y(C) = \mathcal{C}(\_, C)$.

Definition 3.94 (Grothendieck topology) *A* Grothendieck topology J *on a category* $\mathcal{C}$ *is a function* J *which assigns to each object* $C \in \mathcal{C}$ *a collection* $J(C)$ *of sieves on* C*, in such a way that the following conditions are satisfied:*

(1) [Identity cover] For every object C *the maximal sieve* $\{f \mid cod(f) = C\}$ *is in* $J(C)$*,*

(2) [Stability] If $R \in J(C)$ *and* $f : B \to C$ *then the sieve* $f^*(R) = \{g : A \to B \mid f \circ g \in R\}$ *is in* $J(B)$*,*

(3) [Transitivity] If $R_1 \in J(C)$ *and* R_2 *is any sieve on* C *such that* $f^*(R_2) \in J(B)$ *for all* $f : B \to C \in R_1$*, then* $R_2 \in J(C)$*.*

If $S \in J(C)$*, one says that* S *is a* covering sieve*, or that* S covers C*.*

Intuitively, condition (1) states that the sieve generated by the identity arrow is a cover, condition (2) states that given a cover of an object and a substructure of that object, the restriction of the cover to the substructure is a cover of the substructure, and condition (3) states that covers of covers are again covers.

Definition 3.95 (Site) *A* site *is a pair* $(\mathcal{C}, J)$*, consisting of a (small) category* $\mathcal{C}$ *and a Grothendieck topology* J *on* $\mathcal{C}$*.*

In the case of ordinary topological spaces, one usually describes an open cover of U as just a family $\{U_i \mid i \in I\}$ of open subsets of U with union $\bigcup U_i = U$; such a family is not necessarily a sieve, but it generates a sieve – namely, the collections of all open sets $V \subseteq U$ with $V \subseteq U_i$ for some U_i. In the more general context of a category with pullbacks, this way of generating a covering sieve can be carried out in terms of a so-called basis for a Grothendieck topology.

Definition 3.96 (Basis) *A* basis *for a Grothendieck topology on a category* $\mathcal{C}$ *with pullbacks is a function* K *which assigns to each object* C *a collection* $K(C)$ *consisting of families of morphisms with codomain* C*, such that:*

(1) If $f : C' \to C$ is an isomorphism, then $\{f : C' \to C\} \in K(C)$;

(2) If $\{f_i : C_i \to C \mid i \in I\} \in K(C)$, then for any morphism $g : D \to C$ the family of pullbacks $\{\pi_2 : C_i \times_C D \to D \mid \in I\}$ is in $K(D)$;

(3) If $\{f_i : C_i \to C \mid i \in I\} \in K(C)$, and if for each $i \in I$ one has a family $\{g_{ij} : D_{ij} \to C_i \mid j \in I_i\} \in K(C_i)$, then the family of composites $\{f_i \circ g_{ij} : D_{ij} \to C \mid i \in I, j \in I_i\}$ is in $K(C)$.

The elements R of $K(C)$ are called covering families *or* covers *of C.*

Note that a Grothendieck topology J is not always a basis because condition (1) above may not be satisfied. A basis K generates a Grothendieck topology J by $(S \in J(C) \Leftrightarrow \exists R \in K(C) R \subseteq S)$.

Let $\mathcal{C}$ be a category and J a Grothendieck topology on $\mathcal{C}$. A sieve S on C is called *closed* (for J) if and only if for every arrow $f : D \to C$ in $\mathcal{C}$,

$$f^*(S) \in J(D) \text{ if and only if } f \in S.$$

3.6.3 Sheaves on a Site

The main difference between the definition that will be given in this section and the definitions 3.85 and 3.86 given in section 3.6.1 is the fact that here the topology on the index set $\Omega(I)$ is replaced with a more general category $\mathcal{C}$.

Definition 3.97 (Presheaf) *A* presheaf *on a category $\mathcal{C}$ is a contravariant functor from $\mathcal{C}$ to the category of sets* Sets.

A sheaf is a presheaf that satisfies an additional "gluing" condition.

Definition 3.98 (Sheaf) *A* sheaf *on a site $(\mathcal{C}, J)$ is a presheaf $F : \mathcal{C}^{op} \to \mathsf{Sets}$ such that for every object C of $\mathcal{C}$ and every covering sieve $R \in J(C)$, each morphism $R \to F$ in $\mathsf{Sets}^{\mathcal{C}^{op}}$ has a unique extension to a morphism $\mathsf{Hom}_{\mathcal{C}}(-, C) \to F$.*

Remark 3.31 *If F is a presheaf $F \in \mathsf{Sets}^{\mathcal{C}^{op}}$, and $R \in J(C)$ is a cover for C, a* matching family *of elements of F is a function that assigns to every element $f : D \to C$ of R an element $x_f \in F(D)$ such that $F(g)(x_f) = x_{fg}$. An* amalgamation *of such a matching family is an element $x \in F(C)$ with $F(f)(x) = x_f$ for every $f : D \to C \in R$.*
Then the previous definition states that F is a sheaf if and only if every matching family for any cover of any object of $\mathcal{C}$ has a unique amalgamation.

This can be also expressed by requiring that for every object C of $\mathcal{C}$ and each cover $R \in J(C)$ the following diagram is an equalizer:

$$F(C) \xrightarrow{e} \prod_{f \in R} F(dom(f)) \underset{a}{\overset{p}{\rightrightarrows}} \prod_{\substack{f,g; f \in R, \\ dom(f) = cod(g)}} F(dom(g)).$$

Let K be a basis for a topology on a category $\mathcal{C}$ with pullbacks, and J the Grothendieck topology generated by K. In this case, the sheaves for J can be described in terms of the basis K as follows: Given a K-cover $R = \{f_i : C_i \to C \mid i \in I\}$ of C, a family of elements $x_i \in F(C_i)$ is said to be *matching* for R iff $F(\pi^1_{ij})(x_i) = F(\pi^2_{ij})(x_j)$ for every $i, j \in I$, where π^1 and π^2 are the projections from the pullback:

$$\begin{tikzcd}
C_i \times_C C_j \arrow[r, "\pi^2_{ij}"] \arrow[d, "\pi^1_{ij}"'] & C_j \arrow[d, "f_j"] \\
C_i \arrow[r, "f_i"'] & C
\end{tikzcd} \tag{3.2}$$

An *amalgamation* for $\{x_i\}_{i \in I}$ is then an $x \in F(C)$ with $F(f_i)(x) = x_i$ for every $i \in I$.

Proposition 3.32 (cf. [MLM92]) *Let F be a presheaf on $\mathcal{C}$. Then F is a sheaf for J if and only if for any cover $\{f_i : C_i \to C \mid i \in I\}$ in the basis K, any matching family $\{x_i\}_i$ has a unique amalgamation.*

Let $\mathcal{C}$ be a small category and J a Grothendieck topology on $\mathcal{C}$. Let $\mathsf{Sh}(\mathcal{C}, \mathsf{J})$ be the full subcategory of $\mathsf{Sets}^{\mathcal{C}^{op}}$ consisting of the sheaves with respect to J.

Theorem 3.33 (cf. [MLM92]) *The inclusion functor $\iota : \mathsf{Sh}(\mathcal{C}, \mathsf{J}) \to \mathsf{Sets}^{\mathcal{C}^{op}}$ has a left adjoint $a : \mathsf{Sets}^{\mathcal{C}^{op}} \to \mathsf{Sh}(\mathcal{C}, \mathsf{J})$ called the* associated sheaf functor. *Moreover, the functor a commutes with finite limits.*

The composition $a \circ i : \mathsf{Sh}(\mathcal{C}, \mathsf{J}) \to \mathsf{Sh}(\mathcal{C}, \mathsf{J})$ is naturally isomorphic to the identity functor.

3.6.4 Topoi

In what follows we present elementary conditions (or axioms) that turn a category $\mathcal{E}$ into a topos.

Definition 3.99 (Subobject Classifier) *If $\mathcal{C}$ is a category with a terminal object 1, then a* subobject classifier *for $\mathcal{C}$ is a $\mathcal{C}$-object Ω together with a $\mathcal{C}$-morphism $true : 1 \to \Omega$ that satisfies the following axiom:*

Ω-axiom *For every monic $f : A \hookrightarrow E$ there exists a unique $\mathcal{C}$-morphism $char_A : E \to \Omega$ (denoted also $char_f$) such that the following diagram is a pullback:*

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, "!"'] & E \arrow[d, "char_A"] \\
1 \arrow[r, "true"'] & \Omega
\end{tikzcd} \tag{3.3}$$

The morphism $char_A$ is called the *characteristic morphism* of the subobject A of E.

Note that when a subobject classifier exists in a category, it is unique up to isomorphism.

Theorem 3.34 (cf. [MLM92]) *Let $f : A \hookrightarrow E$ and $g : B \hookrightarrow E$ be two subobjects of E. Then $f \simeq g$ if and only if $char_A = char_B$.*

Consequence 3.35 (cf. [MLM92]) *The assignment of $char_A$ to $f : A \hookrightarrow E$ establishes a one-one correspondence between the collection $\mathsf{Sub}(E)$ of subobjects of an object E, and the collection $\mathsf{Hom}_{\mathcal{C}}(E, \Omega)$ of arrows from E to Ω.*

Definition 3.100 (Topos) *An* elementary topos *is a category $\mathcal{E}$ such that*

(1) $\mathcal{E}$ is finitely complete,
(2) $\mathcal{E}$ is finitely co-complete,
(3) $\mathcal{E}$ has exponentiation,
(4) $\mathcal{E}$ has a subobject classifier.

Remark The properties (1) and (3) constitute the definition of *cartesian closed* categories. The condition (2) is implied by the combination (1), (3) and (4) (cf. [MLM92]). Thus, a topos is a cartesian closed category with a subobject classifier.

In what follows we give the main properties of topoi that will be used in the thesis.

Definition 3.101 (Image) *Let $f : A \to B$ be a morphism in $\mathcal{C}$. The* image of f, $\mathsf{Im}_{(f)}$ *is a monomorphism $m : M \to B$ such that there is a unique $e : A \to M$ with $f = m \circ e$ such that the following universality property is satisfied:*

$$\begin{tikzcd}
A \arrow[rr, "f"] \arrow[dr, "e"] \arrow[ddr, "\forall g"'] & & B \\
& M \arrow[ur, "m"] \arrow[d, "\exists! \varphi"] & \\
& \forall N \arrow[uur, "\forall h \mathsf{mono}"'] &
\end{tikzcd}$$

For every N, for every monomorphism $h : N \to B$ and every $g : A \to N$ such that $f = h \circ g$ there is a unique $\varphi : M \to N$ such that $m = h \circ \varphi$ and $g = \varphi \circ e$.

Proposition 3.36 *In a topos, every monomorphism is an equalizer.*

Proof: From the definition of a subobject classifier, every monomorphism is the equalizer of the following diagram:

$$\begin{tikzcd}
M \arrow[r, "m"] & N \arrow[r, "!"] \arrow[rr, bend right, "char_m"'] & 1 \arrow[r, "true"] & \Omega
\end{tikzcd}$$

Proposition 3.37 *In a topos, every arrow f has an image m and factors as $f = m \circ e$ (with e epimorphism).*

The proof can be found in [MLM92], p.185. Since we will need it later, we point out the categorical constructions necessary for determining m and e:

Let $f : A \to B$. Let $x, y : B \to C$ be such that

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "f"'] & B \arrow[d, "y"] \\ B \arrow[r, "x"] & C \end{tikzcd}$$

is a pushout. Let $m : M \to B$ be the equalizer of $x, y : B \to C$. By the universality property of the equalizer, there is a unique $e : A \to M$ such that $f = m \circ e$. Hence, the image of f is obtained by pushouts and equalizers in the topos $\mathcal{E}$.

Proposition 3.38 *For each object A in a topos the partially ordered set $\mathsf{Sub}(A)$ of subobjects of A is a lattice. Moreover, for each arrow $k : A \to B$, taking pullback along k defines a morphism $k^{-1} : \mathsf{Sub}(B) \to \mathsf{Sub}(A)$ of partially ordered sets (i.e. a functor). This functor has as right adjoint the functor $\exists_k$ which sends each subobject S of A to its image under k in B, and a left adjoint $\forall_k$.*

Since this will be used in what follows, we sketch here how the intersection and union are obtained in $\mathsf{Sub}(A)$. The intersection of two subobjects $S, T \hookrightarrow A$ is the pullback in the Diagram3.4:

$$\begin{tikzcd} S \cap T \arrow[r] \arrow[d] & T \arrow[d] \\ S \arrow[r] & A \end{tikzcd} \tag{3.4}$$

The union of two subobjects $S, T \hookrightarrow A$ is obtained as follows: We first form the coproduct $S + T$ of S and T in the topos. The arrow $S + T \to A$ obtained by the universality property of the coproduct has an image M as shown in Diagram 3.5. M is a subobject of A, which contains both S and T. It is not difficult to see that M satisfies the properties of a g.l.b. hence $M = S \cup T$.

$$\begin{tikzcd} S + T \arrow[dr] & & T \arrow[ll] \arrow[dd] \\ & M \arrow[dr] & \\ S \arrow[uu] \arrow[rr] & & A \end{tikzcd} \tag{3.5}$$

3.6.5 Geometric Morphisms

The definition of a "map" between two topoi is based on the examples of sheaves on topological spaces. Let X and Y be topological spaces and $\mathsf{Sh(X)}$, $\mathsf{Sh(Y)}$ the topoi of sheaves on X, resp. Y.

A continuous function $f : X \to Y$ gives rise to a pair of adjoint functors: an *inverse image functor* f^* and a *direct image functor* f_*, with f^* left adjoint to f_*, as follows:

$$\begin{tikzcd} \mathsf{Sh(X)} \arrow[r, shift left, "f_*"] & \mathsf{Sh(Y)} \arrow[l, shift left, "f^*"] \end{tikzcd} \qquad f_* \vdash f^*. \tag{3.6}$$

The direct image functor is defined by composition with f^{-1}, namely if $F : \Omega(X)^{op} \to \mathsf{Sets}$ is a sheaf on X and U is any open subset of Y, then $f_*(F)(U) = F(f^{-1}(U))$. The inverse image functor is usually defined in terms of the étale spaces corresponding to the sheaves: if $p : E \to Y$ is étale, then $f^*(E \xrightarrow{p} Y)$ is the étale space over X defined by pullback along f, as in the following diagram:

$$\begin{tikzcd} f^*(E) \arrow[r] \arrow[d] & E \arrow[d, "p"'] \\ X \arrow[r, "f"'] & Y \end{tikzcd} \tag{3.7}$$

From this definition it follows that f^* preserves finite limits, i.e. it is left exact. Additionally, since $f_* \vdash f^*$ it follows that f^* preserves all colimits and f_* preserves all limits.

Definition 3.102 (Geometric Morphism) *A* geometric morphism $f : \mathcal{E} \to \mathcal{F}$ *between the topoi* $\mathcal{E}$ *and* $\mathcal{F}$ *is a pair of functors* $f^* : \mathcal{E} \to \mathcal{F}$ *and* $f_* : \mathcal{F} \to \mathcal{E}$ *such that* f^* *is left adjoint to* f_* *and* f^* *is left exact. Then* f_* *is called the* direct image *part of* f *and* f^* *the* inverse image part *of the geometric morphism.*

3.6.6 Morphisms of Sites

Definition 3.103 *Let* (C, J) *and* (D, K) *be sites. A functor* $\phi : \mathsf{C} \to \mathsf{D}$ preserves covers *if and only if for every covering sieve* $\mathcal{S} \in J(C)$ *the sieve* $\phi(\mathcal{S})$ *generated by* $\{\phi(u) \mid u : C' \to C \in \mathcal{S}\}$ *is a covering sieve of* $\phi(C)$ *in* D.

Definition 3.104 (Morphism of Sites) *Let* (C, J) *and* (D, K) *be sites. Suppose that* C *and* D *are closed under finite limits. A functor* $\phi : \mathsf{C} \to \mathsf{D}$ *is a* morphism of sites *if* ϕ *preserves finite limits and covers.*

Theorem 3.39 ([MLM92], Th.2, p.409) *For categories* C *and* D *with finite limits, any morphism of sites* $\phi : (\mathsf{C}, J) \to (\mathsf{D}, K)$ *induces a geometric morphism* $f : \mathsf{Sh}(\mathsf{D}, \mathsf{K}) \to \mathsf{Sh}(\mathsf{C}, \mathsf{J})$*; the direct image functor* $f_* : \mathsf{Sh}(\mathsf{D}, \mathsf{K}) \to \mathbf{Sh}(\mathbf{C}, \mathbf{J})$ *sends a sheaf* F *on* (D, K) *to the composition* $f_*(F) = F \circ \phi$*, and the inverse image functor* $f^* : \mathsf{Sh}(\mathsf{C}, \mathsf{J}) \to \mathsf{Sh}(\mathsf{D}, \mathsf{K})$ *sends a sheaf* G *on* (C, J) *to the tensor product* $G \otimes_C A_\phi$ *where*

$$A_\phi = a \circ y \circ \phi : C \xrightarrow{\phi} D \xrightarrow{y} \mathsf{Sets}^{\mathsf{D}^{op}} \xrightarrow{a} \mathsf{Sh}(\mathsf{D}, \mathsf{K}).$$

If the sites (C, J) and (D, K) do not have finite limits a similar theorem can be formulated.

Definition 3.105 (Covering Lifting Property) *A functor* $\pi : \mathsf{D} \to \mathsf{C}$ *is said to have the* covering lifting property *(clp) if for any object* D *of* D *and any* J*-cover* $\mathcal{S}$ *of* $\pi(D)$*, there exists a* K*-cover* $\mathcal{R}$ *of* D *such that* $\pi(\mathcal{R}) = \{\pi(u) \mid u \in \mathcal{R}\} \subseteq \mathcal{S}$.

In other words, π has clp if for every object D of D, every cover of the image of D is refined by the image of a cover of D itself.

Theorem 3.40 ([MLM92], Th.4, p.412) *Let* (C,J) *and* (D,K) *be sites, and let* $\pi : \mathsf{D} \to \mathsf{C}$ *and* $\phi : \mathsf{C} \to \mathsf{D}$ *be functors such that* π *is left adjoint to* ϕ*. If* π *has the covering lifting property, or equivalently, if* ϕ *preserves covers, then there is an induced geometric morphism* $f : \mathsf{Sh}(\mathsf{D},\mathsf{K}) \to \mathsf{Sh}(\mathsf{C},\mathsf{J})$*, with inverse and direct image functors described, for sheaves* F *on* (C,J) *and* G *on* (D,K) *by* $f^*(F) = a(F \circ \pi)$ *and* $f_*(G) = G \circ \phi$*.*

In fact just the functor $\pi : \mathsf{D} \to \mathsf{C}$ alone suffices to give a geometric morphism of sheaves, provided π has the covering lifting property.

Theorem 3.41 ([MLM92], Th.5, p.412) *Let* $\pi : \mathsf{D} \to \mathsf{C}$ *be a functor having the covering lifting property. Then* π *induces a flat and continuous functor* $A_\pi : \mathsf{C} \to \mathsf{Sh}(\mathsf{D},\mathsf{K})$*, defined by* $A_\pi(C) = a \circ C(\pi -, C)$*, and hence a geometric morphism* $f\mathsf{Sh}(\mathsf{D},\mathsf{K}) \to \mathsf{Sh}(\mathsf{C},\mathsf{J})$*, with inverse image functor* $f^*(F) \simeq a(F \circ \pi)$ *for any sheaf* F *on* C*.*

3.6.7 Geometric Logic

Let $\mathcal{L}$ be a many-sorted first-order language given by a collection of "sorts" or "types", collections of relation symbols and of function symbols together with their arities. Starting from the language $\mathcal{L}$ one can inductively define terms and atomic formulae; from atomic formulae one can build up more complicated formulae using the connectives $\vee, \wedge, \Rightarrow, \neg$ and quantifiers for any sort X.

For such a first order language $\mathcal{L}$ one can define an interpretation of $\mathcal{L}$ in a topos $\mathcal{E}$ by associating an object X^M of $\mathcal{E}$ to every sort X in the language, a subobject $R^M \subseteq X_1^M \times \ldots \times X_n^M$ to every relation symbol $R \subseteq X_1 \times \ldots \times X_n$ of $\mathcal{L}$ and an arrow $f^M : X_1^M \times \ldots \times X_n^M \to Y^M$ in $\mathcal{E}$ to each function symbol $f : X_1 \times \ldots \times X_n \to Y$.

Given such an interpretation M of $\mathcal{L}$ in a topos $\mathcal{E}$, one can define for each term $t(x_1, \ldots, x_n)$ of sort Y an arrow $t^M : X_1^M \times \ldots \times X_n^M \to Y^M$, and for every formula $\phi(x_1, \ldots, x_n)$ with free variables $FV(\phi) \subseteq \{x_1, \ldots, x_n\}$ (where x_i is of sort X_i), a subobject $\{(x_1, \ldots, x_n) \mid \phi(x_1, \ldots, x_n)\} \subseteq X_1^M \times \ldots \times X_n^M$.

For example, if $\phi(x_1, \ldots x_n) = R(t_1, \ldots, t_k)$ for some relation symbol R, then the subobject $\{(x_1, \ldots, x_n) \mid R(t_1, \ldots, t_k)\}$ is the pullback of the subobject R^M of $X_1 \times \ldots \times X_n$ along $\left\langle t_1^M, \ldots, t_k^M \right\rangle$.

We interpret the conjunction of two formulae by forming the pullback of the corresponding subobjects; similarly disjunctions are interpreted as unions in $\mathcal{E}$ (see Theorem 3.38 and the subsequent comments). To interpret $\forall$ and $\exists$ recall that for each arrow $k : A \to B$, taking pullback along k defines a morphism $k^{-1} : \mathsf{Sub}(B) \to \mathsf{Sub}(A)$ of partially ordered sets (i.e. a functor). This functor has a right adjoint $\exists_k : \mathsf{Sub}(A) \to \mathsf{Sub}(B)$ and a left adjoint $\forall_k : \mathsf{Sub}(A) \to \mathsf{Sub}(B)$. Thus, quantifiers of the language $\mathcal{L}$ can be interpreted by these adjoints:

$$\{(x_1, \ldots, x_n) \mid (\forall x : X)\phi(x_1, \ldots, x_n, x)\}^M = \forall_\pi(\{\{(x_1, \ldots, x_n, x) \mid \phi(x_1, \ldots, x_n, x)\}^M),$$

$$\{(x_1, \ldots, x_n) \mid (\exists x : X)\phi(x_1, \ldots, x_n, x)\}^M = \exists_\pi(\{\{(x_1, \ldots, x_n, x) \mid \phi(x_1, \ldots, x_n, x)\}^M),$$

where $\pi : X_1^M \times \ldots \times X_n^M \times X^M \to X_1^M \times \ldots \times X_n^M$ is the projection.

Definition 3.106 (Coherent (Geometric) Formulae) *We call* coherent formulae *(called also* geometric formulae*) those formulae built up from atomic formulae using only the connectives $\vee$ and $\wedge$ and the quantifier $\exists$.*

Definition 3.107 (Coherent (Geometric) Axioms) *We call* coherent axioms *(called also* geometric axioms*) are formulae of the form $(\forall x_1, \ldots, x_n)(\phi \Rightarrow \psi)$ where ϕ and ψ are coherent formulae.*

We say that a coherent axiom $(\forall x_1, \ldots, x_n)(\phi \Rightarrow \psi)$ is satisfied in a given interpretation M of L in a topos $\mathcal{E}$ if $\{(x_1, \ldots, x_n) \mid \phi\}^M$ is a subobject of $\{(x_1, \ldots, x_n) \mid \psi\}^M$ in $\mathcal{E}$.

Let now $f : \mathcal{F} \to \mathcal{E}$ be a geometric morphism. Then the inverse image functor f^* yields for every interpretation M of $\mathcal{L}$ in $\mathcal{E}$ an interpretation f^*M in $\mathcal{F}$.

The functor f^* preserves arbitrary colimits and finite limits. Therefore, it preserves equalizers, intersections (obtained by pullback), unions (obtained as images of coproducts, where images are obtained by pushouts and equalizers) and image factorization (obtained by by pushouts and equalizers); hence it preserves the interpretation of any coherent formula. Thus, for every formula built up from atomic formulae using only the connectives $\vee$ and $\wedge$ and the connective $\exists$:

$$f^*(\{(x_1, \ldots, x_n) \mid \phi\}^M) = \{(x_1, \ldots, x_n) \mid \phi\}^{f^*M}.$$

In general f^* will not preserve the interpretation of universally quantified formulae. Nevertheless, one can see that for every coherent axiom

$$\Phi = (\forall x_1, \ldots, x_n)(\phi \Rightarrow \psi)$$

with ϕ and ψ coherent formulae, if Φ is satisfied in a given interpretation M of $\mathcal{L}$ in the topos $\mathcal{E}$ then Φ is satisfied in the induced interpretation f^*M in $\mathcal{F}$.

To see this, let M be an interpretation of L in $\mathcal{E}$ and f a geometric morphism. Then Φ is satisfied in the interpretation M of L in $\mathcal{E}$ if and only if $\{(x_1, \ldots, x_n) \mid \phi\}^M$ is a subobject of $\{(x_1, \ldots, x_n) \mid \psi\}^M$. Since f^* preserves all finite limits it also preserves pullbacks, so

$$f^*(\{(x_1, \ldots, x_n) \mid \phi\}^M) \leq f^*(\{(x_1, \ldots, x_n) \mid \psi\}^M) \text{ is a subobject.}$$

Using the fact that $f^*(\{(x_1, \ldots, x_n) \mid \phi\}^M) = \{(x_1, \ldots, x_n) \mid \phi\}^{f^*M}$ and $f^*(\{(x_1, \ldots, x_n) \mid \psi\}^M) = \{(x_1, \ldots, x_n) \mid \psi\}^{f^*M}$ it follows that

$$\{(x_1, \ldots, x_n) \mid \phi\}^{f^*M} \leq \{(x_1, \ldots, x_n) \mid \psi\}^{f^*M} \text{ is a subobject.}$$

The direct image functor f_* being right adjoint preserves limits, but it does not normally preserve unions or images, so we cannot expect it to preserve the validity of coherent axioms.

We will briefly explain which are the formulae whose interpretations are preserved by direct image functors. It is easy to see that the interpretation of a conjunction of atomic formulas is preserved by direct image functors. Existential quantification is not always preserved. We analyze once more the way existential quantifiers are interpreted. Let $\phi(x_1, \ldots, x_n, x)$ be a formula over $\mathcal{L}$. Then

$$\{(x_1, \ldots, x_n) \mid (\exists x)\phi(x_1, \ldots, x_n, x)\}^M = \exists_\pi(\{(x_1, \ldots, x_n, x) \mid \phi(x_1, \ldots, x_n, x)\})^M$$

where π is the corresponding projection function.

By the definition of $\exists_\pi$, $\exists_\pi(\{(x_1, \ldots, x_n, x) \mid \phi(x_1, \ldots, x_n, x)\}^M$ is the image of $\pi \circ \iota$

$$\{(x_1, \ldots, x_n, x) \mid \phi(x_1, \ldots, x_n, x)\}^M \xrightarrow{\iota} X_1^M \times \ldots \times X_n^M \times X^M \xrightarrow{\pi} X_1^M \times \ldots \times X_n^M.$$

The image of $\pi \circ \iota$ can be obtained as the equalizer of two arrows obtained by the the pushout of $\pi \circ \iota$ with itself. In the particular case when $\pi \circ \iota$ is a monomorphism, the image of $\pi \circ \iota$ is $\{(x_1, \ldots, x_n, x) \mid \phi(x_1, \ldots, x_n, x)\}^M$. The fact that $\pi \circ \iota$ is a monomorphism reflects the situation described intuitively (and informal) in what follows: "the value of x in X^M with the property that $\phi(x_1, \ldots, x_n, x)$ is uniquely determined by the values of $x_1, \ldots, x_2, \ldots, x_n$".

In such cases, the image factorization of $\pi \circ \iota$ is preserved by any functor that preserves monomorphisms. This shows that a certain amount of quantification is preserved by direct image functors.

A formula ϕ is called *cartesian* relative to a given theory $\mathbb{T}$ if it is constructed from atomic formulae using only conjunction and existential quantification over "$\mathbb{T}$-provably unique" variables (i.e. variables whose values, in any model of $\mathbb{T}$, are uniquely determined by the values of the remaining free variables). Cartesian axioms (relative to a given theory $\mathbb{T}$) are similarly defined: they are axioms of the form $(\forall x)(\phi(x) \Rightarrow \psi(x))$ where the formulae ϕ and ψ are cartesian relatively to $\mathbb{T}$. We say that a theory is cartesian if its axioms can be ordered such that each is cartesian relative to those which precede it. Then it follows that models of cartesian theories are preserved by direct image functors.

Chapter 4

A Brief Overview of Related Results

In this chapter we review concepts and results that are directly linked to our own results, that will be presented in the thesis, as well as other related work. We begin by presenting some basic results on sheaves of algebras and the Priestley duality for distributive lattices. These will be used in Chapter 5, and a result due to Davey [Dav73] concerning a method of constructing a sheaf whose stalks are quotients of a given (universal) algebra will be used in Chapter 8, in the study of the behavior of interacting systems.

We briefly point out how both the sheaf representation of algebras and the Priestley representation for distributive lattices lead to fibered structures. We then present the basic results in many-valued resolution. We end by presenting various models for the study of concurrency: first classical approaches are considered, then the method of logical fiberings due to J. Pfalzgraf, and then other models based on sheaves and presheaves.

The results contained in this chapter are in general well-known and can be found in the literature. My contribution consists in organizing the information, pointing out the fact that in both Priestley and sheaf representation fibered structures appear, and in extending a result concerning unification in discriminator varieties due to [Bur92] from a single equation, as appears in [Bur92], to systems of equations (Theorem 4.2).

4.1 Representations of Algebras

4.1.1 Sheaves of Algebras

The stalks and the sets of global sections of a given sheaf may have an algebraic structure (for example in algebraic geometry they may carry a ring or group structure; in model theory sheaves with algebraic structure and their model-theoretic properties are studied [Wer75]).

We present here basic definitions and results concerning sheaves with algebraic structure. More information about the subject can be found for example in [Dav73], [Wer75], [KC79], [Joh82], [MLM92].

Let $\mathcal{A}$ be an algebraic variety (i.e. a class of algebras closed under homomorphic images, subalgebras and products). The variety $\mathcal{A}$ can be described by its signature Σ and by a set Id of identities.

Given an arbitrary category $\mathcal{C}$ with finite products, we can interpret the notion of an object with an $\mathcal{A}$-structure in $\mathcal{C}$ by associating to every operation symbol $\sigma \in \Sigma$ with arity n a morphism $\sigma_A : A^n \to A$ (where A is the object carrying the $\mathcal{A}$-structure), and interpreting every equation as the statement on the commutativity of an appropriate diagram.

A homomorphism between two $\mathcal{A}$-algebras A and B in the category $\mathcal{C}$ is a morphism $h \in \mathsf{Hom}_{\mathcal{C}}(A, B)$ such that for every operation symbol $\sigma \in \Sigma$ with arity n, the following diagram commutes

$$\begin{tikzcd} A^n \arrow[r, "\sigma_A"] \arrow[d, "h^n"'] & A \arrow[d, "h"'] \\ B^n \arrow[r, "\sigma_B"] & B \end{tikzcd} \tag{4.1}$$

The category obtained this way will be denoted $\mathcal{AC}$.

Let $\mathcal{C}, \mathcal{D}$ be categories with finite products and $F : \mathcal{C} \to \mathcal{D}$ a functor preserving finite products. Then for any algebraic variety $\mathcal{A}$, F lifts to a functor $\mathcal{A}F : \mathcal{AC} \to \mathcal{AD}$.

Let $\mathcal{A}$ be an algebraic variety. We know that for every topological space I, the category $\mathsf{Sh}(I)$ has finite limits (in particular finite products). Therefore we can construct the category $\mathcal{A}\mathsf{Sh}(I)$, as described above.

Proposition 4.1 (cf. [Joh82]) *Let $F \in \mathsf{Sh}(I)$ be a sheaf. Assume that F carries an $\mathcal{A}$-structure. Then for every $U \in \Omega(I)$, $F(U)$ has an $\mathcal{A}$-structure, and for every $V \subseteq U$, $\rho^U_V : F(U) \to F(V)$ is a $\mathcal{A}$-morphism. Conversely, if for every $U \in \Omega(I)$, $F(U)$ has an $\mathcal{A}$-structure, and for every $V \subseteq U$, $\rho^U_V : F(U) \to F(V)$ are $\mathcal{A}$-morphisms, then one can give an $\mathcal{A}$-structure to F.*

Proof: A sheaf $F \in \mathcal{A}\mathsf{Sh}(I)$ carries an $\mathcal{A}$-structure if and only if for every $\sigma \in \Sigma$ with arity n, there is a morphism in $\mathsf{Sh}(I)$ (i.e. a natural transformation) $\sigma_F : F^n \to F$, and the diagrams corresponding to the identities that characterize $\mathcal{A}$ commute.

Therefore, if $F \in \mathcal{A}\mathsf{Sh}(I)$ carries an $\mathcal{A}$-structure, it follows that for every open set $U \in \Omega(I)$, $\sigma_F(U) : F^n(U) \to F(U)$ for every $\sigma \in \Sigma$; and all diagrams corresponding to the identities that characterize $\mathcal{A}$ commute at U.

As $F^n(U) = F(U)^n$ it follows that for every $U \in \Omega(I)$, $F(U)$ carries an $\mathcal{A}$-structure, and by the fact that for every $\sigma \in \Sigma$, σ_F is a natural transformation it follows that for every $V \subseteq U$, $\rho^U_V : F(U) \to F(V)$ is a $\mathcal{A}$-morphism. $\square$

Therefore we may regard an $\mathcal{A}$-algebra in $\mathsf{Sh}(I)$ as a sheaf on I with values in the variety $\mathcal{A}$.

Proposition 4.2 (cf. [Joh82]) *Let $F \in \mathsf{Sh}(I)$ be a sheaf. Assume that F carries an $\mathcal{A}$-structure. Then its stalks form an indexed family of $\mathcal{A}$-algebras.*

Proof: For every $i \in I$, the functor $\mathsf{Stalk}_i : \mathsf{Sh}(I) \to \mathsf{Sets}$ which sends F to F_i preserves finite limits, and hence induces a functor $\mathcal{A}\mathsf{Sh}(I) \to \mathcal{A}$. □

The converse is not true in general: we need to know in addition that the algebra operations defined on the stalks "fit together continuously".

For example, let α be a binary operation on A. Let the corresponding operations in the fibers be $\alpha_{A_i} : A_i \times A_i \to A_i$. These operations can be combined to give a map $\alpha_{D(A)} : D(A) \times_I D(A) \to D(A)$, where $D(A) = \coprod_{i \in I} A_i$, $\pi : D(A) \to I$ the canonical projection and $D(A) \times_I D(A) = \{(y, y') \in D(A) \mid \pi(y) = \pi(y')\}$ is the fibered product. The morphism $\alpha_{D(A)}$ is defined component-wise.

These remarks lead to the following definition:

Definition 4.1 *A* sheaf of algebras *over a topological space I is a triple (F, f, I) where:*

(1) F and I are topological spaces (F is called sheaf space and I base space).

(2) The map $f : F \to I$ is a local homeomorphism[1].

(3) For every $i \in I$, $F_i = f^{-1}(i)$ forms an algebra, and all the stalks are of the same type, Σ.

(4) For every operation symbol $\sigma \in \Sigma$, with $a(\sigma) = n$, the induced mapping $\overline{\sigma} : F^{<n>} \to F$ is continuous, where $F^{<n>} = \{(a_1, \dots, a_n) \in F^n \mid f(a_1) = \dots = f(a_n)\}$ (with the topology induced by the product topology of F^n), and $\overline{\sigma}(a_1, \dots, a_n) = \sigma_{F_i}(a_1, \dots, a_n)$ if $f(a_j) = i$ for all $j = 1, \dots, n$.

A Construction by Davey

Let A be an algebra of similarity type Σ, let $(\theta_i)_{i \in I}$ be a family of congruences on A, and let τ be a topology on I. The following problem was addressed and solved in [Dav73]: In which situation does a sheaf exist with fibers $A_i = A/\theta_i$ such that for every $a \in A$ the map $[a] : I \to \coprod_{i \in I} A_i$ is a global section? We briefly present the main results from [Dav73].

Two constructions are possible:

Construction 1 ([Dav73]): Let (F_A, f, I) be defined by $F_A = \coprod_{i \in I} A/\theta_i$, and $f : F_A \to I$ be the natural projection. Assume that a subbasis for the topology on F_A is $\{[a](U) \mid U \in \tau, a \in A\}$, where $[a](U) = \{[a](i) \mid i \in U\} = \{[a]_{\theta_i} \mid i \in U\}$.

Construction 2 ([Dav73]): Let $G_A : \tau \to \Sigma Alg$ be defined on objects by $G_A(U) = A/\theta_U$, where $\theta_U = \bigwedge_{i \in U} \theta_i$ and on morphisms, for every $V \subseteq U$ by the canonical morphism $G_A(U) = A/\theta_U \to A/\theta_V = G_A(V)$, $a_{\theta_U} \mapsto a_{\theta_V}$.

Let $G_i = \varinjlim_{i \in U} G_A(U)$ be the fibers, and for every $i \in I$ let $g_i : G_i \to A_i$ be the unique morphism that arises from the universality property of the colimit. $g_i(\rho_i^U(a)) = a_{\theta_i}$ for every $U \in \tau$ and every $i \in I$.

[1] A map $f : F \to I$ is a local homeomorphism if for every point $x \in F$ there exists a neighborhood U of x in F such that $f(U)$ is open and $f : U \to f(U)$ is a homeomorphism.

It is easy to see that G_A is a presheaf, but it is not necessarily a sheaf. Let (SG_A, g, I) be the associated sheaf.

Note that in the first construction, the stalk at i is isomorphic to A_i, but (F_A, f, I) might be not a sheaf space. In the second construction, (SG_A, g, I) is a sheaf space, but $g_i : G_i \to A_i$ may not be an isomorphism.

Theorem 4.3 ([Dav73]) *The following conditions are equivalent:*

(1) If $[a]_{\theta_i} = [b]_{\theta_i}$ then there is an open neighborhood U of i such that for every $j \in U$, $[a]_{\theta_j} = [b]_{\theta_j}$.

(2) (F_A, f, I) is a sheaf of algebras.

(3) For every $i \in I$, $g_i : G_i \to A_i$ is an isomorphism.

Definition 4.2 (S-topology, [Dav73]) *If $(\theta_i)_{i \in I}$ is a family of congruences on an algebra A, then any topology on I that satisfies (1) is called an* S-topology.

Corollary 4.4 ([Dav73]) *Assume that the topology on I is an S-topology with respect to the family of congruences $(\theta_i)_{i\in I}$. Then (F_A, f, I) and (SG_A, g, I) are isomorphic sheaves of algebras for which*

(1) The stalk at i is isomorphic to $A_i = A/\theta_i$,

(2) The map $\alpha : A \to \Gamma(I, F_A)$ defined by $\alpha(a) = ([a]_{\theta_i})_{i\in I}$ is a homomorphism,

(3) In $A \xrightarrow{\alpha} \Gamma(I, F_A) \leq \prod_{i\in I} A/\theta_i \xrightarrow{p_i} A/\theta_i$:

(i) $p_i \circ \alpha$ is an epimorphism, and
(ii) A is a subdirect product of the family $(A/\theta_i)_{i\in I}$ if and only if $\bigwedge_{i\in I} \theta_i = \Delta_A$ (i.e. if and only if α is a monomorphism).

The coarsest S-topology can be constructed as follows:

Lemma 4.5 (cf. [Dav73, KC79]) *Let $A \hookrightarrow \prod_{i\in I} A_i \xrightarrow{p_i} A_i$ be a subdirect product. The coarsest S-topology on I is the topology generated by the sets $E(a, b) = \{i \in I \mid p_i(a) = p_i(b)\}$ as a subbasis.*

We briefly present a number of results, to round up the previous considerations.

Lemma 4.6 ([KC79]) *Suppose that $A \hookrightarrow \prod_{i\in I} A_i \xrightarrow{p_i} A_i$ is a subdirect product and let τ_1, τ_2 be two topologies on I. If $\tau_1 \subseteq \tau_2$ and τ_1 contains the equalizer topology induced by A, then $\Gamma(F_A, (I, \tau_1)) \subseteq \Gamma(F_A, (I, \tau_2))$.*

Note that, even if the topology on I is an S-topology, it is not always the case that A is isomorphic to the algebra of global sections $\Gamma(F_A, I)$. The following results due to Davey (cf. [Dav73]) show in which case A is isomorphic to an algebra of global sections of a sheaf with fibers $A_i = A/\theta_i$, for $i \in I$.

Definition 4.3 (Global Family, cf. [Dav73]) *A family $(c_i)_{i\in I}$ of elements of A is said to be* global *with respect to $(\theta_i)_{i\in I}$ if for every $i \in I$ there exist $a^i_1, \ldots, a^i_n, b^i_1, \ldots, b^i_n \in A$ such that:*

(i) $(a^i_j, b^i_j) \in \theta_i$ for every $j = 1, \ldots, n$,

(ii) If $(a^i_j, b^i_j) \in \theta_k$ for every $j = 1, \ldots, n$ then $(c_k, c_i) \in \theta_k$.

Theorem 4.7 ([Dav73]) *Let $(\theta_i)_{i\in I}$ be a family of congruences on an algebra A and assume that A is subdirect product of $(A/\theta_i)_{i\in I}$. Endow I with its coarsest S-topology. Then $\alpha : A \to \Gamma(I, F_A)$ is an isomorphism if and only if for every family of elements $(c_i)_{i\in I}$ global with respect to $(\theta_i)_{i\in I}$, there is a $c \in A$ with $(c, c_i) \in \theta_i$ for every $i \in I$.*

4.1.2 Sheaf Representation Theorems in Universal Algebra

Let A be an algebra and $\{f_i : A \to A_i \mid i \in I\}$ a subdirect representation of A (i.e. such that f_i is onto for every $i \in I$ and the canonical homomorphism $f : A \to \prod_{i\in I} A_i$ is injective).

The following construction (for further details see [Wer75]) leads to a representation of A by an algebra of sections over a sheaf of algebras.

(a) Define a topology τ on I such that all sets of the form $\{i \in I \mid f_i(a) = f_i(b)\}$ for $a, b \in A$ are open in I.

(b) Let $S = \coprod_{i\in I} A_i$ and $f : S \to I$ the canonical map. Endow S with a topology such that all sets $\{f_i(a) \mid i \in I\}$ with $a \in A$, are open in S.

(c) For every $a \in A$ let $[a] : I \to S$ be defined by $[a](i) = f_i(a)$.

Then:
(1) $S = (S, f, I)$ is a sheaf of algebras over I (*the standard sheaf of* A),
(2) For every $a \in A$ the map $[a] : I \to S$ is a section,
(3) The mapping $[] : A \to \Gamma S$ is an injective homomorphism (*the standard sheaf representation of* A).

A Topological Representation

In this section we present results due to Werner [Wer75], see also [Dav73] or [BS81].

Let A be the member of a discriminator variety. Let $\mathsf{Con}(A)$ be the congruence lattice of A, with greatest element $\nabla = A \times A$ and smallest element $\triangle = \{(a, a) \mid a \in A\}$.

We define a topological space called the *spectrum* of A, which has the set $Spec(A) = \{\theta \in \mathsf{Con}(A) \mid \theta \text{ maximal } \} = \{\theta \in \mathsf{Con}(A) \mid \text{ for all } \rho, \theta \subseteq \rho \Rightarrow \theta = \rho \text{ or } \rho = \nabla\}$ as underlying set and is endowed with the equalizer topology, i.e. the topology generated by the sets $E(a, b) = \{\theta \in Spec(A) \mid (a, b) \in \theta\}$ and their complements $D(a, b) = \{\theta \in Spec(A) \mid (a, b) \notin \theta\}$.

Lemma 4.8 ([Wer75]) *The sets $E(a,b)$ and $D(a,b)$ form a basis of clopen sets for the equalizer topology on $Spec(A)$ (they form a Boolean algebra).*

Lemma 4.9 ([Wer75]) *Let $X \subseteq Spec(A)$ be a set of maximal congruences. Then the following conditions are equivalent:*

(1) The family $\{\pi_\theta : A \to A/\theta \mid \theta \in X\}$ of canonical projections is a (faithful) subdirect representation of A,

(2) $\bigcap\{\theta \mid \theta \in X\} = \triangle$,

(3) $X \cup \{\nabla\}$ is a dense subset of $Spec(A)$.

Theorem 4.10 ([Wer75]) *For every algebra A in a discriminator variety, the topological space $Spec(A)$ is a Boolean space and its dual $Spec(A)^*$, the Boolean algebra of all clopen subsets of $Spec(A)$, is the set $\{E(a,b) \mid a,b \in A\} \cup \{D(a,b) \mid a,b \in A\}$.*

Definition 4.4 (Standard Sheaf Associated with an Algebra) *Let $\mathcal{V}$ be a discriminator variety, $A \in \mathcal{V}$. The standard sheaf construction yields a sheaf $S(A) = (\coprod_{\theta \in Spec(A)} A/\theta, f, Spec(A))$ over $Spec(A)$.*

Theorem 4.11 ([Wer75]) *Let $\mathcal{V}$ be a discriminator variety, $A \in \mathcal{V}$ and let $S(A)$ be the standard sheaf associated to A. The standard sheaf representation $[\ ] : A \to \Gamma S(A)$ which associates with every $a \in A$ the section $[a] : Spec(A) \to \coprod_{\theta \in Spec(A)} A/\theta$, defined for every $\theta \in Spec(A)$ by $[a](\theta) = [a]_\theta$ is an isomorphism.*

A similar sheaf representation theorem by considering for every $A \in \mathcal{V}$ the *proper spectrum* $Spec_0(A) = Spec(A) \backslash \{\nabla\}$. $Spec_0(A)$ is not a Boolean space in general (it is a Boolean space iff ∇ is an isolated point of $Spec(A)$ i.e. when ∇ is a compact congruence). For details concerning the standard sheaf construction in this case cf. [Wer75] (it is shown that in this case A has a representation as the algebra of all sections with compact support).

4.1.3 Applications: Unification in Discriminator Varieties

Let $\mathcal{V}$ be a variety of algebras (of signature Σ) and let $p(x_1, \ldots, x_n), q(x_1, \ldots, x_n)$ be two terms in $T_\Sigma(\{x_1, \ldots, x_n\})$.

Definition 4.5 ($\mathcal{V}$-unifier)

(1) A $\mathcal{V}$-unifier *of p and q is a substitution $\sigma : \{x_1, \ldots, x_n\} \to \mathsf{T}_\Sigma(U)$ defined for every $i, 1 \leq i \leq n$, by $\sigma(x_i) = t_i(u_1, \ldots, u_m)$, such that $\mathcal{V}$ satisfies $p(t_1, \ldots, t_n) = q(t_1, \ldots, t_n)$. A $\mathcal{V}$-unifier of p and q will be denoted in what follows by $x_i \leftarrow t_i(u_1, \ldots, u_m), 1 \leq i \leq n$.*

(2) *A given $\mathcal{V}$-unifier $\sigma : \{x_1,\dots,x_n\} \to \mathsf{T}_\Sigma(U)$, $x_i \leftarrow t_i(u_1,\dots,u_m), 1 \leq i \leq n$, is* more general *than another $\mathcal{V}$-unifier $\sigma' : \{x_1,\dots,x_n\} \to \mathsf{T}_\Sigma(U')$, $x_i \leftarrow t'_i(u'_1,\dots,u'_k), 1 \leq i \leq n$, if there is a substitution $\theta : \{u_1,\dots,u_m\} \to \mathsf{T}_\Sigma(U')$, $u_j \leftarrow t''_j(u'_1,\dots,u'_k), 1 \leq j \leq m$, such that $\sigma' = \theta \circ \sigma$, i.e. such that $\mathcal{V}$ satisfies*

$$t'_i(u'_1,\dots,u'_m) = t_i(t''_1(u'_1,\dots,u'_k),\dots,t''_m(u'_1,\dots,u'_k)).$$

The notion 'more general than' determines a preorder on the $\mathcal{V}$-unifiers of p and q. If every pair of $\mathcal{V}$-unifiable terms has a most general $\mathcal{V}$-unifier, which is more general than all $\mathcal{V}$-unifiers of the pair, then we say that the variety has *unitary unification.* We know that discriminator varieties have unitary unification (cf. [Bur92]).

In the following, $X = \{x_i \mid i \in \mathbb{N}\}$, $\hat{X} = \{\hat{x}_i \mid i \in \mathbb{N}\}$, and $\tilde{X} = \{\tilde{x}_i \mid i \in \mathbb{N}\}$, will be disjoint sets of variables contained in a countably infinite set $U = \{u_i \mid i \in \mathbb{N}\}$. If p is a term in variables from X, $\hat{p}$ (resp. $\tilde{p}$) will denote the term in variables from $\hat{X}$ (resp. $\tilde{X}$) obtained by replacing each x_i in p by $\hat{x}_i$ (resp. $\tilde{x}_i$).

We can alternatively regard a unifier of $p(x_1,\dots,x_n)$ and $q(x_1,\dots,x_n)$ as a morphism

$$\sigma : \mathsf{F}_\mathcal{V}(\{x_1,\dots,x_n\}) \to \mathsf{F}_\mathcal{V}(U)$$

such that $\sigma([p]) = \sigma([q])$ ($[p],[q]$ denote the equivalence classes of the terms p resp. q in $\mathsf{F}_\mathcal{V}(\{x_1,\dots,x_n\})$).

We can reformulate the relation "is more general than" on the set of unifiers of $p(x_1,\dots,x_n)$ and $q(x_1,\dots,x_n)$ as follows: Let $\sigma_1, \sigma_2 : \mathsf{F}_\mathcal{V}(\{x_1,\dots,x_n\}) \to \mathsf{F}_\mathcal{V}(U)$ be unifiers of $p(x_1,\dots,x_n)$ and $q(x_1,\dots,x_n)$. We say that σ_1 is more general than σ_2 (denoted by $\sigma_1 \leq \sigma_2$) iff there is a substitution $\tau : \mathsf{F}_\mathcal{V}(U) \to \mathsf{F}_\mathcal{V}(U)$ such that $\sigma_2 = \tau \circ \sigma_1$.

Note that the existence of a most general unifier does not depend on the names of the variables (the problem of the names can be solved by composing with an appropriate substitution that "renames" the variables).

For the sake of simplicity, in what follows we will denote the equivalence class of the term p (resp. $\hat{p}$, $\tilde{p}$) in $F_V(X)$ (resp. $F_V(\hat{X})$, $F_V(\tilde{X})$) again by p (resp. $\hat{p}$, $\tilde{p}$) instead of $[p]$ (resp. $[\hat{p}]$, $[\tilde{p}]$). From the context it will be clear when we consider the terms and when their equivalence classes.

Theorem 4.12 ([Bur92]) *Let $\mathcal{V}$ be a discriminator variety with switching term $s(x,y,u,v)$ on the simple algebras in $\mathcal{V}$. Let $p(x_1,\dots,x_n), q(x_1,\dots,x_n)$ be two terms that are unifiable in $\mathcal{V}$ and let $r_1,\dots,r_n$ be terms in variables from X such that $\mathcal{V}$ satisfies $p(r_1,\dots,r_n) = q(r_1,\dots,r_n)$. Then the substitution $x_i \leftarrow s(\hat{p},\hat{q},\hat{x}_i,\hat{r}_i), i = 1,\dots,n$, is a $\mathcal{V}$-unifier of p and q that is more general that any other $\mathcal{V}$-unifier of p and q.*

The theorem above (proved in [Bur92]) can be viewed as an extension of Löwenheim's reproductive solutions of Boolean equations. Although in [Bur92] it is pointed out that every system of equations in a discriminator variety reduces to one equation, we think that it is also of interest to obtain a generalization of Theorem 4.12 to systems of equations, exactly as in the case of Boolean equations (for Löwenheim's theorem that gives the form of reproductive solutions of systems of Boolean equations see e.g. [Rud74] p.978, Th.2.12).

Theorem 4.13 *Let $\mathcal{V}$ be a discriminator variety with switching term $s(x,y,u,v)$ on the simple algebras of $\mathcal{V}$. Let $p_1(x_1,\ldots,x_n), p_2(x_1,\ldots,x_n), q_1(x_1,\ldots,x_n)$ and $q_2(x_1,\ldots,x_n)$ be terms such that there are terms $r_1,\ldots,r_n$ in variables from X such that $r=(r_1,\ldots,r_n)$ is a solution in $\mathcal{V}$ of the system of equations*

$$\begin{cases} p_1(x) = q_1(x) \\ p_2(x) = q_2(x) \end{cases} \tag{4.2}$$

i.e. $\mathcal{V} \models p_i(r) = q_i(r)$ for $i=1,2$.

Let $\sigma_1 : F_{\mathcal{V}}(\{x_1,\ldots,x_n\}) \to F_{\mathcal{V}}(U)$, defined by $\sigma_1(x_i) = s(\hat{p}_1,\hat{q}_1,\hat{x}_i,\hat{r}_i)$, be a most general unifier for $p_1(x_1,\ldots,x_n)$ and $q_1(x_1,\ldots,x_n)$; and let σ_2 : $F_{\mathcal{V}}(\{\hat{x}_1,\ldots,\hat{x}_n\}) \to F_{\mathcal{V}}(U)$, defined by $\sigma_2(\hat{x}_i) = s(\tilde{p}_2,\tilde{q}_2,\tilde{x}_i,\tilde{r}_i)$, be a most general unifier for $\hat{p}_2$ and $\hat{q}_2$. Then the substitution

$$\sigma = \sigma_2 \circ \sigma_1 : F_{\mathcal{V}}(x_1,\ldots,x_n) \to F_{\mathcal{V}}(U)$$

is a most general solution of the system of equations, i.e. has the property that

(1) $\sigma(p_i) = \sigma(q_i)$, for $i=1,2$, and

(2) For every $\mu : F_{\mathcal{V}}(x_1,\ldots,x_n) \to F_{\mathcal{V}}(U)$ such that $\mu(p_i) = \mu(q_i)$ for $i=1,2$, there exists a substitution $\tau : F_{\mathcal{V}}(U) \to F_{\mathcal{V}}(U)$ such that $\tau \circ \sigma = \mu$.

Proof: We first show that σ is a solution of the system (4.2). It is easy to see that $\sigma(p_1) = \sigma_2(\sigma_1(p_1)) = \sigma_2(\sigma_1(q_1)) = \sigma(q_1)$. In order to show that $\sigma(p_2) = \sigma(q_2)$, note first that for every maximal congruence ρ on $F_{\mathcal{V}}(U)$, the quotient $F_{\mathcal{V}}(U)/\rho$ is a simple algebra, hence, by the hypothesis, $s(x,y,u,v)$ is a switching term on $F_{\mathcal{V}}(U)/\rho$. Therefore, for every maximal congruence ρ on $F_{\mathcal{V}}(U)$ we have:

$$[\sigma_1(x_i)]_\rho = s([\hat{p}_1]_\rho,[\hat{q}_1]_\rho,[\hat{x}_i]_\rho,[\hat{r}_i]_\rho) \\ = \begin{cases} [\hat{x}_i]_\rho & \text{if } [p_1(\hat{x}_1,\ldots,\hat{x}_n)]_\rho = [q_1(\hat{x}_1,\ldots,\hat{x}_n)]_\rho \\ [\hat{r}_i]_\rho & \text{otherwise} \end{cases}$$

Hence,

$$[\sigma_1(p_2(x_1,\ldots,x_n))]_\rho = \begin{cases} [p_2(\hat{x}_1,\ldots,\hat{x}_n)]_\rho & \text{if } [p_1(\hat{x}_1,\ldots,\hat{x}_n)]_\rho = [q_1(\hat{x}_1,\ldots,\hat{x}_n)]_\rho \\ [p_2(\hat{r}_1,\ldots,\hat{r}_n)]_\rho & \text{otherwise} \end{cases},$$

and

$$[\sigma_1(q_2(x_1,\ldots,x_n))]_\rho = \begin{cases} [q_2(\hat{x}_1,\ldots,\hat{x}_n)]_\rho & \text{if } [p_1(\hat{x}_1,\ldots,\hat{x}_n)]_\rho = [q_1(\hat{x}_1,\ldots,\hat{x}_n)]_\rho \\ [q_2(\hat{r}_1,\ldots,\hat{r}_n)]_\rho & \text{otherwise} \end{cases}.$$

Therefore, taking into account that σ_2 is an unifier of $\hat{p}_2$ and $\hat{q}_2$, and r is a solution of $\hat{p}_2 = \hat{q}_2$, it follows that for every maximal congruence ρ we have:
$[\sigma(p_2(x_1,\ldots,x_n))]_\rho = [\sigma_2(\sigma_1(p_2(x_1,\ldots,x_n)))]_\rho =$

$$= \begin{cases} [\sigma_2(p_2(\hat{x}_1, \ldots, \hat{x}_n))]_\rho & \text{if } [p_1(\hat{x}_1, \ldots, \hat{x}_n)]_\rho = [q_1(\hat{x}_1, \ldots, \hat{x}_n)]_\rho \\ [\sigma_2(p_2(\hat{r}_1, \ldots, \hat{r}_n))]_\rho & \text{otherwise} \end{cases} =$$
$$= \begin{cases} [\sigma_2(q_2(\hat{x}_1, \ldots, \hat{x}_n))]_\rho & \text{if } [p_1(\hat{x}_1, \ldots, \hat{x}_n)]_\rho = [q_1(\hat{x}_1, \ldots, \hat{x}_n)]_\rho \\ [\sigma_2(q_2(\hat{r}_1, \ldots, \hat{r}_n))]_\rho & \text{otherwise} \end{cases} =$$
$$= [\sigma(q_2(x_1, \ldots, x_n))]_\rho.$$

From the fact that in a discriminator variety the intersection of all maximal congruences is Δ (cf. Section 3.1.4) it follows that $\sigma(p_2(x_1, \ldots, x_n)) = \sigma(q_2(x_1, \ldots, x_n))$.

We now show that σ is the most general such substitution:
Let $\mu : \mathsf{F}_\mathcal{V}(\{x_1, \ldots, x_n\}) \to \mathsf{F}_\mathcal{V}(U)$ be a substitution such that $\mu(p_1) = \mu(q_1)$ and $\mu(p_2) = \mu(q_2)$. We want to show that there exist a substitution $\tau : \mathsf{F}_\mathcal{V}(U) \to \mathsf{F}_\mathcal{V}(U)$ such that $\mu = \tau \circ \sigma$.

Let $n : \mathsf{F}_\mathcal{V}(\tilde{X}) \to \mathsf{F}_\mathcal{V}(X)$ be the renaming substitution given by $n(\tilde{x}_i) = x_i$. Then for every $i = 1, \ldots, n$ we have

$$\begin{aligned} \mu(n(\sigma(x_i))) &= \mu(n(\sigma_2(\sigma_1(x_i)))) = \\ &= \mu(n(\sigma_2(s(\hat{p}_1, \hat{q}_1, \hat{x}_i, \hat{r}_i)))) = \\ &= \mu(n(s(\hat{p}_1, \hat{q}_1, \hat{x}_i, \hat{r}_i)[\hat{x}_j \leftarrow s(\tilde{p}_2, \tilde{q}_2, \tilde{x}_j, \tilde{r}_j), j = 1, \ldots, n])) = \\ &= \mu(s(\hat{p}_1, \hat{q}_1, \hat{x}_i, \hat{r}_i)[\hat{x}_j \leftarrow s(p_2, q_2, x_j, r_j), j = 1, \ldots, n]) = \\ &= s(\hat{p}_1, \hat{q}_1, \hat{x}_i, \hat{r}_i)[\hat{x}_j \leftarrow s(\mu(p_2), \mu(q_2), \mu(x_j), \mu(r_j)), j = 1, \ldots, n] = \\ &= s(\hat{p}_1, \hat{q}_1, \hat{x}_i, \hat{r}_i)[\hat{x}_j \leftarrow \mu(x_j), j = 1, \ldots, n] = \\ &= s(\mu(p_1), \mu(q_1), \mu(x_i), \mu(r_i)) = \mu(x_i). \end{aligned}$$

As $(\mu \circ n) \circ \sigma$ and μ coincide on the generators x_i of $\mathsf{F}_\mathsf{V}(\{x_1, \ldots, x_n\})$, it follows that $(\mu \circ n) \circ \sigma = \mu$. □

The application of Theorem 4.13 can be iterated, so our generalization enables us to solve systems with any number of equations.

The following theorem is a consequence of the fact that in a discriminator variety, for every primitive positive sentence ϕ, we have $\mathsf{F}_\mathcal{V}(\mathsf{n}) \models \phi$ iff $S_n \models \phi$ (where S_n is the class of ($\leq n$)-generated simple algebras in $\mathcal{V}$). This follows from the sheaf representation theorem for discriminator varieties, and from the fact that the stalks of a n-generated Boolean product are ($\leq n$)-generated.

Theorem 4.14 ([Bur92]) *Let $\mathcal{V}$ be a discriminator variety, and let S_n be the class of ($\leq n$)-generated simple algebras in $\mathcal{V}$. Assume that the language of $\mathcal{V}$ contains constants. Then $x_i \leftarrow r_i, 1 \leq i \leq n$, (where r_i are ground terms) is a $\mathcal{V}$-unifier of p and q iff $S_0 \models p(r_1, \ldots, r_n) = q(r_1, \ldots, r_n)$.*

If the language of $\mathcal{V}$ does not contain constants, then a similar result holds, with S_0 replaced by S_1. Additionally, in this case it is not required that the terms r_i are ground terms.
Remark: In practical situations the application of this method may generate very long terms.

4.1.4 Priestley Duality for Distributive Lattices

In this section we briefly present the duality theorem for distributive lattices due to Priestley [Pri70, Pri72].

Definition 4.6 (Priestley Space) *A* Priestley Space *is an ordered topological space $(X, \leq, \tau)$ with the property that*

1) (X, τ) is compact,

2) For every $x, y \in X$, if $x \not\geq y$ then there is a clopen order ideal $U \subseteq X$ such that $x \in U$ and $y \notin U$ (i.e. X is totally order disconnected).

Let D_{01} be the category of distributive lattices, having as objects the distributive lattices with 0, 1, and as morphisms the lattice morphisms, and let P be the category of Priestley spaces, having as objects compact totally order disconnected spaces and as morphisms continuous order preserving maps between these spaces.

The Priestley duality theorem for distributive lattices with 0 and 1 can be stated as follows (for details see also [DP90]).

Theorem 4.15 (Priestley) *The functors*

$$D_{01} \xrightarrow{D} P \qquad P \xrightarrow{E} D_{01}$$

defined on objects by:

$$D(A) = \mathsf{Hom}_{D_{01}}(A, \{0,1\}) \qquad E(X) = \mathsf{Hom}_P(X, \{0,1\})$$

and on morphisms by:

$$f : A_1 \to A_2 \qquad\qquad h : X_1 \to X_2$$

$$D(f) : D(A_2) \to D(A_1) \qquad E(h) : E(X_2) \to E(X_1)$$

$$D(f)(\phi) = \phi \circ f \qquad\qquad E(h)(\psi) = \psi \circ h.$$

define a dual equivalence between the category D_{01} of distributive lattices and the category P of Priestley spaces.

More precisely we have:

(1) For every lattice $L \in D_{01}$, the space $\mathsf{Hom}_{D_{01}}(A, \{0,1\})$ (with the order defined pointwise and the topology generated by the sets $X_a = \{f \mid f(a) = 1\}$ and $X \backslash X_a = \{f \mid f(a) = 0\}$ as a subbasis) is a Priestley space,

(2) For every Priestley space $X = (X, \leq, \tau)$, $\mathsf{Hom}_P(X, \{0,1\})$ is a distributive lattice,

(3) For every lattice $L \in D_{01}$, the map $\eta_A : A \to E(D(A))$ defined by $\eta_A(a) = \{f : A \to \{0,1\} \mid f \text{ is a 0,1-lattice morphism with } f(a) = 1\}$, is an isomorphism of 0,1-lattices,

(4) For every Priestley space $X \in P$, the map $\varepsilon_X : X \to D(E(X))$, defined by $\varepsilon_X(x) = \{h : X \to \{0,1\} \mid h \text{ is continuous, order-preserving, and } h(x) = 1\}$ is an isomorphism of Priestley spaces,

(5) For every morphism of lattices $f : A_1 \to A_2$, $D(f)$ is continuous and order-preserving,

(6) For every $h : X_1 \to X_2$ continuous and order-preserving, $E(h)$ is a morphism of lattices,

(7) The maps $D : D_{01}(A_1, A_2) \to P(D(A_2), D(A_1))$ and $E : P(X_1, X_2) \to D_{01}(E(X_2), E(X_1))$ are bijections and the following diagrams commute:

$$\begin{tikzcd} A_1 \arrow[r, "f"] \arrow[d, "\eta_1"'] & A_2 \arrow[d, "\eta_2"] \\ E(D(A_1)) \arrow[r, "E(D(f))"] & E(D(A_2)) \end{tikzcd} \qquad \begin{tikzcd} X_1 \arrow[r, "h"] \arrow[d, "\varepsilon_1"'] & X_2 \arrow[d, "\varepsilon_2"] \\ D(E(X_1)) \arrow[r, "D(E(h))"] & D(E(X_2)) \end{tikzcd}$$

Note that for every distributive lattice L, $\mathsf{Hom}_{D_{01}}(L, \{0,1\})$ is in bijective correspondence with the family of all prime filters of L; similarly, for every Priestley space X, $\mathsf{Hom}_P(X, \{0,1\})$ is in bijective correspondence with the set of clopen order-ideals of X. Intuitively it is sometimes better to refer to the elements of $D(L)$ as prime filters; technically, proofs are often shorter if considering the elements of $D(L)$ as $0,1$-morphisms of lattices from L to $\{0,1\}$.

We also note that if L is a Boolean algebra then every prime filter is maximal, hence the order on $D(L)$ is discrete. In this case the Priestley representation theorem reduces to the Stone representation theorem (cf. Theorem 3.4). If L is a finite distributive lattice then the topology on $D(L)$ turns out to be the discrete topology. In this case the Priestley representation theorem reduces to the representation theorem for finite distributive lattices due to Birkhoff (cf. Theorem 3.2).

In Section 5.3.1 we will discuss the possibility of extending the Priestley duality theorem for distributive lattices with additional operators.

4.1.5 Sheaf Representation and Priestley Representation seen as Fiberings

We now point out the basic ideas of the two representation theorems described above, namely the sheaf representation theorem and the Priestley representation theorem. Both these representation theorems can be seen as "decompositions" of the algebra as an indexed family of simpler algebras, such that the index space has a "good" structure. We present the main ideas in what follows. Details will be given in Section 5.1.3, where the ideas above will be particularized for the case of SHn-algebras.

Sheaf Representation Theorem

The sheaf representation theorem for discriminator varieties states that every algebra A in a discriminator variety $\mathcal{V}$ is isomorphic to the algebra of continuous functions $f : I \to \coprod_{\rho \in I} A_i$, where I a "base space" (the set of all maximal congruences of A, including ∇) and for every $\rho \in I$, $A_\rho = A/\rho$.

Hence, one can put in evidence a fibered structure defined by A, namely the "base space" I (a topological space, with no relation) and "fibers" (simple algebras in the variety $\mathcal{V}$).

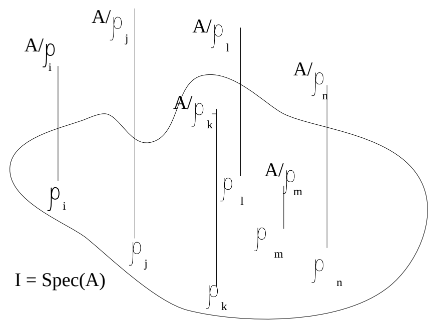


Figure 4.1: The Standard Sheaf Associated to the algebra A

Priestley Representation Theorem

The Priestley representation theorem for the variety of distributive lattices with 0 and 1 states that every distributive lattice A is isomorphic to the lattice of continuous and order-preserving functions $f : I \to \{0, 1\}$, where I a "base space" (the set of all prime filters of A).

Hence, one can put in evidence a fibered structure defined by A, namely the "base space" I (an ordered topological space) and "fibers" (all equal to the 2-element lattice).

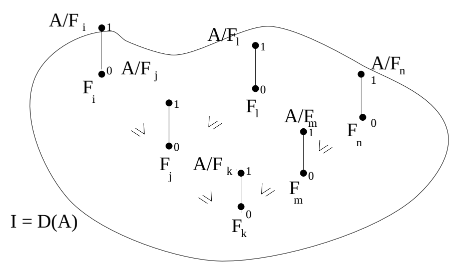


Figure 4.2: The Priestley space associated to the distributive lattice A

Note the similarity between this idea and the idea on which the notion of "Logical Fiberings" (due to Jochen Pfalzgraf) is based (cf. [Pfa91]): in both cases we can put in evidence a "base space" I and "fibers" (namely, simple algebras in $\mathcal{V}$ or, respectively, the lattices $\{0, 1\}$).

A comparison between the two representation theorems

Let $A = (A, \{\sigma_A\}_{\sigma \in \Sigma}$ be a Σ-algebra in a discriminator variety $\mathcal{V}$ of algebras with a distributive lattice underlying structure (i.e. such that $\{\vee, \wedge, 0, 1\} \subseteq \Sigma$, and $(A, \vee_A, \wedge_A, 0_A, 1_A)$ is a distributive lattice).

	Sheaf Representation Theorem	**Priestley Duality Theorem**
Index set:	maximal congruences w.r.t. the signature Σ (including also ∇)	maximal congruences w.r.t. the signature $\{0, 1, \vee, \wedge\}$
Subbasis for topology	$E(a, b)$, $D(a, b)$	$X_a = E(x, 1)$, $X \backslash X_a = E(x, 0) = D(x, 1)$
Order:	discrete	defined pointwise
Fibers:	all simple algebras in $\mathcal{V}$ and the one-point algebra	$\{0, 1\}$ (all simple distributive lattices with $0, 1$)

Additional operators on A may define additional relations on the Priestley dual of A, $D(A)$. In Section 5.1.5 we will present this for the particular case of SHn-algebras, and in Section 5.3.1 we will make some more general remarks.

Note that if A is a Boolean algebra then the sheaf representation and the Priestley representation for A coincide.

4.2 Resolution in Many-Valued Logics

We begin by pointing out the main idea the of the method for automated theorem proving in many-valued logics based on resolution due to Baaz and Fermüller [Baa92, BF92, BF95], as presented in [BF95].

Analyzing the method of resolution it turns out that it is a "two-level" approach: the first level consists in translation of arbitrary formulae to clause form, whereas the actual proving by resolution is a "logic-free" process. In the representation of clauses for classical logic no "logical symbols" appear, with the exception of negation. But even the rôle of negation is not that of a logical operator: literals containing the negation signs can be thought of as being endowed with the truth value "false", whereas literals that do not contain the negation sign can be considered endowed with the truth value "true". This suggested an extension of the procedure to arbitrary finitely-valued logics (described by the tables of the operators and of the quantifiers): literals are in this case atomic formulas endowed with truth values; and it turns out that a similar resolution procedure can be defined.

Also in this case (and in even more general cases, as for example in the case of logic systems described by their Gentzen-type calculus cf. e.g. [Min90]) automated theorem proving procedures by resolution can be given. They are essentially two-level approaches. The first level consists in the translation of arbitrary formulae of any chosen logic into clause form syntax. The clause syntax itself has to be considered as "logic-free". The translations to clause form can be described as derivations in a calculus consisting of logic-specific transformation rules.

The second level consists of the application of a logic-independent resolution principle (e.g. for many-valued clauses) that is a straightforward generalization of the classical 2-valued resolution principle.

This also allows to transfer many refinements of resolution that are essential for efficient theorem proving to the many-valued case.

We start with a slightly different notion of literal, taken, as the rest of this short introduction, from [BF95].

Definition 4.7 (Literal) *Let W denote the set of truth values (in the classical case,* true *or* false*). A* literal *is an atomic formula equipped with a truth value, denoted by L^w.*

Clauses, substitutions, and (most general) unifiers are defined just as in the classical case.

Definition 4.8 (Resolvent) *A clause C_3 is the* resolvent *of C_1 and C_2 if $C_1 = D_1 \cup \{P^u\}$, $C_2 = D_2 \cup \{Q^v\}$, for two different truth values $u \neq v \in W$, and $C_3 = \sigma(D_1 \cup D_2)$, where σ is the m.g.u. of P and Q.*

The proof that the empty clause $\Box$ is derivable from $\mathcal{C}$ if and only if $\mathcal{C}$ is unsatisfiable in its Herbrand universe (i.e., there is no interpretation satisfying $\mathcal{C}$) can be found in [BF92].

We remark here that the completeness of the calculus is preserved when applying the following reduction rules on the set of already derived clauses:

- $\mathcal{C} \cup \{D\} \Rightarrow \mathcal{C}$ if $\sigma(C) \subseteq D$ for a clause $C \in \mathcal{C}$ (the *subsumption rule*);
- $\mathcal{C} \cup \{D\} \Rightarrow \mathcal{C}$ if $\{L^w \mid w \in W\} \subseteq D$ (the *tautology rule*).

In [Häh90, Häh91, Häh93] Hähnle introduced semantic tableaux systems that can be used to implement a generic theorem prover which performs efficiently in a variety of finitely valued logics. The key idea was to enhance the formula language in such a way that it is possible to keep track of the valuations still to be considered at a certain step of the proof. The technical device was the use of truth value sets as signs or prefixes in front of the formulae. In [Häh94], Hähnle presents a general satisfiability-preserving transformation of formulae from arbitrary finitely-valued logics into a clause form based on signed atomic formulae (these clause form translations are structure-preserving), together with a suitable definition of a clause language and a *signed* resolution rule. The main idea is that the literals in the clauses are labeled with sets of truth values; in formulating a resolution principle for this case it turns out to be necessary to formulate a *merging rule* for the truth values (in addition to the standard binary resolution rule). Hähnle shows (using semantic trees) that the procedure thus defined is complete.

Then [Häh94] considers so-called *regular logics* (logics in which all sentences can be expressed in a signed clause form, where all clauses are *regular*, i.e. have all signs of the form $\boxed{\geq j} := \{j, \ldots, \top\}$ or $\boxed{\leq j} := \{\bot, \ldots, j\}$) and gives a complete resolution rule for these logics.

Regular logics have the property that the truth tables of their connectives can be characterized geometrically (for details see [Häh91]).

In [Häh96b] a many-valued version of Anderson and Bledsoe's excess literal technique is proposed, which works particularly well with regular clauses. It is shown that this method can easily be used to prove completeness of a version of semantic clash resolution for regular clauses, namely:

Many-valued negative hyperresolution

$$\frac{\left\{\boxed{\leq i_1}\ p_1\right\} \cup D_1, \ldots, \left\{\boxed{\leq i_n}\ p_n\right\} \cup D_n, \left\{\boxed{\geq j_1}\ p_1, \ldots, \boxed{\geq j_n}\ p_n\right\} \cup E}{D_1 \cup \ldots \cup D_n \cup E}$$

provided $n \geq 1, i_l < j_l$ for all $1 \leq l \leq n$, and $D_1, \ldots, D_n, E$ are negative.

The proof of the completeness of the procedure closely follows the proof given in the original paper in [AB70], and can be found in [Häh96b].

We would also like to mention [Häh96a] where a concise axiomatization of a broad class of generalized quantifiers in many-valued logics – so-called distribution quantifiers – is presented. It is shown that for certain lattice-based quantifiers relatively small axiomatizations can be obtained in a schematic way, by providing an explicit link between skolemized signed formulae and filters/ideals in Boolean set lattices. Hähnle shows that for many "naturally" defined quantifiers the inverse images of sets of truth values for these quantifiers have representations as disjunctive normal form combinations of filters and ideals.

In Section 5 we will present a method for automated theorem proving in some classes of many-valued logics based on distributive algebras with operators that behave "well" with respect to join and/or meet. For the first-order logics we only take the existential and universal quantifiers into account.

Our method is an extension of the method for many-valued signed hyperresolution for regular logics given in [Häh96b], in the sense that in the definition of regular logics [Häh96b] one of the assumptions is the existence of a *total ordering* on the set of truth values. We use the canonical lattice order in the lattice of truth values (which in general is not a total order). The difference between our approach and the above mentioned approaches consists in the fact that we use signed clauses of the form $\boxed{x}\ p^t$ or $\boxed{x}\ p^f$, where the signs are "possible worlds" (which, in the cases considered are of the form $\uparrow x$ where $x \in A$ is a join-irreducible truth value).

It turns out that under our assumptions and with our definition of literal and clause these types of labels are sufficient for formulating an automated theorem proving procedure based on negative hyperresolution.

Our method also has the advantage of clarifying possible links between the use of "sets as signs" and interpretations in Kripke models (in fact the idea of the algorithm occurred to us when studying Kripke models for SHn-logics).

4.3 Models for Cooperating Agents and Concurrency

We continue by presenting some existing approaches to modeling cooperating agents scenarios and concurrency. We begin with a succinct presentation of some classical approaches to concurrency, and then continue with approaches based on notions as fiberings, fiber bundles, presheaves and sheaves.

4.3.1 Classical Approaches to Concurrency

We will present first some selected classical approaches to concurrency, namely transition systems, Petri nets, and trace languages. In the end we will briefly present a new approach due to Pratt [Pra94].

Transition Systems

Transition systems are a commonly used model of computation. They provide the basic operational semantics for Milner's Calculus of Communicating Systems [Mil80, Mil89]. In this theory, processes are defined by algebraic equations and evolve into other processes by performing some actions. The theory is described in detail in the above references. We give here only the basic definitions and show in which way the constructions of complex transition systems from simpler ones can be seen as universal in a category of transition systems. For details we refer to [Mil80, Mil89, WN93].

Transition systems are models in which the notion of a *state* of the system is taken as primitive. The behavior of the system is described by a set of *actions*; the notions of causality and consequence are captured in a transition relation.

Definition 4.9 (Transition Systems) *A* transition system *is a tuple* $T = (S, i, L, \mathsf{Tran})$*, where*

(1) S *is a set of states,*
(2) $i \in S$ *is the initial state,*
(3) L *is a set of labels,*
(4) $\mathsf{Tran} \subseteq S \times L \times S$ *(the transition relation).*

Notation Let $T = (S, i, L, \mathsf{Tran})$ be a transition system. We write $s \xrightarrow{a} s'$ to indicate that $(s, a, s') \in \mathsf{Tran}$.

In what follows, given a transition system T_i, if not specified otherwise we will denote by S_i, i_i, L_i, Tran_i its set of states, initial state, set of labels and transitions respectlvely.

A transition system models a process whose transitions represent the process's atomic actions while the labels are action names.

For technical reasons it is convenient to introduce *idle transitions*, associated to any state.

Definition 4.10 (Idle Transitions) *Let* $T = (S, i, L, \mathsf{Tran})$ *be a transition system. Let* $*$ *be a distinguished symbol that does not belong to* L*. An* idle transition *of* T *consists of* $(s, *, s)$*, where* $s \in S$*. We define* $\mathsf{Tran}_* := \mathsf{Tran} \cup \{(s, *, s) | s \in S\}$ *and* $T_* = (S, i, L \cup \{*\}, \mathsf{Tran}_*)$*.*

Remark The distinguished symbol $*$ corresponds to the *empty action*. The idle transitions model the fact that after an empty action the state remains unchanged.

Idle transitions help to give a simpler definition for the morphisms between transition systems. Basically, morphisms between transition systems can be understood as expressing the partial simulation (or refinement) of one process by another one.

Definition 4.11 (Morphisms) *Let* T_1 *and* T_2 *be two transition systems. A* morphism *from* T_1 *to* T_2*,* $f : T_1 \longrightarrow T_2$*, is a pair* $f = (\sigma, \lambda)$*, where* $\sigma : S_1 \longrightarrow S_2$*,* $\lambda : L_1 \longrightarrow L_2$*,* $\sigma(i_1) = i_2$ *and* $(s, a, s') \in \mathsf{Tran}_1$ *implies* $(\sigma(s), \lambda(a), \sigma(s')) \in \mathsf{Tran}_2$*.*

Proposition 4.16 (The Category of Transition Systems) *Transition systems and their morphisms form a category* TS *in which the composition of two morphisms* $f_1 = (\sigma_1, \lambda_1) : T_0 \longrightarrow T_1$ *and* $f_2 = (\sigma_2, \lambda_2) : T_1 \longrightarrow T_2$ *is defined by* $f_2 \circ f_1 = (\sigma_2 \circ \sigma_1, \lambda_2 \circ \lambda_1) : T_0 \longrightarrow T_2$*, and the identity morphism for a transition system* $T = (S, i, L, \mathsf{Tran})$ *has the form* $(1_S, 1_L)$*, where* 1_S *is the identity function on states* S *and* 1_L *is the identity function on the labeling set* L *of* T*.*

Constructions of more complex transition systems starting from simpler ones can be seem as universal constructions in a category of transition systems (in this case, the universal properties will characterize the constructions up to isomorphism).

It is easy to show that all finite products exist in the category of transition systems. In fact the category TS has all products).

The notion of product models the behavior of a family of agents that are acting independently, in parallel. In order to model situations in which a process can behave like one of several alternative processes, we can use co-products (sums) or fibered sums in the category of transition systems.

Asynchronous Transition Systems

The idea on which asynchronous transition systems are based is simple enough: the transition systems are extended by specifying which transitions are mutually independent.

Definition 4.12 *An* asynchronous transition system *consists of* $(S, i, E, I, \mathsf{Tran})$ *where* (S, i, E, Tran) *is a transition system,* $I \subseteq E \times E$*, the* independence relation *is a irreflexive, symmetric relation on the set* E *of events such that:*

(1) If $e \in E$ *then* $\exists s, s' \in S$ *with* $(s, e, s') \in \mathsf{Tran}$*,*

(2) If $(s, e, s') \in \mathsf{Tran}$ *and* $(s, e, s'') \in \mathsf{Tran}$ *then* $s' = s''$*,*

(3) If $e_1 \; I \; e_2$ *and* $(s, e_1, s_1), (s, e_2, s_2) \in \mathsf{Tran}$ *then for some* u*,* $(s_1, e_2, u), (s_2, e_1, u) \in \mathsf{Tran}$*,*

(4) If $e_1 \; I \; e_2$ *and* $((s, e_1, s_1), (s_1, e_2, u) \in \mathsf{Tran}$ *then for some* s_2*,* $(s, e_2, s_2), (s_2, e_1, u) \in \mathsf{Tran}$*.*

Definition 4.13 (Morphisms) *Let* T_1, T_2 *be two asynchronous transition systems,* $T_1 = (S_1, i_1, E_1, I_1, \mathsf{Tran}_1)$ *and* $T_2 = (S_2, i_2, E_2, I_2, \mathsf{Tran}_2)$*. A* morphism *from* T_1 *to* T_2 *is a morphism of transition systems* $(\sigma, \eta) : (S_1, i_1, E_1, \mathsf{Tran}_1) \to (S_2, i_2, E_2, \mathsf{Tran}_2)$ *such that if* $e \; I_1 \; e'$ *and* $\eta(e), \eta(e')$ *are both defined then* $\eta(e) \; I_2 \; \eta(e')$*.*

The category A of asynchronous transition systems has as objects the asynchronous transition systems and as morphisms, morphisms of asynchronous transition systems.

The category A of asynchronous transition systems has categorical constructions as products and coproducts that essentially generalize those of transition systems.

Petri Nets

Petri nets are one of the oldest models for concurrent processes [Pet62a, Pet62b]. They are a powerful tool for modeling asynchronous parallel processes and are frequently used in modeling cooperating agents scenarios. We present here one of the numerous variants in which Petri nets appear in the literature. We only give the definitions and the basic properties, and show that constructions of Petri nets can be seen as universal in a category of Petri nets. For further details we refer to e.g. [Rei85, MOM91].

Petri nets are models in which the behavior of a system is described by a set of *events*; the notions of causality and consequence are described by precondition and postcondition maps.

Definition 4.14 (Petri Net) *A* Petri net *is a tuple* $N = (B, M, E, pre, post)$*, where*

(1) B *is a set of* conditions*,*

(2) M *is a nonempty subset of* B *(the initial marking),*

(3) E *is a set of* events*,*

(4) $pre : E \longrightarrow \mathcal{P}(B)$ *is the* precondition map *such that for all* $e \in E$*,* $pre(e)$ *is nonempty,*

(5) *post* $: E \longrightarrow \mathcal{P}(B)$ *is the* postcondition map *such that for all* $e \in E$, *post*(e) *is nonempty.*

The initial marking consists of a subset of conditions that are imagined to hold initially. A *marking*, i.e. a subset of conditions, formalizes a notion of *global state*, by specifying the conditions that hold. Markings can change when events occur.

As in the case of transition systems, it is, for technical reasons, often useful to extend events by an *idling event*. As in the case of transition systems, an idling event has no pre- and post-conditions and does not change the current state.

Definition 4.15 (Idling Event) *Let* $N = (B, M, E, pre, post)$ *be a Petri net. Let* $*$ *be a distinguished symbol that does not belong to* E*, which will be called the* idling event. *Define* $E_* = E \cup \{*\}$. *We extend the pre- and post-condition maps to* $*$ *by taking* $pre(*) = post(*) = \emptyset$. *We define* $N_* = (B, M, E_*, pre, post)$.

Definition 4.16 (Transition) *Let* $N = (B, M, E, pre, post)$ *be a Petri net. For* $M_1, M_2 \subseteq B$ *and* $e \in E_*$, *define*

$$M_1 \xrightarrow{e} M_2 \quad \text{if and only if} \quad \begin{array}{l} pre(e) \subseteq M_1, \\ post(e) \subseteq M_2 \text{ and} \\ M_1 \setminus pre(e) = M_2 \setminus post(e). \end{array}$$

Definition 4.17 (Independence of Events) *Let* $N = (B, M, E, pre, post)$ *be a Petri net. Two events* $e_1, e_2 \in E_*$ *are* independent *if*

$$[pre(e_1) \cup post(e_1)] \cap [pre(e_2) \cup post(e_2)] = \emptyset.$$

In what follows, given a Petri net N_i, we will denote by B_i, M_i, E_i, pre_i and $post_i$ its set of conditions, initial marking, pre- and post-condition maps respectively.

Definition 4.18 (Morphisms of Petri Nets) *Let* N_1 *and* N_2 *be two Petri nets. A* morphism *from* N_1 *to* N_2, $g = (\beta, \eta) : N_1 \longrightarrow N_2$, *consists of a relation* $\beta \subseteq B_1 \times B_2$ *and a partial function* $\eta : E_1 \longrightarrow E_2$, *such that the inverse relation* β^{-1} *is a partial function from* B_2 *to* B_1, *and*

$$\beta M_1 = M_2,$$

$$\beta(pre(e)) = pre(\eta(e)), \text{ and}$$

$$\beta(post(e)) = post(\eta(e)).$$

Proposition 4.17 (The Category of Petri Nets) *Petri nets and their morphisms form a category* PN *in which the composition of two morphisms* $(\beta_1, \eta_1) : N_0 \longrightarrow N_1$, *and* $(\beta_2, \eta_2) : N_1 \longrightarrow N_2$ *is defined by* $(\beta_1, \eta_1) \circ (\beta_2, \eta_2) = (\beta_2 \circ \beta_1, \eta_2 \circ \eta_1) : N_0 \longrightarrow N_2$ *and the identity morphism for a Petri net* $N = (B, M, E, pre, post)$ *is* $(1_B, 1_E)$, *where* 1_B *is the identity relation on conditions and* 1_E *is the identity function on events.*

As for transition systems, one can define (in categorical terms) several constructions on Petri nets, such as product (which corresponds to a "synchronization operation" on independent nets) and coproduct.

Event Structures

Event structures were developed as an attempt to link Petri net theory and domain theory. In a Petri net a state is given by a marking, but the same marking can be reached after several different sets of transitions. Thus, the information theoretic content of a marking is rather obscure. Event structures remedy this by making the state of a system be exactly the set of actions that have occurred so far.

In a Petri net multiple occurrences of the same action can occur, so if a state just recorded which actions had occurred and some actions occurred several times, this information will be forgotten. Therefore, the concept of *event* or occurrence of an action was introduced. An event is an action that occurs at most once in an execution.

Definition 4.19 ([WN93]) *An event structure is a structure* $(E, \leq, \sharp)$ *consisting of a set* E *of* events *which are partially ordered by* $\leq$*, the* causal dependency relation*, and a binary symmetric irreflexive relation* $\sharp \subseteq E \times E$*, the* conflict relation*, which satisfy:*

$$\{e' \mid e' \leq e\} \text{ is finite },$$

$$\text{If } e \sharp e' \text{ and } e' \leq e'' \text{ then } e \sharp e'',$$

for all $e, e', e'' \in E$.

Two events $e, e' \in E$ are concurrent (e co e') iff $\neg(e \leq e' \vee e' \leq e \vee e \sharp e')$.

As explained before, we can define a notion of *state* of an event structure $(E, \leq, \sharp)$.

Definition 4.20 ([WN93]) *Let* $(E, \leq, \sharp)$ *be an event structure. Define its configuration* $\mathcal{D}(E, \leq, \sharp)$ *to consist of those subsets* $x \subseteq E$ *which are:*

(1)[Conflict free] $\forall e, e' \in x,\ \neg(e \sharp e')$,

(2)[Downwards-closed] $\forall e, e' \in E$, *if* $e \leq e'$ *and* $e' \in x$ *then* $e \in x$.

Definition 4.21 (Morphisms of Event Structures) *Let* $ES_1 = (E_1, \leq_1, \sharp_1)$ *and* $ES_2 = (E_2, \leq_2, \sharp_2)$ *be event structures. A* morphism *from* ES_1 *to* ES_2 *is a partial function* $\eta : E_1 \to E_2$ *that satisfies*

$$\begin{aligned} \text{If } x \in \mathcal{D}(ES_1) \text{ then } & (1)\ \eta x \in \mathcal{D}(ES_2) \\ & (2)\ \forall e, e' \in x, \text{ if } \eta(e), \eta(e') \text{ are both defined} \\ & \quad \text{and } \eta(e) = \eta(e') \text{ then } e = e'. \end{aligned}$$

A morphism $\eta : ES_1 \to ES_2$ of event structures expresses how behavior in ES_1 determines behavior in ES_2. It can be shown (see e.g. [WN93]) that morphisms of event structures preserve the concurrency relation.

The category ES of event structures has as objects event structures and as morphisms morphisms of event structures. This category has products and coproducts, useful in modeling parallel composition and nondeterministic sums.

Trace Languages

In this section we introduce the basic notions of trace theory and we state some elementary results that will be useful in the next sections. We will mainly follow [Die90].

Traces or languages have been a popular way of representing behavior of processes. The behavior of a process is characterized entirely by the set of its observations or traces. This approach has been very succesful in studying sequential behavior, where the behavior of an automaton is identified with the set of strings it accepts. It has been extended to concurrent processes by regarding the parallel execution of two processes as the "shuffle" of their languages. Traces can be combined with one another using the various operations on strings, giving a nice algebraic theory of processes.

On a certain level of abstraction we may say that the setting of a concurrent system is given by a set of atomic actions X, together with a specification of which actions can be performed independently or concurrently. Such a specification is given by an independence relation $I \subseteq X \times X$. For technical reasons we will assume that I is irreflexive and symmetric (i.e. no action can act concurrently to itself and independence is mutual).

Actions which are not independent are called dependent. A concurrent process in this abstraction is a labelled graph where the labels of nodes are actions and edges represent an ordering (in time) between dependent actions. No edges are drawn between independent actions.

After giving this general intuitive presentation of traces, we start with the basic definitions.

Definition 4.22 (Dependence Alphabet) *A* dependence alphabet *is a pair (X, D), where X is an alphabet and $D \subseteq X \times X$ is a reflexive and symmetric relation, the dependence relation. The complement I of D is irreflexive and symmetric; it is called the* independence relation*.*

Let $=_c$ be the equivalence relation on X^* generated by all the pairs of the form $(uabv, ubav)$, with $u, v \in X^*, (a, b) \in I$. It is easy to see that $=_c$ is a congruence on X^*.

The quotient monoid $X^*/=_c$ is denoted by $M(X, D)$. This monoid has the property that for every $a, b \in X$, if $(a, b) \in I$ then $[a]_{=_c} \cdot [b]_{=_c} = [b]_{=_c} \cdot [a]_{=_c}$.

These types of monoids were first studied by P. Cartier and D. Foata [CF69]. They were introduced in computer science in connection with the analysis of safe net systems by A. Mazurkiewicz [Maz77].

Definition 4.23 *The monoid $M(X, D)$ is* the free partially commutative monoid *generated by the dependence alphabet (X, D).*

The previous definition is motivated by the following universality property satisfied by $M(X, D)$.

Theorem 4.18 (Freeness, cf. [Die90]) *For any monoid M and any mapping $f : X \to M$ such that for every $(a, b) \in I$, $f(a)f(b) = f(b)f(a)$ there exists a unique morphism of monoids $\overline{f} : M(X, D) \to M$ which extends f.*

Definition 4.24 *The category* FPCM *of free partially commutative monoids is the full subcategory of the category* Mon *of monoids, which has the free partially commutative monoids as its objects.*

Traces are equivalence classes of words; in order to avoid ambiguity it is useful to have normal forms. The next theorem, due to Foata (cf. [Die90], [CF69]) defines a normal form for traces.

Let $\mathcal{F}$ be the set of finite non-empty subsets of pairwise independent letters, $\mathcal{F} = \{F \subseteq X \mid F$ finite, non-empty and $\forall a, b \in F$, if $a \neq b$ then $(a, b) \in I\}$.

Each element F of $\mathcal{F}$ is called an *elementary step*. Every elementary step F yields a trace $[F] \in M(X, D)$, where $[F] = \prod_{a \in F} a$ (since all letters in F are pairwise independent, the product is well defined: the order in which the product is computed is not important).

Theorem 4.19 (Foata Normal Form, [CF69]) *Let $t \in M(X, D)$ be a trace. There exists a unique sequence of elementary steps $F_1, \ldots, F_r$, $r \geq 0$, $F_i \in \mathcal{F}$ for all $i = 1, \ldots, r$ such that $t = [F_1] \ldots [F_r]$ and for all $b \in F_i, 2 \leq i \leq r$ there is some $a \in F_{i-1}$ with $(a, b) \in D$.*

Note that every dependence alphabet can be seen as an undirected graph, having the set X as set of vertices and an edge between any pair of different dependent vertices (so the set of edges is $E = D \backslash \{(x, x) \mid x \in X\}$). Conversely, given any undirected graph $G = (X, E)$, G corresponds to the dependence alphabet (X, D) where $D = E \cup \{(x, x) \mid x \in X\}$. The monoid $M(X, D)$ will be then the freely partially commutative monoid associated to the graph G, and will be denoted $M(G)$.

Definition 4.25 (Graph Morphism) *A* graph morphism $h : (X_1, E_1) \to (X_2, E_2)$ *is an application* $h : X_1 \to X_2$ *such that:*

(1) For every $x \in X_2$, $h^{-1}(x)$ is finite,

(2) For every $(x, y) \in E_1$, $(h(x), h(y)) \in E_2$.

Definition 4.26 (The Category of Graphs) *The category* Grph *of undirected graphs has as objects undirected graphs and as morphisms graph morphisms.*

Proposition 4.20 (cf. [Die90]) *Let M be the mapping that associates*

(1) with every graph $G = (X, E)$ the free partially commutative monoid $M(G)$,

(2) with every morphism of graphs $h : G_1 \to G_2$, the unique monoid morphisms $M(h) : M(G_2) \to M(G_1)$, with the property that for every $y \in X_2$, $M(h)(y) = \prod_{x \in h^{-1}(y)} x$.

Then M defines a contravariant functor from Grph *to* FPCM.

Theorem 4.21 ([Fis86]) *Let $h : G_1 \to G_2$ be a morphism of undirected graphs and $M(h) : M(G_2) \to M(G_1)$ be the associated homomorphism of free partially commutative monoids. Then it holds:*

(1) *The homomorphism $M(h)$ is surjective if and only if h is injective,*

(2) *The homomorphism $M(h)$ is injective if and only is h is surjective on vertices and edges.*

Corollary 4.22 (General Embedding Theorem, cf. [Die90]) *Let G be an undirected graph and let $\{G_j \mid j \in J\}$ be a finite family of subgraphs. For $j \in J$ let $\pi_j : M(G) \to M(G_j)$ denote the canonical projection and let $\pi : M(G) \to \prod_{j\in J} M(G_j)$ be the homomorphism into the direct product given by $\pi(t) = (\pi_j(t))_{j\in J}$. Then the mapping π is injective if and only if $G = \bigcup_{j\in J} G_j$.*

Note that if $\{M_j \mid j \in J\}$ is *any* family of non-trivial free partially commutative monoids then $\prod_{j\in J} M_j$ is free partially commutative if and only if J is finite. The direct product $\prod_{j\in J} M(G_j)$ is, in general, not a free partially commutative monoid .

If the family $\{G_j \mid j \in J\}$ is not finite, then – under the assumption that for every vertex x of G there are at most finitely many $j \in J$ such that x is a vertex of G_j – it follows that there is an injective morphism $M(G) \hookrightarrow \bigoplus_{j\in J} M(G_j)$, where $\bigoplus_{j\in J} M(G_j) = \{(m_j)_{j\in J} \mid m_j \in M(G_j) \text{ for all } j \in J, m_j = \varepsilon \text{ a.e.}^2\}$ is the so-called *weak product* of the family $\{M(G_j)\}_{j\in J}$ (see e.g. [Die90]).

Definition 4.27 ([Die90]) *Let $(X_1, D_1), \ldots, (X_k, D_k)$ be dependence alphabets and let $M_i = M(X_i, D_i)$ be the corresponding free partially commutative monoids, $i = 1, \ldots, k$. Then the synchronization of $M_1, \ldots, M_k$ is defined by*

$$M_1 || \ldots || M_k = M(\bigcup_{i=1}^{k} X_i, \bigcup_{i=1}^{k} D_i).$$

Let $L_i \subseteq M_i, i = 1, \ldots, k$. Then the synchronization of $L_1, \ldots, L_k$ is defined by

$$L_1 || \ldots || L_k = \{t \in M_1 || \ldots || M_k \mid p_i(t) \in L_i \textit{ for } i = 1, \ldots, k\},$$

where $p_i : M_1 || \ldots || M_k \to M_i$ denotes the canonical projection, $i = 1, \ldots, k$.

Theorem 4.23 ([Die90], [MP86], [CM85]) *Let $\{G_j \mid j \in J\}$ be a finite family of dependence graphs and $G = \bigcup_{j\in J} G_j$. For every $j \in J$ let $M_j = M(G_j)$, $M = M(G)$, and let $p_j : M \to M_j$, $p^i_{ij} : M_i \to M(G_i \cap G_j)$ be the canonical projections.*

The following assertions are equivalent:

(1) *The canonical embedding $\pi : M \to \{(m_j)_{|j\in J} \mid p^i_{ij}(m_i) = p^j_{ij} \text{ for all } i, j \in J\}$ is an isomorphism.*

(2) *Every chordless cycle in the graph G is a cycle in a subgraph G_j for some $j \in J$.*

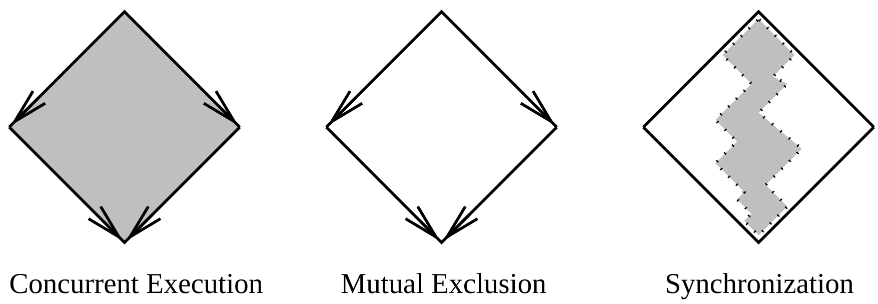
[2] a.e. means *almost everywhere*

Higher Dimensional Automata

Higher dimensional automata are a generalization of automata proposed by Pratt [Pra91] to model non-interleaving concurrency. Standard finite automata are drawn as points representing states and directed arcs representing transitions, hence all the elements are 0 and 1-dimensional objects. Pratt generalizes this to allow elements of any finite dimension, where an n-dimensional object stands for a transition representing the concurrent occurrence of n actions.

Computation can be viewed as a path through such an automaton. Concurrent execution of a and b, $a||b$, is represented as a square whose surface is "filled in", and mutual exclusion $ab \sqcup ba$ as a square whose interior is "empty" (so one has to follow the edges, doing one of a, b at one time).

Communication can be modeled abstractly as "eroding" some of the interior surface. When two processes synchronize, they must both be at some fixed stage in their execution simultaneously, i.e. their execution trajectory must pass through a point. Monotonicity of computation is modeled by the fact that certain parts of the square are illegal. Asynchronous communication is modeled as eroding the area where the message was received before transmission. More communication erodes more area, in the extreme case leading to a single path of execution.



It is possible to generalize the concept of a computation from a path to a set of paths (as a first approximation for a path one can take take a homotopy class, i.e. a set of paths where each path can be deformed into another without jumping over holes). For a geometric approach that uses tools from algebraic geometry and homotopy theory we refer to [Gou95].

Higher dimensional automata are rather difficult to specify because the specifications are quite long. However they are able to control information in a better way than other models, allowing forgetting of useless information.

Partially Ordered Multisets

Partially ordered multisets are the simplest non-interleaving model of processes: instead of ordering events linearly as in a trace, they are ordered partially. Partially ordered multisets are posets with a labeling function which labels every event with an action. Thus partially ordered multisets generalize strings, which are labeled traces.

Partially ordered multisets as a model for concurrency have been studied in [Pra82, Pra86, Gis88]. A process is modeled as a set of partially ordered multisets, and a run executes one of the partially ordered multisets of this set.

Formulae from first order or temporal logic can be used to specify partially ordered multisets, and the algebraic and logical specifications may be mixed.

Partially ordered multisets have been generalized to metric process models [Cre91] and measured sets [Cas91].

Geometric Automata

We give here a very brief presentation of geometric automata, taken from [Gup94]. Geometric automata were introduced by Gunawardena in 1991 as a generalization of event structures. They are based on a syntactic approach to causality. A geometric automaton consists of a set of events each of which is associated with a condition, a boolean formula on the events. Executing an event means changing its value from 0 to 1, and events are executed one at a time when their conditions are satisfied.

This approach is very declarative in its essence: for every event we have to state under which conditions it can happen.

Geometric automata are an interleaving model of concurrency, since one event is executed at a time.

Chu Spaces

A new model of concurrency are the Chu spaces [Pra94, Gup94]. The idea behind the definition of Chu spaces is very simple, namely that the central notions for all (computer) systems are the notion of *state* (a snapshot of a system at any time) and the fact that systems can move from one state to another doing certain *actions*. So a system is a pair of sets: the set X of states it can be in and the set A of events (occurrences of transitions) that can happen during its execution. The state of a system carries the history of a system, namely the set of events that have occurred so far.

Definition 4.28 *A* Chu space *is a binary relation between two sets A and X. We write it as a triple (A, X, R) where $R : A \times X \to \{0,1\}$ gives the binary relation as a characteristic function of a subset of $A \times X$.*

We can think of A as the set of events and X as the set of states of the process represented by the Chu space. Then $R(a, x)$ tells us whether event a has occurred in state x.

Chu spaces first arose as an instance of a general construction, called Chu's construction that originated from the study of so-called "(*)-autonomous categories" by Barr and Chu in 1979.

Definition 4.29 (Chu Maps) *Let (A, X, R) and (B, Y, S) be two Chu spaces. A Chu map between them consists of a pair of functions $f : A \to B$ and $g : Y \to X$ such that for any $a \in A$ and any $y \in Y$, $S(f(a), y) = R(a, g(y))$. This condition is called the adjointness condition for Chu maps, and f, g will be called the left and right adjoint respectively.*

Chu spaces and Chu maps form a category denoted Chu. Some of the mathematical structure of this category is further explored in [Pra93], where Pratt shows how to embed various categories fully and faithfully in Chu.

4.3.2 Links Between These Models

In [WN93] the links between event structures, trace languages, Petri nets and asynchronous transition systems are studied. In [Gup94] the links between Chu spaces and other models for concurrency are considered. In what follows we only state the main ideas. For details we refer to [WN93] and [Gup94] respectively.

$$\begin{tikzcd} \mathsf{ES} & & \mathsf{TL} \arrow[ll, "\Phi_1\,\mathsf{corefl.}"'] \\ & & \\ & & \mathsf{A} \arrow[uu, "\Phi_2\,\mathsf{corefl.}"'] \arrow[uull, "\Phi\,\mathsf{corefl.}"] \arrow[r, shift left] & \mathsf{PN} \arrow[l, shift left, "\mathsf{adjunction}"] \end{tikzcd}$$

To every trace language one can associate an event structure in a canonical way . This representation theorem extends to a coreflection between the categories of event structures and trace languages. (A coreflection is an adjunction in which the unit is a natural isomorphism.) The coreflection expresses the sense in which the model of event structures is “embedded” in the model of trace languages.

The existence of this coreflection also makes it possible to construct explicitly the product on event structures, which is not easy to define directly. For more details see [WN93].

An asynchronous transition system determines a trace language in a canonical way. This mapping extends to a coreflection between the category A of asynchronous transition systems and the category TL of trace languages.

This coreflection does not extend to an adjunction from TL to A. For more details see [WN93].

A coreflection between event structures and asynchronous transition systems follows by composing the coreflections between event structures and trace languages and that between trace languages and asynchronous transition systems.

There is an adjunction between the categories A of asynchronous transition systems and the category PN of Petri nets. For details see for example [WN93].

In [Gup94] Chu spaces and other models of concurrency are compared. It is shown that Chu spaces can mimic the behavior of event structures, i.e. that to every event structure $(E, \leq, \sharp)$ we can associate a Chu space. Further results on the link between event structures and Chu spaces can be found in [Plo93].

Since Chu spaces embed event structures, they can represent any behavior that safe Petri nets can model.

Geometric automata cannot be encoded by Chu spaces. The reason is that in a geometric automata, the enabling formula only needs to be true at the instant an event is taking place, and it can become false later.

4.3.3 Approaches Based on Multi-Modal Logic

The main idea of these approaches is to associate with each action Ac executed by an agent Ag a modal connective $[Ag, Ac]$ with the formula

$$[Ag, Ac]\alpha$$

being read as:

> "If the agent Ag executes the action Ac and the action ends, then α is true in the resulting state."

In this way one obtains a multi-modal language, with a set of modal operators indexed by the set of actions and the set of agents. The logic can be also extended with the (deontic) predicates *Per*, for permission, *Obl* for obligation and *Res*, to restrict the execution of actions. For an extensive study of this logic we refer to [Cos90]. A semantics based on the Kripke's possible world semantics was provided in [Cos90].

Intuitively, a Kripke model $(W, (R_{[Ag,Ac]})_{Ag \in Agents, Ac \in Actions}, V)$ consists of a set W of possible worlds (one can think of them as being states), a family $(R_{[Ag,Ac]})_{Ag \in Agents, Ac \in Actions}$ of relations on W, and a valuation V defined on the cartesian product between the set of propositional formulas (considered pre-defined) and the set of possible worlds, such that for every formula ϕ, $V(\phi, x) \in \{T, F\}$ is the truth value of formula ϕ at state x. For details we refer to [Cos90].

We would also like to mention *dynamic logic* (also known as *the logic of programs*) The language of regular first-order dynamic logic was introduced by Pratt in 1976. The name "dynamic logic" was given to the language by Harel in 1977. For an introduction to dynamic logic we refer to [Har84].

4.3.4 Fibered Models

Logical Fiberings

The approach to modeling robotics scenarios presented in this section originates from [Pfa91], and was worked out on a small example in [PS92]. For a brief discussion, see also [Pfa93]. We will introduce some notations in order to be able to describe the main idea of the logical fiberings approach. The definitions given here are taken with minor differences from [PS95]. All notations used are in correspondence with [Pfa91, PS92].

Let $(A_i)_{i \in I}$ be an indexed system of sets (with a given indexing set I).

An *abstract fibering* is a triple $\xi = (A, \pi, I)$, where A is the disjoint union of the A_i, denoted by $\coprod_{i \in I} A_i$, and $\pi : A \longrightarrow I$ is the canonical projection from A

to I defined as $\pi(a) = i$ for all $a \in A_i$. The index set I is called the *base space*, A the *total space*, π the *display map*, and for every i, we call $A_i = \pi^{-1}(i)$ the *fiber over* i.

The sets A and I can be endowed for example with a topology, with relations, or with an algebraic structure. The map π should respect the corresponding structure of the spaces, for example it has to be a continuous map if we work in the category of topological spaces. The above definition is the most general notion of a fiber bundle (or abstract fibering).

Let $\xi = (A, \pi, I)$ be an abstract fibering. A *global section* $s : I \to A = \coprod_{i \in I} A_i$ is a map such that $\pi \circ s = \mathrm{id}_I$. (This entails $s(i) \in \pi^{-1}(i)$ for all i in I). Let $\xi = (A, \pi, I)$ be an abstract fibering and let $U \subset I$ be a subset of I. A map $s_U : U \to A$ is called a *local section* (with respect to U) if $\pi \circ s_U = \mathrm{id}_U$.

A *morphism* between two fiberings (A, π_1, I) and (B, π_2, I) is a map $f : A \longrightarrow B$ such that the following diagram is commutative:

$$\begin{tikzcd} & I & \\ A \arrow[ur, "\pi_1"] \arrow[rr, "f"] & & B \arrow[ul, "\pi_2"'] \end{tikzcd}$$

i.e. for every $a \in A$: if $a \in A_i$ then $f(a) \in B_i$.

Definition 4.30 (Logical Fiberings) A *logical fibering* is a tuple $\xi = (E, \pi, I, L)$, where E (the total space) and I (the indexing set, also called the base space) are arbitrary sets, and L (denoting the *typical fiber* modeling every fiber $\pi^{-1}(i)$, for all $i \in I$), is taken to be a classical first order logical space.

The simplest form of a fibering is the so-called *trivial fibering*, $\xi = (E, \pi, I, F)$, where $E = I \times F$, π is the first projection, and the fiber over $i \in I$ is $\pi^{-1}(i) = \{i\} \times F$.

For logical fiberings, this corresponds to a *parallel system of logics* L_i over an index set I (serving as base space) for which the typical fiber F is a classical first order logic L. Within each fiber $L_i = \pi^{-1}(i)$, the reasoning processes can run independently and in parallel. Also communication between the fibers can be modeled.

A characteristic feature of a classical fiber bundle is the so-called local triviality property. A locally trivial fiber bundle is composed of parts that locally have a simple structure, in the sense that they are of the type of a product bundle $U_i \times F \to U_i$. Here, the U_i, subsets of the base space I, form a covering of I. The "constraints" arising from forming the entire bundle are modeled by so-called transition functions. They formally describe how the local parts are patched together in all those cases where the covering sets have a non-empty intersection. Each particular fiber $\pi^{-1}(i)$ obtains its structure from the "typical fiber" F. We now give the formal definition of the concept of local triviality.

Definition 4.31 (Locally Trivial Fiberings) A fibering $\xi = (E, \pi, I, F)$ is called *locally trivial* with respect to a covering $\{U_j\}_{j \in J}$ of the base space I, if the following diagram is commutative

$$\begin{tikzcd}
\pi^{-1}(U_j) \arrow[rr, "\Phi_j"] \arrow[dr, "\pi"'] & & U_j \times F \arrow[dl, "p_1"] \\
& U_j &
\end{tikzcd}$$

Φ_j is an isomorphism in the corresponding category, where

$$\Phi_j = (\pi, \phi_j), \qquad \phi_j : \pi^{-1}(U_j) \to F \qquad \text{(a morphism).}$$

For $i \in U_j$, $\phi_{j,i} : \pi^{-1}(i) \overset{\cong}{\to} F$ is the fiber isomorphism induced by Φ_j (this gives π^{-1} its fiber structure).

A product bundle (trivial logical fibering) is given by $E = I \times L$ and $\pi : E \to I$, the projection to the first component. Thus, the fiber over i is $\pi^{-1}(i) = \{i\} \times L =: L_i$, for $i \in I$. We will also call this a *free parallel system*, sometimes denoted by $\mathcal{L}^I$.

The 2-valued subsystems L_i, $i \in I$ are equipped with *local truth values* $\Omega_i = \{T_i, F_i\}$. The set of *(global) truth values* Ω^I for the whole fibering is the disjoint union $\coprod_{i \in I} \Omega_i$.

In such logical systems there are many ways to form logical operations by combining classical logical connectives locally in each L_i and then putting them together in the form of "logical vectors" like $(\phi_i)_{i \in I}$. Furthermore, we can model "system changes" in the sense that we shift logical information (formulas) from one subsystem L_i to another L_j. This corresponds to model communication between fibers (seen as logical state spaces of corresponding agents). For a more formal treatment of such univariate operations and the formation of logical expressions in a logical fibering we refer to [Pfa91].

A basic operation for parallel systems $\mathcal{L}^I$ is the mapping of a local pair (x_i, y_i) in $L_i \times L_i$ $(i \in I)$ into different subsystems L_j, L_k, etc.

In [Pfa91], a classification of all such bivariate operations, called *transjunctions*, is given. A *transjunction* can be represented by its *truth value matrix*, a mapping from a bivariate truth table within a fixed local system L_i into a bivariate truth table where the T and F values (occurring in the truth table within L_i) are *distributed* over four value sets Ω_α, Ω_β, Ω_γ, and Ω_δ, corresponding to the subsystems L_α, L_β, L_γ, and L_δ respectively. The classification of [Pfa91] is based on the type of the truth table under consideration — to which classical connective it corresponds when omitting the indices.

Logical Fiberings and Applications to Modeling Control in Systems of Cooperating Agents

The idea of logical fiberings has been applied to model control for small concrete examples [PSS95, PSS96a]. It turned out that the logical fiberings approach is particularly suitable for modeling communication and interaction between cooperating agents, due to the possibility to switch between a local and a global point of view which is typical for this framework.

We now illustrate the ideas presented above on a simple example, adapted from [Pfa93] (see also [DPSS91]). More complex examples (including also error-handling), as well as the relation between planning and modeling with logical fiberings can be found in [PSS95] and [PSS96a].

Let R_0, R_1, R_2 be three robots performing the following task:
R_0 receives a work piece a and a work piece b. He checks whether a and b are well-positioned on the table. If a and b are well-positioned then R_0 performs an assembly task, and the work piece r obtained from assembling a and b is placed on the table. The pieces of type a are furnished by R_1: if there are pieces of type a in stock, and if no r is placed on the table, R_1 brings a piece of type a and puts it on the assembly bench of R_0. The pieces of type b are furnished by R_2: if there are pieces of type b in stock, and if no r is on the table, R_2 brings a piece of type b to the assembly bench of R_0. After R_0 has assembled a and b and the result r has been placed on the table, r is transported to the stock by R_1 together with R_2.

The states of the system consisting of the cooperating agents R_0, R_1, R_2 can be described by specifying whether pieces of type a (resp. b) are left in stock, and whether a piece of type a (b, r) is placed on the table.

We will use the following truth values to correspond to the actions (TV stands for “Truth Value” and LS for “Logical Subsystem”):

TV	LS	Action
T_0	L_0	R_0 remains inactive
F_0	L_0	R_0 performs the assembly task
T_1	L_1	R_1 remains inactive
F_{11}	L_1	R_1 brings pieces of type a
F_{12}	L_1	R_1 transports the result
T_2	L_2	R_2 remains inactive
F_{21}	L_2	R_2 brings pieces of type b
F_{22}	L_2	R_2 transports the result

Note that L_1 and L_2 are 3-valued logical systems, and L_0 is 2-valued.

As in [PSS95] we now define the m-transjunction Θ by its truth table, below. In order to render the situation pictorially, we put the values of the first four logical propositions horizontally and the last three vertically. Impossible combinations of the truth values of the variables have been deleted. We see that there are several cases in which the resulting truth value is not uniquely determined; this corresponds to a choice of actions for the agents in the scenario. In what follows, $F_\vee$ stands for $F_{11} \vee F_{21}$, $T_\wedge$ for $T_1 \wedge T_2$ and $F_\wedge$ for $F_{12} \wedge F_{22}$.

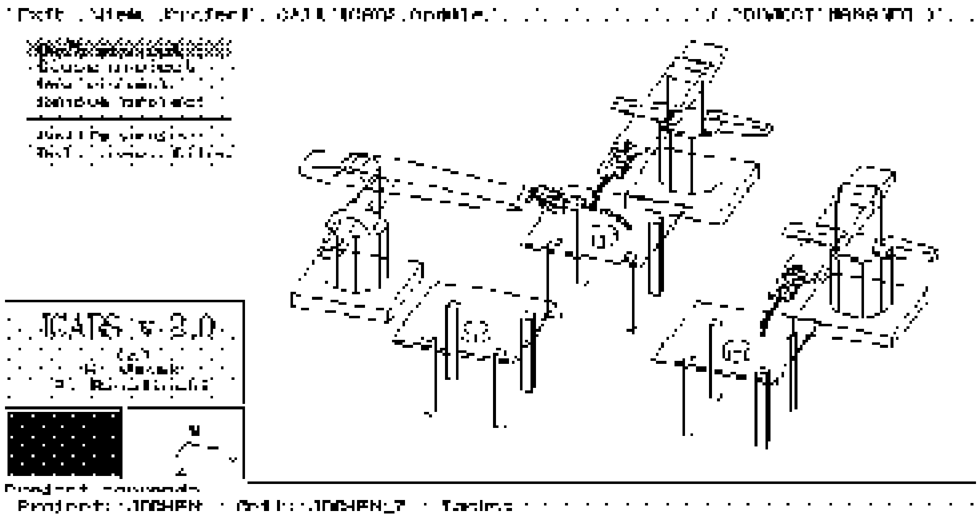


Figure 4.3: Animation of a Scenario with ICARS

Θ		$ontable_a$	0	1	0	1
		$ontable_b$	0	0	1	1
$instock_a$	$instock_b$	$ontable_r$				
$m(m>0)$	$n(n>0)$	0	$F_\vee$	F_{21}	F_{11}	F_0
$m(m>0)$	0	0	F_{11}	T_2	F_{11}	F_0
0	$n(n>0)$	0	F_{21}	F_{21}	T_1	F_0
0	0	0	$T_\wedge$	T_2	T_1	F_0
$m(m\geq 0)$	$n(n\geq 0)$	1	$F_\wedge$	$F_\wedge$	$F_\wedge$	$F_\wedge$

Moreover, every individual action R_{ij} can be seen as a morphism on the corresponding local section, thereby regulating the new values of the control propositions.

Note that in this example if we take the overall, "global" view, the model obtained is similar to the classical model (used for example in planning). Transjunctions are applied on the initial global section, until the goal situation (corresponding section) is reached.

Taking a "local view" however changes things, as robots may be forced to carry out conflicting actions necessitating communication. This communication is modeled by taking global sections in such cases. In the example discussed, conflicts can only arise between R_1 and R_2, suggesting to group the two together as an autonomous unit. What we obtain then is just an optimized version of the "global" approach. For more complicated examples, this concept of localizing the processing of data and parallelizing the decision on which actions to perform next may greatly improve the efficiency of planning and control.

A demonstrator was implemented in the frame of the MEDLAR projects [PSSS95]. It is based on the theoretical results concerning logical fiberings, and uses for animation the ICARS system (developed by W. Jacak and P. Rogalinski).

This system uses wireframe representations for the robots (3-D color graphics) and carries out path planning and collision detection modules before performing any movement the robots have been ordered to make. The animation is driven by the combination of the corresponding interacting logic and planning modules, based on the logical fiberings approach; the logical controller is verified by a Prolog program. A copy of the screen is given in Figure 4.3.

Sheaf Theoretical Models

From the approaches to concurrency based on presheaf and sheaf theory we will briefly present the approaches of Monteiro and Pereira [MP86], Goguen [Gog92], and Winskel and Cattani [Win96, CW96].

The starting point of the approach of Monteiro and Pereira [MP86] is the observation that the structure of a set of concurrent systems (unlike the structure of a single system) is determined by the way the systems are connected and by the way the connections carry interactions among systems. This form of interaction can be described by imposing that individual systems have "individual locations" and that interacting systems occupy "overlapping locations". Furthermore, such overlaps are considered "sites of interaction". This leads to the notion that locations are closed under finite "intersections".

Any activity is seen as taking place at some specific location. Since one can distinguish a "global" location (where the activity of all system components occurs), a "null" location (where no activity occurs) and a location for every set of component systems, this suggests that the set of locations should be closed under finite intersections and arbitrary unions – so the set of locations can be modeled as a topological space or, more abstractly, as a complete Heyting algebra.

Behaviors can be described as elements of the free monoid Σ^* over some event vocabulary Σ. In this way however the information concerning the contribution of the component parts to the behavior is lost. Monteiro and Pereira had the idea of extending the notion of behavior monoids to a structure of localized events, namely the notion of a sheaf of monoids over a complete Heyting algebra of event locations. In [MP86] they study this type of sheaves of monoids associated to concurrent systems. They show that it has the same kind of universal and functorial properties as the free monoids. This definition of sheaves of behavior monoids provides the setting for the definition of sheaves of processes interacting at the intersection of their locations. The sheaves of processes over a complete Heyting algebra have a complete lattice structure that allows the solution of process equations by fixpoint methods.

In [Gog92] Goguen uses concepts from sheaf theory to explain phenomena in concurrent systems, as object, inheritance, deadlock, and non-interference. The approach is very general: it applies not only to concurrent object oriented systems, but also to systems of differential equations, electrical circuits, hardware description languages and much more.

The objects are modeled by sheaves $F : \mathcal{B} \rightarrow \mathsf{Sets}$ over a "base for observation" that contains certain space-time domains. F associates with every domain of observation U (of space-time) a set $F(U)$ of attributes of the given object observed at U. The possible domains U are partially ordered by inclusion, and typically are closed under finite intersections and arbitrary unions, i.e. they form a topological space.

The sheaf condition says that any set of pairwise consistent local observations can be "glued together" into a unique observation over the union of their domains. This sheaf condition appears to be satisfied by all behaviors in all naturally arising systems from Computer Science; however it is not satisfied by certain properties of systems such as for instance the fairness property (cf. [Gog92] p.169).

The main points made in [Gog92] about the relationship between sheaves and objects are the following:

- Objects give rise to sheaves,
- Inheritance relations correspond to sheaf morphisms,
- Systems are diagrams of sheaves,
- Colimits in the category of diagrams of sheaves correspond to interconnecting systems,
- The behavior of a system is given by the limit of its diagram.

These ideas have been applied to Petri Nets by Lilius [Lil93]. The ideas from [Gog92] have been further developed by Malcolm in [Mal94], where a formalization of object classes and systems of objects is given, in order to study basic properties of ways in which systems of objects may be interconnected. He defines an adjunction between PO-systems (functors $S : \mathcal{C}^{op} \rightarrow \mathsf{Obj}$, where $\mathcal{C}$ is a partially ordered set) and sheaves of objects (PO-systems $S : \mathcal{C}^{op} \rightarrow \mathsf{Obj}$ where $\mathcal{C}$ is a complete Heyting algebra) and expresses the hope that, by using a more general notion of sheaf as a functor on a category with a Grothendieck topology, an adjunction between system specifications and sheaves of objects can be obtained. In [Cîr95] Cîrstea shows that transition systems and sheaves can be related by means of an adjunction between the corresponding categories. This is used in order to give a sheaf-theoretic formalization of the distributed semantics for FOOPS developed in [Cîr95].

[Win96] and [CW96] investigate presheaf models for processes.

[Win96] is concerned with the study of presheaf models for process calculi with value passing, and [CW96] with modeling process constructions on presheaves, showing that these preserve open maps, and with transferring such results to traditional models for processes.

Intuitively, process calculi are modeled as presheaves over a small category P considered a "path-category": a model like a transition system or a labelled event structure gives rise to a presheaf $F : \mathsf{P}^{op} \rightarrow \mathsf{Sets}$ associating with every path object P in P the set $F(P)$ of paths (in the chosen model) with "shape" P.

Presheaf models for concurrency turn out to include interleaving models like synchronization trees and independence models like labelled event structures, as well as contributing to a general definition of bisimulation based on so-called open maps.

Roughly speaking, open maps in a category M of models (that can be $\mathsf{TS}, \mathsf{PN}, \mathsf{ES}$ etc. or a fiber in any of these categories) are morphisms with the property that any extension of a computation path in the codomain can be matched by an extension of its domain.

Formally, whenever for $m : P \to Q$ a morphism in P a diagram

$$\begin{tikzcd} P \arrow[r, "p"] \arrow[d, "m"'] & X \arrow[d, "f"] \\ Q \arrow[r, "q"'] & Y \end{tikzcd}$$

in M commutes (meaning that the path $f \circ p$ in Y can be extended via m to a path q in Y), then there is a morphism p' such that in the diagram

$$\begin{tikzcd} P \arrow[r, "p"] \arrow[d, "m"'] & X \arrow[d, "f"] \\ Q \arrow[r, "q"'] \arrow[ur, "p'"] & Y \end{tikzcd}$$

$p' \circ m = p$ and $f \circ p' = q$ (meaning that the path p can be extended via m to a path p' in X which matches q). When the morphism p satisfies this property we say that it is P-open.

[Win96] studies denotational semantics in presheaf models. This is shown to correspond to operational semantics in the sense that bisimulation obtained from the so-called "open maps" is proved to coincide with bisimulation as defined traditionally from the operational semantics. A presheaf model and denotational semantics are proposed for a language allowing process-passing.

[CW96] models process constructions on presheaves showing that these preserve open maps, and transfers such results to traditional models of processes. They show that a wide range of left Kan extensions between categories of presheaves preserve open maps; a particular Kan extension is shown to coincide with a refinement operation on event structures. [CW96] explains (by general arguments) why the operations of a presheaf model preserve open maps and why for specific presheaf models the operations coincide with those traditional models.

Chapter 5

Fibered Representation and Universal Algebra

In this chapter we present some common features of known representation theorems for certain classes of algebras with a lattice reduct, and a method for automated theorem proving in certain many-valued logics, based on the Priestley dual of the algebra of truth values.

We begin with a detailed presentation of the ideas for the case of SHn-logics, since the idea occured to us when working on this topic. Since in Section 4.1.2 we already discussed the sheaf representation theorems for discriminator varieties and its application, we will only mention this type of representation theorems in the case of SHn-algebras and will focus on representation theorems induced by the Priestley representation theorem.

We first present a duality theorem induced by the Priestley duality theorem (an extension of the results in [Itu83]). The Priestley representation theorems suggested a method for automated theorem proving in SHn-logics. This method is presented in Section 5.2. A more general case is discussed in Section 5.3.1, where we analyze the conditions under which a theorem proving procedure similar to that described in Section 5.2 holds. In Section 5.3.3 the automated theorem proving procedure is extended to more general classes of logics; we also take into account first-order logics.

After presenting the ideas in this general framework, we illustrate this general procedure for the case of:

- P_{mn}-logics (sound and complete with respect to the subvariety P_{mn} of the variety of Ockham algebras),
- $SHKn$-logics (sound and complete with respect to the variety of $SHKn$-algebras (Lukasiewicz-Moisil algebras of order n)).

We end this chapter with the description of an implementation of the procedure.

5.1 A Motivating Example: SHn-logics

The language of SHn-logics is a propositional language, whose formulae are built from propositional variables taken from a set Var, with operations $\vee$ (disjunction), $\wedge$ (conjunction), $\Rightarrow$ (intuitionistic implication), $\sim, \neg$ (a De Morgan resp. an intuitionistic negation), and a family $\{S_i \mid i = 1, \ldots, n-1\}$ of unary operations.

The following Hilbert style axiomatization of SHn-logics is taken from [Itu82], see also [IO96].

Axioms:

(A1) $a \Rightarrow (b \Rightarrow a)$
(A2) $(a \Rightarrow (b \Rightarrow c)) \Rightarrow ((a \Rightarrow b) \Rightarrow (a \Rightarrow c))$
(A3) $(a \wedge b) \Rightarrow a$
(A4) $(a \wedge b) \Rightarrow b$
(A5) $(a \Rightarrow b) \Rightarrow ((a \Rightarrow c) \Rightarrow (a \Rightarrow (b \wedge c)))$
(A6) $a \Rightarrow (a \vee b)$
(A7) $b \Rightarrow (a \vee b)$
(A8) $(a \Rightarrow c) \Rightarrow ((b \Rightarrow c) \Rightarrow ((a \vee b)c))$
(A9) $\sim\sim a \leftrightarrow a$
(A10) $S_i(a \wedge b) \leftrightarrow S_i(a) \wedge S_i(b)$
(A11) $S_i(a \Rightarrow b) \leftrightarrow (\bigwedge_{k=i}^{n} S_k(a) \Rightarrow S_k(b))$
(A12) $S_i(S_j(a)) \leftrightarrow S_j(a)$, for every $i, j = 1, \ldots, n-1$
(A13) $S_1(a) \Rightarrow a$
(A14) $S_i(\sim a) \leftrightarrow \sim S_{n-i}a$, for $i = 1, \ldots, n-1$
(A15) $S_1(a) \vee \neg S_1(a)$, where $\neg(a) = a \Rightarrow \sim (a \Rightarrow a)$

Inference rules:

$$\text{(R1)} \ \frac{a, \quad a \Rightarrow b}{b}$$

$$\text{(R2)} \ \frac{a \Rightarrow b}{\sim b \Rightarrow \sim a}$$

$$\text{(R3)} \ \frac{a \Rightarrow b}{S_1(a) \Rightarrow S_1(b)}$$

5.1.1 An Algebraic Semantics for SHn-logics

Definition 5.1 *An abstract algebra $A = (A, 0, 1, \wedge, \vee, \Rightarrow, \neg, \sim, S_1, \ldots, S_{n-1})$ is said to be a* symmetric Heyting algebra of order n *(SHn-algebra for short) if:*

(1) $(A, 0, 1, \wedge, \vee, \Rightarrow, \neg)$ is a Heyting algebra,

(2) $\sim$ is a De Morgan negation on A,

(3) For every $x, y \in A$ and for all $i, j \in \{1, \dots, n-1\}$, the following equations hold:

(S1) $S_i(a \wedge b) = S_i(a) \wedge S_i(b)$,

(S2) $S_i(a \Rightarrow b) = (\bigwedge_{k=i}^{n} S_k(a) \Rightarrow S_k(b))$,

(S3) $S_i(S_j(a)) = S_j(a)$, *for every* $i, j = 1, \dots, n-1$,

(S4) $S_1(a) \vee a = a$,

(S5) $S_i(\sim a) = \sim S_{n-i}a$, *for* $i = 1, \dots, n-1$,

(S6) $S_1x \vee \neg S_1x = 1$, *with* $\neg x = x \Rightarrow 0$.

From the above definition, from the fact that the class of Heyting algebras are equationally definable as well as from the fact that the De Morgan property can be expressed equationally, it follows that the class of symmetric Heyting algebras of order n is a variety, which will be denoted $\mathsf{SH_n}$.

We quote without proof the following properties which are true in every SHn-algebra (cf. [Itu83]):

(S7) $S_i1 = 1, S_i0 = 0$,
(S8) $S_i(x \vee y) = S_ix \vee S_iy$,
(S9) If $S_ix = S_iy$ *for all* $i = 1, \dots, n-1$, *then* $x = y$,
(S10) $x \leq y$ *if and only if* $S_ix \leq S_iy$,
(S11) $S_1x \leq S_2x \leq \dots S_{n-1}x$,
(S12) $x \leq S_{n-1}x$,
(S13) $S_ix \wedge \neg S_ix = 0$,
(S14) $S_ix \vee \neg S_ix = 1$,
(S15) $S_i(\neg x) = \neg S_{n-1}x$,
(S16) $x \wedge S_{i+1}y \leq y \vee S_i(x)$.

Definition 5.2 *Let $n \geq 2$ and let S_{n^2} be the cartesian product $L_n \times L_n$ where $L_n = \{0, \frac{1}{n-1}, \dots, \frac{n-1}{n-1}\}$. Consider the following operations on S_{n^2}:*
(1) $\vee, \wedge$ *defined pointwise,*
(2) $\sim (x, y) = (1 - y, 1 - x)$ *for every* $(x, y) \in S_{n^2}$,
(3) $S_i(x, y) = (S_i(x), S_i(y))$, *where*

$$S_i(\frac{j}{n-1}) = \begin{cases} 1 & \textit{if } i + j \geq n, \\ 0 & \textit{if } i + j < n, \end{cases}$$

(4) $(x_1, x_2) \Rightarrow (y_1, y_2) = (x_1 \Rightarrow y_1, x_2 \Rightarrow y_2)$, *where $\Rightarrow$ is the Heyting relative pseudocomplementation on L_n*[1].

[1]The Heyting relative pseudocomplementation on L_n is defined by: $x \Rightarrow y$ is the largest element z of L_n such that $x \wedge z \leq y$. Hence, $x \Rightarrow y = \begin{cases} 1 & \text{if } x \leq y \\ y & \text{if } x > y \end{cases}$. The pseudocomplementation induced by $\Rightarrow$ is defined by $\neg x = x \Rightarrow 0$. Hence, $\neg x = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x > 0 \end{cases}$.

It is easy to see that $S_{n^2} = (L_n \times L_n, (0,0), (1,1), \vee, \wedge, \Rightarrow, \sim, S_1, \ldots, S_{n-1})$ is a symmetric Heyting algebra of order n [Itu83].

Proposition 5.1 ([Itu83]) *Every symmetric Heyting algebra of order n is isomorphic to a subdirect product of a family of subalgebras of S_{n^2}.*

It follows that the variety SHn of symmetric Heyting algebras of order n is generated by S_{n^2}.

Lemma 5.2 *The variety* SHn *is a discriminator variety.*

Proof: We know that the variety of SH-algebras of order n is generated by the algebra S_{n^2}. We show that S_{n^2} has a discriminator term. Let $d(x) = S_1(x) \wedge \neg \sim S_1(x)$ be a term in the language of SHn-algebras with one variable x. We know that for every $a = (\frac{i}{n-1}, \frac{j}{n-1}) \in S_{n^2}$,

$$S_1(\frac{i}{n-1}, \frac{j}{n-1}) = \begin{cases} (1,1) & \text{if } 1+i \geq n \quad \text{and } 1+j \geq n, \\ (1,0) & \text{if } 1+i \geq n \quad \text{and } 1+j < n, \\ (0,1) & \text{if } 1+i < n \quad \text{and } 1+j \geq n, \\ (0,0) & \text{if } 1+i < n \quad \text{and } 1+j < n, \end{cases}$$

$$\sim S_1(\frac{i}{n-1}, \frac{j}{n-1}) = \begin{cases} (0,0) & \text{if } 1+i \geq n \quad \text{and } 1+j \geq n, \\ (1,0) & \text{if } 1+i \geq n \quad \text{and } 1+j < n, \\ (0,1) & \text{if } 1+i < n \quad \text{and } 1+j \geq n, \\ (1,1) & \text{if } 1+i < n \quad \text{and } 1+j < n, \end{cases}$$

$$\neg \sim S_1(\frac{i}{n-1}, \frac{j}{n-1}) = \begin{cases} (1,1) & \text{if } 1+i \geq n \quad \text{and } 1+j \geq n, \\ (0,1) & \text{if } 1+i \geq n \quad \text{and } 1+j < n, \\ (1,0) & \text{if } 1+i < n \quad \text{and } 1+j \geq n, \\ (0,0) & \text{if } 1+i < n \quad \text{and } 1+j < n. \end{cases}$$

Therefore $S_1(a) \wedge \neg \sim S_1(a)$ is $(1,1)(= 1_{S_{n^2}})$ if $a = (1,1) = 1_{S_{n^2}}$ and is $0_{S_{n^2}}$ otherwise. Hence, $d(a) = \begin{cases} 1_{S_{n^2}} & \text{if } a = 1_{S_{n^2}} \\ 0_{S_{n^2}} & \text{otherwise} \end{cases}$. Therefore we are in the situation described by Example (1) of Section 3.1.4. Hence, a discriminator term for S_{n^2} is $t(x,y,z) = [z \wedge d(x \vee y \Rightarrow x \wedge y)] \vee [x \wedge (d(x \vee y \Rightarrow x \wedge y) \Rightarrow 0)]$.
□

5.1.2 A Kripke Semantics for SHn-logics

In [IO96] a Kripke-style semantics for SHn-logic is given. We recall here the basic definitions and results given in [IO96].

Definition 5.3 (SHn-frame, cf. [IO96]) *A* SHn-frame *is a system* $K = (W, R, \{s_i \mid i = 1, \ldots, n-1\}, g)$ *where:*

(K0) W is a nonempty set (of states), R is a binary relation on W and all s_i and g are functions on W, such that:

(K1) R reflexive,

(K2) R *transitive,*

(K3) $R(x,y)$ *implies* $R(g(y),g(x))$,

(K4) $g(s_i(x)) = s_{n-i}(g(x))$, *for all* $i = 1,\ldots,n-1$,

(K5) $g(g(x)) = x$,

(K6) $s_j(s_i(x)) = s_j(x)$, *for all* $i,j = 1,\ldots,n-1$,

(K7) $R(s_1(x),x)$,

(K8) $R(x,s_{n-1}(x))$,

(K9) $R(s_i(x),s_j(x))$, *for all* $i \leq j$,

(K10) $R(x,y)$ *implies* $R(s_i(x),s_i(y))$ *and* $R(s_i(y),s_i(x))$,

(K11) $R(s_i(y),y)$ *and* $R(y,s_i(y))$ *imply* $s_i(y) = y$, *for all* $i = 1,\ldots,n-1$,

(K12) $R(x,s_i(x))$ *or* $R(s_{i+1}(x),x)$, *for all* $i = 1,\ldots,n-1$, *(for* $n \geq 3$*).*

Definition 5.4 (SHn-model, cf. [IO96]) *An* SHn-model *based on a frame* K *is a system* $M = (K,m)$ *such that* $m : \mathsf{Var} \to \mathcal{P}(W)$ *is a meaning function that assigns subsets of states to propositional variables and satisfies the following condition:*
(HER) $\qquad$ $R(x,y)$ *and* $x \in m(p)$ *imply* $y \in m(p)$.

Definition 5.5 ([IO96]) *We say that an* SHn*-model* M *satisfies a formula* ϕ *at the state* x *(denoted by* $M\overset{r}{\models}_x\phi$*) if the following conditions are satisfied:*

$M\overset{r}{\models}_x p$	*if and only if*	$x \in m(p)$, $\qquad$ *for* $p \in \mathsf{Var}$,
$M\overset{r}{\models}_x \phi \vee \psi$	*if and only if*	$M\overset{r}{\models}_x\phi$ *or* $M\overset{r}{\models}_x\psi$,
$M\overset{r}{\models}_x \phi \wedge \psi$	*if and only if*	$M\overset{r}{\models}_x\phi$ *and* $M\overset{r}{\models}_x\psi$,
$M\overset{r}{\models}_x \phi \Rightarrow \psi$	*if and only if*	*for all* y, *if* $R(x,y)$ *and* $M\overset{r}{\models}_y\phi$ *then* $M\overset{r}{\models}_y\psi$,
$M\overset{r}{\models}_x \neg\phi$	*if and only if*	*for all* y, *if* $R(x,y)$ *then* $M \overset{r}{\not\models}_y\phi$,
$M\overset{r}{\models}_x S_i(\phi)$	*if and only if*	$M\overset{r}{\models}_{s_i(x)}\phi$,
$M\overset{r}{\models}_x \sim\phi$	*if and only if*	$M \overset{r}{\not\models}_{g(x)}\phi$.

A formula ϕ *is true in an* SHn*-model* M *(denoted* $M \overset{r}{\models} \phi$*) if and only if for every* $x \in W$, $M\overset{r}{\models}_x\phi$. *If* $M = (K,m)$, *we will sometimes use the notation* $K\overset{r}{\models}_m\phi$.

The formula ϕ *is true in a* SHn*-frame* K *(denoted by* $K \overset{r}{\models} \phi$*) if and only if it is true in every* SHn*-model based on* K. *The formula* ϕ *is* SHn-valid *if and only if it is true in every* SHn*-frame.*

A formula ϕ *is a semantical consequence of a set of formulae* Γ *(denoted by* $\Gamma \overset{r}{\models} \phi$*) if and only if for every model* M, *if all the formulae from* Γ *are true in* M, *then* ϕ *is true in* M.

The following theorems are proved in [IO96]:

Proposition 5.3 ([IO96]) *In every SHn-frame, if $R(s_i(x), y)$ then there exists $j \geq i$ and there exists $z \in W$ such that $R(x, z)$ and $y = s_j(z)$.*

Proposition 5.4 ([IO96]) *Given a model $M = (K, m)$, the meaning function m can be extended to all formulae by $m(\phi) = \{x \in W \mid M \overset{r}{\models}_x \phi\}$. For every model M and for every formula ϕ, this extension has the property*
(HER) *If $R(x, y)$ and $x \in m(\phi)$ then $y \in m(\phi)$.*

Proposition 5.5 ([IO96]) $X \vdash_{SHn} \phi$ *if and only if* $X \overset{r}{\models} \phi$.

5.1.3 Decomposition

We now point out the basic ideas of the main known representation theorems for SHn-algebras, namely the sheaf representation and the Priestley representation. They will be discussed in detail in Section 5.1.4 and Section 5.1.5. These representations can be seen as "decompositions" of the algebra as an indexed family of simpler algebras, such that the index space has a good structure.

For a given SHn-algebra A, we fix a base set and give a representation of A in terms of continuous functions with domain I. We distinguish several possibilities:

Case 1: I is the set of all maximal congruences of SHn-algebras (including ∇). In this case we obtain a sheaf representation theorem which is the particularization of the sheaf representation theorem given by Werner (which holds for every discriminator variety) to the variety of SHn-algebras.

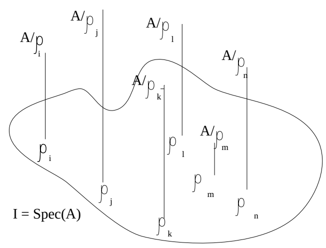


Figure 5.1: The Standard Sheaf Associated to the SHn-algebra A

We point out that in this case the "fibers" are subalgebras of S_{n^2}, and the "base space" is a topological space, with no relation.

Case 2: I is the set of all maximal congruences of A as a distributive lattice. In this case I is the Priestley space of A. We know that A is isomorphic (as a distributive 0,1-lattice) with the lattice of clopen order-filters of I (i.e. with the lattice of all continuous order-preserving functions from I to $\{0,1\}$). In Section 5.1.5 we will show how the Priestley duality theorem can be extended to a duality theorem for SHn-algebras (a topological representation theorem appears for SHn-algebras is given in [Itu83]): The SHn-algebra operations $(\Rightarrow, \neg, \sim, S_1, \dots, S_n)$ on A define a family of relations on I. From this family of relations, new operations $\Rightarrow', \neg', \sim', S_1', \dots, S_n'$ can be defined on the lattice of clopen order-filters of I.

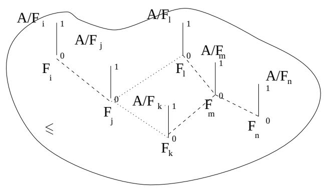


Figure 5.2: The Priestley space associated to the SHn-algebra A

We point out that in this case all the "fibers" are isomorphic to the 2-element lattice $\{0,1\}$ and the "base space" is a topological space endowed with a partial order and relations corresponding to the "non-lattice"-operations on A.

The situations described above seem to be two extreme possibilities. This suggests that one might consider several other possibilities. We may for instance take I to be the set of all maximal congruences of A seen as a De Morgan algebra, or the set of all maximal congruences of A seen as a symmetric Heyting algebra.

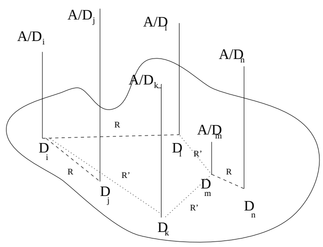


Figure 5.3: Space associated to the SHn-algebra A

In certain situations, the operations not included in the subsignature with respect to which the elements of I are congruences may induce additional relations on I. In this paper we will only focus on the first two (classical) cases. The "intermediate" possibilities will be subject for future work. Note the similarity between this idea and the idea on which the notion of "Logical Fiberings" (due to Jochen Pfalzgraf) is based (cf. [Pfa91] and Section 4.1.5): we can put in evidence a "base space" I and "fibers" (namely, in cases 1 and 2, simple SHn-algebras or, respectively, the lattices $\{0,1\}$).

5.1.4 Sheaf Representations for SHn-algebras and Applications

In Section 5.1.1 we noted that the variety of SH-algebras of order n is a discriminator variety. We can therefore apply the theoretical results described in Section 4.1.2 to this variety.

We first want to determine for every SHn-algebra A the topological space $Spec(A)$ of all maximal congruences of A. The answers are taken from [Itu83].

It is easy to see that the kernel D of a homomorphism $f : A \to B$ of SHn algebras has the following properties:

(D1) $1 \in D$,

(D2) If $a, a \Rightarrow b \in D$ then $b \in D$,

(D3) If $a \Rightarrow b \in D$ then $\sim b \Rightarrow \sim a \in D$,

(D4) If $a \in D$ then $S_1(a) \in D$.

Definition 5.6 (Deductive System) *A subset D of a SHn algebra A is a* deductive system *if it satisfies the conditions (D1), (D2), (D3), (D4).*

Definition 5.7 (Maximal Deductive System) *A deductive system D is* maximal *if it is proper and it is not a proper subset of any other proper deductive system.*

Remark 5.6 ([Itu83]) *The ordered set of all congruences of A is isomorphic to the set of all deductive systems of A, ordered by inclusion.*

It follows that there is a bijective correspondence between the set $Spec(A)$ of maximal congruences on A and the set of all maximal deductive systems of A.

Theorem 5.7 ([Itu83]) *The following statements are true in every SHn-algebra A:*

(1) 1 is the meet of all maximal deductive systems in A,

(2) For a deductive system D in A the following conditions are equivalent:

(i) A/D is simple,
(ii) D is maximal.

In conclusion, every non-degenerate SHn-algebra A is isomorphic to a subdirect product of a family of simple SHn-algebras.

Proposition 5.8 ([Itu83]) *The following statements are true in the variety of SHn-algebras:*

(1) If D is a maximal deductive system in an SHn-algebra A, then A/D is an SHn-algebra isomorphic to a subalgebra of S_{n^2}.

(2) *The simple SHn-algebras are exactly the subalgebras of S_{n^2}.*

(3) *The subdirectly irreducible SHn-algebras are the same as the subalgebras of S_{n^2}.*

Definition 5.8 (Standard Sheaf) *Let A be an SHn-algebra. The standard sheaf associated to A is the sheaf $S(A) = (\coprod_{\theta \in Spec(A)} A/\theta, f, Spec(A))$ over $Spec(A)$.*

Since the variety of SHn-algebras is a discriminator variety, by Theorem 4.11 it follows that the standard representation $[\ ] : A \to \Gamma S(A)$ defined for every $\theta \in Spec(A)$ by $\overline{a}(\theta) = [a]_\theta$ is an isomorphism. This shows that A is isomorphic to the algebra of global sections of a sheaf having as base space the boolean space of all maximal congruences on A (or equivalently all maximal deductive systems – including A), and having as fibers subalgebras of S_{n^2} (and also the one-point algebra, corresponding to ∇).

Applications: Theorem Proving and Solving Equations (Unification)

1. Theorem Proving

By Theorem 5.8, the variety of SHn-algebras is generated by the algebra S_{n^2}. Since propositional SHn-logic is sound and complete with respect to the variety of SHn-algebras, it follows that a formula ϕ in the language of the SHn-logic is a theorem if and only if $SHn \models \phi = 1$. Using the fact that the variety of SHn-algebras is generated by the algebra S_{n^2} it follows that

$$SHn \vdash \phi \text{ if and only if } S_{n^2} \models \phi.$$

Thus, a method for checking that a given formula is a SHn-theorem is to show that for every assignment $f : \mathsf{Var} \to S_{n^2}$, $\overline{f}(\phi) = 1$.

The complexity of this procedure is $\mathcal{O}(n^2)^m$, where m is the number of variables that occur in ϕ.

Another method would be to show that the equation $\phi = 1$ has as solutions all the combinations of truth values in S_{n^2}. In what follows we will consider the application of the sheaf representation theorem for SHn-algebras in solving equations (unification).

2. Solving Equations; Unification

We apply the theorems presented in Section 4.1.3 to the variety SHn of SHn-algebras.

Lemma 5.9 *The free SHn-algebra with 0 generators is the 2-element SHn-algebra. There is only one simple algebra with 0 generators, namely the 2-element SHn-algebra.*

Proof: The free SHn-algebra with 0 generators contains 0 and 1, and is closed under all the operations. But it is known that in every SHn-algebra $\sim 0 = 1, \sim 1 = 0, \neg 0 = 1, \neg 1 = 0, S_i(0) = 0, S_i(1) = 1, 0 \Rightarrow 1 = 1$ and $1 \Rightarrow 0 = 0$. Hence, $\{0, 1\}$ is closed under all the operations.

We know (see e.g. [Itu83]) that the simple SHn-algebras are exactly the subalgebras of S_{n^2}. Let S be a simple algebra with 0 generators. Then $S \subseteq S_{n^2}$ is closed under all the operations and $(0,0), (1,1) \in S$. By the definition of the operations in S_{n^2} it is easy to see that the set $\{(0,0), (1,1)\}$ is closed under all the operations of SHn-algebra. Hence, $S = \{(0,0), (1,1)\}$. □

In what follows we will denote the SHn-algebra with 0 generators by L_2. Note that if p and q are unifiable then they have at least one ground unifier.

Theorem 4.14 specializes to the variety of SHn-algebras as follows.

Corollary 5.10 *Let* SHn *be the variety of SHn-algebras. Then:*

(1) $x_i \leftarrow r_i, 1 \leq i \leq n$, *(where r_i are ground terms), is a unifier of p and q in* SHn *if and only if* $L_2 \models p(r_1, \ldots, r_n) = q(r_1, \ldots, r_n)$.

(2) In this case, the substitution $\sigma : \mathsf{T}_\Sigma(X) \to \mathsf{T}_\Sigma(\hat{X})$ *defined by* $x_i \leftarrow s(\hat{p}, \hat{q}, \hat{x}_i, \hat{r}_i)$, $(i = 1, \ldots, n)$ *is a most general unifier for p and q in* SHn.

Theorems 4.12 and 4.14 show how a most general unifier for two terms in the language of the variety of SHn-algebras can be found:

1) Find a solution in L_2 (the simple SHn-algebra with 0 generators),

2) If no solution is found then by Theorem 4.14 we know that the terms are not unifiable,

3) If there is a (ground) solution in L_2, use Theorem 4.12 to construct a most general unifier.

5.1.5 Priestley Duality for the Variety of SHn-algebras

In [Itu83] Iturrioz gives a topological representation theorem for SHn-algebras. However, in [Itu83], the nature of morphisms in the category of corresponding Priestley spaces with operators, and the functorial aspect of the duality are not discussed. Here we present the construction in detail. This detailed presentation aims at capturing the main features of this type of representations, and at making clear which links exist between the algebraic models and their duals.

We start from the Priestley duality for distributive lattices. Heyting algebras have additional operations as $\Rightarrow$ and $\neg$. An extension of the Priestley duality to Heyting algebras can be found in [Gol89]. We present it below, and then extend it to a duality theorem for SHn-algebras.

Priestley Duality for Heyting Algebras

Let HAlg be the category of Heyting algebras and HSp the category of Heyting spaces, having as objects those Priestley spaces $(X, \leq, \tau)$ with the additional property

$(H1)$ For every $U \subseteq X$ clopen, $\downarrow U$ is also clopen,

and as morphisms maps $\phi : (X_1, \leq_1, \tau_1) \to (X_2, \leq_2, \tau_2)$ which are continuous, order-preserving and additionally satisfy the following condition:

$(H2)$ $\{y \mid \phi(z) \leq y\} = \{\phi(x) \mid z \leq x\}$,

in other words, the order-filter generated in X_2 by $\phi(z)$ is the image by ϕ of the order-filter generated by z in X_1.

Remarks:

(1) The condition $(H1)$ assures that every element U in $E(X)$ has a pseudocomplement, namely $X \setminus \downarrow U$. It is easy to see that the dual of every Heyting algebra satisfies $(H1)$.

(2) A relative pseudocomplement in $E(X)$ is defined (see e.g. [Gol89]) for every U, V clopen order-filters by

$$U \Rightarrow V = \{x \in X \mid \text{ if } x \leq y \text{ and } y \in U \text{ then } y \in V\}$$

In terms of continuous, order-preserving functions this can be reformulated as:

For $h_1, h_2 \in E(X)$,

$$(h_1 \Rightarrow h_2)(x) = 1 \quad \text{if and only if} \quad \text{for every } y \geq x, \text{ if } h_1(y) = 1 \text{ then } h_2(y) = 1.$$

(3) Condition $(H2)$ ensures that the image $E(\phi)$ of a morphism in HSp preserves the pseudocomplement. It is easy to see that condition $(H2)$ is satisfied by every image $D(f)$ of a morphism in HAlg. It is easy to see that, assuming that ϕ is order-preserving, condition $(H2)$ is equivalent to the following condition[2]:

$(H2')$ If $\phi(z) \leq y$ then there exists x with $(z \leq x$ and $\phi(x) = y)$.

It can be proved (see e.g. the remark in [DP90], pp.205) that the Priestley duality induces a dual equivalence between the category of Heyting algebras and the category of Heyting spaces. We give the outline of the proof:

Proof: We have the following functors:

$$\mathsf{HAlg} \xrightarrow{D} \mathsf{HSp} \qquad \mathsf{HSp} \xrightarrow{E} \mathsf{HAlg}$$

defined on objects by:

$$D(A) = \mathsf{Hom}_{D_{01}}(A, L_2) \qquad E(X) = \mathsf{Hom}_P(X, \{0,1\})$$

and on morphisms by:

$$f : A_1 \to A_2 \qquad\qquad h : X_1 \to X_2$$

[2]condition $(H2')$ states that ϕ is a bounded morphism with respect to the relation $\leq$ on X_1 and resp. X_2. Details on bounded morphisms will be given in Section 5.3.2.

$$D(f) : D(A_2) \to D(A_1) \qquad E(h) : E(X_2) \to E(X_1)$$
$$D(f)(\phi) = \phi \circ f \qquad E(h)(\psi) = \psi \circ h.$$

Facts:

(1) For every Heyting algebra A, $D(A)$ is a Priestley space and for every clopen set $U \subseteq D(A)$, $\downarrow U$ is also clopen (because $E(D(A))$ and A are isomorphic as 0,1-lattices, hence $E(D(A))$ is pseudocomplemented).

(2) For every Heyting space $(X, \leq, \tau)$, $E(X)$ is a Heyting algebra.

(3) For every Heyting algebra A, $E(D(A)) \simeq A$ as Heyting algebras. For every Heyting space X, $D(E(X)) \simeq X$ as Heyting spaces.

(4) For every $f : A_1 \to A_2$ morphism of Heyting algebras, $D(f) : D(A_2) \to D(A_1)$ satisfies $(H2)$.

(5) For every $h : X_1 \to X_2$, continuous, order-preserving and satisfying $(H2)$, $E(h)$ is a morphism of Heyting algebras (in particular $E(h)(a \Rightarrow b) = E(h)(a) \Rightarrow E(h)(b)$).

(6) $D : \mathsf{HAlg}(A_1, A_2) \to \mathsf{HSp}(D(A_2), D(A_1))$ and $E : \mathsf{HSp}(X_1, X_2) \to \mathsf{HAlg}(E(X_2), E(X_1))$ are bijections and the following diagrams commute:

$$\begin{tikzcd}
A_1 \arrow[r, "f"] \arrow[d, "\eta_1"'] & A_2 \arrow[d, "\eta_2"] \\
E(D(A_1)) \arrow[r, "E(D(f))"] & E(D(A_2))
\end{tikzcd}
\qquad
\begin{tikzcd}
X_1 \arrow[r, "h"] \arrow[d, "\epsilon_1"'] & X_2 \arrow[d, "\epsilon_2"] \\
D(E(X_1)) \arrow[r, "D(E(h))"] & D(E(X_2))
\end{tikzcd}$$

Priestley Duality for SHn-algebras

Every SHn-algebra has in particular a Heyting algebra structure. Therefore it seems reasonable to look for a suitable subcategory of the category of Heyting spaces which is dually equivalent to the category of SHn-algebras.

A natural idea is to associate for every SHn-algebra A a relation (or function) on the Heyting space corresponding to A to every non-Heyting operation of A.

Let $A = (A, \vee, \wedge, 0, 1, \Rightarrow, \neg, \sim, S_1, \ldots, S_{n-1})$ be a SHn algebra. Let $D(A) = \mathsf{Hom}_{D_{01}}(A, \{0, 1\})$ and consider the following functions on $D(A)$:

(1) $g : D(A) \to D(A)$ defined for every $f : A \to \{0, 1\}$ by $g(f) : A \to \{0, 1\}$ where $g(f)(a) = 1$ if and only if $f(\sim a) = 0$,

(2) For every $i = 1, \ldots, n-1$, a map $s_i : D(A) \to D(A)$ defined for every $f : A \to \{0, 1\}$ by $s_i(f) = f \circ S_i$ (i.e. $s_i(f)(a) = 1$ if and only if $f(S_i(a)) = 1$).

Remark: The definitions can also be given in terms of prime filters. In this case we have:

(1′) $g : D(A) \to D(A)$ is defined for every prime filter $F \in D(A)$ by $g(F) = \{x \in A \mid \sim x \notin F\}$ (the Bialynicki-Birula and Rasiowa involution associated to $\sim$), (2′) For every $i = 1, \ldots, n-1$, the map $s_i : D(A) \to D(A)$ is defined for every prime filter $F \in D(A)$ by $s_i(F) = S_i^{-1}(F)$.

In what follows we will usually prefer the morphism notation since it makes the proofs shorter. Intuitively however, the filter notion turns out to be more appropriate.

Lemma 5.11 *The maps g and s_i are well-defined and continuous.*

Proposition 5.12 *Let $K = (D(A), \leq, \tau, g, s_1, \ldots, s_{n-1})$ be the dual of the SHn-algebra A endowed with the operations defined above. The following properties are fulfilled:*

(1) If $f_1 \leq f_2$ then $g(f_2) \leq g(f_1)$,
(2) $g(s_i(f)) = s_{n-i}(g(f))$,
(3) $g(g(f)) = f$,
(4) $s_j(s_i(f)) = s_j(f)$,
(5) $s_1(f) \leq f$,
(6) $f \leq s_{n-1}(f)$,
(7) $s_i(f) \leq s_j(f)$ for every $i \leq j$,
(8) If $f_1 \leq f_2$ then $s_i(f_1) \leq s_i(f_2)$ and $s_i(f_2) \leq s_i(f_1)$ (i.e. they are equal),
(9) If $s_i(f_1) \leq f_2$ then $s_i(f_2) \leq f_2$,
(10) For all $i = 1, \ldots, n-1$, $f \leq s_i(f)$ or $s_{i+1}(f) \leq f$.

Proof: In what follows a is an arbitrary element of A and f, f_1, f_2 are elements in $D(A)$.

(1) We have $g(f_2)(a) = 1$ if and only if $f_2(\sim a) = 0$. Assume that $g(f_2) = 1$, i.e. $f_2(\sim a) = 0$. Since $f_1(\sim a) \leq f_2(\sim a)$ it follows $f_1(\sim a) = 0$, hence $g(f_1)(a) = 1$.

(2) We have $g(s_i(f))(a) = 1$ if and only if $s_i(f)(\sim a) = 0$, if and only if $f(S_i(\sim a)) = 0$; and $s_{n-i}(g(f))(a) = 1$ if and only if $g(f)(S_{n-i}(a)) = 1$ if and only if $f(\sim S_{n-i}(a)) = 0$. But in A, $S_i(\sim a) = \sim (S_{n-i}(a))$. Hence, $g(s_i(f))(a) = 1$ if and only if $s_{n-i}(g(f))(a) = 1$.

(3) $g(g(f))(a) = 1$ if and only if $g(f)(\sim a) = 0$ if and only if $f(\sim\sim a) = 1$ if and only if $f(a) = 1$ (because for every $a \in A$, $\sim\sim a = a$).

(4) $s_j(s_i(f))(a) = s_i(f)(S_j(a)) = f(S_i(S_j(a))) = f(S_j(a)) = s_j(f)(a)$.

(5) If $s_1(f)(a) = 1$ then $f(S_1(a)) = 1$ and, since for every $a \in A$, $S_1(a) \leq a$ (by $(S4)$) and f is increasing, we have $f(a) = 1$.

(6) Follows from the fact that in A, $a \leq S_{n-1}(a)$ (by $(S12)$).

(7) Follows from the fact that if $i \leq j$ then $S_i(a) \leq S_j(a)$ for every $a \in A$ (by $(S11)$).

(8) Assume first that $s_i(f_1)(a) = 1$. Then $f_1(S_i(a)) = 1$, and since $f_1(S_i(a)) \leq f_2(S_i(a))$ it follows that $s_i(f_2)(a) = f_2(S_i(a)) = 1$.

Assume now that $s_i(f_2)(a) = 1$, i.e. $f_2(S_i(a)) = 1$. We know that for every $a \in A$, $\neg S_i a \vee S_i a = 1$ (by $(S14)$). Therefore $f_1(\neg S_i a \vee S_i a) = f_1(\neg S_i a) \vee f_1(S_i a) = 1$. Therefore either $f_1(\neg S_i a) = 1$ or $f_1(S_i a) = 1$. Assume that $f_1(S_i a) = 0$. Then $f_1(\neg S_i a) = 1$ hence $f_2(\neg S_i a) = 1$, and since $f_2(S_i(a)) = 1$ it follows that $f_2(0) = 1$ which is false. Therefore, $f_1(S_i a) = 1$.

(9) If $s_i(f_1) \leq f_2$ then using (8) we have $s_i(f_2) \leq s_i(s_i(f_1)) = s_i(f_1) \leq f_2$.

(10) Assume that for some $i = 1, \ldots, n-1$ neither $f \leq s_i(f)$ nor $s_{i+1}(f) \leq f$. This means that $f(a) = 1$ and $s_i(f)(a) = f(S_i(a)) = 0$ for some $a \in A$, and $f(b) = 0$ and $s_{i+1}(f)(b) = f(S_{i+1}(b)) = 1$ for some $b \in A$. Therefore $f(a \wedge S_{i+1}(b)) = 1$ and $f(b \vee S_i(a)) = 0$. But this is impossible because for every $a, b \in A$, $a \wedge S_{i+1}(b) \leq b \vee S_i(a)$ (by $(S16)$). □

Lemma 5.13 *Let X be an arbitrary set and let $K = (X, \leq, g, s_1, \ldots, s_{n-1})$ be such that $\leq$ is a partial order on X and $g, s_1, \ldots, s_{n-1}$ are functions on X that satisfy the properties (1)-(10). Then K is a SHn-frame, i.e. satisfies the properties (K1)-(K12) from Definition 5.3.*

Proof: Since $\leq$ is a partial order, $(K1)$, $(K2)$ and $(K11)$ are true. $(K3)$ – $(K10)$ and $(K12)$ follow from the properties (1)-(10). □

It follows in particular that the dual space associated to every SHn-algebra by this procedure are in particular special SHn-frames in the sense of the definition given in [IO96], where the reflexive and transitive relation is a partial order[3].

Lemma 5.14 *Let $f : A_1 \to A_2$ be a morphism of SHn-algebras. Then $D(f) : D(A_2) \to D(A_1)$ is continuous, order-preserving and commutes with the operations $g, s_1, \ldots, s_{n-1}$. Moreover, $D(f)$ satisfies condition $(H2)$.*

Proof: The fact that $D(f)$ is continuous and order-preserving follows from the Priestley duality. It is easy to see that $D(f)(g(h)) = g(D(f)(h))$ and $D(f)(S_i(h)) = S_i(D(f)(h))$. The fact that $D(f)$ satisfies condition $(H2)$ follows from the extension of Priestley duality to Heyting algebras presented in Section 5.1.5. □

Conversely, let $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ be a space such that:
(1) $(X, \leq, \tau)$ is a Heyting space,
(2) $g, s_1, \ldots, s_{n-1}$ are continuous,

[3]However, the choice of more general Kripke-style structures in [IO96], where the relation R is only required to be reflexive and transitive, probably offers more advantages. We have in mind the Kripke models for intuitionistic logic. It is known that intuitionistic propositional logic is complete for (finite) Kripke models over trees. Some models are however needlessly complicated. By filtration (respectively by selective filtration) simpler models can be obtained, but this procedure does not preserve all the properties of models, in particular that of being a tree.

(3) $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ satisfies the conditions (1)-(10) listed above.

By Lemma 5.13 it follows that such a space is a SHn-frame, with $R = \leq$.

Let $E(X) = \{f : X \to \{0, 1\} \mid f$ continuous and order-preserving$\}$ (equivalently, $E(X)$ can be seen as the set of clopen order-filters of X). We know that $E(X)$ is a pseudocomplemented lattice.

Proposition 5.15 *On the lattice $E(X)$ the following operations can be defined:*

(1) $\vee : E(X) \times E(X) \to E(X)$ is defined by $h_1 \vee h_2(x) = max\{h_1(x), h_2(x)\}$. (In terms of clopen order-filters, $\vee$ is the union),

(2) $\wedge : E(X) \times E(X) \to E(X)$ is defined by $h_1 \wedge h_2(x) = min\{h_1(x), h_2(x)\}$. (In terms of clopen order-filters, $\wedge$ is the intersection),

(3) $\Rightarrow: E(X) \times E(X) \to E(X)$ is defined by $h_1 \Rightarrow h_2(x) = 1$ if and only if for every $y \geq x$, if $h_1(y) = 1$ then $h_2(y) = 1$. (In terms of clopen order-filters, $U_1 \Rightarrow U_2 = \{f \in X \mid f \leq g$ and $g \in U_1$ implies $g \in U_2\}$),

(4) $\sim: E(X) \to E(X)$ is defined for every continuous order-preserving $h : X \to \{0, 1\}$ by $\sim h : X \to \{0, 1\}$, $\sim h(x) = 1$ if and only if $g(h(x)) = 0$. (In terms of clopen order-filters: $\sim U = X \backslash g^{-1}(U)$).

(5) For every $i = 1, \ldots, n-1$, $S_i : E(X) \to E(X)$ is defined for every continuous order-preserving $h : X \to \{0, 1\}$ by $S_i(h) = h \circ s_i$. (In terms of clopen order-filters: $S_i(U) = s_i^{-1}(U)$).

The algebra $(E(X), \emptyset, X, \cap, \cup, \Rightarrow, \neg, \sim, S_1, \ldots, S_{n-1})$ is a SHn-algebra.

Proof: From the extension of the Priestley duality to Heyting algebras presennted in Section 5.1.5, it follows that $(E(X), \emptyset, X, \cap, \cup, \Rightarrow, \neg)$ is a Heyting algebra. We will show that the axioms of a SHn-algebra are satisfied.

De Morgan Laws:

$\sim\sim h = h$ *Proof*: For every $h \in E(X)$, $\sim\sim h(x) = 1$ if and only if $\sim h(g(x)) = 0$ if and only if $h(g(g(x))) = 1$. Since $g(g(x)) = x$ it follows that for every $x \in X$, $\sim\sim h(x) = h(x)$.

$\sim (h_1 \vee h_2) = \sim h_1 \wedge \sim h_2$ *Proof*: For $h_1, h_2 \in E(X)$, $\sim (h_1 \vee h_2)(x) = 1$ if and only if $(h_1 \vee h_2)(g(x)) = 0$ if and only if $max\{h_1(g(x)), h_2(g(x))\} = 0$ if and only if $h_1(g(x)) = h_2(g(x)) = 0$ if and only if $\sim h_1(x) = \sim h_2(x) = 1$ if and only if $\min\{\sim h_1(x), \sim h_2(x)\} = 1$.

SHn **Laws:**

$S_i(h_1 \wedge h_2) = S_i(h_1) \wedge S_i(h_2)$ *Proof*: $S_i(h_1 \wedge h_2)(x) = (h_1 \wedge h_2)(s_i(x)) =$
$= min\{h_1(s_i(x)), h_2(s_i(x))\} = S_i(h_1) \wedge S_i(h_2)(x)$.

$S_i(h_1 \Rightarrow h_2) = \bigwedge_{k=i}^{n-1} S_k(h_1) \Rightarrow S_k(h_2)$ *Proof*: By the definition of the relative pseudocomplement it follows that for every $k \geq i$, $S_i(a \Rightarrow b) \leq [S_k(a) \Rightarrow S_k(b)]$. Indeed, $S_k(a \Rightarrow b) \leq (S_k(a) \Rightarrow S_k(b))$ because $S_k(a \Rightarrow b) \wedge S_k(a) \leq S_k(b)$. We therefore have $S_i(a \Rightarrow b) \leq S_k(a \Rightarrow b) \leq [S_k(a) \Rightarrow S_k(b)]$ for every k.

Conversely: Let x be such that $(\bigwedge_{k=i}^{n-1} S_k(h_1) \Rightarrow S_k(h_2))(x) = 1$. We know that $(\bigwedge_{k=i}^{n-1} S_k(h_1) \Rightarrow S_k(h_2))(x) = 1$ if and only if for every $k \geq i$, $[S_k(h_1) \Rightarrow S_k(h_2)](x) = 1$ (i.e. if and only if for every $k \geq i$ and every $y \geq x$, if $h_1(s_k(y)) = 1$ then $h_2(s_k(y)) = 1$). Assume that $S_i(h_1 \Rightarrow h_2)(x) = (h_1 \Rightarrow h_2)(s_i(x)) = 0$. Then for some $z \geq s_i(x)$ we have $h_1(z) = 1$ and $h_2(z) = 0$. The space X is in particular a SHn-frame, hence from Lemma 5.3, if $z \geq s_i(x)$ then there exist $k \geq i$ and $z' \geq x$ such that $z = s_k(z')$. Hence, for some $k \geq i$ and some $z' \geq x$ we have $h_1(s_k(z')) = 1$ and $h_2(s_k(z')) = 0$. Contradiction.

$S_i(S_j(h)) = S_j(h)$ *Proof*: $S_i(S_j(h))(x) = S_j(h)(s_i(x)) = h(s_j(s_i(x))) = h(s_j(x))$ (we used property (4) of SHn-spaces).

$S_1(h) \vee h = h$ *Proof*: $(S_1(h) \vee h)(x) = max\{S_1(h)(x), h(x)\} = max\{h(s_1(x)), h(x)\} = h(x)$ (since by property (5) of SHn-spaces $s_1(x) \leq x$ and h is order-preserving).

$S_i(\sim h) = \sim S_{n-i}(h)$ *Proof*: $S_i(\sim h)(x) = 1$ if and only if $\sim h(s_i(x)) = 1$, i.e. if and only if $h(g(s_i(x))) = 0$. By property (2) of SHn-spaces, $g(s_i)(x) = s_{n-i}(g(x))$. Hence, $S_i(\sim h)(x) = 1$ if and only if $h(s_{n-i}(g(x))) = 0$, i.e. if and only if $\sim S_{n-i}h(x) = 1$.

$S_1(h) \vee \neg S_1(h) = 1$ *Proof*: Assume that this is not true, i.e. that $(S_1(h) \vee \neg S_1(h))(x) = 0$ for some x. It follows that $0 = (S_1(h) \vee \neg S_1(h))(x) = max\{S_1(h(x)), \neg S_1(h)(x)\}$, hence $S_1(h(x)) = \neg S_1(h)(x) = 0$. We know that $\neg S_1(h)(x) = 0$ if and only if there is some $y \geq x$ such that $S_1(h)(y) = 1$. But, from property (8) of SHn-spaces, if $y \geq x$, then $S_1(h)(y) \leq S_1(h)(x)$. Contradiction.

□

Definition 5.9 *The category* SHnSp *of SHn*-spaces *has as*

Objects: *spaces $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ such that:*
(1) $(X, \leq, \tau)$ is a Heyting space,
(2) $g, s_1, \ldots, s_{n-1}$ are continuous,
(3) $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ satisfies the conditions (1)-(10) listed above.

Morphisms: *continuous order-preserving mappings that*
(1) satisfy the condition $(H2')$ and
(2) preserve the operations $g, s_1, \ldots, s_{n-1}$.

Lemma 5.16 *Let $f : X_1 \to X_2$ be a morphism of SHn-spaces. Then $E(f) : E(X_2) \to E(X_1)$ defined by $E(f)(h) = h \circ f$ is a morphism of SHn-algebras.*

Proof: The fact that $E(f)$ is a morphism of Heyting algebras follows from the extension of the Priestley duality to Heyting algebras presented in Section 5.1.5. It only remains to show that it commutes with the operators $\{\sim, S_1, \ldots, S_{n-1}\}$.

We know that $E(f)(\sim h)(x) = 1$ if and only if $\sim h(f(x)) = 0$, i.e. if and only if $h(g(f(x))) = 0$. Since $h(g(f(x))) = h(f(g(x))) = E(f)(h)(g(x))$, it follows that $E(f)(\sim h) = \sim E(f)(h)$.

For every $i = 1, \ldots, n-1$, $E(f)(S_i h) = (S_i \circ h) \circ f = S_i \circ (h \circ f) = S_i E(f)(h)$. □

Corollary 5.17 *The Priestley duality induces a dual equivalence between the category of SHn-algebras and the category of SHn-spaces.*

5.1.6 Link Between Algebraic Semantics and Kripke-style Semantics

We begin by making some remarks concerning morphisms of Kripke frames. Then we compare the algebraic models of SHn-logic and the Kripke-style models.

In Section 5.1.5 we defined a category SHnSp, having as objects SHn-spaces and as morphisms continuous order-preserving mappings that satisfy condition $(H2)$ and preserve the operations $g, s_1, \ldots, s_{n-1}$. In Section 5.1.5 we noted that condition $(H2)$ is equivalent with the "boundness" condition $(H2')$:

$(H2')$ If $\phi(z) \leq y$ then there exists x with $(z \leq x$ and $\phi(x) = y)$.

This indicates a way of defining morphisms between SHn-frames. Given two Kripke frames $K_1 = (W_1, R_1, \{s_i^1 \mid i = 1, \ldots, n-1\}, g^1)$ and $K_2 = (W_2, R_2, \{s_i^2 \mid i = 1, \ldots, n-1\}, g^2)$, a morphism from K_1 to K_2, $f : K_1 \to K_2$ is a map $f : W_1 \to W_2$ such that:

- If $(x_1, y_1) \in R_1$ then $(f(x_1), f(y_1)) \in R_2$,
- If $(f(x_1), y_2) \in R_2$ then there exists a y_1 with $(x_1, y_1) \in R_1$ and $f(y_1) = y_2$.

Note the similarity between this notion of morphism and that of "p-morphism" or "bounded morphism" from modal logic. More considerations in this direction will be made in Section 5.3.2.

We now say some words on the link between the satisfiability notion in the algebraic models (SHn-algebras) and the relational models (Kripke frames) of SHn-logic. The duality theorem stated in Section 5.1.5 together with the soundness and completeness of SHn-logic with respect to the algebraic semantics given in [Itu83], as well as with respect to the Kripke-style semantics given in [IO96], suggests that there might be a link between satisfiability in the two different types of models. In what follows we establish some results in this direction.

Let ϕ be a formula in the language of the SHn-propositional logic with variables Var.

For a given Kripke frame $K = (X, R, g, s_1, \ldots, s_{n-1})$, we will denote the family of all hereditary subsets of X (with respect to the relation R) by $\mathcal{H}(X)$.

Below, we present in parallel the notions of satisfiability in algebraic models and in Kripke models.

Algebraic Models	**Kripke Models**
A: SHn-algebra	$K = (X, R, g, s_1, \dots, s_{n-1})$ SHn-frame
for $f : \mathsf{Var} \to A$; $A \overset{a}{\models}_f \phi$ iff $\overline{f}(\phi) = 1$	for $v : \mathsf{Var} \to \mathcal{H}(X)$ $K \overset{r}{\models}_v \phi$ iff $\{x \mid K \overset{r}{\models}_{v,x} \phi\} = X$
$A \overset{a}{\models} \phi$ iff for every $f : \mathsf{Var} \to A$, $A \overset{a}{\models}_f \phi$	$K \overset{r}{\models} \phi$ iff for every $v : \mathsf{Var} \to \mathcal{H}(X)$, $K \overset{r}{\models}_v \phi$

Lemma 5.18 (cf. [IO96]) *Let $K = (X, R, g, s_1, \dots, s_{n-1})$ be a Kripke frame, and let $\mathcal{H}(X)$ be the family of all hereditary subsets of X (with respect to the relation R). Let $m : \mathsf{Var} \to \mathcal{H}(X)$ be a meaning function such that for every $p \in \mathsf{Var}$, $m(p)$ is a hereditary set. Then the extension of m to formulae has also as values hereditary sets.*

We now consider topological Kripke-style models corresponding to the SHn-spaces:

Lemma 5.19 *Let $X = (X, \leq, \tau, g, s_1, \dots, s_{n-1})$ be a SHn-space and let $m : \mathsf{Var} \to \mathcal{P}(X)$ be a meaning function such that for every $p \in \mathsf{Var}$, $m(p)$ is a clopen order-filter of X. Then the extension of m to formulae has also as values clopen order-filters of X.*

Proof: The extension of m to formulae is $\overline{m}(\phi) = \{x \in X \mid X \overset{r}{\models}_{m,x} \phi\}$. Obviously, $\overline{m}(0) = \emptyset$ and $\overline{m}(1) = X$. We show by structural induction that for every ϕ, $\overline{m}(\phi)$ is a clopen order-filter.

The property is obvious for every $\phi \in \mathsf{Var}$ and for 0 and 1. Let ϕ be a formula. We assume that the property is true for all subformulae of ϕ and show that it is also true for ϕ. We distinguish the following cases:

Case 1: $\phi = \phi_1 \wedge \phi_2$.

By the definition of the satisfiability relation we have $x \in \overline{m}(\phi_1 \wedge \phi_2)$ if and only if $x \in \overline{m}(\phi_1)$ and $x \in \overline{m}(\phi_2)$. Hence, $\overline{m}(\phi_1 \wedge \phi_2) = \overline{m}(\phi_1) \wedge \overline{m}(\phi_2)$. Since by the induction hypothesis $\overline{m}(\phi_1)$ and $\overline{m}(\phi_2)$ are both clopen order-filters it follows that $\overline{m}(\phi_1 \wedge \phi_2)$ is a clopen order-filter.

Case 2: $\phi = \phi_1 \vee \phi_2$.

Analogously to Case 1 we can prove that $\overline{m}(\phi_1 \vee \phi_2) = \overline{m}(\phi_1) \vee \overline{m}(\phi_2)$. Since by the induction hypothesis $\overline{m}(\phi_1)$ and $\overline{m}(\phi_2)$ are both clopen order-filters it follows that $\overline{m}(\phi_1 \vee \phi_2)$ is a clopen order-filter.

Case 3: $\phi = \phi_1 \Rightarrow \phi_2$.

By the definition of the satisfiability relation we know that $x \in \overline{m}(\phi_1 \Rightarrow \phi_2)$ if and only if for all y, if $y \geq x$ and $y \in \overline{m}(\phi_1)$ then $y \in \overline{m}(\phi_2)$. Hence, $\overline{m}(\phi_1 \Rightarrow \phi_2) = \{x \in X \mid \text{ for all } y, \text{ if } y \geq x \text{ and } y \in \overline{m}(\phi_1) \text{ then } y \in \overline{m}(\phi_2)\} = \overline{m}(\phi_1) \Rightarrow \overline{m}(\phi_2)$. Since by the induction hypothesis $\overline{m}(\phi_1), \overline{m}(\phi_2)$ are clopen order-filters, it follows that $\overline{m}(\phi_1) \Rightarrow \overline{m}(\phi_2)$ is a clopen order-filter (see also [Gol89]).

Case 4: $\phi = \neg\phi_1$. Follows from Case 3.

Case 5: $\phi = S_i\phi_1$.

We know that $x \in \overline{m}(S_i\phi_1)$ if and only if $s_i(x) \in \overline{m}(\phi_1)$, i.e. if and only if $x \in s_i^{-1}(\overline{m}(\phi_1))$. Hence, $\overline{m}(S_i\phi_1) = s_i^{-1}(\overline{m}(\phi_1))$. Since $\overline{m}(\phi_1)$ is a clopen order-filter, and s_i is continuous and order-preserving, it follows that also $\overline{m}(S_i\phi_1)$ is a clopen order-filter.

Case 6: $\phi = \sim \phi_1$.

We have $x \in \overline{m}(\sim \phi_1)$ if and only if $g(x) \notin \overline{m}(\phi_1)$, i.e. if and only if $x \in X \backslash g^{-1}(\overline{m}(\phi_1)) = \sim \overline{m}(\phi_1)$

By the induction hypothesis $\overline{m}(\phi_1)$ is a clopen order-filter, hence $g^{-1}(\overline{m}(\phi_1))$ is a clopen order-ideal and therefore $X \backslash g^{-1}(\overline{m}(\phi_1))$ is a clopen order-filter. □

Lemma 5.19 is a simple consequence of the fact that the algebra of clopen order-filters of X, $\mathsf{ClopenOF}(X)$, is a SHn-algebra. In this case, from the universality property of the free algebra $\mathsf{Fma}(\mathsf{Var})$ it follows that every $m : \mathsf{Var} \to \mathsf{ClopenOF}(X)$ extends in a unique way to a homomorphism of SHn-algebras $\overline{m} : \mathsf{Fma}(\mathsf{Var}) \to \mathsf{ClopenOF}(X)$.

Lemma 5.20 *Let A be a SHn-algebra and $D(A)$ its dual. Let $\eta_A : A \to \mathsf{ClopenOF}(D(A))$ be the canonical isomorphism given by the Priestley representation.*

(1) Let $f : \mathsf{Var} \to A$. Then $A \overset{a}{\models}_f \phi$ if and only if $D(A) \overset{r}{\models}_{m_f} \phi$, where $m_f : \mathsf{Var} \to \mathcal{P}(D(A))$ is defined by $m_f(p) = \eta_A(f(p)) = \{h \in D(A) \mid h(f(p)) = 1\}$.

(2) Let $m : \mathsf{Var} \to \mathcal{P}(D(A))$ be such that for every $p \in \mathsf{Var}$, $m(p)$ is a clopen order-filter, and let $f_m : \mathsf{Var} \to A$ be defined by $f_m(p) = \eta_A^{-1}(m(p))$. Then for every $p \in \mathsf{Var}$, $D(A) \overset{r}{\models}_m \phi$ if and only if $A \overset{a}{\models}_{f_m} \phi$.

Proof: (1) Note first that for every $f : \mathsf{Var} \to A$, the associated meaning function $m_f = \eta_A \circ f$ extends to a morphism $\overline{m}_f$ of SHn-algebras from $\mathsf{Fma}(\mathsf{Var})$ to the set of clopen order-filters of $D(A)$, $\mathsf{ClopenOF}(D(A))$, defined by $\overline{m}_f(\phi) = \{h \in D(A) \mid D(A) \overset{r}{\models}_{m_f,h} \phi\}$. From the fact that $\overline{f} : \mathsf{Fma}(\mathsf{Var}) \to A$ is the unique morphism that extends f and that A and $\mathsf{ClopenOF}(D(A))$ are isomorphic, it follows that $\overline{m}_f(\phi) = \overline{\eta_A \circ f}(\phi) = \eta_A \circ \overline{f}(\phi) = \{h \in D(A) \mid h(\overline{f}(\phi)) = 1\}$ (or, in terms of prime filters: $\overline{m}_f(\phi) = \{F \mid \overline{f}(\phi) \in F\}$). We know that $A \overset{a}{\models}_f \phi$ if and only if $\overline{f}(\phi) = 1$ and $D(A) \overset{r}{\models}_{m_f} \phi$ if and only if $\overline{m}_f(\phi) = D(A)$, and from the form of $\overline{m}_f(\phi)$ the equivalence follows easily.

(2) It is easy to see that the unique morphism $\overline{f}_m : \mathsf{Fma}(\mathsf{Var}) \to A$ which extends f_m is $\eta_A^{-1} \circ \overline{m}$. Therefore, $\overline{f}_m(\phi) = 1$ if and only if $\eta_A^{-1} \circ \overline{m}(\phi) = 1$ if and only if $\overline{m}(\phi) = D(A)$. □

Corollary 5.21 *Let A be a SHn-algebra and $D(A)$ its dual, and let $\phi \in \mathsf{Fma}(\mathsf{Var})$ be a formula of the SHn-logic. Then:*

(1) If $D(A) \overset{r}{\models} \phi$ then $A \overset{a}{\models} \phi$,

(2) If A is finite and $A \overset{a}{\models} \phi$, then $D(A) \overset{r}{\models} \phi$.

Proof: (1) Assume that $D(A) \overset{r}{\models} \phi$. We show that $A \overset{a}{\models} \phi$.

Let $f : \mathsf{Var} \to A$ be an assignment of truth values. Let $\eta_A : A \to \mathsf{ClopenOF}(D(A))$ be the canonical isomorphism between A and the set of clopen order-filters of $D(A)$. Let $m : \mathsf{Var} \to \mathsf{ClopenOF}(D(A))$ be defined for every $p \in \mathsf{Var}$ by $m(p) = \eta_A(f(p))$. m has as values order-filters of $D(A)$, hence is a meaning function. From Lemma 5.19, the extension $\overline{m}$ of m to formulae has also as values clopen order-filters of $D(A)$. For every formula ϕ, $\overline{m}(\phi) = \eta_A(\overline{f}(\phi))$. Since $D(A) \overset{r}{\models} \phi$ it follows that $\overline{m}(\phi) = D(A)$. Therefore, by the definition of η_A it follows that $\overline{f}(\phi) = 1$.

(2) Assume that A is finite, and $A \overset{a}{\models} \phi$. We show that $D(A) \overset{r}{\models} \phi$.

Let $m : \mathsf{Var} \to \mathcal{P}(D(A))$ be a meaning function that has as values order-filters of $D(A)$. Since A is finite, the topology on $D(A)$ is discrete, hence the set of clopen order-filters coincides in this case with the set of order-filters. The extension $\overline{m}$ of m to formulae has also as values (clopen) order-filters of $D(A)$. Let $\eta_A : A \to \mathsf{ClopenOF}(D(A))$ be the canonical isomorphism between A and the set of order-filters of $D(A)$. Let $f : \mathsf{Var} \to A$ be defined by $f = \eta_A^{-1} \circ m$. Since $A \overset{a}{\models} \phi$, it follows that $\overline{f}(\phi) = 1$. It is easy to see that $\overline{f} = \overline{(\eta_A^{-1} \circ m)} = \eta_A^{-1} \circ \overline{m}$. Therefore, it follows that $\overline{m}(\phi) = D(A)$. □

Note that, if A is infinite, the fact that $A \overset{a}{\models} \phi$ does not imply in general that $D(A) \overset{r}{\models} \phi$.

Proposition 5.22 *Let ϕ be a formula in the language of the SHn-propositional logic with variables* Var. *The following are equivalent:*

(1) $\vdash_{SHn} \phi$,

(2) For every relational model K, $K \overset{r}{\models} \phi$,

(3) $D(S_{n^2}) \overset{r}{\models} \phi$,

(4) $S_{n^2} \overset{a}{\models} \phi$,

(5) For every SHn-algebra A, $A \overset{a}{\models} \phi$.

Proof: (1) $\Rightarrow$ (2) follows from [IO96], Proposition 5.1; (2) $\Rightarrow$ (3) is immediate because $D(S_{n^2})$ is a relational model; (3) $\Rightarrow$ (4) follows from Corollary 5.21; (4) $\Rightarrow$ (5) follows from the fact that the variety of SHn-algebras is generated by S_{n^2}; (5) $\Rightarrow$ (1) follows from the completeness of SHn-logic with respect to SHn-algebras. □

Corollary 5.23 *Let ϕ be a formula in the language of the SHn-propositional logic with variables* Var. *Then $\vdash_{SHn} \phi$ if and only if $D(S_{n^2}) \overset{r}{\models} \phi$.*

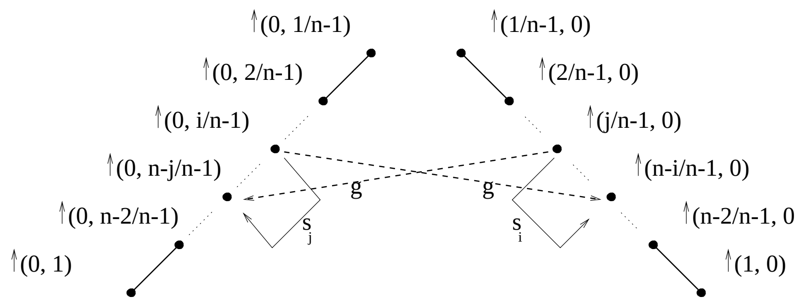


Figure 5.4: The Priestley Dual of S_{n^2}

Remark 5.24 The Priestley dual of S_{n^2}, $D(S_{n^2})$ is isomorphic to the ordered set of the join-irreducible elements of S_{n^2}, namely with the ordered set $D(S_{n^2}) = \{(0, \frac{i}{n-1}) \mid i = 1, \dots, n-1\} \cup \{(\frac{i}{n-1}, 0) \mid i = 1, \dots, n-1\}$ with the order defined pointwise (these elements correspond to the prime filters $\uparrow (0, \frac{i}{n-1}), i = 1, \dots, n-1$ resp. $\uparrow (\frac{i}{n-1}, 0), i = 1, \dots, n-1$). The additional operations $g, s_1, \dots, s_{n-1}$ are defined by:

(1a) $g(\uparrow (0, \frac{i}{n-1})) = \{(x,y) \in S_{n^2} \mid\sim (x,y) \not\in\uparrow (0, \frac{i}{n-1})\} = \{(x,y) \in S_{n^2} \mid 1 - x < \frac{i}{n-1}\} = \{(x,y) \in S_{n^2} \mid x \geq \frac{n-i}{n-1}\} = \uparrow (\frac{n-i}{n-1}, 0)$;

(1b) $g(\uparrow (\frac{i}{n-1}, 0)) = \{(x,y) \in S_{n^2} \mid\sim (x,y) \not\in\uparrow (\frac{i}{n-1}, 0)\} = \{(x,y) \in S_{n^2} \mid 1 - y < \frac{i}{n-1}\} = \{(x,y) \in S_{n^2} \mid y \geq \frac{n-i}{n-1}\} = \uparrow (0, \frac{n-i}{n-1})$;

(2) $s_j(\uparrow (0, \frac{i}{n-1})) = \{(\frac{k}{n-1}, \frac{l}{n-1}) \in S_{n^2} \mid S_j(\frac{k}{n-1}, \frac{l}{n-1}) \geq (0, \frac{i}{n-1})\}$.

$$\text{Since } S_j(\tfrac{k}{n-1}, \tfrac{l}{n-1}) = \begin{cases} (1,1) & \text{if } j+k \geq n, \quad \text{if } j+l \geq n \\ (1,0) & \text{if } j+k \geq n, \quad \text{if } j+l < n \\ (0,1) & \text{if } j+k < n, \quad \text{if } j+l \geq n \\ (0,0) & \text{if } j+k < n, \quad \text{if } j+l < n \end{cases}$$

it is easy to see that $S_j(\frac{k}{n-1}, \frac{l}{n-1}) \geq (0, \frac{i}{n-1})$ if and only if $j + l \geq n$, i.e. if and only if $l \geq n - j$. Thus, $s_j(\uparrow (0, \frac{i}{n-1})) = \uparrow (0, \frac{n-j}{n-1})$. Similarly, $s_j(\uparrow (\frac{i}{n-1}, 0)) = \uparrow (\frac{n-j}{n-1}, 0)$.

5.2 Automated Theorem Proving in SHn-logics

In what follows we present an approach to automated theorem proving that uses the Priestley dual of the algebra of truth values, in cases such a duality holds.

The general procedure – that can be applied in cases in which the logic is sound and complete with respect to a variety $\mathcal{V}$ of algebras that have an underlying distributive lattice structure, with the property that $\mathcal{V}$ is generated by a finite number of finite algebras, and such that the Priestley duality extends to a dual equivalence between $\mathcal{V}$ (seen as a category) and a corresponding category of relational models – will be presented in section 5.3.1.

We will first illustrate the ideas for the case of SHn-logics.

Our approach is in some sense inspired by the approach presented in [Häh94, Häh96b], but is different in that we exploit the fact that the algebraic model has a dual with less elements, and use it in order to improve efficiency (e.g. the number of clauses that are generated).

The main idea of our approach is to use signed literals, where the signs are "possible worlds", i.e. elements of $D(S_{n^2})$ (corresponding to prime filters of truth values) instead of truth values (as done in [BF92, BF95]) or arbitrary sets of truth values (as done in [Häh94, Häh96b]). The idea of using "valuations in $\{0, 1\}$" instead of values is not new. It appears already for instance in [Sco73] for the case of Lukasiewicz logics.

In what follows, for a given meaning function $m : \mathsf{Var} \to \mathcal{O}D(S_{n^2})$ we use the following notation:

$\boxed{x}\ \phi^t$ means "ϕ is true at x" in the interpretation m (i.e. $D(S_{n^2}) \overset{r}{\models}_{m,x} \phi$)

$\boxed{x}\ \phi^f$ means "ϕ is false at x" in the interpretation m (i.e. $D(S_{n^2}) \overset{r}{\not\models}_{m,x} \phi$)

where x is a "possible world", i.e. an element of $D(S_{n^2})$.

We point out that expressions of the type $\boxed{x}\ \phi^t$ respectively $\boxed{x}\ \phi^f$ are very similar to the "positive and negative regular formulae" of the form $\boxed{\geq i}\ \phi$ respectively $\boxed{\leq i}\ \phi$ introduced in [Häh96b] for many-valued logics where the set of values is $\{0, \frac{1}{n-1}, \dots, \frac{n-2}{n-1}, 1\}$.

The only difference is that in [Häh96b] totally ordered sets are considered, whereas we consider labelling over duals of finite distributive lattices. In the particular case of totally ordered lattices of truth values, Hähnle's notions of positive and negative literal are recovered: $\boxed{\geq i}\ \phi$ corresponds to $\boxed{x}\ \phi^t$ and $\boxed{\leq i - 1}\ \phi$ to $\boxed{x}\ \phi^f$, where $x = \uparrow i$, which justifies the terminology "literal with positive polarity" for literals with sign $\boxed{\geq i}$, resp. "literal with negative polarity" for literals with sign $\boxed{\leq i}$, as used in [Häh96b].

In order to define an automated theorem proving procedure, we need:

1) An efficient procedure for translation into clause form,

2) A resolution procedure (which has to be proved sound and complete).

In [Häh96b], Hähnle defines a procedure for transforming a signed formula in a set of signed clauses, and then shows how the signed clauses can be transformed into regular clauses – to which resolution is applied in order to show that they are unsatisfiable.

In what follows we sketch a procedure similar to the one described [Häh96b], by which, given a SHn-formula ϕ, a set Φ of clauses can be obtained such that Φ is unsatisfiable if and only if the given formula ϕ is a theorem.

The advantage of the method presented here is that, instead of using the whole set of values in S_{n^2}, as the general approach would require, it uses the space dual to S_{n^2} which has only $2(n-1)$ elements. In Section 5.3.1 we will show that this method can be extended to more general many-valued logics, sound and complete for classes of algebras for which a Priestley-type duality is known, and whose variety of models is generated by finitely many finite algebras.

5.2.1 An Efficient Translation into Clause Form

As noticed in [Häh94], there are several main obstacles when clausal normal forms are to be used in a generalized context:

(1) Normal forms can become exponentially long with respect to the length of the input formula when "naive" algorithms are used,

(2) The normalized input has no resemblance with the original formula,

Another obstacle can be the fact that many non-classical logics do not have "internal" normal forms.

These problems can be solved by using a *structure-preserving* clause form translation.

The procedure that we present in what follows has been inspired by [Häh94], which deals with short conjunctive normal forms for finitely valued logics; it presents and discusses structure-preserving translations to clause form. The central idea behind structure-preserving clause form translations is to introduce additional atoms (resp. predicate letters in the case of first-order logic), which serve as abbreviations for subformulae of the input. It remains to translate the formulae that represent the definitions of the new literals. In classical logic this is a classical translation procedure, called "translation to definitional form" or "structure-preserving translation" in the literature (see e.g. [Ede92] for a detailed description of this translation strategy).

We begin with some definitions. For the sake of simplicity, we only present the propositional case here.

Let Var be a countably infinite set of variables; in what follows, all the variables belong to Var.

Definition 5.10 *Let $x \in D(S_{n^2})$ be a "possible world" and p be an atom (in the propositional case, a propositional variable). Then* $\boxed{x}\, p^t$ *is a* positive literal (with sign $\boxed{x}$) *and* $\boxed{x}\, p^f$ *is a* negative literal (with sign $\boxed{x}$). *A set of (positive or negative) signed literals is called a* (signed) clause. *A* formula in signed conjunctive normal form (CNF) *is a finite set of (signed) clauses. (In the first-order case we require that the clauses in a formula have disjoint variables.).*

Definition 5.11

(1) A propositional positive literal $\boxed{x}\, p^t$ is satisfiable if for some meaning function $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$, p is true in m at x.

(2) A propositional negative literal $\boxed{x}\, p^f$ is satisfiable if for some meaning function $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$, p is false in m at x.

(3) A propositional signed clause is satisfiable if and only if at least one of its literals is satisfiable.

(4) A signed formula Φ is satisfiable if and only if all clauses in Φ are simultaneously satisfiable by the same interpretation.

Note that if $\boxed{y}\, p^t$ at m and $y \leq x$ then $\boxed{x}\, p^t$ at m.

Let ϕ be a propositional SHn-formula, $\phi \in \mathsf{Fma}(\mathsf{Var})$.

Lemma 5.25

(1) ϕ is a SHn-theorem if and only if there is no valuation m such that ϕ is false at $\uparrow (0,1)$ in m or ϕ is false at $\uparrow (1,0)$ in m.

(2) ϕ is not a SHn-theorem if and only if for at least one valuation m ϕ is false at $\uparrow (0,1)$ in m or ϕ is false at $\uparrow (1,0)$ in m.

Proof: (1) We know that ϕ is a theorem if and only if for every valuation $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$ and every $x \in D(S_{n^2})$, ϕ is true at x in m.

Assume that ϕ is a SHn-theorem. Then there is no valuation m such that ϕ is false at $\uparrow (0,1)$ in m or ϕ is false at $\uparrow (1,0)$ in m. Assume now that there is no valuation m' such that ϕ is false at $\uparrow (0,1)$ in m' or ϕ is false at $\uparrow (1,0)$ in m'. Let m be an arbitrary valuation $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$, and let $x \in D(S_{n^2})$. Because of the form of $D(S_{n^2})$ it follows that $x \geq \uparrow (0,1)$ or $x \geq \uparrow (1,0)$. Since ϕ is true at both $\uparrow (0,1)$ and $\uparrow (1,0)$, it follows that ϕ is true at x.

(2) follows immediately from (1). □

For every formula ϕ we can introduce a new propositional variable p_ϕ.

Lemma 5.26 *The formula ϕ is a theorem if and only if there is no valuation m such that $\overline{m}(\phi) = m(p_\phi)$ and* $\boxed{\uparrow (0,1)}\, p^f_\phi$ *or* $\boxed{\uparrow (1,0)}\, p^f_\phi$ *at m.*

Definition 5.12 *Let ϕ_1, ϕ_2 be two formulae, and let $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$ be a valuation. We say that ϕ_1 and ϕ_2 are equivalent in m (denoted $\phi_1 \equiv_m \phi_2$) if they are true at the same states, i.e. if $\overline{m}(\phi_1) = \overline{m}(\phi_2)$.*

We say that $\phi_1 \equiv \phi_2$ is satisfiable if there is a valuation m such that $\phi_1 \equiv_m \phi_2$.

Lemma 5.27 *The relation $\equiv$ has the following properties:*

(1) Let m be a valuation, and ϕ_1, ϕ_2 be formulae such that $\phi_1 \equiv_m \phi_2$. Then for every $x \in D(S_{n^2})$, $(\boxed{x}\ \phi_1^f \vee \boxed{x}\ \phi_2^t) \wedge (\boxed{x}\ \phi_1^t \vee \boxed{x}\ \phi_2^f)$ at m.

(2) $\phi_1 \equiv \phi_2$ is satisfiable if and only if there exists a valuation m such that for all $x \in D(S_{n^2})$, $(\boxed{x}\ \phi_1^f \vee \boxed{x}\ \phi_2^t) \wedge (\boxed{x}\ \phi_1^t \vee \boxed{x}\ \phi_2^f)$ at m.

Proof: (1) Let m be a valuation, and ϕ_1, ϕ_2 be formulae such that $\phi_1 \equiv_m \phi_2$. Then for every $x \in D(S_{n^2})$, $\boxed{x}\ \phi_1^t$ at m if and only if $\boxed{x}\ \phi_2^t$ at m, and $\boxed{x}\ \phi_1^f$ at m if and only if $\boxed{x}\ \phi_2^f$ at m.

This is equivalent to saying that for every $x \in D(S_{n^2})$ ϕ_1 is true at x or ϕ_2 is false at x, and ϕ_1 is false at x or ϕ_2 is true at x at m, i.e. $(\boxed{x}\ \phi_1^f \vee \boxed{x}\ \phi_2^t) \wedge (\boxed{x}\ \phi_1^t \vee \boxed{x}\ \phi_2^f)$ at m.

(2) Follows immediately from (1). □

We can therefore reduce the task of proving that a formula ϕ is a SHn-theorem to the task of proving that for no valuation $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$ we have $\boxed{\uparrow (0,1)}\ p_\phi^f$ or $\boxed{\uparrow (1,0)}\ p_\phi^f$ at m and $p_\phi \equiv_m \phi$.

By Lemma 5.27, $p_\phi \equiv_m \phi$ if and only if for every $x \in D(S_{n^2})$, $(\boxed{x}\ \phi^f \vee \boxed{x}\ p_\phi^t) \wedge (\boxed{x}\ \phi^t \vee \boxed{x}\ p_\phi^f)$ at m.

Corollary 5.28 *A formula ϕ is a SHn-theorem if and only if there is no valuation m such that*

$$\begin{cases} (1) & \boxed{\uparrow (0,1)}\ p_\phi^f \vee \boxed{\uparrow (1,0)}\ p_\phi^f, & \text{at } m, \\ (2_x) & (\boxed{x}\ \phi^f \vee \boxed{x}\ p_\phi^t) \wedge (\boxed{x}\ \phi^t \vee \boxed{x}\ p_\phi^f) & \text{at } m. \\ & \text{for every } x \in D(S_{n^2}) & \end{cases}$$

We did not yet obtain a set of signed clauses that is unsatisfiable if and only if ϕ is a theorem. It can be seen that in (2_x), expressions of the form $\boxed{x}\ \phi^t$, and $\boxed{x}\ \phi^f$ still occur. We will show how we can recursively eliminate these expressions.

Lemma 5.29 *Let m be a valuation, and let $\diamond$ be a binary and ∇ a unary operator. Then:*

(1) Every formula $\psi = \psi_1 \diamond \psi_2$ is equivalent in m to a formula of the form $p_{\psi_1} \diamond p_{\psi_2}$ where $p_{\psi_i} \equiv_m \psi_i$ for $i = 1, 2$;

(2) *Every formula* $\psi = \nabla\psi_1$ *is equivalent in* m *to a formula of the form* ∇p_{ψ_1}, *where* $p_{\psi_1} \equiv_m \psi_1$.

Moreover, for every $x \in D(S_{n^2})$,

(3) $\boxed{x}\ (\psi_1 \diamond \psi_2)^t$ *at* m *if and only if* $\boxed{x}\ (p_{\psi_1} \diamond p_{\psi_2})^t$ *at* m, *where* $p_{\psi_1} \equiv_m \psi_1$ *and* $p_{\psi_2} \equiv_m \psi_2$;

(4) $\boxed{x}\ (\nabla\psi_1)^t$ *at* m *if and only if* $\boxed{x}\ (\nabla p_{\psi_1})^t$ *at* m, *where* $p_{\psi_1} \equiv_m \psi_1$.

In conclusion, ϕ is a SHn-theorem if and only if the following set of formulae (in classical logic) is unsatisfiable:

$$\begin{cases} \boxed{\uparrow(0,1)}\ p^f_\phi \vee \boxed{\uparrow(1,0)}\ p^f_\phi, & \\ p_\psi \equiv \psi & \text{for every subformula } \psi \text{ of } \phi. \end{cases}$$

Corollary 5.30 *ϕ is a SHn-theorem if and only if the following conjunction of formulae is unsatisfiable:*

$$(\ \boxed{\uparrow(0,1)}\ p^f_\phi \vee \boxed{\uparrow(1,0)}\ p^f_\phi\ )\wedge$$

$$\wedge \bigwedge_{x\in D(S_{n^2})} \bigwedge_{\substack{\psi \text{ subformula of } \phi \\ \psi=\psi_1\diamond\psi_2}} (\ \boxed{x}\ p^f_\psi \vee \boxed{x}\ (p_{\psi_1} \diamond p_{\psi_2})^t\ ) \wedge (\ \boxed{x}\ p^t_\psi \vee \boxed{x}\ (p_{\psi_1} \diamond p_{\psi_2})^f\ )\wedge$$

$$\wedge \bigwedge_{x\in D(S_{n^2})} \bigwedge_{\substack{\psi \text{ subformula of } \phi \\ \psi=\nabla\psi_1}} (\ \boxed{x}\ p^f_\psi \vee \boxed{x}\ (\nabla p_{\psi_1})^t\ ) \wedge (\ \boxed{x}\ p^t_\psi \vee \boxed{x}\ (\nabla p_{\psi_1})^f\ ).$$

Lemma 5.31 *For any given valuation* m *the following holds:*

(Disj t)	$\boxed{x}\ (p_1 \vee p_2)^t$	*iff*	$(\ \boxed{x}\ p^t_1) \vee (\ \boxed{x}\ p^t_2)$.
(Disj f)	$\boxed{x}\ (p_1 \vee p_2)^f$	*iff*	$(\ \boxed{x}\ p^f_1) \wedge (\ \boxed{x}\ p^f_2)$.
(Conj t)	$\boxed{x}\ (p_1 \wedge p_2)^t$	*iff*	$(\ \boxed{x}\ p^t_1) \wedge (\ \boxed{x}\ p^t_2)$.
(Conj f)	$\boxed{x}\ (p_1 \wedge p_2)^f$	*iff*	$(\ \boxed{x}\ p^f_1) \vee (\ \boxed{x}\ p^f_2)$.
$(S_j\ t)$	$\boxed{x}\ (S_j(p_1))^t$	*iff*	$\boxed{s_j(x)}\ p^t_1$.
$(S_j\ f)$	$\boxed{x}\ (S_j(p_1))^f$	*iff*	$\boxed{s_j(x)}\ p^f_1$.
$(\sim,\ t)$	$\boxed{x}\ (\sim(p_1))^t$	*iff*	$\boxed{g(x)}\ p^f_1$.
$(\sim,\ f)$	$\boxed{x}\ (\sim(p_1))^f$	*iff*	$\boxed{g(x)}\ p^t_1$.
$(\Rightarrow,\ t)$	$\boxed{x}\ (p_1 \Rightarrow p_2)^t$	*iff*	$\forall y \geq x,\ \boxed{y}\ p^f_1 \vee \boxed{y}\ p^t_2$.
$(\Rightarrow,\ f)$	$\boxed{x}\ (p_1 \Rightarrow p_2)^f$	*iff*	$\boxed{m}\ p^t_1$, *where* $m = \max\{y \mid y \geq x\}$, $\boxed{x}\ p^f_2$ *and* $\forall x_1, x_2 \geq x, x_1 \neq x_2, \boxed{x_1}\ p^t_1 \vee \boxed{x_2}\ p^f_2$.
$(\neg,\ t)$	$\boxed{x}\ (\neg p)^t$	*iff*	$\forall y \geq x,\ \boxed{y}\ p^f$.
$(\neg,\ f)$	$\boxed{x}\ (\neg p)^f$	*iff*	$\exists y \geq x$ *with* $\boxed{y}\ p^t$.

Proof:

(*Disj* t) : We know that $\boxed{x}\ (p_1 \vee p_2)^t$ if and only if $x \in \overline{m}(p_1 \vee p_2)$ if and only if $(x \in m(p_1)$ or $x \in m(p_2))$ if and only if $(\ \boxed{x}\ p^t_1) \vee (\ \boxed{x}\ p^t_2)$.

($Disj\ \ f$) : Similarly, $\boxed{x}\ (p_1 \vee p_2)^f$ if and only if $x \notin \overline{m}(p_1 \vee p_2)$ if and only if ($x \notin m(p_1)$ and $x \notin m(p_2)$) if and only if $(\boxed{x}\ p_1^f) \wedge (\boxed{x}\ p_2^f)$.

($Conj\ \ t$,$Conj\ \ f$): We know that $\boxed{x}\ (p_1 \wedge p_2)^t$ if and only if $x \in \overline{m}(p_1 \wedge p_2) = \overline{m}(p_1) \cap \overline{m}(p_2)$ if and only if $(\boxed{x}\ p_1^t) \wedge (\boxed{x}\ p_2^t)$. The second part follows similarly.

($S_j\ t, S_j f$) : $\boxed{x}\ (S_j(p_1))^t$ if and only if $x \in \overline{m}(S_j(p_1))$ if and only if $s_j(x) \in m(p_1)$.

($\sim t, \sim f$) : $\boxed{x}\ (\sim (p_1))^t$ if and only if $x \in \overline{m}(\sim (p_1))$ if and only if $g(x) \notin m(p_1)$.

($\Rightarrow t$) : By Lemma 5.19, $\boxed{x}\ (p_1 \Rightarrow p_2)^t$ if and only if $x \in \overline{m}(p_1 \Rightarrow p_2)$ if and only if $\forall y \geq x$ if $\boxed{y}\ p_1^t$ then $\boxed{y}\ p_2^t$ at m, if and only if $\forall y \geq x\ \boxed{y}\ p_1^f \vee \boxed{y}\ p_2^t$ at m.

($\Rightarrow f$) : We know that $\boxed{x}\ (p_1 \Rightarrow p_2)^f$ if and only if for some $y \geq x$, $\boxed{y}\ p_1^t$ and $\boxed{y}\ p_2^f$. By distributivity, the formula $\bigvee_{y \geq x}(\boxed{y}\ p_1^t \wedge \boxed{y}\ p_2^f)$ can alternatively be written as

$$\bigwedge_{\substack{S_1, S_2, S_1 \cap S_2 = \emptyset \\ S_1 \cup S_2 = \{y \mid y \geq x\}}} \left(\bigvee_{y_1 \in S_1} \boxed{y_1}\ p_1^t \vee \bigvee_{y_2 \in S_2} \boxed{y_2}\ p_2^f \right)$$

It is easy to see that, since in our case for every state x the set $\{y \mid y \geq x\}$ is finite and totally ordered, it follows that for every non-empty set $S_1 \subseteq \{y \mid y \geq x\}$, S_1 contains a maximal element $\mathsf{max}(\mathsf{S}_1)$ and $\bigvee_{y_1 \in S_1} \boxed{y_1}\ p_1^t$ if and only if $\boxed{\mathsf{max}(S_1)}\ p_1^t$. Similarly, for every non-empty $S_2 \subseteq \{y \mid y \geq x\}$, S_2 contains a minimal element $\mathsf{min}(\mathsf{S}_2)$ and $\bigvee_{y_2 \in S_2} \boxed{y_2}\ p_2^f$ if and only if $\boxed{\mathsf{min}(S_2)}\ p_2^f$. Additionally, since $S_1 \cap S_2 = \emptyset$, $\mathsf{max}(\mathsf{S}_1) \neq \mathsf{min}(\mathsf{S}_2)$. Hence,

$$\boxed{x}\ (p_1 \Rightarrow p_2)^f$$

$$\text{iff}$$

$$\boxed{\mathsf{max}\{y \mid y \geq x\}}\ p_1^t \wedge \bigwedge_{\substack{S_1, S_2 \neq \emptyset, S_1 \cap S_2 = \emptyset \\ S_1 \cup S_2 = \{y \mid y \geq x\}}} \left(\boxed{\mathsf{max}(S_1)}\ p_1^t \vee \boxed{\mathsf{min}(S_2)}\ p_2^f\right) \wedge \boxed{x}\ p_2^f,$$

i.e. iff $\boxed{\mathsf{max}\{y \mid y \geq x\}}\ p_1^t \wedge \bigwedge_{\substack{y_1, y_2 \geq x, \\ y_1 \neq y_2}} \left(\boxed{y_1}\ p_1^t \vee \boxed{y_2}\ p_2^f\right) \wedge \boxed{x}\ p_2^f$.

($\neg\ t$, $\neg\ f$): Follows from the fact that $\neg p = p \Rightarrow 0$. □

We can use Consequence 5.30 and Lemma 5.31 in order to obtain a conjunctive normal form for ϕ. The rules necessary for eliminating the operators are the following (where L is a signed literal):

$$Disj\ (t): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ (p_1 \vee p_2)^t}{\left\{L, \boxed{\uparrow(0,i)}\ p_1^t, \boxed{\uparrow(0,i)}\ p_2^t\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ (p_1 \vee p_2)^t}{\left\{L, \boxed{\uparrow(i,0)}\ p_1^t, \boxed{\uparrow(i,0)}\ p_2^t\right\}}$$

$$Disj\ (f): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ (p_1 \vee p_2)^f}{\left\{L, \boxed{\uparrow(0,i)}\ p_1^f\right\} \wedge \left\{L, \boxed{\uparrow(0,i)}\ p_2^f\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ (p_1 \vee p_2)^f}{\left\{L, \boxed{\uparrow(i,0)}\ p_1^f\right\} \wedge \left\{L, \boxed{\uparrow(i,0)}\ p_2^f\right\}}$$

$$Conj\ (t): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ (p_1 \wedge p_2)^t}{\left\{L, \boxed{\uparrow(0,i)}\ p_1^t\right\} \wedge \left\{L, \boxed{\uparrow(0,i)}\ p_2^t\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ (p_1 \wedge p_2)^t}{\left\{L, \boxed{\uparrow(i,0)}\ p_1^t\right\} \wedge \left\{L, \boxed{\uparrow(i,0)}\ p_2^t\right\}}$$

$$Conj\ (f): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ (p_1 \wedge p_2)^f}{\left\{L, \boxed{\uparrow(0,i)}\ p_1^f, \boxed{\uparrow(0,i)}\ p_2^f\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ (p_1 \wedge p_2)^f}{\left\{L, \boxed{\uparrow(i,0)}\ p_1^f, \boxed{\uparrow(i,0)}\ p_2^f\right\}}$$

$$S_j(t): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ S_j(p)^t}{\left\{L, \boxed{\uparrow(0, \frac{n}{n-1} - j)}\ p^t\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ S_j(p)^t}{\left\{L, \boxed{\uparrow(\frac{n}{n-1} - j, 0)}\ p^t\right\}}$$

$$S_j(f): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ S_j(p)^f}{\left\{L, \boxed{\uparrow(0, \frac{n}{n-1} - j)}\ p^f\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ S_j(p)^f}{\left\{L, \boxed{\uparrow(\frac{n}{n-1} - j, 0)}\ p^f\right\}}$$

$$\sim(t): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ \sim p^t}{\left\{L, \boxed{\uparrow(\frac{n}{n-1} - i, 0)}\ p^f\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ \sim (p)^t}{\left\{L, \boxed{\uparrow(0, \frac{n}{n-1} - i)}\ p^f\right\}}$$

$$\sim(f): \quad \frac{L \vee \boxed{\uparrow(0,i)}\ \sim p^f}{\left\{L, \boxed{\uparrow(\frac{n}{n-1} - i, 0)}\ p^t\right\}} \qquad \frac{L \vee \boxed{\uparrow(i,0)}\ \sim (p)^f}{\left\{L, \boxed{\uparrow(0, \frac{n}{n-1} - i)}\ p^t\right\}}$$

$$\Rightarrow (t): \quad \frac{L \vee \boxed{\uparrow (0,i)}\,(p_1 \Rightarrow p_2)^t}{\bigwedge_{j \leq i} \left\{L, \boxed{\uparrow (0,j)}\,p_1^f, \boxed{\uparrow (0,j)}\,p_2^t\right\}}, \qquad \frac{L \vee \boxed{\uparrow (i,0)}\,(p_1 \Rightarrow p_2)^t}{\bigwedge_{j \leq i} \left\{L, \boxed{\uparrow (j,0)}\,p_1^f, \boxed{\uparrow (j,0)}\,p_2^t\right\}},$$

$$\Rightarrow (f): \quad \frac{L \vee \boxed{\uparrow (0,i)}\,(p_1 \Rightarrow p_2)^f}{\begin{array}{c}\left\{L, \boxed{(0, \frac{1}{n-1})}\,p_1^t\right\} \wedge \\ \bigwedge_{j \neq j' \leq i} \left\{L, \boxed{\uparrow (0,j)}\,p_1^t, \boxed{\uparrow (0,j')}\,p_2^f\right\} \wedge \\ \left\{L, \boxed{\uparrow (0,i)}\,p_2^f\right\},\end{array}} \qquad \frac{L \vee \boxed{\uparrow (i,0)}\,(p_1 \Rightarrow p_2)^f}{\begin{array}{c}\left\{L, \boxed{(\frac{1}{n-1}, 0)}\,p_1^t\right\} \wedge \\ \bigwedge_{j \neq j' \leq i} \left\{L, \boxed{\uparrow (j,0)}\,p_1^t, \boxed{\uparrow (j',0)}\,p_2^f\right\} \wedge \\ \left\{L, \boxed{\uparrow (i,0)}\,p_2^f\right\},\end{array}}$$

$$\neg(t): \quad \frac{L \vee \boxed{\uparrow (0,i)}\,\neg p^t}{\bigwedge_{j \leq i} \left\{L, \boxed{\uparrow (0,j)}\,p^f\right\}}, \qquad \frac{L \vee \boxed{\uparrow (i,0)}\,\neg p^t}{\bigwedge_{j \leq i} \left\{L, \boxed{\uparrow (j,0)}\,p^f\right\}},$$

$$\neg(f): \quad \frac{L \vee \boxed{\uparrow (0,i)}\,\neg p^f}{\left\{L, \boxed{\uparrow (0,j)}\,p^t, j \leq i\right\}} \qquad \frac{L \vee \boxed{\uparrow (i,0)}\,\neg p^f}{\left\{L, \boxed{\uparrow (j,0)}\,p^t, j \leq i\right\}}$$

After performing this translation, from any formula ϕ we obtain a formula Φ in clause form, containing "literals" of the form $\boxed{x}\,p^t$ and $\boxed{x}\,p^f$ where p is a variable and x a possible world.

From Corollary 5.30 and Lemma 5.31 it follows that ϕ is a theorem if and only if Φ is unsatisfiable.

Proposition 5.32

(1) *The number of clauses generated from a given formula ϕ is $O(n^3 l)$, where l is the number of subformulae of ϕ.*

(2) *If the formula ϕ does not contain the connective $\Rightarrow$, then the number of clauses generated from ϕ is $O(n^2 l)$, where l is the number of subformulae of ϕ.*

(3) *If the formula ϕ does not contain the connectives $\Rightarrow$ and $\neg$, then the number of clauses generated from ϕ is $O(nl)$, where l is the number of subformulae of ϕ.*

Proof: The number of clauses generated from a given formula ϕ is

$$1+\sum_{\psi \text{ subformula of } \phi}\ \sum_{x\in D(S_{n^2})} \mid \text{clauses}(\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t)\mid + \mid \text{clauses}(\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f)\mid.$$

(1) The maximal number of clauses is generated by the subformulae of the form $\psi = \psi_1 \Rightarrow \psi_2$. In this case, for every $x \in D(S_{n^2})$ the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) is less than or equal to $card(\{y \mid y \geq x\}$, and the number of clauses generated by ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) is less than or equal to $2 + card(\{(x_1, x_2) \mid x_1 \neq x_2, x_1, x_2 \geq x\})$. Thus, the number of clauses generated from a given formula ϕ has as upper bound $1 + 2l\Sigma_{i=1}^{n-1}(i + i(i-1) + 2) = 1 + 4l + 2l\Sigma_{i=1}^{n-1} i^2 = 1 + 4l + 2l\frac{(n-1)(n-2)(2n-3)}{6}$. Hence, the number of clauses generated from a given formula ϕ is $O(n^3 l)$.

(2) If the formula ϕ does not contain the operator $\Rightarrow$, then the maximal number of clauses is generated by the subformulae of the form $\psi = \neg\psi_1$. In this case, for every $x \in D(S_{n^2})$ the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) is equal to the number of elements in $D(S_{n^2})$ smaller than x; ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) gives rise to only one clause. If the subformula does not have the form $\neg\psi$, then the sum of the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) and the number of clauses generated by ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) is at most 3. Thus, the number of clauses generated from ϕ has as upper bound $1 + 2l\Sigma_{i=1}^{n-1}(i+1) = 1 + 2l\Sigma_{i=2}^{n} i = 1 + l(n^2 + n - 2)$. Thus, in this case the number of clauses generated from ϕ is $O(n^2 l)$.

(3) Assume that ϕ does not contain the connectives $\Rightarrow$ and $\neg$. In this case, for every $x \in D(S_{n^2})$ the sum of the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) and the number of clauses generated by ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) is at most 3. Thus, the number of clauses generated from ϕ has as upper bound $1 + 6l(n-1)$, so it is $O(nl)$. □

Example 5.1 *Find a clause form for $\phi = (S_1 x \vee \neg S_1 x)$.*

Proof: We introduce the following renamings: $p = S_1 x$, $q = \neg p$, $r = p \vee q$.

Therefore ϕ is a tautology if and only if the conjunction of the following formulae is unsatisfiable:

$\boxed{\uparrow (0,1)}\, r^f \vee \boxed{\uparrow (1,0)}\, r^f$, and for all $\alpha \in D(S_{n^2})$

$\boxed{\alpha}\, r^f \vee \boxed{\alpha}\, (p \vee q)^t$, $\boxed{\alpha}\, r^t \vee \boxed{\alpha}\, (p \vee q)^f$,

$\boxed{\alpha}\, q^f \vee \boxed{\alpha}\, \neg p^t$, $\boxed{\alpha}\, q^t \vee \boxed{\alpha}\, \neg p^f$,

$\boxed{\alpha}\, p^f \vee \boxed{\alpha}\, S_1(x)^t$, $\boxed{\alpha}\, p^t \vee \boxed{\alpha}\, S_1(x)^f$.

We have therefore the following set of clauses:

$$\left\{\boxed{\uparrow(0,1)}\ r^f, \boxed{\uparrow(1,0)}\ r^f\right\},$$

$$\left\{\boxed{\uparrow(0,i)}\ r^f, \boxed{\uparrow(0,i)}\ p^t, \boxed{\uparrow(0,i)}\ q^t\right\},$$

$$\left\{\boxed{\uparrow(0,i)}\ r^t, \boxed{\uparrow(0,i)}\ p^f\right\},$$

$$\left\{\boxed{\uparrow(i,0)}\ r^t, \boxed{\uparrow(i,0)}\ p^f\right\},$$

$$\left\{\boxed{\uparrow(0,i)}\ q^f, \boxed{\uparrow(0,j)}\ p^f\right\},$$

$$\left\{\boxed{\uparrow(i,0)}\ q^f, \boxed{\uparrow(j,0)}\ p^f\right\},$$

$$\left\{\boxed{\uparrow(0,i)}\ p^f, \boxed{\uparrow(0,1)}\ x^t\right\},$$

$$\left\{\boxed{\uparrow(i,0)}\ p^f, \boxed{\uparrow(1,0)}\ x^t\right\},$$

and, for all i and $j \leq i$ (where applicable):

$$\left\{\boxed{\uparrow(i,0)}\ r^f, \boxed{\uparrow(i,0)}\ p^t, \boxed{\uparrow(i,0)}\ q^t\right\},$$

$$\left\{\boxed{\uparrow(0,i)}\ r^t, \boxed{\uparrow(0,i)}\ q^f\right\},$$

$$\left\{\boxed{\uparrow(i,0)}\ r^t, \boxed{\uparrow(i,0)}\ q^f\right\},$$

$$\left\{\boxed{\uparrow(0,i)}\ q^t, \boxed{\uparrow(0,j)}\ p^t, \forall j \leq i\right\}$$

$$\left\{\boxed{\uparrow(i,0)}\ q^t, \boxed{\uparrow(j,0)}\ p^t, \forall j \leq i\right\}$$

$$\left\{\boxed{\uparrow(0,i)}\ p^t, \boxed{\uparrow(0,1)}\ x^f\right\},$$

$$\left\{\boxed{\uparrow(i,0)}\ p^t, \boxed{\uparrow(1,0)}\ x^f\right\}.$$

5.2.2 A Resolution Procedure

We now continue by showing that we can formulate a version of negative hyperresolution in this more general context, inspired by the method described in [Häh94, Häh96b].

Negative Hyperresolution

$$\frac{\left\{\boxed{x_1}\ p_1^f\right\} \cup D_1, \ldots, \left\{\boxed{x_n}\ p_n^f\right\} \cup D_n, \left\{\boxed{y_1}\ p_1^t, \ldots, \boxed{y_n}\ p_n^t\right\} \cup E}{D_1 \cup \ldots \cup D_n \cup E}$$

provided that $n \geq 1$, $y_i \leq x_i$ for all $i = 1, \ldots, n$ and $D_1, \ldots, D_n, E$ are negative.

It is easy to see that if $\Box$ can be derived from Φ by a finite number of applications of many-valued negative hyperresolution then Φ is unsatisfiable (this follows easily from the fact that if m is a model for the negative clauses $C_1, \ldots C_n$ and for the positive clause C then m is a model of any of their resolvents).

In order to prove the completeness of many-valued negative hyperresolution we show that the proof presented in [Häh96b], which was taken virtually unaltered from [AB70], works as well in this case.

Theorem 5.33 *Let Φ be an unsatisfiable set of clauses. Then $\Box$ can be derived from Φ by a finite number of applications of many-valued negative hyperresolution.*

Proof: Let $nl(\Phi)$ be the total number of literals in Φ and $nc(\Phi)$ the total number of clauses in Φ. It is obvious that $nl(\Phi) \geq nc(\Phi)$. We will proceed by induction on the difference $k(\Phi) = nl(\Phi) - nc(\Phi)$. If $\Box \in \Phi$ then the conclusion is obvious. Therefore in what follows we will assume that $\Box \notin \Phi$. We distinguish the following cases

Case 1. $k(\Phi) = 0$: In this case $nl(\Phi) = nc(\Phi)$. Since $\Box \notin \Phi$, Φ must consist only of unit clauses. Since Φ is unsatisfiable, there must exist two clauses $\{\boxed{\alpha}\, p^t\}$ and $\{\boxed{\beta}\, p^f\}$ in Φ such that $\alpha \leq \beta$.

(In order to prove that the last statement is true, assume that for every two clauses C_1, C_2 in Φ, if $C_1 = \boxed{\alpha}\, p^t$ and $C_2 = \boxed{\beta}\, p^f$ then $\alpha \not\leq \beta$. In this case we can construct a valuation $m : \mathsf{Var} \to \mathcal{O}(D(S_{n^2}))$ that satisfies Φ. Indeed, for every $p \in \mathsf{Var}$ let $m(p) = D(A)$ if p does not occur in the clauses in Φ, and $m(p) = \uparrow \{\alpha_m \mid \alpha_m$ is minimal element of $\{\alpha \mid \boxed{\alpha}\, p^t \in \Phi\}\}$. It is easy to see that m is a meaning function, and that m satisfies Φ: if $\boxed{\alpha}\, p^t \in \Phi$ then $\alpha \in m(p)$, whereas if $\boxed{\beta}\, p^f \in \Phi$ then $\beta \notin m(p)$, since otherwise we would have $\beta \geq \alpha_m$, with $\boxed{\alpha_m}\, p^t \in \Phi$.)

Then $\Box$ can be derived from the clauses $\{\boxed{\alpha}\, p^t\}$ and $\{\boxed{\beta}\, p^f\}$ where $\alpha \leq \beta$ by negative hyperresolution.

Case 2. $k(\Phi) > 0$ (i.e. $nl(\Phi) > nc(\Phi)$).

Subcase 2a: Assume that all non-positive clauses consist of one literal. Then it follows that all negative literals in Φ appear in unit clauses. Since Φ is unsatisfiable, there is a positive clause in Φ which immediately produces the empty clause with suitable negative unit clauses.

(In order to prove the last statement, assume that for every positive clause C and every negative clauses $C_1, \dots, C_n$, C does not produce immediately the empty clause with $C_1, \dots, C_n$. We can construct a model that satisfies Φ as follows: Let $m : \mathsf{Var} \to \mathcal{O}(D(A))$ be defined by $m(p) = D(A)$ for every p that does not occur in a negative clause; $m(p) = \uparrow \{\alpha \mid \boxed{\beta}\, p^f \in \Phi$ implies $\beta \not\geq \alpha\}$.

It is easy to see that m is a valuation that makes true all negative clauses as well as all clauses that contain variables which do not occur in any negative clause. Let $C = \{\boxed{y_1}\, p_1^t, \dots, \boxed{y_n}\, p_n^t\}$ be a clause that only contains variables that appear in negative clauses. From the assumption, the empty clause cannot be generated by hyperresolution from C and other negative clauses. Therefore, there exists an index $i \in \{1, \dots, n\}$ such that for every negative clause C' in Φ that contains the variable p_i, $C' = \{\boxed{x'}\, p_i^f\}$ such that $x' \not\geq y_i$. But in this case $y_i \in m(p)$, hence m makes C true.)

Subcase 2b: There is a non-positive non-unit clause $C = \{\boxed{\beta}\, p^f\} \cup D \in \Phi$ with $D \neq \Box$. Let $\Phi' = \Phi \backslash \{C\}$, $\Phi_1 = \Phi' \cup D$, and $\Phi_2 = \Phi' \cup \{\{\boxed{\beta}\, p^f\}\}$. Since Φ is unsatisfiable it follows that both Φ_1, Φ_2 are unsatisfiable. Moreover, Φ_1 and Φ_2 contain the same number of clauses as Φ, but they have less literals. Hence, $k(\Phi_1) < k(\Phi)$ and $k(\Phi_2) < k(\Phi)$. Since Φ_1 and Φ_2 are both unsatisfiable, by the induction hypothesis, $\Box$ can be deduced from Φ_1 and $\Box$ can be deduced from Φ_2 by hyperresolution. Consider a hyperresolution deduction of $\Box$ from Φ_1. If we replace each occurrence of D by C in this proof then we obtain a valid hyperresolution deduction with last clause $\square$ or $\{\,\boxed{\beta}\ p^f\}$. If the last clause is $\square$, we have already a hyperresolution deduction of $\square$ from Φ. If the last clause is $\{\,\boxed{\beta}\ p^f\}$, then its deduction from Φ can be extended to a deduction of $\square$ from Φ using the fact that there is a deduction of $\square$ from Φ_2. $\square$

Example 5.2 *Consider the example presented before. The set of clauses can be shown to be unsatisfiable by negative hyperresolution:*

$$\begin{array}{ccc}
\left\{\boxed{\uparrow(0,1)}\, r^t, \boxed{\uparrow(0,1)}\, q^f\right\} & \left\{\boxed{\uparrow(0,1)}\, r^f, \boxed{\uparrow(1,0)}\, r^f\right\} & \left\{\boxed{\uparrow(0,1)}\, r^t, \boxed{\uparrow(0,1)}\, p^f\right\} \\
\left\{\boxed{\uparrow(0,1)}\, q^f, \boxed{\uparrow(1,0)}\, r^f\right\} & \left\{\boxed{\uparrow(0,1)}\, p^f, \boxed{\uparrow(1,0)}\, r^f\right\} & \left\{\boxed{\uparrow(0,1)}\, p^t, \boxed{\uparrow(0,1)}\, x^f\right\} \\
 & \left\{\boxed{\uparrow(0,1)}\, x^f, \boxed{\uparrow(1,0)}\, r^f\right\} & \left\{\boxed{\uparrow(0,i)}\, p^f, \boxed{\uparrow(0,1)}\, x^t\right\} \forall i \\
\left\{\boxed{\uparrow(0,1)}\, q^f, \boxed{\uparrow(1,0)}\, r^f\right\} & \left\{\boxed{\uparrow(0,i)}\, p^f, \boxed{\uparrow(1,0)}\, r^f\right\} \forall i & \left\{\boxed{\uparrow(0,1)}\, q^t, \boxed{\uparrow(0,i)}\, p^t, \forall i \leq 1\right\} \\
\left\{\boxed{\uparrow(1,0)}\, r^t, \boxed{\uparrow(1,0)}\, q^f\right\} & \left\{\boxed{\uparrow(1,0)}\, r^f\right\} & \left\{\boxed{\uparrow(1,0)}\, r^t, \boxed{\uparrow(1,0)}\, p^f\right\} \\
\left\{\boxed{\uparrow(1,0)}\, q^f\right\} & \left\{\boxed{\uparrow(1,0)}\, p^f\right\} & \left\{\boxed{\uparrow(1,0)}\, p^t, \boxed{\uparrow(1,0)}\, x^f\right\} \\
 & \left\{\boxed{\uparrow(1,0)}\, x^f\right\} & \left\{\boxed{\uparrow(i,0)}\, p^f, \boxed{\uparrow(1,0)}\, x^t\right\} \forall i \\
\left\{\boxed{\uparrow(1,0)}\, q^f\right\} & \left\{\boxed{\uparrow(i,0)}\, p^f\right\} \forall i & \left\{\boxed{\uparrow(1,0)}\, q^t, \boxed{\uparrow(i,0)}\, p^t, \forall i \leq 1\right\} \\
 & \square &
\end{array}$$

5.3 A General Approach

Results on sheaf representation for discriminator varieties and the applications to unifications have been given in Section 4.1.2 and Section 4.1.3. Therefore, in what follows we will only focus on applications of the Priestley representation theorem to automated theorem proving.

In this section we will show that the ideas on which the procedure for automated theorem proving presented before is based can be applied without major modifications to a wide class of logics, namely the class of those logics that are sound and complete with respect to a variety $\mathcal{V}$ of algebras with a distributive lattice reduct and operators, with the additional property that $\mathcal{V}$ is generated by one finite algebra A, and such that the Priestley Duality for distributive lattices extends to a dual equivalence between $\mathcal{V}$ (seen as a category) and an appropriate category of ordered Priestley spaces endowed with additional relations.

We begin with some theoretical considerations that show the correctness of our approach, and then we will give a similar general resolution procedure.

We will end by illustrating the method by means of two examples.

5.3.1 Theoretical Considerations

We will begin by analyzing the propositional case. In Section 5.3.6 we will also consider many-valued first-order logics.

Let $\mathcal{L}$ be a propositional logic which satisfies the following properties:

(P1) $\mathcal{L}$ is sound and complete with respect to a variety of algebras $\mathcal{V}_{\mathcal{L}}$ generated by finitely many finite algebras, i.e. such that $\mathcal{V}_{\mathcal{L}} = HSP(A_1, \dots, A_n)$. In other words,

$$\begin{array}{lll} \vdash_{\mathcal{L}} \phi & \text{if and only if} & \mathcal{V}_{\mathcal{L}} \models \phi = 1 \\ & \text{if and only if} & \text{for } i = 1, \dots, n, A_i \models \phi = 1. \end{array}$$

(P2) The algebras in $\mathcal{V}_{\mathcal{L}}$ have an underlying distributive lattice structure, and

(P3) The Priestley duality induces a full duality between the variety $\mathcal{V}_{\mathcal{L}}$ and a subcategory of the category of Priestley spaces (possibly with additional operators), that we will denote by $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$.

For every algebra $A \in \mathcal{V}_{\mathcal{L}}$ we will denote its dual by $D(A)$, and the isomorphism between A and the set $\mathsf{ClopenOF}(D(A))$ of clopen order-filters of $D(A)$ by $\eta_A : A \to \mathsf{ClopenOF}(D(A))$.

Topological Relational Models

The satisfiability relation $\overset{a}{\models}$ for the algebras of $\mathcal{V}_{\mathcal{L}}$ induces a satisfiability relation $\overset{rc}{\models}$ for the elements in $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ as follows:

In order to define $\overset{rc}{\models}$ we will require that all the meaning functions have as values clopen order-filters in K. Let K be an object of $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$. Let $m : \mathsf{Var} \to \mathsf{ClopenOF}(K)$ be a meaning function. Since we assumed that there is a dual equivalence between the categories $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ and $\mathcal{V}_{\mathcal{L}}$ (induced by the Priestley duality), it follows that $K = D(A)$ for some $A \in \mathcal{V}_{\mathcal{L}}$.

$$\begin{tikzcd}
\mathsf{Var} \arrow[r, "m"] \arrow[dr] & \mathsf{ClopenOF}(D(A)) \arrow[r, shift left, "\eta_A^{-1}"] & A \arrow[l, shift left, "\eta_A"] \\
& \mathsf{Fma}(\mathsf{Var}) \arrow[u, "\overline{m}"] \arrow[ur, "\overline{\eta_A^{-1} \circ m}"'] &
\end{tikzcd} \tag{5.1}$$

Definition 5.13 *Let* $\overline{m} : \mathsf{Fma}(\mathsf{Var}) \to \mathsf{ClopenOF}(K) = \mathsf{ClopenOF}(D(A))$ *be defined by* $\overline{m} = \eta_A \circ \overline{\eta_A^{-1} \circ m}$. *We define:*

(1) $D(A) \stackrel{rc}{\models}_{m,x} \phi$ *if and only if* $x \in \overline{m}(\phi)$,

(2) $D(A) \stackrel{rc}{\models}_{m} \phi$ *if and only if* $\overline{m}(\phi) = D(A)$,

(3) $D(A) \stackrel{rc}{\models} \phi$ *if and only if* $D(A) \stackrel{r}{\models}_{m} \phi$ *for every* $m : \mathsf{Var} \to \mathsf{ClopenOF}(D(A))$.

Lemma 5.34 *Let* A *be an algebra in* $\mathcal{V}$. *Then* $A \stackrel{a}{\models} \phi$ *if and only if* $D(A) \stackrel{rc}{\models} \phi$.

Proof: Assume that $A \stackrel{a}{\models} \phi$. Let $m : \mathsf{Var} \to \mathsf{ClopenOF}(D(A))$ be a meaning function such that for every $p \in \mathsf{Var}$, $m(p)$ is a clopen order-filter. Let $f_m : \mathsf{Var} \to A$ be defined by $f_m = \eta_A^{-1} \circ m$. Since $A \stackrel{a}{\models} \phi$ it follows that $\overline{f_m}(\phi) = 1$, hence $\overline{m}(\phi) = \eta_A \circ \overline{\eta_A^{-1} \circ m}(\phi) = \eta_A(1) = D(A)$, i.e. $D(A) \stackrel{rc}{\models}_m \phi$.

Conversely, assume that $D(A) \stackrel{rc}{\models} \phi$. Let $f : \mathsf{Var} \to A$ be an arbitrary assignment. Let $m_f : \mathsf{Var} \to \mathsf{ClopenOF}(D(A))$ be defined by $m_f = \eta_A \circ f$. Since $D(A) \stackrel{rc}{\models} \phi$ it follows that $\overline{m}_f(\phi) = D(A)$, where $\overline{m}_f = \eta_A \circ \overline{\eta_A^{-1} \circ \eta_A \circ f} = \eta_A \circ \overline{f}$. Hence, $\overline{f}(\phi) = 1$. □

Proposition 5.35 *Under the conditions (P1)–(P3) we have*

$$\vdash_{\mathcal{L}} \phi \text{ if and only if for all } i = 1, \ldots, n, D(A_i) \stackrel{rc}{\models} \phi.$$

Proof: By (P1), $\vdash_{\mathcal{L}} \phi$ if and only if for all $i = 1, \ldots, n$, $A_i \stackrel{a}{\models} \phi$. By Lemma 5.34, for every $i = 1, \ldots, n$, $A_i \stackrel{a}{\models} \phi$ if and only if $D(A_i) \stackrel{rc}{\models} \phi$. □

Remark: The result in Lemma 5.34 may seem surprising, since a corresponding theorem does not hold in general for modal logics if one considers satisfiability in modal algebras respectively in Kripke frames. The reason is that we defined a restricted notion of meaning function on the topological relational models $K \in \mathcal{V}_{\mathcal{L}}\mathsf{Sp}$, by imposing that the values of such meaning functions are *clopen* order-filters.

Note that the satisfiability relation $\stackrel{rc}{\models}$ defined this way does not coincide with the notions of satisfiability defined for Kripke models in [IO96] (where the values of the meaning functions are only required to be hereditary sets).

However, for finite spaces $K \in \mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ the notion of satisfiability defined before coincides with the more general notion of satisfiability defined in [IO96]. This happens because in the finite case the topology on the Priestley spaces is discrete (hence, all the order-filters are clopen). By Assumption (P1), the algebras $A_1, \dots, A_n$ are finite, thus every meaning function with as values order-filters has as values actually *clopen* order-filters (in the discrete topology). Thus, in Proposition 5.35 the topological properties of the meaning functions play no rôle. Hence, the satisfiability relation defined on $D(A_i)$, for $i \in \{1, \dots, n\}$ coincides with a more general satisfiability relation analogous to the one defined in [IO96].

5.3.2 Towards a Link Between Algebraic and Relational Models

The consideration in Section 5.1.5 concerning the extension of the Priestley duality theorem to SHn-algebras, as well as similar existing approaches (like for example duality theorems for varieties of Ockham algebras and for θ-valued Łukasiewicz-Moisil algebras), together with duality theorems between certain varieties of modal logics and appropriate categories of topological Kripke models suggest that a general approach to extending Priestley duality theorems to varieties of distributive algebras with operators may be possible. Some steps in this direction have been done by [Gol89], where Priestley duality is used in order to develop a representation theorem for distributive lattices with operators (join hemimorphisms and meet hemimorphisms are considered).

In what follows we extend the results of [Gol89] in that we consider separately operators that are homomorphisms or antimorphisms, and use these results in order to define a general notion of relational (not necessarily topological) models for such logics, and a notion of satisfiability in such models. Thus, in Lemma 5.36, we explicitely consider lattice homomorphisms and lattice antimorphisms besides the join- and meet-hemimorphisms studied in [Gol89]. This will offer a general framework for expressing the automated theorem proof procedure, for proving its correctness and for analyzing its complexity.

We will consider some cases that appear more frequently in practice. Namely, we will consider the situations when the operators of the logic $\mathcal{L}$ can belong to the following classes: $\{\wedge, \vee\}, Lh, La, Jh, Mh$, and Heyting implication and negation Hey.

We assume that the logic $\mathcal{L}$ satisfies the conditions $(P1) - (P3)$, with the remarks that:

(1) The signature of the algebras in $\mathcal{V}_{\mathcal{L}}$ is $\{\wedge, \vee, 0, 1\} \cup \Sigma$, where $\Sigma = Lh \cup La \cup Jh \cup Mh$ (or $\Sigma = \{\Rightarrow, \neg\} \cup Lh \cup La \cup Jh \cup Mh$ if the logic also contains the operation symbols for Heyting implication and negation),

(2) The additional operations on the algebras in $\mathcal{V}_{\mathcal{L}}$ belong to the following classes: lattice homomorphisms (Lh), lattice antimorphisms (La), join hemimorphisms (Jh) and meet hemimorphisms (Mh) (resp. Heyting algebra operations (Hey)). The definitions are given below.

Definition 5.14 *Let A be an algebra with a lattice reduct.*

A lattice antimorphism *on A is a function $k : A \to A$ with $k(0) = 1, k(1) = 0, k(a_1 \vee a_2) = k(a_1) \wedge k(a_2)$ and $k(a_1 \wedge a_2) = k(a_1) \vee k(a_2)$.*
A join hemimorphism *on A is a function $f : A^n \to A$ such that for every $i, 1 \leq i \leq n$,*
$(Jh1)$ $f(a_1, \dots, a_{i-1}, 0, a_{i+1}, \dots, a_n) = 0,$
$(Jh2)$ $f(a_1, \dots, a_{i-1}, b_1 \vee b_2, a_{i+1}, \dots, a_n) =$
$= f(a_1, \dots, a_{i-1}, b_1, a_{i+1}, \dots, a_n) \vee f(a_1, \dots, a_{i-1}, b_2, a_{i+1}, \dots, a_n).$
A meet hemimorphism *on A is a function $g : A^n \to A$ such that for every $i, 1 \leq i \leq n$,*
$(Mh1)$ $g(a_1, \dots, a_{i-1}, 1, a_{i+1}, \dots, a_n) = 1,$
$(Mh2)$ $g(a_1, \dots, a_{i-1}, b_1 \wedge b_2, a_{i+1}, \dots, a_n) =$
$= g(a_1, \dots, a_{i-1}, b_1, a_{i+1}, \dots, a_n) \wedge g(a_1, \dots, a_{i-1}, b_2, a_{i+1}, \dots, a_n).$

The algebras will be denoted by

$$(A, \vee, \wedge, \{h_A\}_{h \in Lh}, \{k_A\}_{k \in La}, \{f_A\}_{f \in Jh}, \{g_A\}_{g \in Lh}).$$

Based on previous papers in which the link between the algebraic and the relational semantics for modal logics is analyzed (as [Lem66a, Lem66b], as well as the general study in [Gol89]), we will indicate a canonical way in which one can associate functions or relations in the dual space and vice-versa. This will provide hints about the definition of a satisfiability relation on these dual spaces, as well as about the translation to clause form.

	$\mathcal{V}$	$\xrightarrow{D}$	$\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$
Lh	$h \in Lh$, i.e. $h_A : A \to A$, lattice homomorphism	$\mapsto$	$D(h_A) : D(A) \to D(A)$ order-preserving $D(h_A)(f) = f \circ h, \forall f : A \to \{0,1\}$ (prime filters: $D(h_A)(F) = \{a \in A \mid h(a) \in F\}$)
La	$k \in La$, i.e. $k_A : A \to A$ lattice antimorphism	$\mapsto$	$D(k_A) : D(A) \to D(A)$ order-reversing $D(k_A)(f) = sw \circ f \circ k, \forall f : A \to \{0,1\}$, where $sw(0) = 1, sw(1) = 0$ (prime filters: $D(k_A)(F) = \{a \in A \mid k(a) \notin F\}$)
Jh	$f \in Jh$, i.e. $f_A : A^n \to A$ join hemimorphism	$\mapsto$	$D(f_A) \subseteq D(A)^{n+1}$, increasing relation $D(f_A)(f_1, \dots, f_n, f_{n+1})$ iff $(f_i(x_i) = 1\ \forall i \in \{1, \dots, n\})$ implies $f_{n+1}(f(x_1, \dots, x_n)) = 1$ (prime filters: $D(f_A)(F_1, \dots, F_n, F_{n+1})$ iff $a_i \in F_i, \forall i \in \{1, \dots, n\}$ implies $f(a_1, \dots, a_n) \in F_{n+1}$.)
Mh	$g \in Mh$, i.e. $g_A : A^n \to A$ meet hemimorphism	$\mapsto$	$D(g_A) \subseteq D(A)^{n+1}$, decreasing relation $D(g_A)(f_1, \dots, f_n, f_{n+1})$ iff $f_{n+1}(g(x_1, \dots, x_n)) = 1$ implies $(\exists i \in \{1, \dots, n\} : f_i(x_i) = 1)$. (prime filters: $D(g_A)(F_1, \dots, F_n, F_{n+1})$ iff $(g(a_1, \dots, a_n) \in F_{n+1}$ implies $\exists i \in \{1, \dots, n\}$ with $a_i \in F_i$.)

We note that in a Priestley space the clopen order-filters and their complements form a subbasis for the topology. It is easy to see that this topology is the join of two topologies:

(1) The upper topology, generated by the set of clopen order-filters as a basis,

(2) The lower topology, generated by the set of complements of clopen order-filters as a basis.

Definition 5.15 *Let $(X, \leq)$ be a partially ordered set and let $R \subseteq X^{n+1}$ be a n-ary relation on X.*

(1) R is an increasing *relation if it has the property that for all $x \in X^n$ and every $y, z \in X$, $R(x, y)$ and $y \leq z$ implies $R(x, z)$.*

(2) R is a decreasing *relation if it has the property that for all $x \in X^n$ and every $y, z \in X$, $R(x, y)$ and $z \leq y$ implies $R(x, z)$.*

Lemma 5.36 *Let A be a distributive lattice with operators in the classes Lh, La, Jh, Mh. Let $(D(A), \leq, \tau)$ be the Priestley dual of A. The following holds:*

(1) If $h \in Lh$ (i.e. $h_A : A \to A$ is a lattice homomorphism), then $D(h_A) : D(A) \to D(A)$ is order-preserving and continuous with respect to the topology τ.

(2) If $k \in La$ (i.e. $k_A : A \to A$ is a lattice antimorphism), then $D(k_A) : D(A) \to D(A)$ is order-reversing and continuous with respect to the topology τ.

(3) If $f \in Jh$ (i.e. $f_A : A^n \to A$ is a join hemimorphism), then $D(f_A) \subseteq D(A)^{n+1}$ is an increasing relation such that for every $F \in D(A)$, $D(f_A)^{-1}(F)$ is closed in the product topology on $D(A)^n$ of the upper topology, and moreover, for every $U_1, \dots, U_n \in \mathsf{ClopenOF}(D(A))$, the set $\{F \mid \exists F_1 \in U_1, \dots, F_n \in U_n : D(f_A)(F_1, \dots, F_n, F)\}$ is clopen.

(4) If $g \in Mh$ (i.e. $g_A : A^n \to A$ is a meet hemimorphism), then $D(g_A) \subseteq D(A)^{n+1}$ is a decreasing relation such that for every $F \in D(A)$, $D(g_A)^{-1}(F)$ is closed in the product topology on $D(A)^n$ of the lower topology, and moreover, for every $U_1, \dots, U_n \in \mathsf{ClopenOF}(D(A))$, the set $\{F \mid \forall F_1, \dots, F_n, D(g_A)(F_1, \dots, F_n) \Rightarrow \exists i : F_i \in U_i\}$ is clopen.

Proof: (1) Let $h_A : A \to A$ be a lattice homomorphism. Let $F_1, F_2 \in D(A)$ be such that $F_1 \subseteq F_2$. Then for every $a \in D(h_A)(F_1) = h_A^{-1}(F_1)$, we have $h_A(a) \in F_1 \subseteq F_2$, hence $a \in D(h_A)(F_2) = h_A^{-1}(F_2)$. Thus, $D(h_A)(F_1) \subseteq D(h_A)(F_2)$.

Since the topology on $D(A)$ is generated by the sets $X_a = \{F \in D(A) \mid a \in F\}$ and $D(A) \backslash X_a = \{F \in D(A) \mid a \notin F\}$ as a subbasis, it is sufficient to show that $D(h_A)^{-1}(X_a)$ and $D(h_A)^{-1}(D(A) \backslash X_a)$ are open. This holds, because $D(h_A)^{-1}(X_a) = \{F \mid D(h_A)(F) \in X_a\} = \{F \mid a \in h_A^{-1}(F)\} = \{F \mid h_A(a) \in$

$F\} = X_{h_A(a)}$ and $D(h_A)^{-1}(D(A)\backslash X_a) = \{F \mid D(h_A)(F) \not\in X_a\} = \{F \mid a \not\in h_A^{-1}(F)\} = \{F \mid h_A(a) \not\in F\} = D(A)\backslash X_{h_A(a)}$.

(2) Let $k_A : A \to A$ be a lattice antimorphism. Let $F_1, F_2 \in D(A)$ be such that $F_1 \subseteq F_2$. Then for every $a \in D(k_A)(F_2) = D(A)\backslash k_A^{-1}(F_2)$, we have $k_A(a) \not\in F_2$, hence, since $F_1 \subseteq F_2$, $k_A(a) \not\in F_1$. Hence, $a \in D(k_A)(F_1) = D(A)\backslash k_A^{-1}(F_1)$. Thus, $D(k_A)(F_2) \subseteq D(k_A)(F_1)$.

In order to show that $D(k_A)$ is continuous it is sufficient to show that $D(k_A)^{-1}(X_a)$ and $D(k_A)^{-1}(X\backslash X_a)$ are open. This holds, because $D(k_A)^{-1}(X_a) = \{F \mid D(k_A)(F) \in X_a\} = \{F \mid a \not\in k_A^{-1}(F)\} = \{F \mid k_A(a) \not\in F\} = D(A)\backslash X_{k_A(a)}$ and $D(k_A)^{-1}(D(A)\backslash X_a) = \{F \mid D(k_A)(F) \not\in X_a\} = \{F \mid a \in k_A^{-1}(F)\} = \{F \mid k_A(a) \not\in F\} = X_{k_A(a)}$.

The properties (3) and (4) are analyzed in [Gol89]. Here, we only point out the main ideas of the proofs; for details concerning these proofs we refer to [Gol89], pp.187-190.

(3) Let $f_A : A^n \to A$ be a join hemimorphism. Let $F_1, \dots, F_n \in D(A)$ and $F \subseteq F'$. Assume that $D(f_A)(F_1, \dots, F_n, F)$. By the definition of $D(f_A)$ this holds if and only if $a_i \in F_i$ for all $i \in \{1, \dots, n\}$ implies $f(a_1, \dots, a_n) \in F$. Let $(a_1, \dots, a_N)$ be such that $a_i \in F_i \; \forall i \in \{1, \dots, n\}$. Since $D(f_A)(F_1, \dots, F_n, F)$, it follows that $f_A(a_1, \dots, a_n) \in F \subseteq F'$, hence that $f_A(a_1, \dots, a_n) \in F'$. Therefore, $D(f_A)(F_1, \dots, F_n, F')$.

In order to show that inverse images of points by $D(f_A)$ are closed, let $F \in D(A)$. To show that $D(f_A)^{-1}(F)$ is closed, let $(G_1, \dots, G_n) \in D(A)^n$ be such that $(G_1, \dots, G_n) \not\in D(f_A)^{-1}(F)$, i.e. (not $D(f_A)(G_1, \dots, G_n, F)$). Then there exist $x_1, \dots, x_n \in A$ such that $x_i \in G_i$ for every $i \leq n$, and $f_A(x_1, \dots, x_n) \not\in F$. In this case $G_i \in X_{x_1}$ for every $i \leq n$. Let $N = X_{x_1} \times \dots \times X_{x_n}$. N is an open neighborhood of $(G_1, \dots, G_n)$ in the product of the upper topology. Let $(F_1, \dots, F_n) \in N$. Then $x_i \in F_i$ for every $i \leq n$, and since $f_A(x_1, \dots, x_n) \not\in F$, it follows that not $D(f_A)(F_1, \dots, F_n)$, hence $(F_1, \dots, F_n) \not\in D(f_A)^{-1}(F)$. This shows that $N \subseteq D(A)^n \backslash D(f_A)^{-1}(F)$.

In order to show that for all clopen order-filters $U_1, \dots, U_n$, the set $\{y \mid \exists x_i \in U_1, \dots, x_n \in U_n : D(f_A)(x_1, \dots, x_n, y)\}$ is also a clopen order-filter note first that it is an order-filter, since $D(f_A)$ is order-increasing. It is easy to see that if f is a join hemimorphism, $X_{f(x_1, \dots, x_n)} = \{F \in D(A) \mid \exists G_1 \in X_{x_1}, \dots, \exists G_n \in X_{x_n} : D(f_A)(G_1, \dots, G_n, F)\}$.

(4) Let $g_A : A^n \to A$ be a meet hemimorphism. Let $F_1, \dots, F_n \in D(A)$ and $F' \subseteq F$. Assume that $D(g_A)(F_1, \dots, F_n, F)$. By the definition of $D(g_A)$ this holds if and only if $g_A(a_1, \dots, a_n) \in F$ implies $a_i \in F_i$ for some $i \in \{1, \dots, n\}$.

Assume that $D(g_A)(F_1, \dots, F_n, F)$. Let $g_A(a_1, \dots, a_n) \in F'$. Since $F' \subseteq F$, $g_A(a_1, \dots, a_n) \in F$, hence $a_i \in F_i$ for some $i \in \{1, \dots, n\}$. Therefore, $D(g_A)(F_1, \dots, F_n, F')$.

The fact that inverse images of points by $D(g_A)$ are closed follows as in (3), taking into account the definition of $D(g_A)$. In order to show that for every clopen order-filters $U_1, \dots, U_n$, the set $\{y \mid \forall x_1, \dots, x_n, D(g_A)(x_1, \dots, x_n) \Rightarrow \exists i : x_i \in U_i\}$ is clopen, note first that it is an order-filter and also that if g is a meet hemimorphism, then

$$X_{f(x_1,\ldots,x_n)} = \{F \in D(A) \mid \forall G_1, \ldots, G_n, (D(g_A)(G_1, \ldots, G_n, F) \text{ implies } \exists i \leq n : G_i \in X_{x_i})\}$$

□

The properties (3) and (4) in Lemma 5.36 are analyzed in [Gol89]. Here we also consider operations as lattice morphisms and antimorphisms, since in this case the operations induced on the dual space are much simpler (order-preserving, respectively order-inverting continuous maps).

Let DLO_Σ be the category of distributive lattices with operators in Σ having:

Objects: Distributive lattices with 0, 1 and with additional operators in Σ.

Morphisms: Lattice morphisms that preserve $0, 1$ and the operators in Σ.

In [Gol89] a category RPS_Σ of relational Priestley spaces is defined, having:

Objects: Relational Priestley Spaces, i.e. spaces of the form $\mathbf{X} = (X, \leq, \tau, \{H_X\}_{H\in Lh}, \{K_X\}_{K\in La}, \{R_X\}_{R\in Jh}, \{Q_X\}_{Q\in Mh})$ where

(1) $(X, \leq \tau)$ is a Priestley space,

(2) for every $H \in Lh$, $H_X : X \to X$ is a continuous order-preserving map,

(3) for every $K \in La$, $K_X : X \to X$, is a continuous order-reversing map

(4) for every $R \in Jh$ $R_X \subseteq X^{n_R+1}$ is an increasing relation such that:

(4a) for every $y \in D(A)$, $R_X^{-1}(y)$ is closed in the product topology on $D(A)^{n_R}$ of the upper topology,

(4b) for every $Y_1, \ldots, Y_n \in \mathsf{ClopenOF}(D(A))$ $\{y \mid \exists x_1 \in Y_1, \ldots, x_n \in Y_n : R_X(x_1, \ldots, x_n, y)\}$ is clopen.

(5) for every $Q \in Mh$, $Q_X \subseteq X^{n_Q+1}$ is a decreasing relation such that:

(5a) for every $y \in D(A)$, $Q_X^{-1}(y)$ is closed in the product topology on $D(A)^n$ of the lower topology,

(5b) for every $Y_1, \ldots, Y_n \in \mathsf{ClopenOF}(D(A))$ $\{y \mid \forall x_1, \ldots, x_n, Q_X(x_1, \ldots, x_n) \Rightarrow \exists i : x_i \in Y_i\}$ is clopen.

Morphisms continuous bounded morphisms, see the following definition.

Definition 5.16 (Bounded Morphism, [Gol89]) *Let $\mathbf{X_1}$ and $\mathbf{X_2}$ be two relational Priestley spaces,*

$$\mathbf{X_1} = (X_1, \leq, \tau_1, \{H_{X_1}\}_{H\in Lh}, \{K_{X_1}\}_{K\in La}, \{R_{X_1}\}_{R\in Jh}, \{Q_{X_1}\}_{Q\in Mh}),$$

$$\mathbf{X_2} = (X_2, \leq, \tau_2, \{H_{X_2}\}_{H \in Lh}, \{K_{X_2}\}_{K \in La}, \{R_{X_2}\}_{R \in Jh}, \{Q_{X_2}\}_{Q \in Mh}).$$

(1) A map $\phi : X_1 \to X_2$ is a morphism *if it preserves the order $\leq$, the operations Lh, La and the relations Jh, Mh in passing from X_1 to X_2:*

(M1) $x \leq y$ implies $\phi(x) \leq \phi(y)$,

(M2) $x = H_{X_1}(y)$ implies $\phi(x) = H_{X_2}(\phi(y))$ for every $H \in Lh \cup La$,

(M3) $R_{X_1}(x_1, \ldots, x_n, x)$ implies $R_{X_2}(\phi(x_1), \ldots, \phi(x_n), \phi(x))$ for every $R \in Jh \cup Mh$.

(2) A morphism is bounded *if for all $z \in X_1$ it satisfies*

(BM1) $R_{X_2}(y_1, \ldots, y_n, \phi(z))$ implies $\exists x_1, \ldots, x_n \in X_1(R_{X_1}(x_1, \ldots, x_n, z)$ and $y_i \leq \phi(x_i)$, for every $1 \leq i \leq n)$, for every $y_1, \ldots, y_n \in X_2$, and every $R \in Jh$,

(BM2) $Q_{X_2}(y_1, \ldots, y_n, \phi(z))$ implies $\exists x_1, \ldots, x_n \in X_1(Q_{X_1}(x_1, \ldots, x_n, z)$ and $\phi(x_i) \leq y_i$, for every $1 \leq i \leq n)$, for every $y_1, \ldots, y_n \in X_2$, and every $Q \in Mh$,

Note that the operations in Lh and La induce relations as follows:
(Lh) For every $H \in Lh$, $\rho_H = \{(F_1, F_2) \mid F_1 = H(F_2)\}$,
(La) For every $K \in La$, $\rho_K = \{(F_1, F_2) \mid F_1 = K(F_2)\}$.
Conditions similar to $(BM1)$ and $(BM2)$ can be stated also for these relations, namely:
$(BMLh)$ $\rho_{H_{X_2}}(G, \phi(F_2))$ implies $\exists F_1$: $\rho_{H_{X_1}}(F_1, F_2)$ and $G = \phi(F_1)$,
$(BMLa)$ $\rho_{K_{X_2}}(G, \phi(F_2))$ implies $\exists F_1$: $\rho_{K_{X_1}}(F_1, F_2)$ and $G = \phi(F_1)$.
However, it is not necessary to explicitly specify conditions similar to $(BM1)$ and $(BM2)$ for the operations in Lh and La because they are already satisfied, as shown by Lemma 5.37.

Lemma 5.37 *Let $\phi : \mathbf{X_1} \to \mathbf{X_2}$ be a morphism of relational Priestley spaces. Let $H \in Lh \cup La$. For every $G \in X_2$, if $H_{X_2}(\phi(F_2)) = G$, then there exists a $F_1 \in X_1$ such that $\phi(F_1) = G$ and $H_{X_1}(F_2) = F_1$.*

Proof: It suffices to take $F_1 = H_{X_1}(F_2)$. The fact that $\phi(F_1) = G$ follows from the fact that ϕ is morphism. □

In [Gol89] it is proved that there is a dual equivalence between the category DLO of distributive lattices with operators and the category RPS of relational Priestley spaces. It is very easy to see that this correspondence can be extended a dual equivalence between the category DLO_Σ of distributive lattices with operators in Σ (i.e. including lattice morphisms and lattice antimorphisms) and the corresponding category RPS_Σ of relational Priestley spaces.

The correspondence between the operations and relations in a relational Priestley space X and the corresponding operations on the lattice of its clopen order-filters $\mathsf{ClopenOF}(X)$ is schematically represented in the next table:

	$\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$	$\stackrel{E}{\to}$	$\mathcal{V}_{\mathcal{L}}$
(Lh)	$H_X : X \to X$, order-preserving	$\mapsto$	$E(H_X) : E(X) \to E(X)$, $E(H_X)(f) = f \circ H_X$, $\forall f \in X \to \{0,1\}$ continuous, order-preserving (order-filters: $E(H_X)(U) = \{x \in X \mid H_X(x) \in U\}$).
(La)	$K_X : X \to X$ order-reversing	$\mapsto$	$E(K_X) : E(X) \to E(X)$, $E(K_X)(f) = sw \circ f \circ K_X$, $\forall f : X \to \{0,1\}$, continuous, order-preserving (order-filters: $E(K_X)(U) = \{x \in X \mid K(x) \notin U\}$).
(Jh)	$R_X \subseteq X^{n+1}$ increasing	$\mapsto$	$E(R_X) : E(X)^n \to E(X)$ $E(R_X)(f_1, \dots, f_n)(x) = 1$ iff ($\exists x_1, \dots, x_n$ s.t. $R_X(x_1, \dots, x_n, x)$ and $f_i(x_i) = 1$ $\forall i \in \{1, \dots, n\}$). (order-filters: $E(R_X)(U_1, \dots, U_n) =$ $= \{x \in X \mid \exists x_1 \in U_1, \dots, x_n \in U_n : R_X(x_1, \dots, x_n, x)\}$).
(Mh)	$Q_X \subseteq X^{n+1}$ decreasing	$\mapsto$	$E(Q_X) : E(X)^n \to E(X)$, $E(Q_X)(f_1, \dots, f_n)(x) = 1$ iff ($\forall x_1, \dots, x_n \in X$ if $Q_X(x_1, \dots, x_n, x)$ then $f_i(x_i) = 1$ for some $i \in \{1, \dots, n\}$). (order-filters: $E(Q_X)(U_1, \dots, U_n) =$ $= \{x \in X \mid \forall x_1, \dots, x_n$ with $Q_X(x_1, \dots, x_n, x), \exists i, x_i \in U_i\}$).

Assume that a given logic $\mathcal{L}$ satisfies conditions $(P1) - (P3)$ given in Section 5.3.1.

Condition $(P3)$ states that the logic $\mathcal{L}$ has the property that the duality between the category DLO_Σ of distributive lattices with operators in Σ and the category RPS_Σ of corresponding relational Priestley spaces restricts to a dual equivalence between $\mathcal{V}_{\mathcal{L}}$ and an appropriate subcategory $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ of RPS_Σ.

The duals of the algebras in $\mathcal{V}$ are therefore Priestley spaces endowed with additional operation and relation symbols, corresponding to the operators in Σ. We will represent the corresponding operations, indexed by the same families: $(X, \leq, \tau, \{H_X\}_{H \in Lh}, \{K_X\}_{K \in La}, \{R_X\}_{R \in Jh}, \{Q_X\}_{Q \in Lh})$.

We will assume that the elements in $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ can be described as relational Priestley spaces that satisfy additional properties (induced by the identities that characterize $\mathcal{V}_{\mathcal{L}}$).

Relational Models

The previous considerations suggest a possible definition for relational models for a logic $\mathcal{L}$ with connectives $\Sigma = \{\vee, \wedge\} \cup Lh \cup La \cup Jh \cup Mh$ which is sound and complete with respect to a variety $\mathcal{V}_{\mathcal{L}}$ of algebras that satisfies $(P3)$.

Our goal in what follows is to explain (in an intuitive way) how the satisfiability relation $\overset{r}{\models}$ on Kripke frames and models can be defined if we know a similar relation $\underset{a}{\models}$ for algebraic models.

Definition 5.17 ($\mathcal{L}$-Frame) *A* $\mathcal{L}$-frame *is a relational structure* $X = (X, \leq, \{H_X\}_{H \in Lh}, \{K_X\}_{K \in La}, \{R_X\}_{R \in Jh}, \{Q_X\}_{Q \in Lh})$ *where for every* $H \in Lh$*,* H_X :

$X \to X$ is an order-preserving map, for every $K \in La$, $K_X : X \to X$ is an order-reversing map, for every $R \in Jh$, $R_X \subseteq X^{n_R+1}$ is an increasing relation, and for every $Q \in Mh$, $Q_X \subseteq X^{n_Q+1}$ is a decreasing relation.

Remark: Let $X = (X, \leq, \tau, \{H_X\}_{H \in Lh}, \{K_X\}_{K \in La}, \{R_X\}_{R \in Jh}, \{Q_X\}_{Q \in Lh})$ be a $\mathcal{L}$-frame. Consider the discrete topology τ on X. The ordered topological space $(X, \leq, \tau)$ is totally order-disconnected, and it is compact if and only if X is finite.

Definition 5.18 ($\mathcal{L}$-Model) *A* $\mathcal{L}$-model *based on a $\mathcal{L}$-frame*

$$X = (X, \leq, \{H_X\}_{H \in Lh}, \{K_X\}_{K \in La}, \{R_X\}_{f \in Jh}, \{Q_X\}_{g \in Lh}$$

is a system $M = (X, m)$, where $m : \mathsf{Var} \to \mathcal{P}(X)$ is a meaning function that assigns to every variable $p \in \mathsf{Var}$ an order-filter of X.

The extension of the meaning function to formulae (that generalizes the way meaning functions are extended to formulae in modal logic, for example) is explained by the following lemma.

Lemma 5.38 *Let X be an ordered relational structure with operations and relations in the classes Lh, La, Jh, Mh. The following hold:*

(1) Every order preserving operation on X, $H \in Lh$ induces a lattice morphism $h_H : \mathcal{O}(X) \to \mathcal{O}(X)$, defined for every order-filter U of X by $h_H(U) = H^{-1}(U)$.

(2) Every order reversing operation on X, $K \in Lh$ induces a lattice morphism $k_K : \mathcal{O}(X) \to \mathcal{O}(X)$ defined for every order-filter U of X by $k_K(U) = \mathcal{O}(X) \backslash k^{-1}(U)$.

(3) Every increasing relation $R \subseteq X^{n+1}$ induces a join hemimorphism $f_R : \mathcal{O}(X)^n \to \mathcal{O}(X)$, defined for every $U_1, \ldots, U_n \in \mathcal{O}(X)$ by

$$f_R(U_1, \ldots, U_n) = \{x \in X \mid \exists x_1 \in U_1, \ldots, x_n \in U_n : R(x_1, \ldots, x_n, x)\}.$$

(4) Every decreasing relation $Q \subseteq X^{n+1}$ induces a meet hemimorphism $g_Q : \mathcal{O}(X)^n \to \mathcal{O}(X)$, defined for every $U_1, \ldots, U_n \in \mathcal{O}(X)$ by

$$g_Q(U_1, \ldots, U_n) = \{x \in X \mid \forall x_1, \ldots, x_n \text{ with } Q(x_1, \ldots, x_n, x), \exists i, x_i \in U_i\}.$$

Proof: (1) Let $H : X \to X$ be order-preserving, and let h_H be defined for every $U \in \mathcal{O}(X)$ by $h_H(U) = H^{-1}(U)$. We show that for every $U \in \mathcal{O}(X)$, $h_H(U) \in \mathcal{O}(X)$. Let $x \leq y$. Assume that $x \in h_H(U) = H^{-1}(U)$. Then $H(x) \in U$. Since H is order-preserving and U an order-filter it follows that $H(y) \in U$, i.e. that $y \in h_H(U)$. This proves that $h_H(U) \in \mathcal{O}(X)$. The fact that h_H is a lattice homomorphism follows immediately.

(2) Let $K : X \to X$ be order-reversing, and let k_K be defined for every $U \in \mathcal{O}(X)$ by $k_K(U) = \mathcal{O}(X) \backslash K^{-1}(U)$. We show that for every $U \in \mathcal{O}(X)$,

$k_K(U) \in \mathcal{O}(X)$. Let $x \leq y$. Assume that $x \in k_K(U)$. Then $x \notin K^{-1}(U)$, hence $K(x) \notin U$. Since K is order-reversing and U an order-filter it follows that $K(y) \notin U$, i.e. that $y \in k_K(U)$. This proves that $k_K(U) \in \mathcal{O}(X)$. The fact that h_H is a lattice antimorphism follows immediately.

(3) Let $R \subseteq X^n$ be an increasing relation. For every $U_1, \dots, U_n \in \mathcal{O}(X)$ let $f_R(U_1, \dots, U_n) = \{x \in X \mid \exists x_1 \in U_1, \dots, x_n \in U_n : R(x_1, \dots, x_n, x)\}$. We show that for every $U_1, \dots, U_n \in \mathcal{O}(X)$, $f_R(U_1, \dots, U_n) \in \mathcal{O}(X)$. Let $x \leq y$. Assume that $x \in f_R(U_1, \dots, U_n)$. Then for some $x_1 \in U_1, \dots, x_n \in U_n$, $R(x_1, \dots, x_n, x)$. Since R is an increasing relation it follows that $R(x_1, \dots, x_n, y)$. Hence we proved that there exist $x_1 \in U_1, \dots, x_n \in U_n$ such that $R(x_1, \dots, x_n, y)$, i.e. that $y \in f_R(U_1, \dots, U_n)$. It is easy to see that for every $i \in \{1, \dots, n\}$, $f_R(U_1, \dots, U_{i-1}, \emptyset, U_{i+1}, \dots, U_n) = \emptyset$. Also,
$f_R(U_1, \dots, U_{i-1}, U_i \cup V_i, U_{i+1}, \dots, U_n) =$

$$\begin{aligned}
&= \{x \in X \mid \exists x_1 \in U_1, \dots, x_i \in U_i \cup V_i, \dots, x_n \in U_n : R(x_1, \dots, x_n, x)\} = \\
&= \{x \in X \mid \exists x_1 \in U_1, \dots, x_i \in U_i, \dots, x_n \in U_n : R(x_1, \dots, x_n, x)\} \cup \\
&\quad \cup \{x \in X \mid \exists x_1 \in U_1, \dots, x_i \in V_i, \dots, x_n \in U_n : R(x_1, \dots, x_n, x)\} = \\
&= f_R(U_1, \dots, U_{i-1}, U_i, U_{i+1}, \dots, U_n) \cup f_R(U_1, \dots, U_{i-1}, V_i, U_{i+1}, \dots, U_n).
\end{aligned}$$

(4) Let $Q \subseteq X^n$ be an decreasing relation. For every $U_1, \dots, U_n \in \mathcal{O}(X)$ let $g_Q(U_1, \dots, U_n) = \{x \in X \mid \forall x_1, \dots, x_n \text{ if } Q(x_1, \dots, x_n, x) \text{ then } \exists i \in \{1, \dots, n\}, x_i \in U_i\}$. We show that for every $U_1, \dots, U_n \in \mathcal{O}(X)$, $g_Q(U_1, \dots, U_n) \in \mathcal{O}(X)$. Let $x \leq y$. Assume that $x \in g_Q(U_1, \dots, U_n)$, i.e. for every $x_1, \dots, x_n$, if $Q(x_1, \dots, x_n, x)$, then there is an $i \in \{1, \dots, n\}$ with $x_i \in U_i$.

Let $x_1, \dots, x_n$ be such that $Q(x_1, \dots, x_n, y)$. Then, by the fact that Q is decreasing, $Q(x_1, \dots, x_n, x)$. This shows that $y \in g_Q(U_1, \dots, U_n)$. It is easy to see that for every $1 \leq i \leq n$, $f_R(U_1, \dots, U_{i-1}, X, U_{i+1}, \dots, U_n) = X$. Also,
$f_R(U_1, \dots, U_{i-1}, U_i \cap V_i, U_{i+1}, \dots, U_n) =$

$$\begin{aligned}
&= \{x \in X \mid \forall x_1, \dots, x_n \text{ if } Q(x_1, \dots, x_n, x) \\
&\quad \text{then either } x_j \in U_j \text{ for some } j \neq i, \text{ or } x_i \in U_i \cap V_i\} = \\
&= \{x \in X \mid \forall x_1, \dots, x_n \text{ if } Q(x_1, \dots, x_n, x) \text{ then } x_j \in U_j \text{ for some } j\} \cap \\
&\quad \cap \{x \in X \mid \forall x_1, \dots, x_n \text{ if } Q(x_1, \dots, x_n, x) \text{ then either} \\
&\qquad x_j \in U_j \text{ for some } j \neq i, \text{ or } x_i \in V_i\} = \\
&= f_R(U_1, \dots, U_{i-1}, U_i, U_{i+1}, \dots, U_n) \cup f_R(U_1, \dots, U_{i-1}, V_i, U_{i+1}, \dots, U_n).
\end{aligned}$$

□

This shows that the set $\mathcal{O}(X)$ of order-filters of X is closed under the operations in Σ, hence it is in particular a distributive lattice with operators.

The extension of the meaning function $m : \mathsf{Var} \to \mathcal{O}(X)$ is the unique morphism of distributive lattices with 0, 1, and operators in Σ, $\overline{m} : \mathsf{Fma}(\mathsf{Var}) \to \mathcal{O}(X)$ that extends m.

Definition 5.19 *We say that a $\mathcal{L}$-model $M = (K, m)$* satisfies *a formula ϕ at the state x (denoted by $M \overset{r}{\models}_x \phi$) if and only if $x \in \overline{m}(\phi)$.*

A formula ϕ is true in a $\mathcal{L}$-model $M = (K, m)$ (denoted by $M \stackrel{r}{\models} \phi$) if and only if $\overline{m}(\phi) = X$.

The formula ϕ is true in a $\mathcal{L}$-frame K (denoted by $K \stackrel{r}{\models} \phi$) if and only if it is true in every $\mathcal{L}$-model based on K.

Lemma 5.39 *Let A be an algebra in $\mathcal{V}$. Assume that $D(A) \stackrel{r}{\models} \phi$. Then $A \models \phi = 1$.*

Proof: Assume that $D(A) \stackrel{r}{\models} \phi$. Let $f : \mathsf{Var} \to A$ be an arbitrary assignment of values to the variables. Let $m_f : \mathsf{Var} \to \mathsf{ClopenOF}(D(A))$ be defined by $m_f(p) = \eta_A \circ f$. The unique extension of m_f to the formulae also has clopen order-filters as values, and by the universality property it is easy to see that $\overline{m_f} = \eta_A \circ \overline{f}$. Since $D(A) \stackrel{r}{\models} \phi$, it follows that $\overline{m_f}(\phi) = D(A)$, hence $\overline{f}(\phi) = 1$. □

The converse does not hold in general. However, it holds if A is finite, as shown in the next lemma.

Lemma 5.40 *Let A be a finite algebra in $\mathcal{V}$. Then $A \models \phi = 1$ if and only if $D(A) \stackrel{r}{\models} \phi$.*

Proof: Assume that $A \models \phi = 1$. We show that $D(A) \stackrel{r}{\models} \phi$. Let $m : \mathsf{Var} \to \mathcal{O}(D(A))$ be a meaning function. Since A is finite, the topology of $D(A)$ is discrete, hence the set of clopen order-filters of $D(A)$ coincides with the set $\mathcal{O}(D(A))$ of order-filters of $D(A)$, hence $m : \mathsf{Var} \to \mathsf{ClopenOF}(D(A))$. Let $f : \mathsf{Var} \to A$ be defined by $f = \eta_A^{-1} \circ m$. Since $A \models \phi = 1$, it follows that $\overline{f}(\phi) = 1$. It is easy to see that $\overline{f} = \eta_A^{-1} \circ \overline{m}$. Thus, $\eta_A^{-1}(\overline{m}(\phi)) = 1$, hence $\overline{m}(\phi) = D(A)$. The converse follows from Lemma 5.39. □

Corollary 5.41 *Assume that the logic $\mathcal{L}$ satisfies conditions (P1), (P2), (P3). Then*

$$\vdash_{\mathcal{L}} \phi \text{ if and only if for all } i = 1, \ldots, n, D(A_i) \stackrel{r}{\models} \phi.$$

5.3.3 Automated Theorem Proving

In this section we will show that the resolution procedure described in Section 5.2 can be extended to other finitely valued logics.

Let $\mathcal{L}$ be a propositional logic with the following properties:

(P1') $\mathcal{L}$ is sound and complete with respect to a variety of algebras $\mathcal{V}$, such that: $\mathcal{V} = HSP(A)$, where A is a finite algebra.

(P2) The algebras in $\mathcal{V}$ are distributive lattices with operators.

(P3) The Priestley duality between the category D_{01} of distributive lattices and the category P of Priestley spaces induces a dual equivalence between the variety $\mathcal{V}$ seen as a category and a corresponding category $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ of Priestley spaces endowed with additional functions and relations.

For every algebra $A \in \mathcal{V}$ we will denote its dual by $D(A)$, and the isomorphism between A and $\mathsf{ClopenOF}(D(A))$ by $\eta_A : A \to \mathsf{ClopenOF}(D(A))$.

For every operation symbol $f \notin \{\vee, \wedge, \Rightarrow, \neg\}$ in the language of $\mathcal{L}$, the corresponding operation or relation in the category $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ will be denoted $D(f)$.

We showed that a satisfiability relation $\overset{r}{\models}$ can be defined on the relational models. As noticed in Section 5.3.2, the satisfiability relation $\overset{a}{\models}$ for the algebras of $\mathcal{V}$ induces a satisfiability relation $\overset{rc}{\models}$ for the elements in $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$. These coincide on finite models. From Corollary 5.41 and assertion $(P1')$ it follows that for every propositional formula ϕ in the logic $\mathcal{L}$,

$$\mathcal{L} \vdash \phi \text{ if and only if } D(A) \overset{r}{\models} \phi.$$

As in the case of SHn-logics, for a given meaning function (interpretation) $m : \mathsf{Var} \to \mathcal{O}(D(A))$, we will introduce the abbreviations:

$\boxed{x}\ \phi^t$ means "ϕ is true at x" in the interpretation m (i.e. $D(A) \overset{r}{\models}_{m,x} \phi$),
$\boxed{x}\ \phi^f$ means "ϕ is false at x" in the interpretation m (i.e. $D(A) \overset{r}{\not\models}_{m,x} \phi$),

where x is a "possible world", i.e. an element of $D(A)$.

Note that if $\boxed{y}\ p^t$ at m and $y \leq x$ then $\boxed{x}\ p^t$ at m, and if $\boxed{y}\ p^f$ at m and $x \leq y$ then $\boxed{x}\ p^f$ at m.

5.3.4 Translation to Clause Form

First we introduce a notion of *signed literals* and *signed clauses.*

Definition 5.20 *Let $x \in D(A)$ be a "possible world" and p be an atom. Then* $\boxed{x}\ p^t$ *is a* positive literal (with sign $\boxed{x}$) *and* $\boxed{x}\ p^f$ *is a* negative literal (with sign $\boxed{x}$)*. A set of (positive or negative) signed literals is called a* (signed) clause. *A* formula in signed conjunctive normal form (CNF) *is a finite set of (signed) clauses. In the first-order case we require that the clauses in a formula have disjoint variables.*

Definition 5.21 *A propositional positive literal $\boxed{x}\ p^t$ is satisfiable if for some meaning function $m : \mathsf{Var} \to \mathcal{O}(D(A))$, p is true in m at x.*

A propositional negative literal $\boxed{x}\ p^f$ is satisfiable if for some meaning function $m : \mathsf{Var} \to \mathcal{O}(D(A))$, p is false in m at x.

A propositional signed clause is satisfiable if and only if at least one of its literals is satisfiable.

A signed formula Φ is satisfiable if and only if all clauses in Φ are simultaneously satisfiable by the same interpretation.

We present a structure-preserving translation to clause form, similar to the one presented in [Häh94].

Let Var be a countably infinite set of variables which will contain all the variables that will be considered in what follows. Let ϕ be a formula in the language of $\mathcal{L}$, $\phi \in \mathsf{Fma}(\mathsf{Var})$.

Lemma 5.42 *The formula ϕ is a theorem in $\mathcal{L}$ if and only if there is no valuation m with the property that ϕ is false at x in m for some minimal element x in $D(A)$.*

Proof: Assume first that ϕ is a theorem in $\mathcal{L}$. Then for every meaning function $m : \mathsf{Var} \to \mathcal{O}(D(A))$ and every $x \in D(A)$, ϕ is true at x in m. This holds in particular for the minimal elements of $D(A)$.

In order to prove the inverse implication, we assume there is no valuation m with the property that ϕ is false at x in m for some minimal element x in $D(A)$. Let $m : \mathsf{Var} \to \mathcal{O}(D(A))$ and $y \in D(A)$ be arbitrary but fixed. Since $D(A)$ is finite, there exists an element x which is minimal in $D(A)$ such that $x \leq y$. By the hypothesis ϕ is true at x in m. Therefore, ϕ is also true at y in m. □

The central idea behind structure-preserving clause form translations is to introduce additional atoms (resp. predicate letters in the case of first-order logic), which serve as abbreviations for subformulae of the input. It remains to translate the formulae that represent the definitions of the new literals.

Let p_ϕ be a new propositional variable introduced for the formula ϕ.

Lemma 5.43 *The formula ϕ is a theorem if and only if there is no valuation m such that $\overline{m}(\phi) = m(p_\phi)$ and $\boxed{x}\, p_\phi^f$ at m for some minimal element $x \in D(A)$.*

Definition 5.22 *Let ϕ_1, ϕ_2 be two formulae, and let $m : \mathsf{Var} \to \mathcal{O}(D(A))$ be a valuation. We say that ϕ_1 and ϕ_2 are equivalent in m (denoted $\phi_1 \equiv_m \phi_2$ if they are true at the same states, i.e. if $\overline{m}(\phi_1) = \overline{m}(\phi_2)$). We say that $\phi_1 \equiv \phi_2$ is satisfiable if there is a valuation m such that $\phi_1 \equiv_m \phi_2$.*

Lemma 5.44 *The relations $\equiv$ and $\equiv_m$ have the following properties:*

(1) Let m be a valuation, and ϕ_1, ϕ_2 be formulae such that $\phi_1 \equiv_m \phi_2$. Then for every $x \in D(A)$, $\boxed{x}\, \phi_1^t$ at m if and only if $\boxed{x}\, \phi_2^t$ at m, and $\boxed{x}\, \phi_1^f$ at m if and only if $\boxed{x}\, \phi_2^f$ at m.

(2) $\phi_1 \equiv \phi_2$ is satisfiable if and only if there exists a valuation m such that for all $x \in D(A)$, $(\boxed{x}\, \psi_1^f \vee \boxed{x}\, \psi_2^t) \wedge (\boxed{x}\, \psi_1^t \vee \boxed{x}\, \psi_2^f)$ at m.

The proof is similar to the proof of Lemma 5.27.

Therefore, we can reduce the task of proving that a formula ϕ is a theorem in $\mathcal{L}$ to the task of proving that for no valuation $m : \mathsf{Var} \to \mathcal{O}(D(A))$ we have $\boxed{x}\, p^f_\phi$ at m for some minimal element $x \in D(A)$ and $p_\phi \equiv_m \phi$.

Corollary 5.45 *A formula ϕ is a theorem if and only if there is no valuation m such that*

$$\begin{cases} (1) & \bigvee_{x \text{ minimal in } D(A)} \boxed{x}\, p^f_\phi & \textit{at } m \\ (2_x) & (\boxed{x}\, \phi^f \vee \boxed{x}\, p^t_\phi) \wedge (\boxed{x}\, \phi^t \vee \boxed{x}\, p^f_\phi) & \textit{at } m \\ & \textit{for every } x \in D(A) & \end{cases}$$

Note that in (2_x), expressions as $\boxed{x}\, \phi^t$ resp. $\boxed{x}\, \phi^f$ occur. In order to obtain a clause form we have to (recursively) eliminate these expressions.

Lemma 5.46 *Let m be a valuation, and let σ be an operation symbol in $\mathcal{L}$ of arity n. Then every formula $\phi = \sigma(\psi_1, \dots, \psi_n)$ is equivalent in m to a formula of the form $\sigma(p_{\psi_1}, \dots, p_{\psi_n})$ where $p_{\psi_i} \equiv_m \psi_i$ for $i = 1, \dots, n$.*

Moreover, for every $x \in D(A)$, $\boxed{x}\, \sigma(\psi_1, \dots, \psi_n)^t$ at m if and only if $\boxed{x}\, \sigma(p_{\psi_1}, \dots, p_{\psi_n})^t$ at m, where $p_{\psi_1} \equiv_m \psi_1$ and $p_{\psi_2} \equiv_m \psi_2$.

Corollary 5.47 *ϕ is a theorem in $\mathcal{L}$ if and only if the following conjunction of formulae is unsatisfiable:*

$$\left(\bigvee_{x \text{ minimal in } D(A)} \boxed{x}\, p^f_\phi \right) \wedge \bigwedge_{x \in D(A)} \bigwedge_{\psi \text{ subformula of } \phi} (\boxed{x} p^f_\psi \vee \boxed{x} \psi^t) \wedge (\boxed{x} p^t_\psi \vee \boxed{x} \psi^f)$$

In order to obtain a conjunctive normal form, we have to analyze the way in which expressions of the form $\boxed{x}\psi^t$ and $\boxed{x}\psi^f$ can be decomposed further, depending on the structure of ψ, taking into accounnt Lemma 5.46.

Lemma 5.48 *The following holds, for any given valuation m:*

(Disj t)	$\boxed{x}\, (p_1 \vee p_2)^t$	*iff*	$(\boxed{x}\, p^t_1) \vee (\boxed{x}\, p^t_2)$.
(Disj f)	$\boxed{x}\, (p_1 \vee p_2)^f$	*iff*	$(\boxed{x}\, p^f_1) \wedge (\boxed{x}\, p^f_2)$.
(Conj t)	$\boxed{x}\, (p_1 \wedge p_2)^t$	*iff*	$(\boxed{x}\, p^t_1) \wedge (\boxed{x}\, p^t_2)$.
(Conj f)	$\boxed{x}\, (p_1 \wedge p_2)^f$	*iff*	$(\boxed{x}\, p^f_1) \vee (\boxed{x}\, p^f_2)$.

For expressing the transformation rules for the other operations we first point out the form of the extension of the meaning function to formulae.

Lemma 5.49 *Let σ be an operation symbol in Σ with arity n. Then*

$$\overline{m}(\sigma(p_1, \dots, p_n)) = \sigma_{\mathcal{O}(D(A))}(m(p_1), \dots, m(p_n)).$$

Proof: We use the fact that A as well as $\mathcal{O}(D(A))$ are algebras of the same similarity type, and that η_A and η_A^{-1} are mutually inverse homomorphisms.

Hence,
$$\begin{aligned}\overline{m}(\sigma(p_1,\dots,p_n)) &= \eta_A \circ \overline{\eta_A^{-1} \circ m}(\sigma(p_1,\dots,p_n)) = \\ &= \eta_A(\overline{\eta_A^{-1} \circ m}(\sigma(p_1,\dots,p_n))) = \\ &= \eta_A(\sigma_A(\eta_A^{-1}(m(p_1)),\dots,\eta_A^{-1}(m(p_n)))) = \\ &= \eta_A(\eta_A^{-1}(\sigma_{\mathcal{O}(D(A))}(m(p_1),\dots,m(p_n)))) = \\ &= \sigma_{\mathcal{O}(D(A))}(m(p_1),\dots,m(p_n)).\end{aligned}$$
□

The way the translation to clause form proceeds further depends on the way the operations are defined on $\mathcal{O}(D(A))$.

For example, assume that the algebras in $\mathcal{V}$ additionally have a Heyting algebra structure, we can use the fact that the elements of $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ are Heyting spaces.

Lemma 5.50 *Assume that the members of $\mathcal{V}$ have a Heyting algebra structure and that the duality between $\mathcal{V}$ and $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$ is induced by the Priestley duality for Heyting algebras. In this case, for any given valuation m:*

$$\begin{array}{llll}
(\Rightarrow,\ t) & \boxed{x}\ (p_1 \Rightarrow p_2)^t & \textit{iff} & \textit{for every } y \geq x,\ \boxed{y}\ p_1^f \vee \boxed{y}\ p_2^t. \\
(\Rightarrow,\ f) & \boxed{x}\ (p_1 \Rightarrow p_2)^f & \textit{iff} & \left(\bigvee_{m \geq x, maximal} \boxed{m}\ p_1^t\right),\ \boxed{x}\ p_2^f \textit{ and} \\
 & & & \bigwedge_{\substack{S_1,S_2 \neq \emptyset, S_1 \cap S_2 = \emptyset \\ S_1 \cup S_2 = \{y \mid y \geq x\}}} \left(\bigvee_{y_M \in S_1, maximal} \boxed{y_M}\ p_1^t\right) \vee \\
 & & & \qquad \left(\bigvee_{y_m \in S_2, minimal} \boxed{y_m}\ p_2^f\right) \\
(\neg,\ t) & \boxed{x}\ (\neg p)^t & \textit{iff} & \textit{for every } y \geq x,\ \boxed{y}\ p^f. \\
(\neg,\ f) & \boxed{x}\ (\neg p)^f & \textit{iff} & \textit{there is a } y, y \geq x \textit{ with } \boxed{y}\ p^t.
\end{array}$$

Proof: It is easy to see that:
$$\begin{aligned}\overline{m}(p_1 \Rightarrow p_2) &= \eta_A(\overline{\eta_A^{-1} \circ m}(p_1 \Rightarrow p_2)) = \\ &= \eta_A(\overline{\eta_A^{-1} \circ m}(p_1) \Rightarrow \overline{\eta_A^{-1} \circ m}(p_2)) = \\ &= \eta_A(\eta_A^{-1}(m(p_1)) \Rightarrow \eta_A^{-1}(m(p_2))) = \\ &= (m(p_1) \Rightarrow m(p_2)) = \\ &= \{x \in D(A) \mid \text{ for all } y, y \geq x \text{ and } y \in m(p_1) \text{ implies } y \in m(p_2)\}.\end{aligned}$$
$(\Rightarrow$, t), $(\neg$, t) and $(\neg$, f) follow immediately from this.

In order to prove $(\Rightarrow$, t) note that $\boxed{x}\ (p_1 \Rightarrow p_2)^f$ if and only if $x \notin \overline{m}(p_1 \Rightarrow p_2)$, which holds if and only if there exists at least one $y \geq x$ such that $y \in m(p_1)$ and $y \notin m(p_2)$.

Thus, $\boxed{x}\ (p_1 \Rightarrow p_2)^f$ if and only if $\bigvee_{y \geq x}(\boxed{y}\ p_1^t \wedge \boxed{y}\ p_2^f)$ if and only if (by distributivity)

$$\bigwedge_{\substack{S_1,S_2,S_1 \cap S_2 = \emptyset \\ S_1 \cup S_2 = \{y \mid y \geq x\}}} \left(\bigvee_{y_1 \in S_1} \boxed{y_1}\ p_1^t \vee \bigvee_{y_2 \in S_2} \boxed{y_2}\ p_2^f\right)$$

Taking into account that for every $S_1 \subseteq \{y \mid y \geq x\}$ $\bigvee_{y_1 \in S_1} \boxed{y_1}\ p_1^t$ if and only if $\bigvee_{y_M \in S_1, maximal} \boxed{y_M}\ p_1^t$, and $\bigvee_{y_2 \in S_2} \boxed{y_2}\ p_2^f$ if and only if $\bigvee_{y_m \in S_2, minimal} \boxed{y_m}\ p_2^f$, we obtain

$$\boxed{x}\ (p_1 \Rightarrow p_2)^f$$

iff

$$\left(\bigvee_{y_M \geq x, maximal} \boxed{y_M}\, p_1^t\right) \wedge$$

$$\bigwedge_{\substack{S_1,S_2\neq\emptyset, S_1\cap S_2=\emptyset\\ S_1\cup S_2=\{y|y\geq x\}}} \left(\bigvee_{y_M\in S_1, maximal}\boxed{y_M}\, p_1^t \vee \bigvee_{y_m\in S_2, minimal}\boxed{y_m}\, p_2^f\right)\wedge \boxed{x}\, p_2^f.$$

□

Lemma 5.51 *The following hold for any valuation m:*

(Lh) Let h be an operator on the logic $\mathcal{L}$ such that $h_A : A \to A$ is a lattice homomorphism. Let $H_A = D(h_A)$ be the corresponding operation on $D(A)$. Then $x \in \overline{m}(h(p))$ iff $H_A(x) \in m(p)$.

(La) Let k be an operator on the logic $\mathcal{L}$ such that $k_A : A \to A$ is a lattice antimorphism. Let $K_A = D(k_A)$ be the corresponding operation on $D(A)$. Then $x \in \overline{m}(h(p))$ if and only if $K_A(x) \notin m(p)$.

(Jh) Let f be an operator on the logic $\mathcal{L}$ such that $f_A : A \to A$ is a join hemimorphism. Let $R_A = D(f_A)$ be the corresponding relation on $D(A)$. Then $x \in \overline{m}(f(p_1, \dots, p_n))$ if and only if $\exists x_1 \in m(p_1), \dots, x_n \in m(p_n)$ such that $(x_1, \dots, x_n, x) \in R_A$.

(Mh) Let g be an operator on the logic $\mathcal{L}$ such that $g_A : A \to A$ is a meet hemimorphism. Let $Q_A = D(g_A)$ be the corresponding relation on $D(A)$. Then $x \in \overline{m}(g(p_1, \dots, p_n))$ if and only if $\forall x_1, \dots, x_n$ such that $(x_1, \dots, x_n, x) \in Q_A$ there exists an $i \in \{1, \dots, n\}$ such that $x_i \in m(p_i)$.

Proof:
(Lh) $\overline{m}(h(p)) = \eta_A(\overline{\eta_A^{-1} \circ m}(h(p)) = \eta_A(h_A(\eta_A^{-1}(m(p)))) =$
$= h_{\mathcal{O}(D(A))}(m(p)) = \{x \in X \mid H_A(x) \in m(p)\}$.
Hence, $x \in \overline{m}(h(p))$ if and only if $x \in h_{\mathcal{O}(D(A))}(m(p))$, if and only if $H_A(x) \in m(p)$.
(La) $\overline{m}(k(p)) = \eta_A(\overline{\eta_A^{-1} \circ m}(k(p)) = \eta_A(k_A(\eta_A^{-1}(m(p)))) = k_{\mathcal{O}(D(A))}(m(p))$.
Hence, $x \in \overline{m}(k(p))$ if and only if $K_A(x) \notin m(p)$.
(Jh) $\overline{m}(f(p_1, \dots, p_n)) = \eta_A(\overline{\eta_A^{-1} \circ m}(f(p_1, \dots, p_n)) =$
$= \eta_A(f_A(\eta_A^{-1}(m(p_1)), \dots, \eta_A^{-1}(m(p_n)))) =$
$= f_{\mathcal{O}(D(A))}(m(p_1), \dots, m(p_n)) =$
$= f_{R_f}(m(p_1), \dots, m(p_n)) =$
$= \{x \in D(A) \mid \exists x_1, \dots x_n$, with $x_1 \in m(p_1), \dots, x_n \in m(p_n)$ and $(x_1, \dots, x_n, x) \in R_f\}$.
The conclusion follows immediately.
(Mh) $\overline{m}(g(p_1, \dots, p_n)) = \eta_A(\overline{\eta_A^{-1} \circ m}(g(p_1, \dots, p_n)) =$
$= \eta_A(g_A(\eta_A^{-1}(m(p_1)), \dots, \eta_A^{-1}(m(p_n)))) =$
$= g_{\mathcal{O}(D(A))}(m(p_1), \dots, m(p_n)) =$
$= g_{Q_f}(m(p_1), \dots, m(p_n)) =$

$$= \{x \in D(A) \mid \forall x_1, \dots x_n, \text{ with } (x_1, \dots, x_n, x) \in R_f, \\ \exists i \in \{1, \dots, n\}, \text{ such that } x_i \in m(p_i)\}. \qquad \Box$$

We therefore obtain a procedure for translation to clause form. The rules for translating disjunction, conjunction, and Heyting negation are those presented in Section 5.2. For translating the Heyting implication we take into account the rules $(\Rightarrow, t)$ and $(\Rightarrow, f)$.

For the other operations we have the following transformation rules:

$$(Lh): \ (h,t) \ \frac{L \vee \boxed{x}\ h(p)^t}{\left\{L, \boxed{H_h(x)}\ p^t\right\}} \qquad (h,f) \ \frac{L \vee \boxed{x}\ h(p)^f}{\left\{L, \boxed{H_h(x)}\ p^f\right\}}$$

$$(La): \ (k,t) \ \frac{L \vee \boxed{x}\ k(p)^t}{\left\{L, \boxed{K_k(x)}\ p^f\right\}} \qquad (k,f) \ \frac{L \vee \boxed{x}\ k(p)^f}{\left\{L, \boxed{K_k(x)}\ p^t\right\}}$$

Consider for example a unary join-hemimorphism $\Diamond : A \to A$ (similar to the modal operator for "possibility"). The rule (Jh) described above specializes in this case to:

$$(\Diamond, t) \ \frac{L \vee \boxed{x}\ \Diamond(p)^t}{\left\{L, \boxed{x_1}\ p_1^t, (x_1, x) \in D(\Diamond)\right\}} \qquad (\Diamond, f): \ \frac{L \vee \boxed{x}\ \Diamond(p)^f}{\bigwedge_{(x_1,x) \in D(\Diamond)} \left\{L, \boxed{x_1}\ p_1^f\right\}}$$

Consider now a unary meet-homomorphism $\Box : A \to A$ (similar to the modal operator for "necessity"). The rule (Mh) specializes in this case to:

$$(\Box, t) \ \frac{L \vee \boxed{x}\ \Box(p)^t}{\bigwedge_{(x_1,x) \in Q\Box} \left\{L, \boxed{x_1}\ p_1^t\right\}} \qquad (\Box, f): \ \frac{L \vee \boxed{x}\ \Box(p)^f}{\left\{L, \boxed{x_1}\ p_1^f, (x_1, x) \in Q\Box\right\}}$$

In general the number of clauses generated by arbitrary meet- and join-hemimorphisms can be quite high. The next proposition gives an estimate of the number of clauses generated from a given formula ϕ (we assume that only unary meet- and join-hemimorphisms occur in ϕ).

Proposition 5.52 *Let n be the number of elements of $D(A)$. The following holds:*

(1) If the formula ϕ only contains the connectives $\wedge, \vee$ and g such that g is interpreted as a lattice morphism or antimorphism on A, then the number of clauses generated from ϕ is $O(ln)$ where l is the number of subformulae of ϕ.

(2) If the formula ϕ only contains the connectives $\wedge, \vee, \neg$ and g such that $\neg$ is interpreted as a Heyting negation on A and g is interpreted as a lattice morphism or antimorphism, (and possibly unary join- or meet-hemimorphisms) then the number of clauses generated from ϕ is $O(ln^2)$.

(3) *If the formula ϕ only contains the connectives $\wedge, \vee, \neg, \Rightarrow$ and g such that $\neg$ is interpreted as a Heyting negation, $\Rightarrow$ as a Heyting implication on A, and g as a lattice morphism or antimorphism, then the number of clauses generated from ϕ is $O(ln2^n)$.*

Proof: The number of clauses generated from a given formula ϕ is

$$1+\sum_{\psi \text{ subformula of } \phi}\ \sum_{x \in D(S_{n^2})} |\text{ clauses}(\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t)| + |\text{ clauses}(\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f)|.$$

(1) Assume that ϕ only contains the connectives $\vee, \wedge$ and some morphisms and/or antimorphisms. In this case, for every $x \in D(S_{n^2})$ the sum of the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) and the number of clauses generated by ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) is at most 3. Thus, the number of clauses generated from ϕ has as upper bound $1 + 3nl$, so it is $O(nl)$.

(2) Assume that ϕ only contains the connectives $\vee, \wedge, \neg$, some morphisms and/or antimorphisms and possibly unary meet- and/or join-hemimorphisms. In this case, for every $x \in D(S_{n^2})$ the sum of the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) and the number of clauses generated by ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) is not larger than $n+1$. Note that rules $(\neg, t), (h, f), (g, t)$ – where h is a meet-hemimorphism and g a join-hemimorphism – introduce maximally n clauses, whereas rules $(\neg, f), (h, t), (g, f)$ introduce only one clause. Thus, the number of clauses generated from ϕ has as upper bound $1+nl(n+1)$, hence it is $O(n^2 l)$.

(3) If also the operator $\Rightarrow$ appears in ϕ, then for every $x \in D(S_{n^2})$ the number of clauses generated by ($\boxed{x}\, p_\psi^f \vee \boxed{x}\, \psi^t$) is less than or equal to the number of elements in $D(S_{n^2})$ smaller than x, and the number of clauses generated by ($\boxed{x}\, p_\psi^t \vee \boxed{x}\, \psi^f$) is less than or equal to the number of subsets of $\{y \mid y \geq x\}$. Thus, the number of clauses generated from a given formula ϕ has as upper bound $1 + nl(n + 2^n + 1)$, hence it is $O(n2^n l)$ (if the order in $D(A)$ has some good properties, as in the case of SHn-logics, less clauses are generated). $\square$

5.3.5 A Resolution Procedure

In this section we show that also in this more general context a hyperresolution procedure can be formulated.

Negative Hyperresolution

$$\frac{\left\{\boxed{x_1}\, p_1^f\right\} \cup D_1, \ldots, \left\{\boxed{x_n}\, p_n^f\right\} \cup D_n, \left\{\boxed{y_1}\, p_1^t, \ldots, \boxed{y_n}\, p_n^t\right\} \cup E}{D_1 \cup \ldots \cup D_n \cup E}$$

provided that $n \geq 1$, $y_i \leq x_i$ for all $i = 1, \ldots, n$ and $D_1, \ldots, D_n, E$ are negative.

The following theorem shows that the negative hyperresolution principle can be applied.

Theorem 5.53 *Let Φ be a set of clauses. Then Φ is unsatisfiable if and only if $\Box$ can be derived from Φ by a finite number of applications of many-valued negative hyperresolution.*

The proof given in Section 5.2.2 holds also in this more general case.

Theorem 5.54 *Let $\mathcal{L}$ be a (finitely valued) propositional logic that has the following properties:*

(1) $\mathcal{L}$ is sound and complete with respect to a variety $\mathcal{V}$ of algebras with a distributive lattice reduct,

(2) The variety $\mathcal{V}$ is generated by a finite algebra A,

(3) The Priestley duality extends to a dual equivalence between the variety $\mathcal{V}$ and a suitably chosen subcategory of the category of Priestley spaces (with additional operators and relations corresponding to the operations in the logic $\mathcal{L}$),

(4) For every formula ϕ in the language of $\mathcal{L}$ there is an equivalent set Φ of clauses signed by elements in $D(A)$.

In this case, a formula ϕ is a theorem if and only if $\Box$ can be derived from the set of clauses Φ by a finite number of applications of many-valued negative hyperresolution.

5.3.6 An Approach to Automated Theorem Proving in First-Order Logic

We briefly present here an extension to first-order logic.

Let $\Sigma = \{\wedge, \vee, \sigma_1, \ldots, \sigma_r\}$ be a signature, where $\wedge, \vee$ are binary operations and for every $i \in \{1, \ldots, r\}$, σ_i has arity n_i. Let A be a finite algebra in a variety $\mathcal{V}$ of Σ-algebras with an underlying distributive algebra structure. Assume that $\mathcal{V}$ is generated by A, and that the Priestley duality for distributive lattices extends to a dual equivalence between $\mathcal{V}$ and a category of suitable Priestley spaces $\mathcal{V}_{\mathcal{L}}\mathsf{Sp}$. The dual of A will be denoted $D(A)$.

In what follows we present a possibility of defining first-order many-valued logics based on A (i.e. having A as a set of truth values).

Syntax

Consider the language of a first-order logic $\mathcal{L}$, consisting of:

(1) An infinite (countable) set X of variables.

(2) A set O of function symbols;

(3) A set P of predicate symbols;

(4) A set $\Sigma = \{\wedge, \vee, \sigma_1, \ldots, \sigma_r\}$ of logical operators, including the binary operators $\wedge, \vee$;

(5) A finite set of (one-place) quantifiers $Q_1, \ldots, Q_k$;

Let $\mathsf{Term}_O(X)$ be the set of terms of the language $\mathcal{L}_A$, i.e. the free O-algebra freely generated by X, and let Term^0_O be the set of *ground terms*[4] of the language $\mathcal{L}_A$. Let $At_{\mathcal{L}}(X)$ be the set of atomic formulae of the language of $\mathcal{L}$, i.e. the set of all the expressions of the form $R(t_1, \ldots, t_n)$ where R is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms, and let $\mathsf{Fma}(\mathcal{L})$ be the free algebra of similarity type $(\wedge, \vee, \sigma_1, \ldots, \sigma_r, (Q_1 x)_{x\in X}, \ldots, (Q_k x)_{x\in X})$ freely generated by the set of atomic formulae of the language of $\mathcal{L}$.

Semantics

The (finite) algebra A can be regarded as a set of "truth values" for the logic $\mathcal{L}$. The operations σ_A for $\sigma \in \Sigma$ can be regarded as "truth tables" in A. The quantifiers can also be defined by "truth tables". Thus:

- For every $\sigma \in \Sigma$ with arity n we associate a truth function $\sigma_A : A^n \to A$.
- For every quantifier Q we associate a truth function $\overline{Q} : \mathcal{P}(A)\backslash\emptyset \to A$.

Referring to a formula $Q(x)\phi(x)$ truth functions can be understood as follows, in a given domain D of interpretation: Let $M \subseteq A$ be the set

$$\{w \mid \exists d \in D \text{ such that } w \text{ is assigned to } \phi(d)\}.$$

The quantifier Q associates a truth value $\overline{Q}(M) \in A$ to each such set.

Definition 5.23 (Frame, Interpretation) *A* frame[5] *for a language (consisting of a set of variables X, a set of operation symbols O, a set of predicate symbols P, a set of logical connectors Σ, and a set of quantifiers $\{Q_1, \ldots, Q_k\}$) and a set of truth values A is a pair (D, I) where:*

(1) D is a non-empty set, the domain, *and*

(2) I is a signature interpretation, i.e. a function assigning a function $I(f) : D^n \to D$ to every n-ary function symbol $f \in O$, and a function $I(R) : D^n \to A$ to every n-ary predicate symbol $R \in P$.

An interpretation *$\mathcal{I}$ for a language $(P, O, \Sigma, \{Q_1, \ldots, Q_k\}, X)$ and a set of truth values A is a triple (D, I, d) where (D, I) is a frame and d is a* variable assignment *$d : X \to D$.*

[4] A ground term is a term that does not contain any variable.

[5] This definition and the name *frame* is taken from [BF95]. Note the difference between this notion of frames and the notion defined in Sections 5.1.2 and 5.3.2.

The idea of a generalization of the algebraic treatment of formulae of propositional calculi to formulae of predicate calculi is due to Mostowski [Mos48] pp.204-207 who introduced an algebraic interpretation of formulae of intuitionistic calculi (the quantifiers $\forall$ and $\exists$ are interpreted as greatest lower bound resp. least upper bound). The approach in [BF95] is an extension of Mostowski's approach in that arbitrary quantifiers (not only $\forall$ and $\exists$) are considered, and they are interpreted as maps from the family of nonempty subsets of A to A.

Examples of frames for a language are the *Herbrand* frames (called *Herbrand structures* in [Häh96b]). The domain of a Herbrand frame is the set $H = \mathsf{Term}^0_O$ of all ground terms (if the language contains no constant one constant is added); the signature interpretation I_H assigns

- to every n-ary function symbol f the map $I_H(f) : H^n \to H$ defined by $I_H(f)(h_1, \ldots, h_n) = f(t_1, \ldots, t_n)$ (seen as a ground term),
- to every n-ary predicate symbol R a map $I_H(R) : H^n \to A$.

Every interpretation $\mathcal{I} = (D, I, d)$ induces a valuation function $v_{\mathcal{I}} : \mathsf{Fma}(\mathcal{L}) \to A$ as follows:

(1) $v_{\mathcal{I}}(x) = d(x)$ for all variables $x \in X$,

(2) $v_{\mathcal{I}}(f(t_1, \ldots, t_n)) = I(f)(v_{\mathcal{I}}(t_1), \ldots, v_{\mathcal{I}}(t_n))$ for all n-ary function symbols $f \in O$, $n \geq 0$,

(3) $v_{\mathcal{I}}(R(t_1, \ldots, t_n)) = I(R)(v_{\mathcal{I}}(t_1), \ldots, v_{\mathcal{I}}(t_n))$ for all n-ary predicate symbols $R \in P$, $n \geq 0$,

(4) $v_{\mathcal{I}}(\sigma(\phi_1, \ldots, \phi_n)) = \sigma_A(v_{\mathcal{I}}(\phi_1), \ldots, v_{\mathcal{I}}(\phi_n))$ for all logical operators $\sigma \in \Sigma$,

(5) $v_{\mathcal{I}}((Qx)\phi) = \overline{Q}(\{w \mid \exists d \in D \text{ s.t. } v_{\mathcal{I}_{x,d}}(\phi) = w\})$ for all quantifiers Q, where $\mathcal{I}_{x,d}$ is identical with $\mathcal{I}$ except for assigning d to the variable x.

A *Herbrand interpretation* is an interpretation in a Herbrand structure.

Example 5.3 *For any Herbrand interpretation $\mathcal{I}$ we have*

$$v_{\mathcal{I}}(\forall x \phi(x)) = \bigwedge \{v_{\mathcal{I}_{x,t}}(\phi) \mid t \text{ ground term}\},$$

$$v_{\mathcal{I}}(\exists x \phi(x)) = \bigvee \{v_{\mathcal{I}_{x,t}}(\phi) \mid t \text{ ground term}\}.$$

Definition 5.24 (Validity, Satisfiability) *A formula ϕ is* valid *in a logic $\mathcal{L}$ iff for all interpretations $\mathcal{I}$ for the language of $\mathcal{L}$ and A, $v_{\mathcal{I}}(\phi) = 1$.*

A formula ϕ is satisfiable *in a logic $\mathcal{L}$ iff there is an interpretation $\mathcal{I}$ for the language of $\mathcal{L}$ and A with $v_{\mathcal{I}}(\phi) = 1$.*

The definition can be formulated in a more general way, allowing a set of "designated elements" $A_d \subseteq A$: a formula ϕ is valid in $\mathcal{L}$ if and only if for all interpretations $\mathcal{I}$ for $\mathcal{L}_A$ and A, $v_{\mathcal{I}}(\phi) \in A_d$, and it is satisfiable in $\mathcal{L}$ if and only if there is an interpretation $\mathcal{I}$ for $\mathcal{L}_A$ and A with $v_{\mathcal{I}}(\phi) \in A_d$. However, in what follows we will focus on the notion of validity resp. satisfiability given in Definition 5.24.

Note that for every $f : \mathsf{Fma}(\mathcal{L}) \to A$, there exists a map $m_f = \eta_A \circ f : \mathsf{Fma}(\mathcal{L}) \to \mathcal{O}(D(A))$. Then $f(\phi) = 1$ if and only if $m_f(\phi) = D(A)$. Reciprocally, for every $m : \mathsf{Fma}(\mathcal{L}) \to \mathcal{O}(D(A))$, we can define $f_m : \mathsf{Fma}(\mathcal{L}) \to A$ by $f_m = \eta_A^{-1} \circ m$. We have $f_m(\phi) = 1$ if and only if $m(\phi) = D(A)$.

This suggests that one can introduce an alternative notion of interpretation, namely as a triple $\mathcal{M} = (D, M, d)$, where $d : X \to D$, $M(f) : D^n \to D$ for every function symbol of arity n and $M(R) : D^n \to \mathcal{O}(D(A))$ for every predicate symbol of arity n. Because of the isomorphism between A and $\mathcal{O}(D(A))$ (since A is finite), these two notions of interpretation are equivalent.

Interpretations in $\mathcal{O}(D(A))$ extend to formulae in a similar way as interpretations in A.

Proposition 5.55 *A formula ϕ is valid in $\mathcal{L}$ if and only if for every interpretation $\mathcal{M} = (D, M, d)$ in $\mathcal{O}(D(A))$, $v_{\mathcal{M}}(\phi) = D(A)$.*

Like in the propositional case, given an interpretation $\mathcal{M} = (D, M, d)$, where $d : X \to D$, $M(f) : D^n \to D$ for every function symbol of arity n and $M(P) : D^n \to \mathcal{O}(D(A))$ for every predicate symbol of arity n, we will now consider signed formulae (in classical logic) of the type:

$$\boxed{\alpha}\ \phi^t \text{ for "}\phi\text{ is true at }\alpha\text{" in } \mathcal{M} \text{ (i.e. } \alpha \in v_{\mathcal{M}}(\phi))$$
$$\boxed{\alpha}\ \phi^f \text{ for "}\phi\text{ is false at }\alpha\text{" in } \mathcal{M} \text{ (i.e. } \alpha \notin v_{\mathcal{M}}(\phi))$$

where α is a "possible world", i.e. an element of $D(A)$.

Note that if $\alpha \leq \beta \in D(A)$ and $\boxed{\alpha}\ \phi^t$ in $\mathcal{M}$ then $\boxed{\beta}\ \phi^t$ in $\mathcal{M}$.

Lemma 5.56 *The formula ϕ is valid in $\mathcal{L}$ if and only if there is no interpretation $\mathcal{M} = (D, M, d)$ of $\mathcal{L}$ in $\mathcal{O}(D(A))$*

$$\bigvee_{\alpha \in D(A),\ minimal} \boxed{\alpha}\ \phi^f \ \text{in } \mathcal{M}.$$

Proof: It follows from the fact that ϕ is not valid if and only if there exists an interpretation $\mathcal{M} = (D, M, d)$ such that $v_{\mathcal{M}}(\phi) \neq D(A)$, i.e. if and only if there exists an interpretation $\mathcal{M} = (D, M, d)$ and an $\alpha \in D(A)$ minimal such that $\alpha \notin v_{\mathcal{M}}(\phi)$. □

Translation to clause form

We first present a procedure for translation to clause form using structure-preserving rules. We present, as in the propositional case, a *structure-preserving* transformation method (the method presented here is inspired by the structure-preserving transformation method to clause form in first-order logic given in [Ede92], see also [BF95] and [Häh94]).

Let ϕ be a formula. The main idea is to introduce for every non-atomic subformula ψ of ϕ a new atomic formula of the form $P_\psi(\overline{x})$, where P_ψ is a new predicate symbol and $\overline{x}$ are the free variables in ψ.

Lemma 5.57 *Let $\mathcal{I} = (D, I, d)$ be an interpretation of the language (X, O, P) of the logic $\mathcal{L}$. Let ϕ be a fomula in $\mathcal{L}$ and let (X, O, P^*) the language obtained by introducing a new predicate symbol P_ψ for every non-atomic subformula ψ of ϕ.*

Let $\mathcal{I}^* = (D^*, I^*, d^*)$ be the interpretation for the language (X, O, P^*) defined inductively as follows:

(1) $D^* = D$,

(2) $I^*(f) = I(f)$ for every function symbol $f \in O$,

(3) $I^*(R) = I(R)$ for every predicate symbol $R \in P$,

For every non-atomic subformula $\psi = \sigma(\psi_1, \dots, \psi_m)$ of ϕ with free variables $\{x_1, \dots, x_n\}$, $I^*(P_\psi) : D^n \to A$ is defined for every $d_1, \dots, d_n \in D$ by $I^*(P_\psi)(d_1, \dots, d_n) = \sigma_A(v_{\mathcal{I}^*_{x_i/d_i}}(P_{\psi_1}(x_1, \dots, x_n)), \dots, v_{\mathcal{I}^*_{x_i/d_i}}(P_{\psi_n}(x_1, \dots, x_n)))$,

For every non-atomic subformula $\psi = (Qx)\psi_1(x, x_1, \dots, x_n)$ of ϕ with free variables $\{x_1, \dots, x_n\}$, $I^*(P_\psi) : D^n \to A$ is defined for every $d_1, \dots, d_n \in D$ by $I^*(P_\psi)(d_1, \dots, d_n) = v_{\mathcal{I}^*{}_{x_i/d_i}}((Qx)P_\psi(x_1, \dots, x_n))$.

(4) $d^* = d : X \to D$.

Then for every subformula ψ of ϕ with free variables $x_1, \dots, x_n$, and for every $d_1, \dots, d_n \in D$, $v_{\mathcal{I}^*_{x_i/d_i}}(P_\psi(x_1, \dots, x_n)) = v_{\mathcal{I}_{x_i/d_i}}(\psi)$.

Proof: We proceed by structural induction on the structure of ψ.

If ψ is an atomic subformula of ϕ of the form $\psi = R(t_1, \dots, t_n)$ then obviously $v_{\mathcal{I}^*}(R(t_1, \dots, t_n)) = v_{\mathcal{I}}(\psi)$.

Moreover, if $x_1, \dots, x_n$ are the free variables in ψ, then for every $d_1, \dots, d_n \in D$, $v_{\mathcal{I}^*_{x_i/d_i}}(R(t_1, \dots, t_n)) = v_{\mathcal{I}_{x_i/d_i}}(\psi)$.

Let now ψ be a subformula of ϕ. Assume that for every subformula ψ' of ψ with free variables $x_1, \dots, x_n$, $v_{\mathcal{I}^*_{x_i/d_i}}(P_{\psi'}(x_1, \dots, x_n)) = v_{\mathcal{I}_{x_i/d_i}}(\psi')$ for every $d_1, \dots, d_n \in D$. We distinguish the following cases:

Case 1: $\psi = \sigma(\psi_1, \dots, \psi_m)$.
In this case $v_{\mathcal{I}^*}(P_\psi(x_1, \dots, x_n)) = I^*(P_\psi)(d(x_1), \dots, d(x_n)) =$
$= \sigma_A(v_{\mathcal{I}^*}(P_{\psi_1}(x_1, \dots, x_n)), \dots, v_{\mathcal{I}^*}(P_{\psi_n}(x_1, \dots, x_n))) =$
$= \sigma_A(v_{\mathcal{I}}(\psi_1), \dots, v_{\mathcal{I}}(\psi_n)) = v_{\mathcal{I}}(\psi)$.

Similarly, for every $d_1, \ldots, d_n \in D$,
$v_{\mathcal{I}^*_{x_i/d_i}}(P_\psi(x_1, \ldots, x_n)) = \sigma_A(v_{\mathcal{I}^*_{x_i/d_i}}(P_{\psi_1}(x_1, \ldots, x_n)), \ldots, v_{\mathcal{I}^*_{x_i/d_i}}(P_{\psi_n}(x_1, \ldots, x_n))) =$
$= \sigma_A(v_{\mathcal{I}_{x_i/d_i}}(\psi_1), \ldots, v_{\mathcal{I}_{x_i/d_i}}(\psi_n)) = v_{\mathcal{I}_{x_i/d_i}}(\psi)$.

Case 2: $\psi = (Qx)\psi_1(x, x_1, \ldots, x_n)$.
In this case $v_{\mathcal{I}^*}(P_\psi(x_1, \ldots, x_n)) = v_{\mathcal{I}^*}((Qx)P_{\psi_1}(x, x_1, \ldots, x_n)) = \overline{Q}(\{w \mid \exists d \in D \text{ s.t. } v_{\mathcal{I}^*{}_{x,d}}(P_{\psi_1}(x, x_1, \ldots, x_n)) = w\} = \overline{Q}(\{w \mid \exists d \in D \text{ s.t. } v_{\mathcal{I}_{x,d}}(\psi_1) = w\} = v_{\mathcal{I}}(\psi)$.

Similarly, for every $d_1, \ldots, d_n \in D$,

$$\begin{aligned}
v_{\mathcal{I}^*_{x_i/d_i}}(P_\psi(x_1, \ldots, x_n)) &= I^*(P_\psi)(d_1, \ldots, d_n) = \\
&= v_{\mathcal{I}^*{}_{x_i/d_i}}((Qx)P_{\psi_1}(x, x_1, \ldots, x_n)) = \\
&= \overline{Q}(\{w \mid \exists d \in D \text{ s.t. } v_{\mathcal{I}^*{}_{x_i/d_i, x/d}}(P_{\psi_1}(x, x_1, \ldots, x_n)) = w\} = \\
&= \overline{Q}(\{w \mid \exists d \in D \text{ s.t. } v_{\mathcal{I}_{x_i/d_i.x/d}}(\psi_1) = w\} = \\
&= v_{\mathcal{I}_{x_i/d_i}}(\psi).
\end{aligned}$$
□

Lemma 5.58 *The following hold:*

(1) *The formula ϕ is valid if and only if for every interpretation $\mathcal{I} = (D, I, d)$, if the following conditions hold:*

  (1a) *for every subformula $\psi(x_1, \ldots, x_n) = \sigma(\psi_1, \ldots, \psi_m)$ of ϕ with free variables $x_1, \ldots, x_n$, $v_{\mathcal{I}_{x_i/d_i}}(P_\psi(x_1, \ldots, x_n)) =$ $\sigma_A(v_{\mathcal{I}_{x_i/d_i}}(P_{\psi_1}(x_1, \ldots, x_n)), \ldots, v_{\mathcal{I}_{x_i/d_i}}(P_{\psi_m}(x_1, \ldots, x_n)))$, for every $d_1, \ldots, d_n \in D$, and*

  (1b) *for every subformula $\psi(x_1, \ldots, x_n) = (Qx)\psi_1(x, x_1, \ldots, x_n)$, $v_{\mathcal{I}_{x_i/d_i}}(P_\psi(x_1, \ldots, x_n)) = v_{\mathcal{I}_{x_i/d_i}}((Qx)P_{\psi_1}(x, x_1, \ldots, x_n))$, for every $d_1, \ldots, d_n \in D$.*

  then P_ϕ is true at $\mathcal{I}$.

(2) *The formula ϕ is satisfiable if and only if there exists an interpretation $\mathcal{I} = (D, I, d)$ such that*

  (2a) *for every subformula $\psi(x_1, \ldots, x_n) = \sigma(\psi_1, \ldots, \psi_m)$ of ϕ with free variables $x_1, \ldots, x_n$, $v_{\mathcal{I}_{x_i/d_i}}(P_\psi(x_1, \ldots, x_n)) =$ $\sigma_A(v_{\mathcal{I}_{x_i/d_i}}(P_{\psi_1}(x_1, \ldots, x_n)), \ldots, v_{\mathcal{I}_{x_i/d_i}}(P_{\psi_m}(x_1, \ldots, x_n)))$, for every $d_1, \ldots, d_n \in D$, and*

  (2b) *for every subformula $\psi(x_1, \ldots, x_n) = (Qx)\psi_1(x, x_1, \ldots, x_n)$, $v_{\mathcal{I}_{x_i/d_i}}(P_\psi(x_1, \ldots, x_n)) = v_{\mathcal{I}_{x_i/d_i}}((Qx)P_{\psi_1}(x, x_1, \ldots, x_n))$, for every $d_1, \ldots, d_n \in D$.*

  and P_ϕ is true at $\mathcal{I}$.

Proof: Let ϕ be a fomula in $\mathcal{L}$. By Lemma 5.57 we know that for every interpretation $\mathcal{I} = (D, I, d)$ and its extension $\mathcal{I}^* = (D, I^*, d)$ to the new introduced predicate symbols, $v_{\mathcal{I}^*_{x_i/d_i}}(P_\phi(x_1, \ldots, x_n)) = v_{\mathcal{I}_{x_i/d_i}}(\phi)$ for every $d_1, \ldots, d_n \in D$.

In particular, $v_{\mathcal{I}^*}(P_\phi(x_1, \ldots, x_n)) = v_{\mathcal{I}}(\phi)$.

(1) Assume that ϕ is valid. Let $\mathcal{I} = (D, I, d)$ be an arbitrary interpretation of the extended language such that the conditions $(1a)$ and $(1b)$ hold. Then $\mathcal{I} = (D, I, d)$ is the extension (in the sense of Lemma 5.57) of its restriction to (X, O, P), hence $v_{\mathcal{I}}(P_\phi(x_1, \dots, x_n)) = v_{\mathcal{I}}(\phi)$. Since ϕ is valid it follows that $v_{\mathcal{I}}(\phi) = 1$. Therefore, $v_{\mathcal{I}}(P_\phi(x_1, \dots, x_n)) = 1$, i.e. P_ϕ is true at $\mathcal{I}$.

Conversely, let ϕ be a formula in $\mathcal{L}$ and assume that for every interpretation $\mathcal{I}$ that satisfies conditions $(1a)$ and $(1b)$, P_ϕ is valid in $\mathcal{I}$. Let $\mathcal{I} = (D, I, d)$ be an interpretation for the language (X, O, P). It is easy to see that the extension $\mathcal{I}^*$ of $\mathcal{I}$ to the language obtained by introducing a new predicate symbol for every subformula ψ of ϕ satisfies $(1a)$ and $(1b)$. It follows then that P_ϕ is true at $\mathcal{I}^*$. But $v_{\mathcal{I}^*}(P_\phi(x_1, \dots, x_n)) = v_{\mathcal{I}}(\phi)$. Thus, ϕ is true in $\mathcal{I}$.

(2) Assume that ϕ is satisfiable. Then there exists an interpretation $\mathcal{I} = (D, I, d)$ such that ϕ is true in $\mathcal{I}$. It is easy to see that the extension $\mathcal{I}^*$ of $\mathcal{I}$ to the language obtained by introducing a new predicate symbol for every subformula ψ of ϕ satisfies $(2a)$ and $(2b)$. Moreover, since $v_{\mathcal{I}^*}(P_\phi(x_1, \dots, x_n)) = v_{\mathcal{I}}(\phi)$, P_ϕ is true at $\mathcal{I}^*$.

Conversely, assume that there exists an interpretation $\mathcal{I} = (D, I, d)$ of the extended language such that the conditions $(2a)$ and $(2b)$ are fulfilled and P_ϕ is true at $\mathcal{I}^*$. Then $\mathcal{I} = (D, I, d)$ is the extension (in the sense of Lemma 5.57) of its restriction to (X, O, P), hence $v_{\mathcal{I}}(P_\phi(x_1, \dots, x_n)) = v_{\mathcal{I}}(\phi)$. Therefore, ϕ is true in $\mathcal{I}$. $\square$

Example 5.4 *Let ϕ be the formula $(\forall x) \sim S_1(p(x))$ in a first-order SHn-logic (based on the algebra S_{n^2} as an algebra of truth values) with a unary predicate symbol p (as explained at the beginning of Section 5.3.6).*

Let P_ϕ be a new predicate symbol with arity 0 corresponding to the formula ϕ (without free variables), $P_{\sim S_1(p(x))}$ a new predicate symbol with arity 1 (corresponding to the subformula $\sim S_1(p(x))$), and $P_{S_1(p(x))}$ a new predicate symbol with arity 1 (corresponding to the subformula $S_1(p(x))$).

Then

- *ϕ is valid if and only if for every interpretation $\mathcal{I}$,*
  (i) $I(P_\phi) = v_{\mathcal{I}}((\forall x) P_{\sim S_1(p(x))}(x))$,
  (ii) for any instantiation $d \in D$, $v_{\mathcal{I}_{x/d}}(P_{\sim S_1(p(x))}(x)) = \sim v_{\mathcal{I}_{x/d}}(P_{S_1(f(x))})$, and
  (iii) for any instantiation $d \in D$, $v_{\mathcal{I}_{x/d}}(P_{S_1(p(x))}(x)) = S_1(v_{\mathcal{I}_{x/d}}(p(x)))$

  imply that P_ϕ is true in $\mathcal{I}$, and

- *ϕ is satisfiable if and only if there is an interpretation $\mathcal{I} = (D, I, d)$ such that*
  (i) $I(P_\phi) = v_{\mathcal{I}}((\forall x) P_{\sim S_1(p(x))}(x))$,
  (ii) for any instantiation $d \in D$, $v_{\mathcal{I}_{x/d}}(P_{\sim S_1(p(x))}(x)) = \sim v_{\mathcal{I}_{x/d}}(P_{S_1(f(x))})$, and
  (iii) for any instantiation $d \in D$, $v_{\mathcal{I}_{x/d}}(P_{S_1(p(x))}(x)) = S_1(v_{\mathcal{I}_{x/d}}(p(x)))$

  and P_ϕ is true in $\mathcal{I}$.

Taking into account the bijective correspondence between models $\mathcal{I} = (D, I, d)$ based on A and models $\mathcal{M} = (D, M, d)$ based on $\mathcal{O}(D(A))$ and the fact that P_ϕ is not true at $\mathcal{M}$ if and only if

$$\bigvee_{\alpha \in D(A),\ \text{minimal}} \boxed{\alpha}\, P^f_\phi \text{ in } \mathcal{M},$$

from Lemma 5.56 we obtain the following immediate corollary.

Corollary 5.59 *The formula ϕ is valid if and only if there exists no interpretation $\mathcal{M} = (D, M, d)$ of $\mathcal{L}$ in $\mathcal{O}(D(A))$ such that*

$$\begin{cases}
\bigvee_{\alpha \in D(A),\ minimal} \boxed{\alpha}\, P^f_\phi & \textit{in } \mathcal{M}, \\[2ex]
\begin{array}{c} P_\psi(x_1, \dots, x_n) \textit{ true at } \alpha \textit{ in } \mathcal{M} \textit{ iff} \\ \sigma(P_{\psi_1}, \dots, P_{\psi_k}) \textit{ true at } \alpha \textit{ in } \mathcal{M} \\ \textit{for every } \alpha \in D(A) \end{array} & \begin{array}{l} \textit{for all subformulae} \\ \psi = \sigma(\psi_1, \dots, \psi_m) \textit{ of } \psi \\ \textit{and every instantiation of the} \\ \textit{free variables } x_1, \dots, x_n \textit{ of } \psi, \end{array} \\[2ex]
\begin{array}{c} P_\psi(x_1, \dots, x_n) \textit{ true at } \alpha \textit{ in } \mathcal{M} \textit{ iff} \\ (Qx)P_{\psi_1}(x, x_1, \dots, x_n) \textit{ true at } \alpha \textit{ in } \mathcal{M} \\ \textit{for every } \alpha \in D(A) \end{array} & \begin{array}{l} \textit{for all subformulae} \\ \psi = (Qx)\psi_1(x) \textit{ of } \phi, \\ \textit{and every instantiation of the} \\ \textit{free variables } x_1, \dots, x_n \textit{ of } \psi. \end{array}
\end{cases}$$

Proposition 5.60 *The formula ϕ is valid if and only if there exists no interpretation $\mathcal{M} = (D, M, d)$ of $\mathcal{L}$ in $\mathcal{O}(D(A))$ such that*

$$\begin{cases}
\bigvee_{\alpha \in D(A),\ minimal} \boxed{\alpha}\, P^f_\phi & \textit{in } \mathcal{M}, \\[2ex]
\begin{array}{c} (\boxed{\alpha}\, P_\psi(x_1, \dots, x_n)^t \vee \boxed{\alpha}\, \sigma(P_{\psi_1}, \dots, P_{\psi_k})^f) \wedge \\ \wedge (\boxed{\alpha}\, P_\psi(x_1, \dots, x_n)^f \vee \boxed{\alpha}\, \sigma(P_{\psi_1}, \dots, P_{\psi_k})^t) \\ \textit{in } \mathcal{M} \\ \textit{for every } \alpha \in D(A) \end{array} & \begin{array}{l} \textit{for all subformulae} \\ \psi = \sigma(\psi_1, \dots, \psi_m) \textit{ of } \phi \\ \textit{and every instantiation of the} \\ \textit{free variables } x_1, \dots, x_n \textit{ of } \psi, \end{array} \\[2ex]
\begin{array}{c} (\boxed{\alpha}\, P_\psi(x_1, \dots, x_n)^t \vee \boxed{\alpha}\, (Qx)P_{\psi_1}(x, x_1, \dots, x_n)^f) \wedge \\ \wedge (\boxed{\alpha}\, P_\psi(x_1, \dots, x_n)^f \vee \boxed{\alpha}\, (Qx)P_{\psi_1}(x, x_1, \dots, x_n)^t) \\ \textit{in } \mathcal{M} \\ \textit{for every } \alpha \in D(A) \end{array} & \begin{array}{l} \textit{for all subformulae} \\ \psi = (Qx)\psi_1(x) \textit{ of } \phi \\ \textit{and every instantiation of the} \\ \textit{free variables } x_1, \dots, x_n \textit{ of } \psi. \end{array}
\end{cases}$$

The situation when $\psi = \sigma(\psi_1, \dots, \psi_n)$ can be handled as in the propositional case, taking into account the properties of σ. For the situation when $\psi = (Qx)\psi_1(x)$, $Q \in \{\forall, \exists\}$, we have the following results (some of them appear – in a different context – also in [Häh96a]).

Lemma 5.61 *Let $\mathcal{I} = (D, I, d)$ be an interpretation of $\mathcal{L}$ in A, and let $a \in A$ be a join-irreducible element. The following properties hold:*

($\forall$1) *$v_\mathcal{I}(\forall x \phi(x, x_1, \dots, x_n)) \geq a$ iff $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \dots, x_n)) \geq a$ for every instantiation $d \in D$ of x.*

($\forall$2) *$v_{\mathcal{I}}(\forall x \phi(x, x_1, \ldots, x_n)) \not\geq a$ iff $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \not\geq a$ for some instantiation $d \in D$ of x, depending on $d(x_1), \ldots, d(x_n)$.*

($\exists$1) *$v_{\mathcal{I}}(\exists x \phi(x, x_1, \ldots, x_n)) \geq a$ iff $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \geq a$ for some instantiation $d \in D$ of x, depending on $d(x_1), \ldots, d(x_n)$.*

($\exists$2) *$v_{\mathcal{I}}(\exists x \phi(x, x_1, \ldots, x_n)) \not\geq a$ iff $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \not\geq a$ for every instantiation $d \in D$ of x.*

Proof: ($\forall$1) and ($\forall$2) follow immediately, taking into account that

$$f(\forall x \phi(x, x_1, \ldots, x_n)) = \bigwedge \{w \mid \exists d \in D \ s.t.\ v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w\}.$$

($\exists$1) For the direct implication, assume that $v_{\mathcal{I}}(\exists x \phi(x)) = \bigvee \{w \mid \exists d \in D \ s.t.\ v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w\} \geq a$. From the fact that A is distributive and for every ϕ the set $\{w \mid \exists d \in D \ s.t.\ v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w\}$ is finite, it follows that $a = \bigvee \{w \mid \exists d \in D \ s.t.\ v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w\} \wedge a = \bigvee \{w \wedge a \mid \exists d \in D \ s.t.\ v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w\}$. By the fact that a is join-irreducible it then follows that $a = w \wedge a$ for some w such that there exists $d \in D$ with $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w$, hence, $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \geq a$ for some $d \in D$ (depending on $d(x_1), \ldots, d(x_n)$. The converse follows immediately since if $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \geq a$ for some $d \in D$, then also $v_{\mathcal{I}}(\exists x \phi(x)) = \bigvee \{w \mid \exists d \in D \ s.t.\ v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) = w\} \geq a$.

($\exists$2) follows immediately from ($\exists$1). $\square$

Lemma 5.62 *Let $\mathcal{M} = (D, M, d)$ be an interpretation of $\mathcal{L}$ in $\mathcal{O}(D(A))$. The following properties hold:*

(1) $\boxed{\alpha}$ $(\forall x \phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ iff $\boxed{\alpha}$ $(\phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ for every instantiation of x.

(2) $\boxed{\alpha}$ $(\forall x \phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ iff $\boxed{\alpha}$ $(\phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ for some some instantiation $f_\phi(d(x_1), \ldots, d(x_n))$ for x, where f_ϕ is a new function symbol.

(3) $\boxed{\alpha}$ $(\exists x \phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ iff $\boxed{\alpha}$ $(\phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ for some instantiation $f_\phi(d(x_1), \ldots, d(x_n))$ for x, where f_ϕ is a new function symbol.

(4) $\boxed{\alpha}$ $(\exists x \phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ iff $\boxed{\alpha}$ $(\phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ for every instantiation of x.

Proof: We know that all the elements $\alpha \in D(A)$ are of the form $\uparrow a$ with a a join-irreducible element in A.

For every $\mathcal{M} = (D, M, d)$, let $v_{\mathcal{M}} : \mathsf{Fma}(\mathcal{L}) \to \mathcal{O}(D(A))$ be its extension to the formulae. Let $\mathcal{I} = (D, I, d)$ be the associated interpretation in A (obtained by composition with η_A^{-1} when necessary). For every $\psi \in \mathsf{Fma}(\mathcal{L})$ we have

$v_{\mathcal{M}}(\psi) = \eta_A(v_{\mathcal{I}}(\psi)) = \{\uparrow a \mid v_{\mathcal{I}}(\psi) \in\uparrow a\} = \{\uparrow a \mid v_{\mathcal{I}}(\psi) \geq a\}$. Thus, $\alpha = \uparrow a \in v_{\mathcal{M}}(\psi)$ if and only if $v_{\mathcal{I}}(\psi) \geq a$.

Thus, $\boxed{\alpha}\ \psi^t$ in $\mathcal{M}$ if and only if $v_{\mathcal{I}}(\psi) \geq a$. Therefore, using Lemma 5.61 we have:

(1) $\boxed{\alpha}\ (\forall x \phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ if and only if $v_{\mathcal{I}}(\forall x \phi(x, x_1, \ldots, x_n)) \geq a$, which, by Lemma 5.61 happens if and only if $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \geq a$ for every instantiation d of x. Thus, $\boxed{\alpha}\ (\forall x \phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ if and only if for every instantiation of x, $\boxed{\alpha}\ (\phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$.

(2) $\boxed{\alpha}\ (\forall x \phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ if and only if $\alpha \notin v_{\mathcal{M}}(\forall x \phi(x, x_1, \ldots, x_n))$. By Lemma 5.61 this happens if and only if $v_{\mathcal{I}_{x/d}}(\phi(x, x_1, \ldots, x_n)) \not\geq a$ for some instantiation d of a, depending on the values of the other free variables in ϕ.

The assertions (3) and (4) follow analogously. □

Let $\mathcal{L}'$ be the first-order language obtained from the language of $\mathcal{L}$ by adding a new r-ary predicate symbol P_ϕ for every formula ϕ in $\mathcal{L}$ with r free variables and a k-ary function symbol f_ϕ for every formula ϕ of $\mathcal{L}$ that starts with a quantifier and has k free variables. The f_ϕ will serve as Skolem functions.

As a consequence of the previous results, the following hold:

- $\boxed{\alpha}\ (\forall x \phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ iff $\boxed{\alpha}\ (\phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ for every instantiation of x.
- $\boxed{\alpha}\ (\forall x \phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ iff $\boxed{\alpha}\ (\phi(f_\phi(x_1, \ldots, x_n), x_1, \ldots, x_n))^f$ in $\mathcal{M}$ (where f_ϕ is a new function symbol).
- $\boxed{\alpha}\ (\exists x \phi(x, x_1, \ldots, x_n))^t$ in $\mathcal{M}$ iff $\boxed{\alpha}\ (\phi(f_\phi(x_1, \ldots, x_n), x_1, \ldots, x_n))^t$ in $\mathcal{M}$ (where f_ϕ is a new function symbol).
- $\boxed{\alpha}\ (\exists x \phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ iff $\boxed{\alpha}\ (\phi(x, x_1, \ldots, x_n))^f$ in $\mathcal{M}$ for every instantiation of x.

If with all operations in A one can associate corresponding relations on the dual set, according to the remarks already done in Section 5.3.1 and Section 5.3.3, these rules allow us starting from a given formula ϕ to construct a set of *signed clauses* Φ such that ϕ is valid if and only if Φ is unsatisfiable.

Definition 5.25 *Let $\alpha \in D(A)$ be a "possible world" and L be an atomic formula. Then $\boxed{\alpha}\ L^t$ is a* positive literal *(with sign $\boxed{\alpha}$) and $\boxed{\alpha}\ L^f$ is a* negative literal *(with sign $\boxed{\alpha}$). A set of (positive or negative) signed literals is called a* (signed) clause. *A* formula in signed conjunctive normal form (CNF) *is a finite set of (signed) clauses. We require that the clauses in a formula have disjoint variables.*

Definition 5.26 *A positive literal $\boxed{\alpha}\ L^t$ is satisfiable if for some interpretation in $\mathcal{O}(D(A))$, $\mathcal{M} = (D, M, d)$, $\alpha \in v_{\mathcal{M}}(L)$. A negative literal $\boxed{\alpha}\ L^f$ is satisfiable if for some interpretation in $\mathcal{O}(D(A))$, $\mathcal{M} = (D, M, d)$, $\alpha \notin v_{\mathcal{M}}(L)$.*

A signed clause is satisfiable if and only if at least one of its literals is satisfiable. A signed formula Φ is satisfiable if and only if all clauses in Φ are simultaneously satisfiable by the same interpretation.

Let Φ be a set of clauses. The *Herbrand universe* $H(\Phi)$ of Φ is the set of variable-free terms that consist of constants and function symbols occurring in Φ. If there is no constant in Φ we introduce a special constant symbol to prevent $H(\Phi)$ from being empty.

A *ground instance* C' of a clause (or atom) C in Φ is a substitution instance of C such that terms in $H(\Phi)$ replace the variables of C.

The *Herbrand base* $A(\Phi)$ of Φ is the set of all ground instances of atoms that occur in clauses in Φ.

An *assignment* in A associates truth values in A with atoms. Alternatively, an assignment in $D(A)$ associates upwards-closed sets of elements in D(A) ("possible worlds") with atoms. Since $\mathcal{O}(D(A))$ and A are isomorphic, the two notions express the same thing. A *complete assignment* in A to a set of atoms K is defined as a set of literals $\{P^{v(P)} \mid P \in K\}$, where $v : K \to A$.

A *complete assignment* in $D(A)$ to a set of atoms K is defined as a set of the form $\{ \boxed{\alpha}\ P^t \mid P \in K, \alpha \in m(P)\} \cup \{ \boxed{\alpha}\ P^f \mid P \in K, \alpha \notin m(P)\}$ where $m : K \to \mathcal{O}(D(A)))$.

An *H-interpretation*[6] of a set of clauses Φ is a complete assignment to $A(\Phi)$. A H-interpretation M satisfies a clause set Φ if and only if for every $C \in \Phi$, all ground instances C' of C are such that $C' \cap M \neq \emptyset$. Φ is *H-unsatisfiable* if there is no H-interpretation that satisfies Φ.

Proposition 5.63 *A set of clauses Φ is unsatisfiable if and only if it is H-unsatisfiable.*

Sketch of the proof: Every H-interpretation corresponds to a frame, as defined in Definition 5.23. Assume that Φ is H-satisfiable, i.e. that there exists a H-interpretation M that satisfies Φ. We can therefore construct a Herbrand interpretation in A $\mathcal{H} = (\mathsf{Term}^0_{\mathsf{O}}, I, d)$ that satisfies Φ (or alternatively a Herbrand interpretation in $\mathcal{O}(D(A))$, $\mathcal{H}_M = (\mathsf{Term}^0_{\mathsf{O}}, M, d)$).

Conversely, let $\mathcal{I} = (D, I, d)$ be an arbitrary interpretation. $\mathcal{I}$ induces an H-interpretation in A, $I_{\mathcal{I}} = \{P^w \mid v_{\mathcal{I}}(P) = w, P \in A(\Phi)\}$ (resp. an H-interpretation in $D(A)$, $M_{\mathcal{I}} = \{ \boxed{\alpha}\ P^t \mid \alpha \in \eta_A(v_{\mathcal{I}}(P)), P \in A(\Phi)\} \cup \{ \boxed{\alpha}\ P^f \mid \alpha \notin \eta_A(v_{\mathcal{I}}(P)), P \in A(\Phi)\}$). $\square$

A Resolution Procedure

We describe also in this case a resolution procedure based on negative hyperresolution.

In what follows we consider clauses signed by elements of $D(A)$. Since $\boxed{\alpha}\ L^t$ implies $\boxed{\beta}\ L^t$ for every $\beta \geq \alpha$, we will delete in a clause all occurrences $\boxed{\beta}\ L^t$ of signed literals with the property that there exists an α with $\beta \geq \alpha$ such that $\boxed{\alpha}\ L^t \in C$. Similarly, $\boxed{\alpha}\ L^f$ implies $\boxed{\beta}\ L^f$ for every $\beta \leq \alpha$. Hence we will

[6] Note the difference we make here between *Herbrand interpretations* (cf. the examples after Definition 5.23) and H-interpretations: Herbrand interpretations apply to formulae and are interpretations over the algebra of all variable-free terms, whereas H-interpretations are defined for clauses, and refer only to the Herbrand universe of a set of clauses. The two notions are very similar, but because of these differences we use here different names for them.

delete in a clause all occurrences $\boxed{\beta}\, L^f$ of signed literals with the property that there exists an α with $\beta \leq \alpha$ such that $\boxed{\alpha}\, L^f \in C$.

Definition 5.27 (Factor) *Let C be a clause. If two or more positive literals of C have a m.g.u. σ, then $\sigma(C)$ (where only the signed literal with the smallest sign is kept) is called a* factor *of C. Likewise, if two or more negative literals of a clause C have a m.g.u. σ, then $\sigma(C)$ (where only the literal with the largest sign is kept) is called a* factor *of C.*

Definition 5.28 (Binary Resolvent) *Let C_1 and C_2 be clauses with no variable in common. Let $\boxed{\alpha}\, L_1^t$ and $\boxed{\beta}\, L_2^f$ be signed literals occurring in C_1 and C_2, respectively. If $\alpha \leq \beta$ and L_1 and L_2 have a m.g.u. σ, then the clause $C_3 := (\sigma(C_1) - \boxed{\alpha}\, \sigma(L_1^t)) \cup (\sigma(C_2) - \boxed{\beta}\, \sigma(L_2^f))$ is called a* binary resolvent *of C_1 and C_2.*

Definition 5.29 (Resolvent) *A* resolvent *of two clauses C_1 and C_2 is one of the following binary resolvents:*

1. *a binary resolvent of C_1 and C_2,*
2. *a binary resolvent of C_1 and a factor of C_2,*
3. *a binary resolvent of a factor of C_1 and C_2,*
4. *a binary resolvent of a factor of C_1 and a factor of C_2.*

Let P be an ordering of predicate symbols. A finite set of clauses

$$\{E_1, \ldots, E_q, N\}, q \geq 1$$

is called a *semantic clash* with respect to P if and only if $E_1, \ldots, E_q$ (called *electrons*) and N (called *nucleus*) satisfy the following conditions:

1. $E_1, \ldots, E_q$ are negative clauses,
2. Let $R_1 = N$. For each $i = 1, \ldots, q$, there is a resolvent R_{i+1} of R_i and E_i,
3. The literal in E_i, which is resolved upon, contains the largest predicate symbol in E_i, $i = 1, \ldots, q$,
4. R_{q+1} is a positive clause.

R_{q+1} is called a *resolvent by hyperresolution* of the semantic clash $\{E_1, \ldots, E_q, N\}$.

This can be schematically represented as follows:

$$\frac{\left\{\boxed{\beta_1}\, L_1^f\right\} \cup D_1, \ldots, \left\{\boxed{\beta_n}\, L_n^f\right\} \cup D_n, \left\{\boxed{\alpha_1}\, L_1^t, \ldots, \boxed{\alpha_n}\, L_n^t\right\} \cup E}{D_1 \cup \ldots \cup D_n \cup E}$$

provided that $n \geq 1$, $\alpha_i \leq \beta_i$ for all $i = 1, \ldots, n$ and $D_1, \ldots, D_n, E$ are negative.

The following theorem states the correctness of the automated theorem proving procedure by hyperresolution:

Theorem 5.64 *For any set of clauses Φ, if the empty clause $\Box$ can be derived from Φ by resolution, then Φ is H-unsatisfiable.*

Proof: Follows from the fact that an H-interpretation that satisfies a set of clauses also satisfies all their factors and their resolvents. But no H-interpretation satisfies the empty clause. $\Box$

The next results prove the completeness of the procedure.

Lemma 5.65 (Lifting Lemma) *If C_1' and C_2' are instances of C_1 and C_2 respectively, and if C' is a resolvent of C_1' and C_2', then there is a resolvent C of C_1 and C_2 such that C' is an instance of C.*

Proof: We rename if necessary the variables in C_1 and C_2 such that the variables in C_1 and C_2 are disjoint. Let $\boxed{\alpha} L_1'^t$ and $\boxed{\beta} L_2'^f$ be the literals resolved upon ($\alpha \leq \beta$), and let the resolvent of C_1' and C_2' be

$$C' = (\sigma(C_1') - \boxed{\alpha}\ \sigma(L_1'^t)) \cup (\sigma(C_2') - \boxed{\beta}\ \sigma(L_2'^f)),$$

where σ is the m.g.u. of L_1' and L_2'.

Since C_1' and C_2' are instances of C_1 and C_2 respectively, there is a substitution θ such that $C_1' = \theta(C_1)$ and $C_2' = \theta(C_2)$ (we used the disjointness of the variables in C_1 and C_2). Let $\boxed{\alpha_1}\ L_1^{1^t}, \ldots, \boxed{\alpha_{r_1}}\ L_1^{r_1{}^t}$ resp. $\boxed{\beta_1}\ L_2^{1^f}, \ldots, \boxed{\beta_{r_2}}\ L_2^{r_2{}^f}$ be the literals in C_1, resp. C_2 corresponding to L_1' resp. L_2' (i.e. such that $\theta(L_i^1) = \ldots = \theta(L_i^{r_i}) = L_i'$, $i = 1, 2$). From the definition of factor clauses it also follows that $\alpha = min\{\alpha_1, \ldots, \alpha_{r_1}\}$ and $\beta = max\{\beta_1, \ldots, \beta_{r_2}\}$.

If $r_i > 1$, let λ_i be a m.g.u. for $\{L_i^1, \ldots, L_i^{r_i}\}$, and let $L_i = \lambda_i(L_i^1)$, $i = 1, 2$. Then L_i is a literal in the factor $\lambda_i(C_i)$ of C_i. If $r_i = 1$, then let λ_i be the identity and $L_i = \lambda_i(L_i^1)$. Let $\lambda = \lambda_1 \cup \lambda_2$ (the variables in C_1, C_2 are disjoint). Since L_1' and L_2' are unifiable, L_1 and L_2 are unifiable. Let γ be the m.g.u. of L_1 and L_2.

$$C = ((\gamma(\lambda(C_1)) - \boxed{min\{\alpha_1, \ldots, \alpha_{r_1}\}}\ \gamma(L_1)) \cup ((\gamma(\lambda(C_2)) - \boxed{max\{\beta_1, \ldots, \beta_{r_2}\}}\ \gamma(L_2))$$

$$= ((\{\gamma(\lambda(L)) \mid L \in C_1\} - \{\boxed{\alpha_1}\ \gamma(\lambda(L_1^1))^t, \ldots, \boxed{\alpha_{r_1}}\ \gamma(\lambda(L_1^{r_1}))^t\}) \cup$$

$$\cup ((\{\gamma(\lambda(L)) \mid L \in C_2\} - \{\boxed{\beta_1}\ \gamma(\lambda(L_2^1))^f, \ldots, \boxed{\beta_{r_2}}\ \gamma(\lambda(L_2^{r_2}))^f\}).$$

C is a resolvent of C_1 and C_2. Clearly, C' is an instance of C since

$$C' = (\sigma(C_1') - \boxed{\alpha}\ \sigma(L_1')) \cup (\sigma(C_2') - \boxed{\beta}\ \sigma(L_2')) =$$

$$= (\sigma(\theta(C_1)) - \{\boxed{\alpha_1}\ \sigma(\theta(L_1^1))^t, \ldots, \boxed{\alpha_{r_1}}\ \sigma(\theta(L_1^{r_1}))^t\})) \cup$$

$$= (\sigma(\theta(C_2)) - \{\boxed{\beta_1}\ \sigma(\theta(L_2^1))^f, \ldots, \boxed{\beta_{r_2}}\ \sigma(\theta(L_2^{r_2}))^f\}),$$

and $\gamma \circ \lambda$ is more general than $\sigma \circ \theta$. $\Box$

Lemma 5.66 *Let P be an ordering on predicate symbols, and let Φ be a finite unsatisfiable set of signed ground clauses. Then $\Box$ can be derived from Φ by a finite number of applications of many-valued negative hyperresolution.*

Proof: Similar to the proof of Theorem 5.33. $\Box$

Theorem 5.67 *Let P be an ordering on predicate symbols, and Φ be a finite unsatisfiable set of signed clauses. Then $\Box$ can be derived from Φ by a finite number of applications of many-valued negative hyperresolution.*

Proof: Since Φ is unsatisfiable, there is a finite unsatisfiable set Φ' of ground instances of clauses in Φ. By Lemma 5.66, $\Box$ can be derived from Φ' by a finite number of applications of many-valued negative hyperresolution. Let D' be the deduction of $\Box$ from Φ'. From the deduction D' we can produce a deduction D by hyperresolution of $\Box$ from Φ as follows.

For any node N of D', let C'_N be the ground clause at node N in D'. Now, attach to each initial node I a clause C_I from Φ such that C'_I is a ground instance of C_I. Then, for each non-initial node N, if clauses have been attached in this way to each of its immediate predecessor nodes and they constitute a semantic clash, attach to N the hyperresolvent of which C'_N is an instance (this is possible because of the lifting lemma, Lemma 5.65). In this way we attach a clause C_N to each node N such that C'_N is a ground instantiation of C_N. The clause attached to the terminal node must be $\Box$, since the clause already there is $\Box$. It is easy to see that the deduction tree, together with the attached clauses is a deduction by hyperresolution of $\Box$ from Φ. This completes the proof. $\Box$

5.4 Examples

5.4.1 P_{mn}-logics

Definition 5.30 (Ockham Algebras) *A* Ockham algebra $A = (L, \vee, \wedge, f, 0, 1)$ *is a distributive lattice with* 0 *and* 1 *with an unary operation f satisfying:*

(O1) $f(0) = 1$,
(O2) $f(1) = 0$,
(O3) $f(x \wedge y) = f(x) \vee f(y)$,
(O4) $f(x \vee y) = f(x) \wedge f(y)$.

Before entering into detail, we indicate some algebraic and logical motivation for the study of Ockham algebras. For details see also [Urq79]. The class of De Morgan lattices[7] arises naturally in the study of logics omitting the paradoxes of material implication. The study of De Morgan lattices and their representations has very much helped in investigating the algebras of propositions that arise from these logics. The Ockham algebras offer a setting for describing a larger class of logics. In logical terms, the goal is to describe the

[7] A De Morgan lattice is a structure $M = (L, \vee, \wedge, \sim 0, 1)$, where $(L, \vee, \wedge, 0, 1)$ is a distributive lattice with 0 and 1, and $\sim$ is a unary operation such that $\sim 1 = 0$, $\sim\sim x = x$ and $\sim (x \wedge y) = \sim x \vee \sim y$ for every $x, y \in L$.

structure of algebras of propositions which lack not only the paradoxes of material implication but also the law of double negation. The resulting theory is quite elegant mathematically and subsumes not only the theory of De Morgan lattices but also that of Stone lattices[8] which have been extensively investigated by lattice theorists.

The class of Ockham algebras is equationally definable, hence it is a variety. We will denote by $\mathcal{K}$ the variety of Ockham algebras.

Definition 5.31 (P_{mn}) *For $m > n \geq 0$, let P_{mn} be the subclass of $\mathcal{K}$ defined by*

$f^m(x) = f^n(x)$ for every $x \in L$ if $m - n$ is even, respectively

$$\left\{ \begin{array}{ll} f^m(x) \vee f^n(x) = 1 & \text{for every } x \in L \\ f^m(x) \wedge f^n(x) = 0 & \text{for every } x \in L \end{array} \right. \quad \text{if } m - n \text{ is odd.}$$

We recall the following results (for details see e.g. [Urq79]).

Let K be the category of Ockham algebras, having as objects Ockham algebras and as morphisms, morphisms of Ockham algebras.

Let O be the category of Ockham spaces, having as

Objects:	structures $(X, \leq, \tau, g)$ where $(X, \leq, \tau)$ is a Priestley space and $g : X \to X$ is a continuous, order reversing map;
Morphisms:	continuous, order preserving maps that preserve the unary operator.

Proposition 5.68 ([Urq79]) *The Priestley duality induces a dual equivalence between the category K and the category O.*

Proposition 5.69 ([Urq79]) *$L \in P_{mn}$ if and only if the dual space $D(L) = (X, \leq, \tau, g)$ of L satisfies $g^m(x) = g^n(x)$ for every $x \in X$.*

Let $\mathsf{P_{mn}}$ be the full subcategory of K whose objects are algebras in P_{mn}, and let $\mathsf{O_{mn}}$ be the full subcategory of O whose objects are those Ockham spaces $(X, \leq, \tau, g)$ that additionally satisfy $g^m(x) = g^n(x)$ for every $x \in X$.

Corollary 5.70 ([Urq79]) *The dual equivalence between the category of Ockham algebras and the category of Ockham spaces induces a dual equivalence between $\mathsf{P_{mn}}$ and $\mathsf{O_{mn}}$.*

Theorem 5.71 ([Urq79]) *If L is an Ockham algebra and $(X, \leq, \tau, g)$ the dual space of L, then L is subdirectly irreducible if and only if $\{x \mid \overline{g^\infty(\{x\})} \neq X\}$ is not dense in X (where $g^\infty(Y)$ stands for $\{g^n(y) \mid n \geq 0, y \in Y\}$).*

[8] A Stone lattice is a structure $S = (L, \vee, \wedge, *, 0, 1)$, where $(L, \vee, \wedge, 0, 1)$ is a distributive lattice with 0 and 1, and $*$ is a unary operation such that $0^* = 1, x \wedge x^* = 0, (x \wedge y)^* = x^* \vee y^*$ for every $x, y \in L$.

If L is finite, then τ is the discrete topology on L, which gives us the following consequence.

Corollary 5.72 ([Urq79]) *If L is a finite Ockham algebra and $(X, \leq, \tau, g)$ the dual space of L, then L is subdirectly irreducible if and only if for some $x \in X$, $g^\infty(\{x\}) = X$.*

For $m > n \geq 0$, let S_{mn} be the structure $(X, \leq, \tau, g)$ defined by setting $X = \{0, 1, \ldots, m-1\}$, with the discrete topology and order, and with $g(k) = k+1$ for $k < m-1$ and $g(m-1) = n$. Now let L_{mn} be the dual algebra of the space S_{mn} (the lattice of all subsets of S_{mn}, with $f(Y) = S_{mn} \backslash g^{-1}(Y)$).

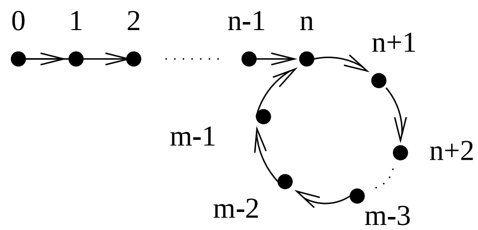


Figure 5.5: S_{mn}

Remark 5.73 *For every $m > n \geq 0$, L_{mn} is subdirectly irreducible.*

Proof: Follows from the fact that $g^\infty(\{0\}) = \{0, 1, \ldots, m-1\} = S_{mn}$. □

In fact, an even stronger result holds:

Proposition 5.74 ([Gol81])

(1) A finite distributive Ockham algebra is simple if and only if it is a subalgebra of L_{m0} for some $m > 0$,

(2) A finite distributive Ockham algebra is subdirectly irreducible if and only if it is a subalgebra of L_{mn} for some $m > n \geq 0$.

Theorem 5.75 ([Urq79]) *Every algebra in P_{mn} is isomorphic to a sublattice of a product of copies of L_{mn}.*

Corollary 5.76 ([Gol81], [AP94],p.6) *The variety P_{mn} is generated by the subdirectly irreducible algebra whose dual is S_{mn}.*

Note that in this case we only know the form of the dual of the algebra that generates the variety P_{mn}.

Let $\mathcal{L}_{mn}$ be a logic with set of variables Var and with connectives $\vee, \wedge$ (binary) and f (unary) that is sound and complete with respect to the variety P_{mn}, i.e. such that for every formula ϕ in the language of $\mathcal{L}_{mn}$,

$$\vdash_{\mathcal{L}_{mn}} \phi \text{ if and only if } P_{mn} \models \phi = 1.$$

Since the variety P_{mn} is generated by the algebra L_{mn}, it follows that

$$\vdash_{\mathcal{L}_{mn}} \phi \text{ if and only if } L_{mn} \models \phi = 1.$$

By the considerations in Section 5.3.1 we can define a relation $\overset{r}{\models}$ as follows:

Let $v : \mathsf{Var} \to \mathcal{O}(S_{mn})$ be a meaning function that has as values order-filters in $S_{mn} = D(L_{mn})$. Let $\eta : L_{mn} \to \mathcal{O}(S_{mn})$ be the canonical bijection between L_{mn} and $\mathcal{O}(D(L_{mn}))$, and let η^{-1} be its inverse. Let $\overline{v} : \mathsf{Fma}(\mathsf{Var}) \to \mathcal{O}(S_{mn})$ be defined by $\overline{v}(\phi) = \eta \circ \overline{(\eta^{-1} \circ v)}$.

We define

$$S_{mn} \overset{r}{\models}_{v,x} \phi \text{ if and only if } x \in \overline{v}(\phi),$$

$$S_{mn} \overset{r}{\models}_{v} \phi \text{ if and only if } \overline{v}(\phi) = S_{mn},$$

$$S_{mn} \overset{r}{\models} \phi \text{ if and only if for every } v : \mathsf{Var} \to \mathcal{O}(S_{mn}), S_{mn} \overset{r}{\models}_{v} \phi.$$

By Proposition 5.35, for every formula ϕ in the logic $\mathcal{L}_{mn}$ we have

$$\mathcal{L}_{mn} \models \phi \text{ if and only if } S_{mn} \overset{r}{\models} \phi.$$

Proposition 5.77 *The following holds:*

(1) $S_{mn} \overset{r}{\models}_{v,x} \phi_1 \wedge \phi_2$ *if and only if* $S_{mn} \overset{r}{\models}_{v,x} \phi_1$ *and* $S_{mn} \overset{r}{\models}_{v,x} \phi_2$,

(2) $S_{mn} \overset{r}{\models}_{v,x} \phi_1 \vee \phi_2$ *if and only if* $S_{mn} \overset{r}{\models}_{v,x} \phi_1$ *or* $S_{mn} \overset{r}{\models}_{v,x} \phi_2$,

(3) $S_{mn} \overset{r}{\models}_{v,x} f(\phi)$ *if and only if* $S_{mn} \overset{r}{\not\models}_{v,g(x)} \phi$.

Proof: (1) and (2) hold since the operations $\vee$ and $\wedge$ in $\mathcal{O}(S_{mn})$ are union resp. intersection.

In order to prove (3) note first that the definition of f in $\mathcal{O}(S_{mn})$ is $f(U) = S_{mn} \backslash g^{-1}(U)$. Hence, $S_{mn} \overset{r}{\models}_{v,x} f(\phi)$ if and only if

$$\begin{aligned}
x \in v(f(\phi)) &= \eta \circ \overline{(\eta^{-1} \circ v)}(f(\phi)) = \\
&= \eta(\overline{(\eta^{-1} \circ v)}(f(\phi))) = \\
&= \eta(f(\overline{(\eta^{-1} \circ v)}(\phi))) = \\
&= f(\eta(\overline{(\eta^{-1} \circ v)}(\phi))) = \\
&= S_{mn} \backslash g^{-1}(\eta(\overline{(\eta^{-1} \circ v)}(\phi))),
\end{aligned}$$

if and only if $g(x) \notin \eta(\overline{(\eta^{-1} \circ v)}(\phi))$.

Thus, $S_{mn} \overset{r}{\models}_{v,x} f(\phi)$ if and only if $S_{mn} \overset{r}{\models}_{v,g(x)} \phi$. □

In this case, it is easy to see that the space S_{mn} is much simpler than the Ockham algebra L_{mn}, which we did not explicitly construct here. However, note that L_{mn} has 2^m elements.

5.4.2 $SHKn$-Logic

Let $SHKn$ be the logic obtained from the SHn logic by adjoining the following axiom:

$$(A16) \qquad (a\wedge \sim a) \Rightarrow (b\vee \sim b).$$

It is easy to see that $SHKn$-logics are sound and complete with respect to the subvariety $SHKn$ of the variety of SHn-algebras, consisting of those SHn-algebras that satisfy the Kleene law:

$$(a\wedge \sim a) \leq (b\vee \sim b).$$

The fact that the Lukasiewicz-Moisil algebras of order n (cf. Definition 3.15) satisfy the Kleene property has been proved in [Sic67] and [Cig70] in two different ways. The fact that every SHn-algebra that satisfies the Kleene law is a Lukasiewicz-Moisil algebra of order n is proved in [Itu82].

Theorem 5.78 ([Itu82]) *For a SHn-algebra A the following conditions are equivalent:*
(1) A satisfies the Kleene law,
(2) A is a Lukasiewicz-Moisil algebras of order n.

In [IO96], in order to obtain a Kripke semantics for $SHKn$-logics, $SHKn$-frames are defined as SHn-frames that satisfy the following condition:

$$(K14) \qquad R(x, g(x)) \text{ or } R(g(x), x).$$

Soundness and completeness of $SHKn$-logics with respect to the class of $SHKn$-frames are proved.

This suggested us to investigate whether the Priestley duality between the category of SHn-algebras and that of SHn-spaces restricts to a dual equivalence between the category of $SHKn$-algebras and a suitable subcategory of SHnSp.

Lemma 5.79 *Let A be an $SHKn$-algebra and let $D(A) = (D(A), \leq, \tau, g, s_1, \ldots, s_n)$ be the SHn-space associated with A by the Priestley duality for SHn-algebras. Then for every $h \in D(A)$, $h \leq g(h)$ or $g(h) \leq h$.*

Proof: Assume that there exists an $h \in \mathsf{Hom}_{D_{01}}(A, \{0,1\})$ in $D(A)$ such that $h \not\leq g(h)$ and $g(h) \not\leq h$. Then there exist two elements of A, say a and b such that $h(a) = 1$, $g(h(a)) = 0$; and $h(b) = 0$, $g(h(b)) = 1$. But $g(h(a)) = 0$ if and only if $h(\sim a) = 1$, and $g(h(b)) = 1$ if and only if $h(\sim b) = 0$.

Therefore, there exist $a, b \in A$ such that $h(a\wedge \sim a) = h(a) \wedge h(\sim a) = 1$ and $h(b\vee \sim b) = h(b) \vee h(\sim b) = 0$. This is a contradiction, because h is a homomorphism of algebras, hence $h(a\wedge \sim a) \leq h(b\vee \sim b)$ for every $a, b \in A$. $\square$

Definition 5.32 *The category* SHKnSp *of* $SHKn$-spaces *has as*

Objects: *spaces* $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ *such that:*
(1) $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ *is an* SHn*-space,*
(2) for every $x \in X$, $x \leq g(x)$ *or* $g(x) \leq x$.

Morphisms: *continuous order-preserving mappings that*
(1) satisfy the condition $(H2')$ *and*
(2) preserve the operations $g, s_1, \ldots, s_{n-1}$.

Lemma 5.80 *Let* $(X, \leq, \tau, g, s_1, \ldots, s_{n-1})$ *be an* $SHKn$*-space. Let* $E(X)$ *be its dual as an* SHn*-space. Then* $E(X)$ *satisfies the Kleene law.*

Proof: Assume that there exist two elements $f_1, f_2 \in E(X) = \mathsf{Hom}_P(X, \{0,1\})$ such that $f_1 \wedge \sim f_1 \not\leq f_2 \vee \sim f_2$. Then there exists an $x \in X$ such that $f_1(x) \wedge \sim f_1(x) = 1$ and $f_2(x) \vee \sim f_2(x) = 0$, i.e. such that $f_1(x) = \sim f_1(x) = 1$ and $f_2(x) = \sim f_2(x) = 0$. By the definition of $\sim$ if follows then that $f_1(x) = 1, f_1(g(x)) = 0, f_2(x) = 0, f_2(g(x)) = 1$. But in X we know that either $x \leq g(x)$ or $g(x) \leq x$, hence we should have either $f_1(x) \leq f_1(g(x))$ and $f_2(x) \leq f_2(g(x))$, or $f_1(g(x)) \leq f_1(x)$ and $f_2(g(x)) \leq f_2(x)$. Contradiction.

This shows that for every $f_1, f_2 \in E(X)$, $f_1 \wedge \sim f_1 \leq f_2 \vee \sim f_2$. □

Theorem 5.81 *The Priestley duality between the category of* SHn*-algebras and that of* SHn*-spaces restricts to a dual equivalence between the category of* $SHKn$*-algebras (which coincides with the category of Lukasiewicz-Moisil algebras) and the category* SHKnSp *of* $SHKn$*-spaces.*

In what follows we will use the term "Lukasiewicz-Moisil algebras" instead of $SHKn$-algebras, because the varieties are the same; the Heyting implication in a Lukasiewicz-Moisil algebra is expressed by $x \Rightarrow y = y \vee \bigwedge_{i=1}^{n-1} (\sim (S_i(x)) \vee S_i(y))$.

Let L_n be the algebra $(\{0, \frac{1}{n-1}, \ldots, \frac{n-2}{n-1}, 1\}, \vee, \wedge, \sim, S_1, \ldots, S_{n-1}, 0, 1)$, where $x \vee y = max(x, y)$, $x \wedge y = min(x, y)$, $\sim x = 1 - x$, and for every $i \in \{1, \ldots, n-1\}$, $S_i(\frac{j}{n-1}) = \begin{cases} 1 & \text{if } i + j \geq n \\ 0 & \text{if } i + j < n \end{cases}$. L_n is a Lukasiewicz-Moisil algebra. On L_n another operation (namely the Heyting implication) can be defined by

$$x \Rightarrow y \ := \ y \vee \bigwedge_{i=1}^{n-1} (\sim (S_i(x)) \vee S_i(y)).$$

It is well-known that the subdirectly irreducible algebras in $\mathcal{L}_n$ are L_n and its subalgebras, each of which is simple. Thus, the variety of Lukasiewicz-Moisil algebras is generated by L_n.

Let $D(\mathrm{L}_n)$ be the Priestley dual of the n-element Lukasiewicz-Moisil algebra. $D(\mathrm{L}_n)$ is isomorphic to the ordered set of the join-irreducible elements of L_n, namely with the ordered set $D(\mathrm{L}_n) = \{\frac{1}{n-1}, \frac{2}{n-1}, \ldots, \frac{n-2}{n-1}, 1\}$ (these elements correspond to the prime filters $\uparrow \frac{i}{n-1}$, $i = 1, \ldots, n-1$). The additional operations $g, s_1, \ldots, s_{n-1}$ are defined by:

$g(\uparrow \frac{j}{n-1}) = \{\frac{k}{n-1} \in \mathrm{L}_n \mid g(\frac{k}{n-1}) \notin \uparrow \frac{j}{n-1}\} = \{\frac{k}{n-1} \in \mathrm{L}_n \mid 1 - \frac{k}{n-1} < \frac{j}{n-1}\} = \uparrow \frac{n-j}{n-1}$;
$s_i(\uparrow \frac{j}{n-1}) = \{\frac{k}{n-1} \in \mathrm{L}_n \mid S_i(\frac{k}{n-1}) \in \uparrow \frac{j}{n-1}\} = \{\frac{k}{n-1} \in \mathrm{L}_n \mid S_i(\frac{k}{n-1}) \geq \frac{j}{n-1}\} = \{\frac{k}{n-1} \in \mathrm{L}_n \mid S_i(\frac{k}{n-1}) = 1\} = \{\frac{k}{n-1} \in \mathrm{L}_n \mid i + k \geq n\} = \{\frac{k}{n-1} \in \mathrm{L}_n \mid \frac{k}{n-1} \geq \frac{n-i}{n-1}\} = \uparrow \frac{n-i}{n-1}$, for every $i = 1, \ldots, n-1$.

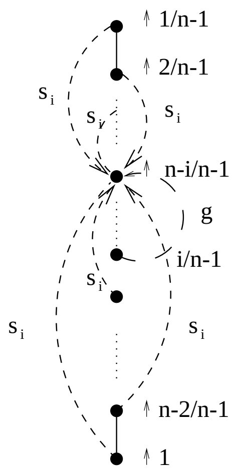


Figure 5.6: $D(\mathrm{L}_n)$: The Priestley Dual of L_n

Since the variety $\mathcal{L}_n$ is generated by the n-element Lukasiewicz-Moisil algebra L_n, it follows that

$$SHKn \models \phi \text{ if and only if } \mathrm{L}_n \models \phi = 1.$$

By the remarks in Section 5.3.1, we can define a relation $\overset{r}{\models}$ as follows:

Let $v : \mathsf{Var} \to \mathcal{O}(D(\mathrm{L}_n))$ be a meaning function that has as values order-filters in $D(\mathrm{L}_n)$. Let $\eta : \mathrm{L}_n \to \mathcal{O}(D(\mathrm{L}_n))$ be the canonical bijection, and let η^{-1} be its inverse. Let $\overline{v} : \mathsf{Fma}(\mathsf{Var}) \to \mathcal{O}(D(\mathrm{L}_n))$ be defined by $\overline{v}(\phi) = \eta \circ \overline{(\eta^{-1} \circ v)}$.

We define

$$D(\mathrm{L}_n) \overset{r}{\models}_{v,x} \phi \text{ if and only if } x \in \overline{v}(\phi),$$

$$D(\mathrm{L}_n) \overset{r}{\models}_{v} \phi \text{ if and only if } \overline{v}(\phi) = D(\mathrm{L}_n),$$

$$D(\mathrm{L}_n) \overset{r}{\models} \phi \text{ if and only if for every } v : \mathsf{Var} \to \mathcal{O}(D(\mathrm{L}_n)), D(\mathrm{L}_n) \overset{r}{\models}_{v} \phi.$$

By Proposition 5.35, for every formula ϕ in the $SHKn$-logic, we have

$$SHKn \models \phi \text{ if and only if } D(\mathrm{L}_n) \overset{r}{\models} \phi.$$

Lemma 5.82 *The following holds:*

(1) $D(L_n) \overset{r}{\models}_{v,x} \phi_1 \wedge \phi_2$ if and only if $D(L_n) \overset{r}{\models}_{v,x} \phi_1$ and $D(L_n) \overset{r}{\models}_{v,x} \phi_2$.

(2) $D(L_n) \overset{r}{\models}_{v,x} \phi_1 \vee \phi_2$ if and only if $D(L_n) \overset{r}{\models}_{v,x} \phi_1$ or $D(L_n) \overset{r}{\models}_{v,x} \phi_2$.

(3) $D(L_n) \overset{r}{\models}_{v,x} S_i(\phi)$ if and only if $D(L_n) \overset{r}{\models}_{v,s_i(x)} \phi$.

(4) $D(L_n) \overset{r}{\models}_{v,x} g(\phi)$ if and only if $D(L_n) \overset{r}{\not\models}_{v,g(x)} \phi$.

Proof: (1) and (2) are obvious. In order to prove (3) note that, taking into account the definition of s_i in $\mathcal{O}(D(\mathrm{L}_n))$ as $s_i(U) = S_i^{-1}(U)$, $D(\mathrm{L}_n) \overset{r}{\models}_{v,x} S_i(\phi)$ if and only if $x \in \overline{v}(S_i(\phi)) = \eta \circ \overline{(\eta^{-1} \circ v)}(S_i(\phi)) = \eta(\overline{(\eta^{-1} \circ v)}(S_i(\phi))) = \eta(S_i(\overline{(\eta^{-1} \circ v)}(\phi))) = S_i(\eta(\overline{(\eta^{-1} \circ v)}(\phi))) = s_i^{-1}(\eta(\overline{(\eta^{-1} \circ v)}(\phi)))$, if and only if $s_i(x) \in \eta(\overline{(\eta^{-1} \circ v)}(\phi)) = \overline{v}(\phi)$. Thus, $D(\mathrm{L}_n) \overset{r}{\models}_{v,x} S_i(\phi)$ if and only if $D(\mathrm{L}_n) \overset{r}{\models}_{v,s_i(x)} \phi$. (4) follows similarly taking into account the definition of g.□

5.5 An Implementation

Theorem proving by resolution is essentially a two stage process. The first step is translating a given assertion to clause form. The second step is the actual proof. The first step is the one which captures the logic; the second is a purely computational, algebraic process, independent of the underlying logic.

The existing implementations of resolution-type methods for many-valued logics normally only present the solution of the first step, namely clause generation. This is the case with the approach of Baaz and Fermüller and that of Hähnle. The second step, the actual proof by resolution is the most time-consuming; it would seem that creating an interface with an existing system in which resolution is implemented (as Otter, Isabelle, etc.) would solve the problem.

In what follows we will present both an algorithm for translating formulae to clause form and an algorithm (based on negative hyperresolution) for the actual proof.

To understand the structure of the method, it is helpful to look at the general structure of the algorithm:

Algorithm for Resolution in Non-Classical Logics

Input: *a formula* ϕ.

Output: □ *if* ϕ *is unsatisfiable.*

Algorithm *:*

> *Find a set* F *of clauses such that* ϕ *is a theorem if and only if* F *is unsatisfiable.*
>
> *Apply many-valued hyperresolution to* F.

Our theorem prover is implemented in SICStus Prolog. According with the remarks above, the automated proof procedure consists of a procedure for the translation to clause form and a proof procedure by negative hyperresolution.

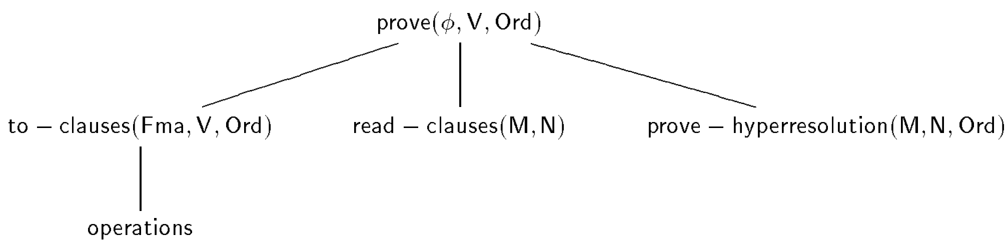
The implementation of the main procedure, **prove**(ϕ, V, Ord), takes as input a formula ϕ, a set of possible worlds V (corresponding to $D(A)$) and an ordering Ord on V. The output is □ if the formula is a theorem.

The data structures used are lists (or atoms, in the case of variables). We represent the data as follows:

- Every propositional variable p is represented as an atom p,
- Every formula of the form $op(t_1, \ldots, t_n)$ is represented as a list of the form $[op, l_1, \ldots, l_n]$, where $l_1, \ldots, l_n$ are the lists that represent, recursively, the formulae $t_1, \ldots, t_n$,
- Every literal of the form $\boxed{x}\ p^t$ (resp. $\boxed{x}\ p^f$) is represented by $[x, p, t]$ (resp. $[x, p, f]$),
- The clauses are represented as lists of literals, such that the positive literals occur first. Thus, a clause is negative if and only if it begins with a negative literal.

We made our implementation in SICStus Prolog as modular as possible. The parts that are logic-specific are either given as an argument to the procedure (as V and Ord) or are supposed to be given separately (as are the definitions of the special operators on $D(A)$).

Roughly, the structure of the **prove**(ϕ, V, Ord) is as follows:



Here are brief descriptions of the procedures:

- **to-clauses** generates the set F of clauses, which are written in a separate file.
- **read-clauses(M, N)** reads the clauses and separates them in the list M of nonnegative clauses and the list N of negative clauses.
- **prove-hyperresolution(M, N, Ord)** applies negative hyperresolution to the clauses $M \cup N$, taking into account that the list M contains the nonnegative clauses and N the negative clauses.
- **operations** contains definitions of the operations on V, corresponding to the specific logic. It is used in generating the clauses, in the procedure **to-clauses**.

These procedures will be presented into more detail in Sections 5.5.1 and 5.5.2.

For a more comfortable manipulation of lists we also defined some procedures of technical nature. They are contained in the source code and we will not describe them here.

5.5.1 Implementation for the Translation to Clause Form

The procedure for translation to clause form is based on Corollary 5.47 and Lemma 5.48; if any other operations, like for example Heyting implication or modal operators are involved, their transition to clause form is based on lemmas such as Lemma 5.50 or Lemma 5.51.

Corollary 5.47 states that ϕ is a theorem if and only if the following conjunction of formulae is unsatisfiable:

$$(\boxed{\uparrow(0,1)}\,p^f_\phi \vee \boxed{\uparrow(1,0)}\,p^f_\phi) \wedge \bigwedge_{x \in D(S_{n^2})} \bigwedge_{\psi \text{ subformula of } \phi} (\boxed{x}\,p^f_\psi \vee \boxed{x}\,\psi^t) \wedge (\boxed{x}\,p^t_\psi \vee \boxed{x}\,\psi^f)$$

The general structure of a procedure for transition to clause form, based on Corollary 5.47, is as follows:

Algorithm for Translation to Clause Form

Input: *a formula* ϕ.

Output: *A set of clauses* F *such that* ϕ *is a theorem if and only if* F *is unsatisfiable.*

Algorithm to-clauses(ϕ, V, Ord):

Let p_ϕ *be a new propositional symbol.*

$C_1 := \{\boxed{x}\, p^f_\phi \mid x$ *minimal in* $D(A)\}$,

$F := \{C_1\}$.

for *all subformulae* ψ *of* ϕ *that are not variables*

do

Let p_ψ *be a new propositional symbol.*

F_1 :=clauses-transform($\boxed{x}\psi^t$),

F_2 :=clauses-transform($\boxed{x}\psi^f$),

$F := F \cup \{\{\boxed{x} p^f_\psi\} \cup C \mid C \in F_1\} \cup \{\{\boxed{x} p^t_\psi\} \cup C \mid C \in F_2\}$.

od

The procedure clauses-transform generates the clauses associated with $\boxed{x}\psi^t$ resp. $\boxed{x}\psi^f$, according to the structure of ψ.

We illustrate the ideas for the case of SHn-logics.

Procedure clauses-transform for SHn-logic

Input: *an expression e of the form $\boxed{x}\psi^t$ or $\boxed{x}\psi^f$, where ψ is not a propositional variable.*

Output: *A set of signed clauses F that is satisfiable if and only of the expression e is satisfiable.*

Algorithm *:*

We distinguish several cases:

if $e = \boxed{x}\ (\psi_1 \vee \psi_2)^t$ **then** $F := \{\{\ \boxed{x}\ p^t_{\psi_1}, \boxed{x}\ p^t_{\psi_2}\}\}$,
if $e = \boxed{x}\ (\psi_1 \vee \psi_2)^f$ **then** $F := \{\{\ \boxed{x}\ p^f_{\psi_1}\}, \{\ \boxed{x}\ p^f_{\psi_2}\}\}$,
if $e = \boxed{x}\ (\psi_1 \wedge \psi_2)^t$ **then** $F := \{\{\ \boxed{x}\ p^t_{\psi_1}\}, \{\ \boxed{x}\ p^t_{\psi_2}\}\}$,
if $e = \boxed{x}\ (\psi_1 \wedge \psi_2)^f$ **then** $F := \{\{\ \boxed{x}\ p^f_{\psi_1}, \boxed{x}\ p^f_{\psi_2}\}\}$,
if $e = \boxed{x}\ S_j(\psi)^t$ **then** $F := \{\{\ \boxed{s_j(x)}\ \psi^t\}\}$,
if $e = \boxed{x}\ S_j(\psi)^f$ **then** $F := \{\{\ \boxed{s_j(x)}\ \psi^f\}\}$,
if $e = \boxed{x}\ \sim(\psi)^t$ **then** $F := \{\{\ \boxed{g(x)}\ \psi^f\}\}$,
if $e = \boxed{x}\ \sim(\psi)^f$ **then** $F := \{\{\ \boxed{g(x)}\ \psi^t\}\}$,
if $e = \boxed{x}\ (\psi_1 \Rightarrow \psi_2)^t$ **then** $F := \{\{\boxed{y}\ {\psi_1}^f, \boxed{y}\ {\psi_2}^t\} \mid y \geq x\}$,
if $e = \boxed{x}\ (\psi_1 \Rightarrow \psi_2)^f$ **then** $F := \{\{\boxed{\mathsf{max}\{y \mid y \geq x\}}\ {\psi_1}^t\}, \{\boxed{x}\ {\psi_2}^f\}\} \cup$
$\{\{\boxed{x_1}\ {\psi_1}^t, \boxed{x_2}\ {\psi_2}^f\} \mid x_1, x_2 \geq x, x_1 \neq x_2\}$,
if $e = \boxed{x}\ \neg(\psi)^t$ **then** $F := \{\{\boxed{y}\ \psi^f\} \mid y \geq x\}$,
if $e = \boxed{x}\ \neg(\psi)^f$ **then** $F := \{\{\boxed{y}\ \psi^t \mid y \geq x\}\}$,

In the case of first-order logics we also have to take quantifiers into consideration. The structure-preserving translation is done exactly as in the case of propositional logic.

By Lemma 5.62 we additionally have the following cases:

if $e = \boxed{x}\ ((\forall u)A(u))^t$ **then** $F := \{\{\boxed{x}\ A(u)^t\}\}$ for every (ground) term u,

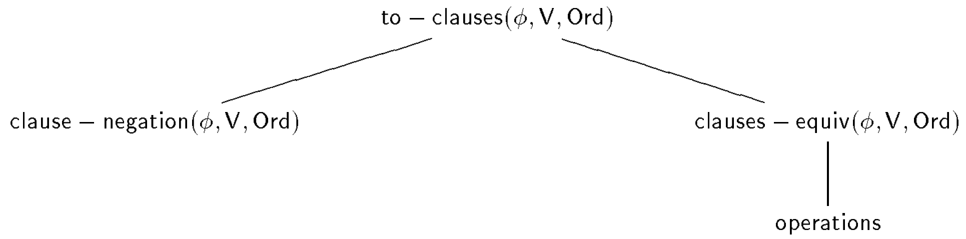
if $e = \boxed{x}\ ((\forall u)A(u))^f$ **then** $F := \{\{\boxed{x}\ A(f(v))^f\}\}$ where f is a new (Skolem) function and v is the set of all variables that appear unbound in $(\forall u)A(u)$.

if $e = \boxed{x}\ ((\exists u)A(u))^t$ **then** $F := \{\{\boxed{x}\ A(f(v))^t\}\}$ where f is a new (Skolem) function and v is the set of all variables that appear unbound in $(\forall u)A(u)$.

if $e = \boxed{x}\,((\exists u)A(u))^f$ **then** $F := \{\{\boxed{x}\,A(u)^f\}\}$ for every (ground) term u.

Our implementation in SICStus Prolog closely follows the algorithm described above. The parts that are logic-specific are either given as an argument to the procedure (as V and Ord) or are supposed to be given separately (as are the definitions of the special operators on $D(A)$).

We briefly describe the structure of our procedure for translation to clause form in more detail.



Here are brief descriptions of the procedures above:

- **clause-negation**(ϕ, V, Ord) generates the clause that is a consequence of the fact that the formula ϕ is supposed to be false, namely that there is some minimal possible world x at which ϕ is false. The clause is $C_1 := \{\boxed{x}\,p^f_\phi \mid x \text{ minimal in } D(A)\}$.
- **clauses-equiv**(ϕ, V, Ord) generates the set F of clauses that are consequences of $\psi \Leftrightarrow p_\psi$ for all subformulae ψ of ϕ.

  The procedure generates all subformulae of ϕ, and for every subformula ψ, according to the structure of the formula and the operations on V (defined in **operations**) it constructs the set of clauses $F = \{\{\boxed{x}p^f_\psi\} \cup C \mid C \in F_1\} \cup \{\{\boxed{x}p^t_\psi\} \cup C \mid C \in F_2\}$, where $F_1 := \mathsf{clauses} - \mathsf{transform}(\boxed{x}\psi^t)$ and $F_2 := \mathsf{clauses} - \mathsf{transform}(\boxed{x}\psi^f)$.

We present some examples:

Example 5.5 *Consider the SHn-logic for $n = 2$. Let $\phi = S_1(p) \vee \neg(S_1(p))$. The algebra S_{2^2} and its dual are represented in Figure 5.7.*

Thus, the dual $D(S_{2^2})$ of S_{2^2} consists of two incomparable elements. We know that $s_1(\uparrow(1,0)) = \uparrow(1,0)$, $s_1(\uparrow(0,1)) = \uparrow(0,1)$, and $g(\uparrow(1,0)) = \uparrow(0,1)$, $g(\uparrow(0,1)) = \uparrow(1,0)$.

We will denote $D(S_{2^2})$ by $V = [a, b]$, where $Ord = []$. The module **operations** *in this case contains the following definitions:*

```
d_s1(a, a).
d_s1(b, b).

d_sim_neg(a, b).
d_sim_neg(b, a).
```

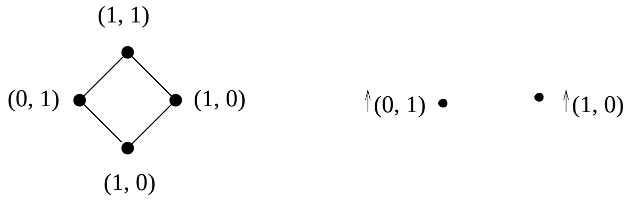


Figure 5.7: S_{2^2} and its Priestley Dual.

corresponding to the fact that $s_1(a) = a, s_1(b) = b, g(a) = b, g(b) = a$.

The following set of clauses is obtained:

```
| ?- to_clauses([or, [s1, p],[neg, [s1, p]]], [a, b], []).

[[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]].

[[a,p_s1_p,t],[a,p_neg_s1_p,t],[a,p_or_s1_p_neg_s1_p,f]].

[[a,p_or_s1_p_neg_s1_p,t],[a,p_s1_p,f]].

[[a,p_or_s1_p_neg_s1_p,t],[a,p_neg_s1_p,f]].

[[a,p_s1_p,f],[a,p_neg_s1_p,f]].

[[a,p_s1_p,t], [a,p_neg_s1_p,t]].

[[a,p,t],[a,p_s1_p,f]].

[[a,p_s1_p,t],[a,p,f]].

[[b,p_s1_p,t],[b,p_neg_s1_p,t],[b,p_or_s1_p_neg_s1_p,f]].

[[b,p_or_s1_p_neg_s1_p,t],[b,p_s1_p,f]].

[[b,p_or_s1_p_neg_s1_p,t],[b,p_neg_s1_p,f]].

[[b,p_s1_p,f],[b,p_neg_s1_p,f]].

[[b,p_s1_p,t], [b,p_neg_s1_p,t]].

[[b,p,t],[b,p_s1_p,f]].

[[b,p_s1_p,t],[b,p,f]].
```

Example 5.6 *Consider now the* SHn*-logic for* $n = 3$. *Let* $\phi = S_1(p) \vee \neg(S_1(p))$ *as above. The algebra* S_{3^2} *and its dual are represented in Figure 5.8.*

We know that $s_1(\uparrow (1,0)) = s_1(\uparrow (\frac{1}{2},0)) = \uparrow (1,0)$ *and* $s_1(\uparrow (0,1)) = s_1(\uparrow (0,\frac{1}{2})) = \uparrow (0,1)$; *moreover,* $s_2(\uparrow (1,0)) = s_1(\uparrow (\frac{1}{2},0)) = \uparrow (\frac{1}{2},0)$ *and* $s_2(\uparrow (0,1)) = s_1(\uparrow (0,\frac{1}{2})) = \uparrow (0,\frac{1}{2})$.

We will denote $D(S_{3^2})$ *by* $V = [a,b,c,d]$, *where* $Ord = [[a,b],[c,d]]$.

The module **operations** *in this case contains the following definitions:*

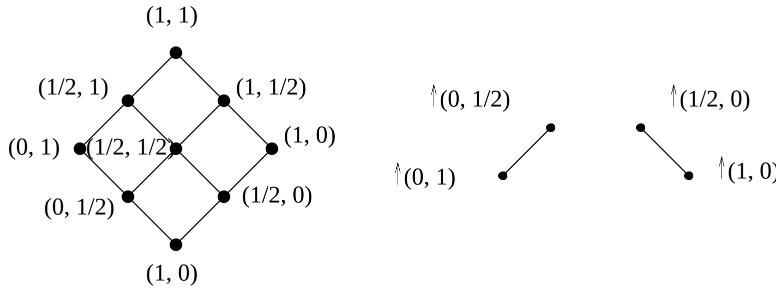


Figure 5.8: S_{3^2} and its Priestley dual.

```
d_s1(X, a) :- X = a; X = b.
d_s1(X, c) :- X = c; X = d.

d_s2(X, b) :- X = a; X = b.
d_s2(X, d) :- X = c; X = d.

d_sim_neg(a, d).
d_sim_neg(d, a).

d_sim_neg(b, c).
d_sim_neg(c, b).
```

corresponding to the fact that $s_1(a) = s_1(b) = a, s_1(c) = s_1(d) = c,$ $s_2(a) = s_2(b) = b, s_2(c) = s_2(d) = d,$ *and* $g(a) = d, g(d) = a, g(b) = c, g(c) = b.$

The following set of clauses is obtained:

```
| ?- to_clauses([or, [s1, p],[neg, [s1, p]]], [a, b, c, d], [[a, b], [c, d]]).

[[a,p_or_s1_p_neg_s1_p,f],[c,p_or_s1_p_neg_s1_p,f]].

[[a,p_s1_p,t],[a,p_neg_s1_p,t],[a,p_or_s1_p_neg_s1_p,f]].

[[a,p_or_s1_p_neg_s1_p,t],[a,p_s1_p,f]].

[[a,p_or_s1_p_neg_s1_p,t],[a,p_neg_s1_p,f]].

[[a,p_s1_p,f],[a,p_neg_s1_p,f]].

[[b,p_s1_p,f],[a,p_neg_s1_p,f]].

[[a,p_s1_p,t], [b,p_s1_p,t], [a,p_neg_s1_p,t]].

[[a,p,t],[a,p_s1_p,f]].

[[a,p_s1_p,t],[a,p,f]].

[[b,p_s1_p,t],[b,p_neg_s1_p,t],[b,p_or_s1_p_neg_s1_p,f]].

[[b,p_or_s1_p_neg_s1_p,t],[b,p_s1_p,f]].
```

```
[[b,p_or_s1_p_neg_s1_p,t],[b,p_neg_s1_p,f]].

[[b,p_s1_p,f],[b,p_neg_s1_p,f]].

[[b,p_s1_p,t], [b,p_neg_s1_p,t]].

[[a,p,t],[b,p_s1_p,f]].

[[b,p_s1_p,t],[a,p,f]].

[[c,p_s1_p,t],[c,p_neg_s1_p,t],[c,p_or_s1_p_neg_s1_p,f]].

[[c,p_or_s1_p_neg_s1_p,t],[c,p_s1_p,f]].

[[c,p_or_s1_p_neg_s1_p,t],[c,p_neg_s1_p,f]].

[[c,p_s1_p,f],[c,p_neg_s1_p,f]].

[[d,p_s1_p,f],[c,p_neg_s1_p,f]].

[[c,p_s1_p,t], [d,p_s1_p,t], [c,p_neg_s1_p,t]].

[[c,p,t],[c,p_s1_p,f]].

[[c,p_s1_p,t],[c,p,f]].

[[d,p_s1_p,t],[d,p_neg_s1_p,t],[d,p_or_s1_p_neg_s1_p,f]].

[[d,p_or_s1_p_neg_s1_p,t],[d,p_s1_p,f]].

[[d,p_or_s1_p_neg_s1_p,t],[d,p_neg_s1_p,f]].

[[d,p_s1_p,f],[d,p_neg_s1_p,f]].

[[d,p_s1_p,t], [d,p_neg_s1_p,t]].

[[c,p,t],[d,p_s1_p,f]].

[[d,p_s1_p,t],[c,p,f]].
```

Example 5.7 *Consider the logic $\mathcal{L}_{mn}$, discussed in Section 5.4.1, with $m = 2$ and $n = 1$. Let $\phi = f^2(p) \vee f(p)$. We know that $g(0) = 1$, $g(1) = 1$. We*

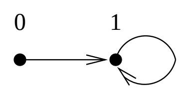


Figure 5.9: The space S_{21}

will denote S_{21} by $[a, b]$, with $Ord = []$. The module **operations** *in this case contains the following definitions:*

```
d_f(a, b).
d_f(b, b).
```

The following set of clauses is obtained:

```
| ?- to_clauses([or, [f, [f, p]], [f, p]], [a, b], []).

[[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]].

[[a,p_f_f_p,t],[a,p_f_p,t],[a,p_or_f_f_p_f_p,f]].

[[a,p_or_f_f_p_f_p,t],[a,p_f_f_p,f]].

[[a,p_or_f_f_p_f_p,t],[a,p_f_p,f]].

[[b,p_f_p,f],[a,p_f_f_p,f]].

[[b,p_f_p,t],[a,p_f_f_p,t]].

[[b,p,f],[a,p_f_p,f]].

[[b,p,t],[a,p_f_p,t]].

[[b,p_f_f_p,t],[b,p_f_p,t],[b,p_or_f_f_p_f_p,f]].

[[b,p_or_f_f_p_f_p,t],[b,p_f_f_p,f]].

[[b,p_or_f_f_p_f_p,t],[b,p_f_p,f]].

[[b,p_f_p,f],[b,p_f_f_p,f]].

[[b,p_f_p,t],[b,p_f_f_p,t]].

[[b,p,f],[b,p_f_p,f]].

[[b,p,t],[b,p_f_p,t]].
```

Example 5.8 *Consider the logic $\mathcal{L}_{mn}$, discussed in Section 5.4.1 with $m = 3$ and $n = 0$. Let $\phi = f^3(p) \vee p$.*

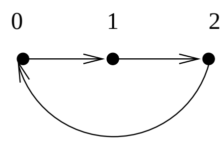


Figure 5.10: The space S_{30}.

We know that $g(0) = 1$, $g(1) = 2$ and $g(2) = 0$. We will denote S_{30} by $[a, b, c]$, with $Ord = []$. The module **operations** *in this case contains the following definitions:*

```
d_f(a, b).
d_f(b, c).
d_f(c, a).
```

The following set of clauses is obtained:

```
| ?- to_clauses([or, [f, [f, [f, p]]], p], [a,b,c], []).

[[a,p_or_f_f_f_p_p,f],[b,p_or_f_f_f_p_p,f],[c,p_or_f_f_f_p_p,f]].

[[a,p_f_f_f_p,t],[a,p,t],[a,p_or_f_f_f_p_p,f]].

[[a,p_or_f_f_f_p_p,t],[a,p_f_f_f_p,f]].

[[a,p_or_f_f_f_p_p,t],[a,p,f]].

[[b,p_f_f_p,f],[a,p_f_f_f_p,f]].

[[b,p_f_f_p,t],[a,p_f_f_f_p,t]].

[[b,p_f_p,f],[a,p_f_f_p,f]].

[[b,p_f_p,t],[a,p_f_f_p,t]].

[[b,p,f],[a,p_f_p,f]].

[[b,p,t],[a,p_f_p,t]].

[[b,p_f_f_f_p,t],[b,p,t],[b,p_or_f_f_f_p_p,f]].

[[b,p_or_f_f_f_p_p,t],[b,p_f_f_f_p,f]].

[[b,p_or_f_f_f_p_p,t],[b,p,f]].

[[c,p_f_f_p,f],[b,p_f_f_f_p,f]].

[[c,p_f_f_p,t],[b,p_f_f_f_p,t]].

[[c,p_f_p,f],[b,p_f_f_p,f]].

[[c,p_f_p,t],[b,p_f_f_p,t]].

[[c,p,f],[b,p_f_p,f]].

[[c,p,t],[b,p_f_p,t]].

[[c,p_f_f_f_p,t],[c,p,t],[c,p_or_f_f_f_p_p,f]].

[[c,p_or_f_f_f_p_p,t],[c,p_f_f_f_p,f]].

[[c,p_or_f_f_f_p_p,t],[c,p,f]].

[[a,p_f_f_p,f],[c,p_f_f_f_p,f]].

[[a,p_f_f_p,t],[c,p_f_f_f_p,t]].

[[a,p_f_p,f],[c,p_f_f_p,f]].

[[a,p_f_p,t],[c,p_f_f_p,t]].

[[a,p,f],[c,p_f_p,f]].

[[a,p,t],[c,p_f_p,t]].
```

It can be seen from the examples above that in the propositional case this method is not always more efficient than the method of direct verification: in Example 5.5 (resp. 5.6) only one propositional variable occurs, hence there are only 4 (resp. 9) possible values for this variable to be tested.

In Example 5.7 and Example 5.8 the advantage of this method consists of the fact that we do not need to compute the algebra L_{21} (resp. L_{30}) and can directly use its dual space. However, this algebras have $2^2 = 4$ (resp. $2^3 = 8$) elements, hence a direct verification is also possible.

In general if there are few variables the method of direct verification is more efficient, whereas if many variables occur our method may be better. Nevertheless, the real advantages of the use of resolution in many-valued theorem proving are in automated theorem proving for the first-order case, where no direct verification by plugging in truth values can be applied. In what follows we will illustrate the way clauses are generated in first-order logic.

Example 5.9 *Consider a first-order version of Example 5.5. Let $A(x)$ be a formula in this logic, containing a free variable x and let $\phi = (\forall x)S_1(A(x)) \vee \neg(S_1(A(x)))$.*

Therefore in this case a clause form for ϕ is:

```
| ?- to_clauses([[forall, x], [or, [s1, [a, x]], [neg, [s1, [a, x]]]]], [a, b],
    []).
[[a,p_forall_x_or_s1_a_x_neg_s1_a_x,f],[b,p_forall_x_or_s1_a_x_neg_s1_a_x,f]].

[[a,[p_or_s1_a_x_neg_s1_a_x,x],t],[a,p_forall_x_or_s1_a_x_neg_s1_a_x,f]].

[[a,p_forall_x_or_s1_a_x_neg_s1_a_x,t],[a,[p_or_s1_a_x_neg_s1_a_x,f1],f]].

[[a,[p_s1_a_x,x],t],[a,[p_neg_s1_a_x,x],t],[a,[p_or_s1_a_x_neg_s1_a_x,x],f]].

[[a,[p_or_s1_a_x_neg_s1_a_x,x],t],[a,[p_s1_a_x,x],f]].

[[a,[p_or_s1_a_x_neg_s1_a_x,x],t],[a,[p_neg_s1_a_x,x],f]].

[[a,[p_s1_a_x,x],f],[a,[p_neg_s1_a_x,x],f]].

[[a,[p_s1_a_x,x],t], [a,[p_neg_s1_a_x,x],t]].

[[a,[p_a_x,x],t],[a,[p_s1_a_x,x],f]].

[[a,[p_s1_a_x,x],t],[a,[p_a_x,x],f]].

[[b,[p_or_s1_a_x_neg_s1_a_x,x],t],[b,p_forall_x_or_s1_a_x_neg_s1_a_x,f]].

[[b,p_forall_x_or_s1_a_x_neg_s1_a_x,t],[b,[p_or_s1_a_x_neg_s1_a_x,f2],f]].

[[b,[p_s1_a_x,x],t],[b,[p_neg_s1_a_x,x],t],[b,[p_or_s1_a_x_neg_s1_a_x,x],f]].

[[b,[p_or_s1_a_x_neg_s1_a_x,x],t],[b,[p_s1_a_x,x],f]].

[[b,[p_or_s1_a_x_neg_s1_a_x,x],t],[b,[p_neg_s1_a_x,x],f]].

[[b,[p_s1_a_x,x],f],[b,[p_neg_s1_a_x,x],f]].
```

```
[[b,[p_s1_a_x,x],t],[b,[p_neg_s1_a_x,x],t]].

[[a,[p_a_x,x],t],[b,[p_s1_a_x,x],f]].

[[b,[p_s1_a_x,x],t],[a,[p_a_x,x],f]].
```

5.5.2 Hyperresolution

The procedure for theorem proving by hyperresolution is an adaptation of the algorithm given in [CL73].

Algorithm for many-valued hyperresolution

Input: *A set F of signed clauses.*

Output: $\square$ *if F is unsatisfiable.*

Algorithm *:*

$M :=$ *The set of all negative clauses in F,*
$N :=$ *The set of all non-negative clauses in F,*

$i := 0$,

$A_0 := \emptyset$, $B_0 := N$,

repeat

while A_i *does not contain* $\square$ *and* $B_i \neq \emptyset$
do

$W_{i+1} :=$ *The set of ordered resolvents of C_1 against C_2, where C_1 is an ordered clause or an ordered factor of an ordered clause in M, C_2 is an ordered clause in B, the resolved literal of C_1 contains the "largest" predicate symbol in C_1, and the resolved literal of C_2 is the "last" literal of C_2.*

$A_{i+1} :=$ *The set of negative ordered clauses in W_{i+1},*

$B_{i+1} :=$ *The set of non-negative ordered clauses in W_{i+1},*

$i := i + 1$,

od

if A_i *contains* $\square$ **then Return**,

$T := A_0 \cup \ldots \cup A_i$, $M := M \cup T$,

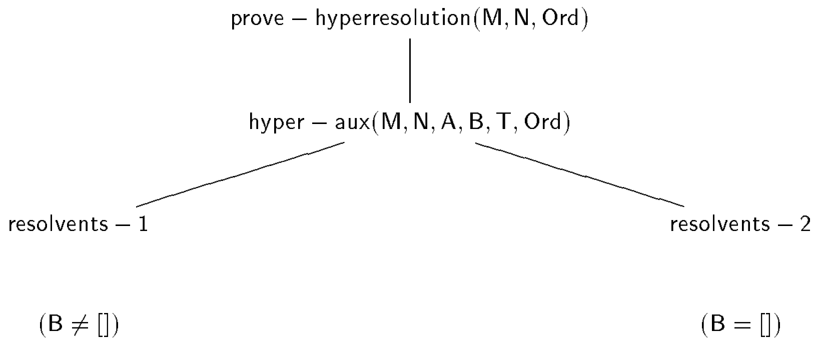
$R :=$ *The set of ordered resolvents of C_1 against C_2, where C_1 is an ordered clause or an ordered factor of an ordered clause in T, C_2 is an ordered clause in N, and the resolved literal in C_1 contains the "largest" predicate symbol in C_1.*

(Note that the resolved literal in C_2 can be any literal in C_2, not necessarily the last literal.)

$A_0 :=$ *The set of all negative literals in R,*

B_0 := *The set of all nonnegative literals in R.*

Our implementation in SICStus Prolog follows the algorithm described above. We briefly describe the structure of our procedure for negative hyperresolution.



Here are brief descriptions of the procedures above:

- **prove-hyperresolution(M, N, Ord)** calls **hyper-aux(M, N, [], N, [], Ord)**
- **hyper-aux(M, N, A, N, T, Ord)** If B is not empty, this procedure calls **resolvents-1** which generates a set W of ordered clauses between M and B. Let $A1$ be the list of negative clauses in W, and $B1$ the list of non-negative clauses in W. $A1$ is added to T to form $T1$ (the list of newly generated negative clauses). Then **hyper-aux** is applied recursively to $M, N, A1, B1, T1$. This continues until either no non-negative clauses are generated or a contradiction is deduced.

  If $B = []$, the resolvents between T and N are generated with **resolvents-2**. Let $A1, B1$ be the lists of negative resp. non-negative clauses generated this way. The procedure **hyper-aux** is applied recursively to $M, N, A1, B1, []$.
- **resolvents-1(M, B, M1, N1, Ord-Pred, Ord)** generates the set of ordered resolvents of C_1 against C_2 where C_1 is an ordered clause or an ordered factor of an ordered clause in M, C_2 is an ordered clause in B, the resolved literal of C_1 contains the "largest" predicate symbol in C_1, and the resolved literal of C_2 is the "last" literal of C_2.
- **resolvents-2(T, N, M1, N1, Ord-Pred, Ord)** generates the set of ordered resolvents of C_1 against C_2 where C_1 is an ordered clause or an ordered factor of an ordered clause in T, C_2 is an ordered clause in N, and the resolved literal of C_1 contains the "largest" predicate symbol in C_1.

Example 5.10 *Consider the SHn-logic for $n = 2$, as described in Example 5.5. Let $\phi = S_1(p) \vee \neg(S_1(p))$. In what follows we present the execution of the* *procedure* prove$(\phi, [\mathsf{a}, \mathsf{b}], [])$. *All the clauses generated at every application of* resolvents-1 *or* resolvents-2 *are explicitly printed.*

```
| ?- prove([or, [s1, p],[neg, [s1, p]]], [a, b], []).
hyper_aux:
M:
1  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

B:
1  [[a,p_s1_p,t],[a,p_neg_s1_p,t],[a,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_or_s1_p_neg_s1_p,t],[a,p_s1_p,f]]
3  [[a,p_or_s1_p_neg_s1_p,t],[a,p_neg_s1_p,f]]
4  [[a,p_s1_p,t],[a,p_neg_s1_p,t]]
5  [[a,p,t],[a,p_s1_p,f]]
6  [[a,p_s1_p,t],[a,p,f]]
7  [[b,p_s1_p,t],[b,p_neg_s1_p,t],[b,p_or_s1_p_neg_s1_p,f]]
8  [[b,p_or_s1_p_neg_s1_p,t],[b,p_s1_p,f]]
9  [[b,p_or_s1_p_neg_s1_p,t],[b,p_neg_s1_p,f]]
10  [[b,p_s1_p,t],[b,p_neg_s1_p,t]]
11  [[b,p,t],[b,p_s1_p,f]]
12  [[b,p_s1_p,t],[b,p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[a,p_neg_s1_p,f]]
6  [[b,p,f],[b,p_neg_s1_p,f]]

B1:

hyper_aux: B = []

T:
1  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[a,p_neg_s1_p,f]]
6  [[b,p,f],[b,p_neg_s1_p,f]]

M:
1  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]
 M1:
1  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[a,p_neg_s1_p,f]]
6  [[b,p,f],[b,p_neg_s1_p,f]]
7  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
8  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
9  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

out of resolvents_2
A1:
1  [[a,p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
```

```
3  [[b,p,f],[a,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
5  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]

B1:
1  [[a,p_neg_s1_p,t],[b,p_or_s1_p_neg_s1_p,f]]
2  [[b,p_neg_s1_p,t],[a,p_or_s1_p_neg_s1_p,f]]
3  [[a,p_s1_p,t],[a,p_or_s1_p_neg_s1_p,f],[a,p,f]]
4  [[b,p_s1_p,t],[b,p_or_s1_p_neg_s1_p,f],[b,p,f]]

hyper_aux:
M:
1  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[a,p_neg_s1_p,f]]
6  [[b,p,f],[b,p_neg_s1_p,f]]
7  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
8  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
9  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

B:
1  [[a,p_neg_s1_p,t],[b,p_or_s1_p_neg_s1_p,f]]
2  [[b,p_neg_s1_p,t],[a,p_or_s1_p_neg_s1_p,f]]
3  [[a,p_s1_p,t],[a,p_or_s1_p_neg_s1_p,f],[a,p,f]]
4  [[b,p_s1_p,t],[b,p_or_s1_p_neg_s1_p,f],[b,p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
2  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
3  [[a,p_or_s1_p_neg_s1_p,f],[a,p,f],[a,p_neg_s1_p,f]]
4  [[b,p_or_s1_p_neg_s1_p,f],[b,p,f],[b,p_neg_s1_p,f]]

B1:

hyper_aux: B = []

T:
1  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
3  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
4  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
5  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
6  [[a,p_or_s1_p_neg_s1_p,f],[a,p,f],[a,p_neg_s1_p,f]]
7  [[b,p_or_s1_p_neg_s1_p,f],[b,p,f],[b,p_neg_s1_p,f]]

M:
1  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
3  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[a,p_neg_s1_p,f]]
6  [[b,p,f],[b,p_neg_s1_p,f]]
7  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
8  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
9  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]
 M1:
1  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
3  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
4  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
5  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
```

```
6  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
7  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
8  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
9  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
10  [[a,p,f],[a,p_neg_s1_p,f]]
11  [[b,p,f],[b,p_neg_s1_p,f]]
12  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

out of resolvents_2
A1:
1  [[b,p_s1_p,f],[a,p,f]]
2  [[b,p_neg_s1_p,f],[a,p,f]]
3  [[a,p_s1_p,f],[b,p,f]]
4  [[a,p_neg_s1_p,f],[b,p,f]]
5  [[a,p_s1_p,f],[a,p,f],[a,p_neg_s1_p,f]]
6  [[b,p_s1_p,f],[b,p,f],[b,p_neg_s1_p,f]]

B1:
1  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]
2  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]

hyper_aux:
M:
1  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
3  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
4  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
5  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
6  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
7  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
8  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
9  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
10  [[a,p,f],[a,p_neg_s1_p,f]]
11  [[b,p,f],[b,p_neg_s1_p,f]]
12  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

B:
1  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]
2  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_neg_s1_p,f]]
2  [[b,p_neg_s1_p,f]]
3  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
5  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]

B1:

hyper_aux: B = []

T:
1  [[b,p_s1_p,f],[a,p,f]]
2  [[a,p_s1_p,f],[b,p,f]]
3  [[a,p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f]]
5  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
7  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]
```

```
M:
1  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
3  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
4  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
5  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
6  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
7  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
8  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
9  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
10  [[a,p,f],[a,p_neg_s1_p,f]]
11  [[b,p,f],[b,p_neg_s1_p,f]]
12  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]
 M1:
1  [[b,p_s1_p,f],[a,p,f]]
2  [[a,p_s1_p,f],[b,p,f]]
3  [[a,p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f]]
5  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
7  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
8  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
9  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
10  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
11  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
12  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
15  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
16  [[a,p,f],[a,p_neg_s1_p,f]]
17  [[b,p,f],[b,p_neg_s1_p,f]]
18  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
19  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
20  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

out of resolvents_2
A1:
1  [[a,p,f],[b,p,f]]

B1:
1  [[b,p_neg_s1_p,t],[a,p,f]]
2  [[a,p_neg_s1_p,t],[b,p,f]]
3  [[a,p_s1_p,t]]
4  [[b,p_s1_p,t]]
5  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]
6  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]

hyper_aux:
M:
1  [[b,p_s1_p,f],[a,p,f]]
2  [[a,p_s1_p,f],[b,p,f]]
3  [[a,p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f]]
5  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
7  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
8  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
9  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
10  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
11  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
12  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
```

```
15  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
16  [[a,p,f],[a,p_neg_s1_p,f]]
17  [[b,p,f],[b,p_neg_s1_p,f]]
18  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
19  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
20  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

B:
1  [[b,p_neg_s1_p,t],[a,p,f]]
2  [[a,p_neg_s1_p,t],[b,p,f]]
3  [[a,p_s1_p,t]]
4  [[b,p_s1_p,t]]
5  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]
6  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[b,p,f]]
2  [[a,p,f]]
3  [[b,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[b,p,f]]
6  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]

B1:

hyper_aux: B = []

T:
1  [[b,p,f]]
2  [[a,p,f]]
3  [[b,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[b,p,f]]
6  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]

M:
1  [[b,p_s1_p,f],[a,p,f]]
2  [[a,p_s1_p,f],[b,p,f]]
3  [[a,p_neg_s1_p,f]]
4  [[b,p_neg_s1_p,f]]
5  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
7  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
8  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
9  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
10  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
11  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
12  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
15  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
16  [[a,p,f],[a,p_neg_s1_p,f]]
17  [[b,p,f],[b,p_neg_s1_p,f]]
18  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
19  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
20  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]
 M1:
1  [[b,p,f]]
2  [[a,p,f]]
3  [[b,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[b,p,f]]
6  [[b,p_s1_p,f],[a,p,f]]
```

```
7  [[a,p_s1_p,f],[b,p,f]]
8  [[a,p_neg_s1_p,f]]
9  [[b,p_neg_s1_p,f]]
10  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
11  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
12  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
13  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
14  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
15  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
16  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
17  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
18  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
19  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
20  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
21  [[a,p,f],[a,p_neg_s1_p,f]]
22  [[b,p,f],[b,p_neg_s1_p,f]]
23  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
24  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
25  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

out of resolvents_2
A1:
1  [[b,p_s1_p,f]]
2  [[a,p_s1_p,f]]

B1:
1  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]
2  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]

hyper_aux:
M:
1  [[b,p,f]]
2  [[a,p,f]]
3  [[b,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[b,p,f]]
6  [[b,p_s1_p,f],[a,p,f]]
7  [[a,p_s1_p,f],[b,p,f]]
8  [[a,p_neg_s1_p,f]]
9  [[b,p_neg_s1_p,f]]
10  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
11  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
12  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
13  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
14  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
15  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
16  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
17  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
18  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
19  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
20  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
21  [[a,p,f],[a,p_neg_s1_p,f]]
22  [[b,p,f],[b,p_neg_s1_p,f]]
23  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
24  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
25  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

B:
1  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]
2  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_neg_s1_p,f],[a,p,f]]
```

```
2  [[b,p_neg_s1_p,f],[b,p,f]]
3  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]

B1:

hyper_aux: B = []

T:
1  [[b,p_s1_p,f]]
2  [[a,p_s1_p,f]]
3  [[a,p_neg_s1_p,f],[a,p,f]]
4  [[b,p_neg_s1_p,f],[b,p,f]]
5  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]

M:
1  [[b,p,f]]
2  [[a,p,f]]
3  [[b,p_or_s1_p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p,f],[b,p,f]]
6  [[b,p_s1_p,f],[a,p,f]]
7  [[a,p_s1_p,f],[b,p,f]]
8  [[a,p_neg_s1_p,f]]
9  [[b,p_neg_s1_p,f]]
10  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
11  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
12  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
13  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
14  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
15  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
16  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
17  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
18  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
19  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
20  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
21  [[a,p,f],[a,p_neg_s1_p,f]]
22  [[b,p,f],[b,p_neg_s1_p,f]]
23  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
24  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
25  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]
 M1:
1  [[b,p_s1_p,f]]
2  [[a,p_s1_p,f]]
3  [[b,p,f]]
4  [[a,p,f]]
5  [[b,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_or_s1_p_neg_s1_p,f]]
7  [[a,p,f],[b,p,f]]
8  [[b,p_s1_p,f],[a,p,f]]
9  [[a,p_s1_p,f],[b,p,f]]
10  [[a,p_neg_s1_p,f]]
11  [[b,p_neg_s1_p,f]]
12  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
15  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
16  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
17  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
18  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
19  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
20  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
21  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
22  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
23  [[a,p,f],[a,p_neg_s1_p,f]]
24  [[b,p,f],[b,p_neg_s1_p,f]]
25  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
```

```
26  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
27  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

out of resolvents_2
A1:

B1:
1  [[b,p_neg_s1_p,t]]
2  [[a,p_neg_s1_p,t]]
3  [[a,p_s1_p,t],[a,p_or_s1_p_neg_s1_p,f],[a,p,f]]
4  [[b,p_s1_p,t],[b,p_or_s1_p_neg_s1_p,f],[b,p,f]]
5  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]
6  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]

hyper_aux:
M:
1  [[b,p_s1_p,f]]
2  [[a,p_s1_p,f]]
3  [[b,p,f]]
4  [[a,p,f]]
5  [[b,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_or_s1_p_neg_s1_p,f]]
7  [[a,p,f],[b,p,f]]
8  [[b,p_s1_p,f],[a,p,f]]
9  [[a,p_s1_p,f],[b,p,f]]
10  [[a,p_neg_s1_p,f]]
11  [[b,p_neg_s1_p,f]]
12  [[b,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
13  [[a,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
14  [[b,p_s1_p,f],[a,p_neg_s1_p,f]]
15  [[a,p_s1_p,f],[b,p_neg_s1_p,f]]
16  [[a,p_neg_s1_p,f],[b,p_neg_s1_p,f]]
17  [[b,p_or_s1_p_neg_s1_p,f],[a,p,f]]
18  [[a,p_or_s1_p_neg_s1_p,f],[b,p,f]]
19  [[a,p_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
20  [[a,p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
21  [[b,p_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
22  [[b,p_neg_s1_p,f],[a,p_or_s1_p_neg_s1_p,f]]
23  [[a,p,f],[a,p_neg_s1_p,f]]
24  [[b,p,f],[b,p_neg_s1_p,f]]
25  [[a,p_or_s1_p_neg_s1_p,f],[b,p_or_s1_p_neg_s1_p,f]]
26  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
27  [[b,p_s1_p,f],[b,p_neg_s1_p,f]]

B:
1  [[b,p_neg_s1_p,t]]
2  [[a,p_neg_s1_p,t]]
3  [[a,p_s1_p,t],[a,p_or_s1_p_neg_s1_p,f],[a,p,f]]
4  [[b,p_s1_p,t],[b,p_or_s1_p_neg_s1_p,f],[b,p,f]]
5  [[b,p_s1_p,t],[a,p_neg_s1_p,f]]
6  [[a,p_s1_p,t],[b,p_neg_s1_p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  []
2  [[b,p_or_s1_p_neg_s1_p,f],[b,p,f],[a,p_neg_s1_p,f]]
3  [[a,p_or_s1_p_neg_s1_p,f],[a,p,f],[b,p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f],[a,p,f],[b,p_or_s1_p_neg_s1_p,f]]
5  [[b,p_or_s1_p_neg_s1_p,f],[b,p,f],[a,p_or_s1_p_neg_s1_p,f]]
6  [[a,p_or_s1_p_neg_s1_p,f],[a,p,f],[a,p_neg_s1_p,f]]
7  [[b,p_neg_s1_p,f],[a,p_neg_s1_p,f]]
8  [[b,p_or_s1_p_neg_s1_p,f],[b,p,f],[b,p_neg_s1_p,f]]

B1:
```

```
Is a theorem: Contradiction is found
```

Note that if we use the structure of $D(S_{n^2})$ we can reduce the number of clauses that are generated, for those formulae that do not contain the De Morgan negation $\sim$. Namely, we can use the fact that $D(S_{n^2})$ consists of two branches and that the transformation rules for the operations in $\{\vee, \wedge, \neg, \Rightarrow\}$ preserve the branch of $D(S_{n^2})$. It is easy to see that it is sufficient to give a refutation for

$$(\boxed{\uparrow(0,1)}\, p_\phi^f \wedge$$

$$\wedge \bigwedge_i \bigwedge_{\substack{\psi \text{ subformula of } \phi \\ \psi = \psi_1 \diamond \psi_2}} (\boxed{(0,i)}\, p_\psi^f \vee \boxed{(0,i)}\, (p_{\psi_1} \diamond p_{\psi_2})^t) \wedge (\boxed{(0,i)}\, p_\psi^t \vee \boxed{(0,i)}\, (p_{\psi_1} \diamond p_{\psi_2})^f) \wedge$$

$$\wedge \bigwedge_i \bigwedge_{\substack{\psi \text{ subformula of } \phi \\ \psi = \nabla \psi_1}} (\boxed{(0,i)}\, p_\psi^f \vee \boxed{(0,i)}\, (\nabla p_{\psi_1})^t) \wedge (\boxed{(0,i)}\, p_\psi^t \vee \boxed{(0,i)}\, (\nabla p_{\psi_1})^f).$$

(i.e. in just one of the branches of $D(S_{n^2})$); for the other branch a similar refutation can be constructed by simply renaming the nodes, and they can be then combined to a refutation by resolution for

$$(\boxed{\uparrow(0,1)}\, p_\phi^f \vee \boxed{\uparrow(1,0)}\, p_\phi^f) \wedge$$

$$\wedge \bigwedge_{x \in D(S_{n^2})} \bigwedge_{\substack{\psi \text{ subformula of } \phi \\ \psi = \psi_1 \diamond \psi_2}} (\boxed{x}\, p_\psi^f \vee \boxed{x}\, (p_{\psi_1} \diamond p_{\psi_2})^t) \wedge (\boxed{x}\, p_\psi^t \vee \boxed{x}\, (p_{\psi_1} \diamond p_{\psi_2})^f) \wedge$$

$$\wedge \bigwedge_{x \in D(S_{n^2})} \bigwedge_{\substack{\psi \text{ subformula of } \phi \\ \psi = \nabla \psi_1}} (\boxed{x}\, p_\psi^f \vee \boxed{x}\, (\nabla p_{\psi_1})^t) \wedge (\boxed{x}\, p_\psi^t \vee \boxed{x}\, (\nabla p_{\psi_1})^f).$$

Example 5.11 *Consider the SHn-logic for $n = 2$, as described in Example 5.5, and let $\phi = S_1(p) \vee \neg(S_1(p))$. If we consider only the branch $\{a\}$ of $D(S_{2^2})$, we obtain:*

```
| ?- read_clauses(M, N, res_1),  prove_hyperresolution(M, N, []).
hyper_aux:
M:
1  [[a,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]

B:
1  [[a,p_s1_p,t],[a,p_neg_s1_p,t],[a,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_or_s1_p_neg_s1_p,t],[a,p_s1_p,f]]
3  [[a,p_or_s1_p_neg_s1_p,t],[a,p_neg_s1_p,f]]
4  [[a,p_s1_p,t],[a,p_neg_s1_p,t]]
5  [[a,p,t],[a,p_s1_p,f]]
6  [[a,p_s1_p,t],[a,p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_s1_p,f]]
2  [[a,p_neg_s1_p,f]]
3  [[a,p,f],[a,p_neg_s1_p,f]]

B1:
```

```
hyper_aux: B = []

T:
1  [[a,p_s1_p,f]]
2  [[a,p_neg_s1_p,f]]
3  [[a,p,f],[a,p_neg_s1_p,f]]

M:
1  [[a,p_or_s1_p_neg_s1_p,f]]
2  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]
 M1:
1  [[a,p_s1_p,f]]
2  [[a,p_neg_s1_p,f]]
3  [[a,p,f],[a,p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]

out of resolvents_2
A1:
1  [[a,p,f]]

B1:
1  [[a,p_neg_s1_p,t]]
2  [[a,p_s1_p,t]]
3  [[a,p_s1_p,t],[a,p_or_s1_p_neg_s1_p,f],[a,p,f]]

hyper_aux:
M:
1  [[a,p_s1_p,f]]
2  [[a,p_neg_s1_p,f]]
3  [[a,p,f],[a,p_neg_s1_p,f]]
4  [[a,p_or_s1_p_neg_s1_p,f]]
5  [[a,p_s1_p,f],[a,p_neg_s1_p,f]]

B:
1  [[a,p_neg_s1_p,t]]
2  [[a,p_s1_p,t]]
3  [[a,p_s1_p,t],[a,p_or_s1_p_neg_s1_p,f],[a,p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  []
2  [[a,p,f]]
3  [[a,p_or_s1_p_neg_s1_p,f],[a,p,f],[a,p_neg_s1_p,f]]

B1:

Is a theorem: Contradiction is found
```

Note that this is specific to the SHn-logics. In what follows we illustrate the general procedure for $\mathcal{L}_{mn}$-logics.

Example 5.12 *Consider the $\mathcal{L}_{mn}$ logic, with $m = 2$ and $n = 1$, as described in Example 5.7. Let $\phi = f^2(p) \vee f(p)$. In what follows we present the execution of the procedure* prove(ϕ, [a, b], []). *All the clauses generated at every application of* resolution-1 *or* resolution-2 *are explicitly printed.*

```
| ?- prove([or, [f, [f, p]], [f, p]], [a, b], []).
hyper_aux:
M:
1  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
```

```
2  [[b,p_f_p,f],[a,p_f_f_p,f]]
3  [[b,p,f],[a,p_f_p,f]]
4  [[b,p_f_p,f],[b,p_f_f_p,f]]
5  [[b,p,f],[b,p_f_p,f]]

B:
1  [[a,p_f_f_p,t],[a,p_f_p,t],[a,p_or_f_f_p_f_p,f]]
2  [[a,p_or_f_f_p_f_p,t],[a,p_f_f_p,f]]
3  [[a,p_or_f_f_p_f_p,t],[a,p_f_p,f]]
4  [[b,p_f_p,t],[a,p_f_f_p,t]]
5  [[b,p,t],[a,p_f_p,t]]
6  [[b,p_f_f_p,t],[b,p_f_p,t],[b,p_or_f_f_p_f_p,f]]
7  [[b,p_or_f_f_p_f_p,t],[b,p_f_f_p,f]]
8  [[b,p_or_f_f_p_f_p,t],[b,p_f_p,f]]
9  [[b,p_f_p,t],[b,p_f_f_p,t]]
10  [[b,p,t],[b,p_f_p,t]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
2  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
3  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
4  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]

B1:
1  [[a,p_f_f_p,t],[b,p,f]]
2  [[b,p_f_f_p,t],[b,p,f]]

hyper_aux:
M:
1  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
2  [[b,p_f_p,f],[a,p_f_f_p,f]]
3  [[b,p,f],[a,p_f_p,f]]
4  [[b,p_f_p,f],[b,p_f_f_p,f]]
5  [[b,p,f],[b,p_f_p,f]]

B:
1  [[a,p_f_f_p,t],[b,p,f]]
2  [[b,p_f_f_p,t],[b,p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:

B1:

hyper_aux: B = []

T:
1  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
2  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
3  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
4  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]

M:
1  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
2  [[b,p_f_p,f],[a,p_f_f_p,f]]
3  [[b,p,f],[a,p_f_p,f]]
4  [[b,p_f_p,f],[b,p_f_f_p,f]]
5  [[b,p,f],[b,p_f_p,f]]
 M1:
1  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
2  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
```

```
3  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
4  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
5  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
6  [[b,p_f_p,f],[a,p_f_f_p,f]]
7  [[b,p,f],[a,p_f_p,f]]
8  [[b,p_f_p,f],[b,p_f_f_p,f]]
9  [[b,p,f],[b,p_f_p,f]]

out of resolvents_2
A1:
1  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_f_p,f],[b,p_f_f_p,f]]
3  [[a,p_f_f_p,f],[b,p_f_p,f]]
4  [[a,p_f_p,f],[b,p_f_p,f]]

B1:

hyper_aux: B = []

T:
1  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_f_p,f],[b,p_f_f_p,f]]
3  [[a,p_f_f_p,f],[b,p_f_p,f]]
4  [[a,p_f_p,f],[b,p_f_p,f]]

M:
1  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
2  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
3  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
4  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
5  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
6  [[b,p_f_p,f],[a,p_f_f_p,f]]
7  [[b,p,f],[a,p_f_p,f]]
8  [[b,p_f_p,f],[b,p_f_f_p,f]]
9  [[b,p,f],[b,p_f_p,f]]
 M1:
1  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_f_p,f],[b,p_f_f_p,f]]
3  [[a,p_f_p,f],[b,p_f_p,f]]
4  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
5  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
6  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
7  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
8  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
9  [[b,p_f_p,f],[a,p_f_f_p,f]]
10  [[b,p,f],[a,p_f_p,f]]
11  [[b,p_f_p,f],[b,p_f_f_p,f]]
12  [[b,p,f],[b,p_f_p,f]]

out of resolvents_2
A1:

B1:
1  [[a,p_f_p,t],[a,p_or_f_f_p_f_p,f],[b,p_f_f_p,f]]
2  [[b,p_f_p,t],[b,p_f_f_p,f]]
3  [[b,p_f_p,t],[a,p_f_f_p,f]]
4  [[b,p,t],[b,p_f_f_p,f]]
5  [[b,p,t],[a,p_f_f_p,f]]
6  [[a,p_f_f_p,t],[a,p_or_f_f_p_f_p,f],[b,p_f_p,f]]
7  [[b,p,t],[b,p_f_p,f]]
8  [[a,p_f_f_p,t],[a,p_f_p,f]]
9  [[b,p_f_f_p,t],[a,p_f_p,f]]
10  [[b,p,t],[a,p_f_p,f]]

hyper_aux:
M:
```

```
1  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_f_p,f],[b,p_f_f_p,f]]
3  [[a,p_f_p,f],[b,p_f_p,f]]
4  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
5  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
6  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
7  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
8  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
9  [[b,p_f_p,f],[a,p_f_f_p,f]]
10  [[b,p,f],[a,p_f_p,f]]
11  [[b,p_f_p,f],[b,p_f_f_p,f]]
12  [[b,p,f],[b,p_f_p,f]]

B:
1  [[a,p_f_p,t],[a,p_or_f_f_p_f_p,f],[b,p_f_f_p,f]]
2  [[b,p_f_p,t],[b,p_f_f_p,f]]
3  [[b,p_f_p,t],[a,p_f_f_p,f]]
4  [[b,p,t],[b,p_f_f_p,f]]
5  [[b,p,t],[a,p_f_f_p,f]]
6  [[a,p_f_f_p,t],[a,p_or_f_f_p_f_p,f],[b,p_f_p,f]]
7  [[b,p,t],[b,p_f_p,f]]
8  [[a,p_f_f_p,t],[a,p_f_p,f]]
9  [[b,p_f_f_p,t],[a,p_f_p,f]]
10  [[b,p,t],[a,p_f_p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_f_f_p,f]]
2  [[b,p_f_f_p,f]]
3  [[b,p_f_f_p,f],[b,p,f]]
4  [[a,p_f_f_p,f],[b,p,f]]

B1:

hyper_aux: B = []

T:
1  [[a,p_f_f_p,f]]
2  [[b,p_f_f_p,f]]
3  [[b,p_f_f_p,f],[b,p,f]]
4  [[a,p_f_f_p,f],[b,p,f]]

M:
1  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_f_p,f],[b,p_f_f_p,f]]
3  [[a,p_f_p,f],[b,p_f_p,f]]
4  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
5  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
6  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
7  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
8  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
9  [[b,p_f_p,f],[a,p_f_f_p,f]]
10  [[b,p,f],[a,p_f_p,f]]
11  [[b,p_f_p,f],[b,p_f_f_p,f]]
12  [[b,p,f],[b,p_f_p,f]]
 M1:
1  [[a,p_f_f_p,f]]
2  [[b,p_f_f_p,f]]
3  [[b,p_f_f_p,f],[b,p,f]]
4  [[a,p_f_f_p,f],[b,p,f]]
5  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
6  [[a,p_f_p,f],[b,p_f_f_p,f]]
7  [[a,p_f_p,f],[b,p_f_p,f]]
8  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
```

```
9  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
10  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
11  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
12  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
13  [[b,p_f_p,f],[a,p_f_f_p,f]]
14  [[b,p,f],[a,p_f_p,f]]
15  [[b,p_f_p,f],[b,p_f_f_p,f]]
16  [[b,p,f],[b,p_f_p,f]]

out of resolvents_2
A1:

B1:
1  [[a,p_f_p,t],[a,p_or_f_f_p_f_p,f]]
2  [[b,p_f_p,t]]
3  [[a,p_f_p,t],[a,p_or_f_f_p_f_p,f],[b,p,f]]
4  [[b,p_f_p,t],[b,p,f]]

hyper_aux:
M:
1  [[a,p_f_f_p,f]]
2  [[b,p_f_f_p,f]]
3  [[b,p_f_f_p,f],[b,p,f]]
4  [[a,p_f_f_p,f],[b,p,f]]
5  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
6  [[a,p_f_p,f],[b,p_f_f_p,f]]
7  [[a,p_f_p,f],[b,p_f_p,f]]
8  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
9  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
10  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
11  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
12  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
13  [[b,p_f_p,f],[a,p_f_f_p,f]]
14  [[b,p,f],[a,p_f_p,f]]
15  [[b,p_f_p,f],[b,p_f_f_p,f]]
16  [[b,p,f],[b,p_f_p,f]]

B:
1  [[a,p_f_p,t],[a,p_or_f_f_p_f_p,f]]
2  [[b,p_f_p,t]]
3  [[a,p_f_p,t],[a,p_or_f_f_p_f_p,f],[b,p,f]]
4  [[b,p_f_p,t],[b,p,f]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  [[a,p_or_f_f_p_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_or_f_f_p_f_p,f],[b,p_f_p,f]]
3  [[a,p_f_p,f]]
4  [[b,p,f]]

B1:

hyper_aux: B = []

T:
1  [[a,p_or_f_f_p_f_p,f],[b,p_f_f_p,f]]
2  [[a,p_or_f_f_p_f_p,f],[b,p_f_p,f]]
3  [[a,p_f_p,f]]
4  [[b,p,f]]

M:
1  [[a,p_f_f_p,f]]
2  [[b,p_f_f_p,f]]
3  [[b,p_f_f_p,f],[b,p,f]]
```

```
4  [[a,p_f_f_p,f],[b,p,f]]
5  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
6  [[a,p_f_p,f],[b,p_f_f_p,f]]
7  [[a,p_f_p,f],[b,p_f_p,f]]
8  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
9  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
10  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
11  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
12  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
13  [[b,p_f_p,f],[a,p_f_f_p,f]]
14  [[b,p,f],[a,p_f_p,f]]
15  [[b,p_f_p,f],[b,p_f_f_p,f]]
16  [[b,p,f],[b,p_f_p,f]]
 M1:
1  [[a,p_f_p,f]]
2  [[b,p,f]]
3  [[a,p_f_f_p,f]]
4  [[b,p_f_f_p,f]]
5  [[b,p_f_f_p,f],[b,p,f]]
6  [[a,p_f_f_p,f],[b,p,f]]
7  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
8  [[a,p_f_p,f],[b,p_f_f_p,f]]
9  [[a,p_f_p,f],[b,p_f_p,f]]
10  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
11  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
12  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
13  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
14  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
15  [[b,p_f_p,f],[a,p_f_f_p,f]]
16  [[b,p,f],[a,p_f_p,f]]
17  [[b,p_f_p,f],[b,p_f_f_p,f]]
18  [[b,p,f],[b,p_f_p,f]]

out of resolvents_2
A1:
1  [[a,p_f_f_p,f],[b,p_f_p,f]]

B1:
1  [[a,p_f_f_p,t],[a,p_or_f_f_p_f_p,f]]
2  [[b,p,t]]
3  [[a,p_f_p,t]]
4  [[b,p_f_p,t]]

hyper_aux:
M:
1  [[a,p_f_p,f]]
2  [[b,p,f]]
3  [[a,p_f_f_p,f]]
4  [[b,p_f_f_p,f]]
5  [[b,p_f_f_p,f],[b,p,f]]
6  [[a,p_f_f_p,f],[b,p,f]]
7  [[a,p_f_f_p,f],[b,p_f_f_p,f]]
8  [[a,p_f_p,f],[b,p_f_f_p,f]]
9  [[a,p_f_p,f],[b,p_f_p,f]]
10  [[a,p_f_f_p,f],[b,p_or_f_f_p_f_p,f]]
11  [[a,p_f_p,f],[b,p_or_f_f_p_f_p,f]]
12  [[b,p_f_f_p,f],[a,p_or_f_f_p_f_p,f]]
13  [[b,p_f_p,f],[a,p_or_f_f_p_f_p,f]]
14  [[a,p_or_f_f_p_f_p,f],[b,p_or_f_f_p_f_p,f]]
15  [[b,p_f_p,f],[a,p_f_f_p,f]]
16  [[b,p,f],[a,p_f_p,f]]
17  [[b,p_f_p,f],[b,p_f_f_p,f]]
18  [[b,p,f],[b,p_f_p,f]]

B:
1  [[a,p_f_f_p,t],[a,p_or_f_f_p_f_p,f]]
```

```
2  [[b,p,t]]
3  [[a,p_f_p,t]]
4  [[b,p_f_p,t]]

resolvents_1 completed; hyper_aux follows

Read clauses starts:
A1:
1  []
2  [[a,p_or_f_f_p_f_p,f]]
3  [[a,p_or_f_f_p_f_p,f],[b,p,f]]
4  [[a,p_or_f_f_p_f_p,f],[b,p_f_f_p,f]]
5  [[b,p_f_p,f]]

B1:

Is a theorem: Contradiction is found
```

We present the execution times for several examples. All the experiments were carried out under SICStus Prolog running on a SUN workstation with 128MB RAM, equipped with a SPARC processor and a SPARC floating point processor with the following parameters: 86.1 MIPS, 10.6 MFLOPS, 44.2 SPECint92, 52.9 SPECfp92. The field "Refined" in the table containing the execution times for the resolution procedure refers to the fact that only one branch of the dual space $D(S_{n^2})$ was used, as illustrated in Example 5.11.

Translation to clause form

Formula	Logic	$\|A\|$	$\|D(A)\|$	Nr.Clauses	Time*
$S_1(p) \vee \neg S_1(p)$	$SH2$	4	2	15	50
	$SH3$	9	4	31	110
	$SH4$	16	6	49	170
	$SH5$	25	8	69	300
	$SH6$	36	10	91	470
$(p \Rightarrow q) \vee (q \Rightarrow p)$	$SH2$	4	2	19	70
	$SH3$	9	6	49	140
	$SH4$	16	6	99	270
$(r \Rightarrow (p \Rightarrow r))$	$SH2$	4	2	13	30
$((p \Rightarrow q) \Rightarrow r) \Rightarrow (r \Rightarrow (p \Rightarrow r))$	$SH3$	9	6	91	380
$f^2(p) \vee f(p)$	$\mathcal{L}_{21}$	4	2	15	40
$f^3(p) \vee p$	$\mathcal{L}_{30}$	8	3	28	60

Resolution

Formula	Logic	$\|A\|$	$\|D(A)\|$	General		Refined	
				Nr.Cl.	Time*	Nr.Cl.	Time*
$S_1(p) \vee \neg S_1(p)$	$SH2$	4	2	15	10480	8	330
	$SH3$	9	4	31	–	16	6369
$(p \Rightarrow q) \vee (q \Rightarrow p)$	$SH2$	4	2	19	–	10	449
	$SH3$	9	4	49	–	25	2079
$(r \Rightarrow (p \Rightarrow r))$	$SH2$	4	2	13	2510	7	240
$((p \Rightarrow q) \Rightarrow r) \Rightarrow (r \Rightarrow (p \Rightarrow r))$	$SH3$	9	4	91	–	46	4899
$f^2(p) \vee f(p)$	$\mathcal{L}_{21}$	4	2	15	4500	–	–

$*$ Time is expressed in milliseconds.

5.6 Comparison with Existing Approaches and Final Remarks

We showed that for some classes of logics (e.g. those satisfying properties (P1) – (P3)) both a procedure for translation to clause from using structure-preserving rules, and a method for proving non-satisfiability of a set of clauses by negative hyperresolution.

The idea of using literals of the form $\boxed{\alpha}P^t$ and $\boxed{\alpha}P^f$ occurred to us when investigating the properties of SHn-logics and when we noticed that they are sound and complete with respect to the finite SHn-frame $D(S_{n^2})$, consisting of all 0,1-lattice homomorphisms[9] from S_{n^2} to $\{0, 1\}$. The work towards finding a procedure for automated theorem proving for this class of logics was inspired by the papers of Baaz and Fermüller [BF95] and Hähnle [Häh94].

We want to point out that our approach is not as general as that presented in [BF95] or [Häh93]. The goal of our approach is to exploit the finer structure of the algebra of truth values of a given logic, and, when this structure allows it, to improve the efficiency of the algorithm (i.e. to reduce the number of clauses that are generated, or to avoid using the truth tables of the operations, and instead use the tables for the corresponding operations on the dual space – with smaller cardinality – and the corresponding transformation rules).

In [BF95] it is shown that, applying structure-preserving rules, the clause form of a formula with k occurrences of at most r-ary operators and m occurrences of quantifiers contains no more than

$$(S) \qquad k|W|^r + m2^{|W|} + 1$$

clauses, if optimal translation rules are used (where W is the set of truth values). Thus, for a formula with k occurrences of at most r-ary operators and no quantifiers, the clause form contains no more than $k|W|^r$ clauses.

For instance it can be seen (by directly checking) that for $\phi = S_1(p) \vee \neg S_1(p)$ the number of clauses generated with this very general method is $n^4 - 4n^3 + 8n^2 - 8n + 4$ if we consider S_{n^2} as a set of truth values. If we only consider the clauses generated by $S_1(p) \vee \neg S_1(p)^{\neq(1,1)}$ and those induced by the definitions of the subformulas $S_i(p)$ and $\neg S_i(p)$ we obtain $6n^2 - 4n + 1$ clauses. Thus, 17 clauses are generated for $n = 2$, 43 for $n = 3$, 81 for $n = 4$, 131 for $n = 5$, 193 for $n = 6$, etc.

The theoretical result of Baaz and Fermüller is fascinating (it establishes a general approach to resolution-based theorem proving for finite-valued quantificational first-order logics, and in the same time presents the resolution procedure in a new light). However, in practical applications, because of the exponential explosion of clauses in the resolution procedure, it is desirable to reduce as much as possible the number of clauses that occur during the process, either by improving the procedure for transformation to clause form, or by using refinements of the resolution procedure.

[9] Note that the idea of using "valuations" instead of values appears already in [Sco73], in the context of Lukasiewicz logics.

In [Häh94] it is shown that in every n-valued logic $\mathcal{L}$, for every signed formula $S : \phi$ there is a CNF-representation Φ of ϕ of length $\mathcal{O}(n^2|\phi|)$ such that $S : \phi$ is valid if and only if the empty clause can be deduced by signed resolution from Φ. This CNF-representation is also computed using structure-preserving translation rules. Moreover, a generalized concept of "polarity" for the defined subformulae is taken into consideration in order to reduce the number of clauses that are generated. It is also pointed out that for a certain class of finitely-valued logics (called regular logics) the signed resolution rule can be simplified: a generalized version of negative hyperresolution is presented that works very well for regular clauses.

Our method described in this section extends the version of negative hyperresolution developed by Hähnle for regular logics to the case when the set of truth values is not linearly ordered. It is efficient especially in the cases when the set of truth values is not linearly ordered, i.e. when the algebra of truth values has considerably more elements than its dual space.

For example, in the case of SHn-logics, the clause form of a formula with l subformulae (i.e. l occurrences of operators) has at most $\mathcal{O}(n^3 l)$, whereas the upper bound of the number of clauses computed in the general case is $\mathcal{O}((n^2)^2 l)$ (because the algebra of truth values in this case has n^2 elements).

The difference is even more considerable in the case of P_{mn}-logics: by our procedure the clause form of a formula with l subformulae has at most 1 + 3 m l clauses, whereas the upper bound of the number of clauses computed in the general case is $\mathcal{O}((2^m)^2 l)$ (because the algebra of truth values has 2^m elements in this case).

In the case of $SHKn$-logics, the method using "regular signs" developed in [Häh94] is essentially the same as the one described here, since $\boxed{\uparrow i}\ P^t = \boxed{\geq i}\ P$ and $\boxed{\uparrow i}\ P^f = \boxed{\leq (i-1)}\ P$.

Running our Prolog program on several tests we noticed that not all the clauses generated were actually used in the resolution process. Preliminary experimental results suggest that the number of generated clauses can be reduced using a concept very similar to the concept of "polarity" introduced by Hähnle. This will be subject for further work.

We would like to make some remarks about Lukasiewicz logics. The Lukasiewicz-Moisil algebras of order n were created by Moisil as an algebraic counterpart for the many-valued logics of Lukasiewicz. However, it turned out that n-valued Lukasiewicz-Moisil algebras are models for the n-valued logics of Lukasiewicz only for $n = 3$ and $n = 4$. Rose showed that for $n \geq 5$ the Lukasiewicz implication (defined by $x \rightarrow_L y = \begin{cases} 1 & \text{if } x \leq y \\ 1 - (x - y) & \text{if } x \geq y \end{cases}$) cannot be expressed in terms of the Lukasiewicz-Moisil algebra operations $\vee, \wedge, \sim, D_i$ on L_n. This can be seen by noticing that for every $n \geq 5$, $S_n = \{0, \frac{1}{n-1}, \frac{n-2}{n-1}, 1\}$ is a subalgebra with respect to the Lukasiewicz-Moisil algebra operations, but $\frac{n-2}{n-1} \rightarrow_L \frac{1}{n-1} = \frac{2}{n-1} \notin S_n$.

In [Cig82] Cignoli introduced so-called *proper Lukasiewicz algebras* of order n and showed that n-valued Lukasiewicz logics are sound and complete with respect to the class of proper Lukasiewicz algebras of order n.

Finding a Priestley-type representation for proper Lukasiewicz algebras of order n is (to our knowledge) still an open problem. Hence, our method cannot yet be applied for Lukasiewicz logics.

We would like to point out that analyzing our proofs above we noticed that the restrictive condition imposed on the logic $\mathcal{L}$ (i.e. that the logic $\mathcal{L}$ is sound and complete with respect to a variety $\mathcal{V}$ of algebras with an underlying distributive lattice structure, such that $\mathcal{V}$ is generated by one finite algebra A and the Priestley duality extends to a dual equivalence between $\mathcal{V}$ and a category $\mathcal{V}\mathsf{Sp}$ of Priestley spaces with operators) can be relaxed.

In the procedure for automated theorem proving described above we did not use all the duals of the algebras in the variety $\mathcal{V}$, but only the dual of the finite algebra A that generates $\mathcal{V}$.

As already said in Section 1.2 it seems that it suffices if $\mathcal{L}$ is sound and complete with respect to a finite Kripke-style frame (a finite set endowed e.g. with an order relation and with additional relations associated to the operations in the logic). We would like to investigate the degree of generality of this approach. In the thesis we decided to keep the initially imposed set of conditions on the logic $\mathcal{L}$ because the Priestley duality for the variety $\mathcal{V}$ furnished an intuitive description of the way such a finite Kripke frame can be constructed. Moreover, it turned out that the duality theorem offers a general framework for describing certain classes of Kripke models for these logics and a way of defining the validity relation $\overset{r}{\models}$ for these Kripke models starting from validity relation $\overset{a}{\models}$ on the algebraic models.

Chapter 6

Towards a Sheaf Semantics for Systems of Interacting Agents

In this chapter we give the main motivation for an approach to modeling interacting agents (robots) based on sheaf theory.

At the beginning, as a motivation for our theoretical study, we illustrate the problems that appear on a simple example, adapted from [Pfa93]. This example leads to a formal definition of a system. We then show how morphisms between systems can be defined in general; thus, we introduce a category SYS of systems. We show that the admissible states and the admissible parallel actions define functors from SYS^{op} to Sets (presheaves), and that the transitions between states defined by the admissible parallel actions define a natural transformation in PreSh(SYS), $Tr : Act \to \Omega^{St \times St}$ or, alternatively, that they define a subpresheaf Tr of $Act \times St \times St$. A natural question arises: is it possible to define a covering relation on SYS that induces a Grothendieck topology on SYS? In the next chapters we will answer this question. However, for the sake of simplicity we do not consider the general case. Instead, in Chapter 7 and Chapter 8 we restrict to the case when the morphisms are inclusions. A general theory that takes arbitrary morphisms between systems into account will be subject for future work.

6.1 A Motivating Example

We begin with a simple example (adapted from [Pfa93]) as a motivation for our theoretical study, for the definitions that will be given, and for the assumptions that will be made.

Let R_0, R_1, R_2, R_3 be four robots performing the following task:

- R_0 receives a work piece a and a work piece b and performs an assembly task. The work piece r obtained from assembling a and b is placed on the assembly bench.
- R_1 furnishes pieces of type a. He checks whether there are pieces of type a left in stock, and whether a piece of type a or an r resulting from assembling

a and b is placed on the assembly bench of R_0. If there are pieces of type a in stock, and if no a or r are placed on the table, R_1 brings a piece of type a to R_0.

- R_2 furnishes pieces of type b. He checks whether there are pieces of type b left in stock, and whether a b or an r is placed on the table. If there are pieces of type b in stock, and no b or r is on the table, R_2 brings a piece of type b to R_0.
- After R_0 has assembled a and b, R_3 receives the result r and transports it to the stock.

Let S be the system resulting from the interaction of these robots. We can assume that the system can be "described" by the interconnected subsystems S_0, S_1, S_2, and S_3, which correspond to the robots R_0, R_1, R_2 and R_3.

6.1.1 States

The states of the system S can be expressed using the *control variables* described in the table below. The set of control variables relevant for system S_0 is $X_0 = \{p_a, p_b, p_r\}$, the one for S_1 is $X_1 = \{s_a, p_a, p_r\}$, for S_2, $X_2 = \{s_b, p_b, p_r\}$, and for S_3, $X_3 = \{p_r\}$. We will assume that the subsystems $S_0, \ldots, S_3$ communicate via common control variables. In Figure 6.1 we show how the control variables are shared, and how they can be used for communication with "external" systems (e.g. Stock-a, Stock-b).

Variable	Description	System
s_a = in-stock-a	"there is at least one piece of type a in stock"	S_1
s_b = in-stock-b	"there is at least one piece of type b in stock"	S_2
p_a = on-table-a	"a piece of type a is on the assembly bench"	S_0,S_1
p_b = on-table-b	"a piece of type b is on the assembly bench"	S_0,S_2
p_r = on-table-res	"the result r is on the assembly bench"	S_0, S_1, S_2, S_3

A *state* of the system S is a possible assignment of truth values to the relevant control variables. We might additionally assume that only some of these assignments are admissible, imposing some *constraints* on the values that can be taken by the control variables. This turns out to be especially useful when the control variables are not independent.

Let us assume for example that in the given system it is not allowed to have a result piece and a piece of type a or b on the working bench at the same time, but it is allowed to have a piece of type a and one of type b. That means that p_a and p_r cannot both be true, and p_b and p_r cannot both be true. This can be expressed by a set of identities on the boolean algebra freely generated by the control variables of the system, in this case $Id = \{p_a \wedge p_r = 0, p_b \wedge p_r = 0\}$.

The agents R_0 and R_1 can "communicate" using the control variables common to these systems, namely the set $X_{01} = \{p_a, p_r\}$. Analogously, the agents R_0 and R_2 can "communicate" using the set $X_{02} = \{p_b, p_r\}$, and R_1 and R_2 using the set $X_{12} = \{p_r\}(= X_3)$. We can therefore assume that the structure of the given systems of cooperating agents as an interconnection of subsystems determines a topology on the set of control variables. Also the set of constraints (if they do not link variables in different subsystems) can be "distributed" over the subsystems in the same way.

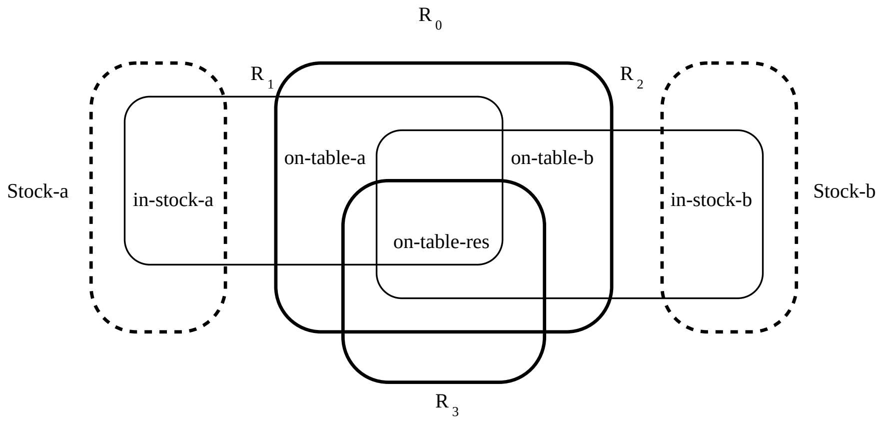


Figure 6.1: Control Variables

Consider the basis $\mathcal{B} = \{X_0, X_1, X_2, X_3, X_{01}, X_{02}\}$, consisting of the sets of control variables corresponding respectively to the subsystems consisting of the robots $R_0, \ldots, R_3$, as well as to their "subsystems" by means of which the communication is done. The corresponding restrictions of the set of constraints are $\{Id_0, Id_1, Id_2, Id_3, Id_{01}, Id_{02}\}$, where $Id_0 = Id$, $Id_1 = \{p_a \wedge p_r = 0\}$, $Id_2 = \{p_b \wedge p_r = 0\}$, and $Id_{01} = Id_{02} = Id_3 = \emptyset$, corresponding to the subsystems mentioned above.

The set of states of the system will be the set of those assignments of truth values to the control variables that satisfy this set of identities. Similarly, the set of states for system S_i (corresponding to the agent R_i) is $St(S_i) = \{s_i : X_i \to \{0,1\} \mid s_i \models Id_i\}$. Thus $St(S_0) = \{s_0 : X_0 \to \{0,1\} \mid s_0(p_a)s_0(p_r) = 0, s_0(p_b)s_0(p_r) = 0\}$, $St(S_1) = \{s_1 : X_1 \to \{0,1\} \mid s_1(p_a)s_1(p_r) = 0\}$, $St(S_2) = \{s_2 : X_2 \to \{0,1\} \mid s_2(p_b)s_2(p_r) = 0\}$, and $St(S_3) = \{s_3 \mid s_3 : X_3 \to \{0,1\}\}$.

It is easy to see that for every family $(s_i)_{i=0,\ldots,3}$ with the property that s_i is a state for the system S_i (corresponding to R_i), and such that for every i, j, s_i and s_j coincide on the common control variables, there is exactly one state of the system, s, such that the restriction of s to the control variables P_i is s_i for every i. This means that the following gluing condition is satisfied:

- For every $\{s_B\}_{B \in \mathcal{B}}$, where $s_B : B \to \{0,1\}$ satisfies the equations in $Id_B =$

$\{e \in Id \mid Var(e) \subseteq B\}$, such that for every $B_1, B_2 \in \mathcal{B}$, $s_{B_1|B_1 \cap B_2} = s_{B_2|B_1 \cap B_2}$, there exists a unique $s : P \to \{0,1\}$ that satisfies the set of constraints Id, such that for every $B \in \mathcal{B}$, $s_{|B} = s_B$.

Since there are typical properties of a sheaf visible, this leads to the idea that the link between local and global states could best be described by sheaves over a suitable topology on the set of control variables (or over a suitable Grothendieck topology on a category of systems) defined by the structure of the given system.

Remark: Note that the gluing property described above does not hold for every topology on the set X of control variables of the system. Consider for example the discrete topology on X. Then X can be covered by the family $\{\{s_a\}, \{s_b\}, \{p_a\}, \{p_b\}, \{p_r\}\}$. Then the following family of assignments of truth values to the control variables: $s_{\{s_a\}}(s_a) = 0$, $s_{\{s_b\}}(s_b) = 0$, $s_{\{p_a\}}(p_a) = 1$, $s_{\{p_b\}}(p_b) = 1$, $s_{\{p_r\}}(p_r) = 1$ agrees on common control variables (because the domains are disjoint), but no information about the constraints can be recovered, hence by "gluing" these mappings together one obtains a map $s : X \to \{0,1\}$ which does not satisfy the set Id of identities. This shows that an appropriate topology on X has to respect the way the constraints are shared between subsystems.

6.1.2 Actions

The system S is also characterized by a set of (atomic) actions. Below we will give the list of the atomic actions (with pre- and postconditions) and the agent that performs them.

Action	Description	Precond.	Postcond.	Agent/Interpr.
A	Assemble a piece of type a with one of type b	$p_a = 1$ $p_b = 1$ $p_r = 0$	$p_a = 0$ $p_b = 0$ $p_r = 1$	R_0: assemble
B_a	Bring a piece of type a	$p_a = 0$ $s_a = 1$ $p_r = 0$	$p_a = 1$ $s_a = 0$ $p_r = 0$	R_1: give-a R_0: receive-a
B_b	Bring a piece of type b	$p_b = 0$ $s_a = 1$ $p_r = 0$	$p_b = 1$ $s_a = 0$ $p_r = 0$	R_2: give-b R_0: receive-b
T_r	Store the result	$p_r = 1$	$p_r = 0$	R_3: receive-r R_0: give-r

We can assume for example that R_1 and R_2 can perform the actions of bringing a piece of type a respectively b in parallel but that R_0 is not allowed to execute in parallel the action of taking a piece of type a and of giving the result to R_3. We also can assume that other actions, as for example **give-a** (by R_1) and **receive-a** (by R_0) have to be executed in the same time. Therefore, they have been "identified" in the larger subsystem under the name B_a. Figure 6.2 shows how actions are shared between subsystems, and some relations between

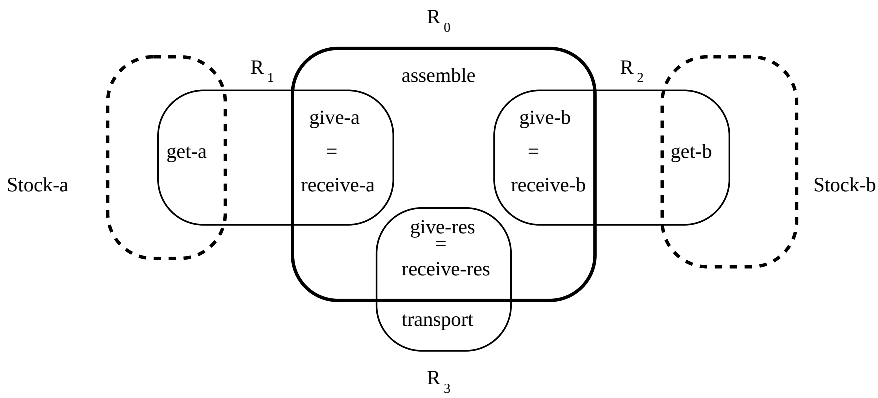


Figure 6.2: Actions

them (for the sake of simplicity we do not consider the actions **get-a**, **get-b** and **transport** in what follows). It is also natural to suppose that some *constraints* are given, expressing which of these actions can be performed in parallel, which cannot, which must be executed in the same time, and so on.

There are many possibilities to specify this kind of constraints. One solution is to consider parallel actions as subsets of the set of atomic actions A, or equivalently by maps $f : A \to \{0, 1\}$, where $f(a) = 1$ means that a is executed and $f(a) = 0$ means that a is not executed. In this case the constraints can be described by imposing restrictions on the combinations of the values (0 or 1) that can be assigned to the atomic actions in an admissible parallel action. This approach is very similar to the one adopted in the description of states (section 6.1.1). We will assume that the constraints can be described by identities on the boolean algebra freely generated by the set of atomic actions A. (On our example, among the constraints are: $\mathsf{give\text{-}a} = \mathsf{receive\text{-}a}$, $\mathsf{give\text{-}b} = \mathsf{receive\text{-}b}$, $\mathsf{give\text{-}res} = \mathsf{receive\text{-}res}$, $\mathsf{give\text{-}res} \wedge \mathsf{receive\text{-}a} = 0$, $\mathsf{give\text{-}res} \wedge \mathsf{receive\text{-}b} = 0$, etc.).

There exist approaches for modeling concurrency where it is necessary to specify which actions can be performed in parallel and which not. One such approach is based on considering a "dependence" relation on the set of (atomic) actions, i.e. a reflexive and symmetric relation $D \subseteq A \times A$: the parallel execution is then only allowed for those actions which are "independent" w.r.t. D. This leads to the study of partial-commutative monoids (cf. [Die90]). The link with this approach is analyzed in Chapter 8.

As when considering states, the structure of the system as an interconnection of subsystems induces a topology on the set of all actions. Correspondingly, the constraints distribute over the subsystems (it seems natural to make the assumption that all the constraints are made "inside" some specified subsystem). A basis $\mathcal{B}$ for this topology is obtained as explained in the study of states, taking the family of all actions corresponding to the subsystems $S_0, \ldots, S_3$, as well as to the "subsystems" by means of which the communication is done (i.e. finite intersections of those sets). In the case where there are constraints that link actions in different subsystems (that could for example be expressed in some special "scheduling systems"), the actions that correspond to these scheduling systems will also be considered elements in the basis. Also in this case a similar gluing property holds:

- For every family of parallel actions $\{f_B\}_{B\in\mathcal{B}}$, where $f_B : A_B \to \{0,1\}$ satisfies the constraints $Id_B = \{e \in Id \mid Var(e) \subseteq B\}$, such that for every $B_1, B_2 \in \mathcal{B}$, $f_{B_1|B_1\cap B_2} = f_{B_2|B_1\cap B_2}$, there exists a unique $f : A \to \{0,1\}$ that satisfies Id such that for every $B \in \mathcal{B}$, $f_{|B} = f_B$.

Note that the fact that the gluing property holds is strongly related to the specific form of the constraints in C_A (boolean equations in our case). If for example a parallel action $f : A \to \{0,1\}$ is allowed if and only if $f^{-1}(1)$ is finite, then the infinite gluing property does not hold.

6.2 Systems

Taking into account the considerations in section 6.1, we will assume that a system S is described by:

- A set X of control variables of the system (where for every control variable $x \in X$, a set V_x of possible values for x is specified), and a set Γ of constraints, specifying which combinations of values for the variables are admissible (i.e. satisfy Γ). An admissible combination of values of the control variables will describe a state of the system S. The set of states of the system will be denoted $St(S)$,

- A set A of atomic actions (where for every $a \in A$, X_a denotes the minimal set of control variables a depends on and $Tr_a \subseteq St(S)_{|X_a} \times St(S)_{|X_a}$ a relation indicating how the values of these variables change when the action a is performed), together with a set C of constraints that shows which actions are incompatible and cannot be performed in parallel, or which actions have to be performed at the same time.

We first say some words about a possible way of modeling interaction (communication) between systems. Consider for instance two agents (robots or production units). Every agent has its own set of control variables (parameters) and its own set of actions, together with a specification of their pre- and postconditions.

In order for the two systems to be able to *communicate* they need a *"dictionary"*, at least for some of these notions. The situation is pictorially represented in Figure 6.3.

In what follows we assume that we can identify those elements that are shown equal by the dictionary, hence we can assume that communicating agents have *common control variables* and *common actions*. We further assume that the sensors of the two agents are compatible, in the sense that the values of the common control variables "sensed" at a given moment of time by two such agents are the same.

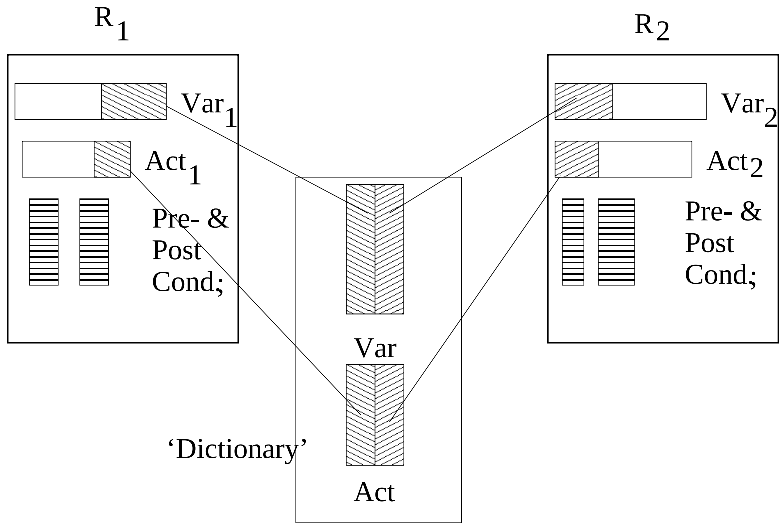


Figure 6.3: Two agents that can communicate.

Therefore, agents that communicate can be seen as part of the system obtained by their interconnection; they can be represented by putting into evidence their common part, identified by the common "dictionary" (cf. Figure 6.3) as illustrated in Figure 6.4.

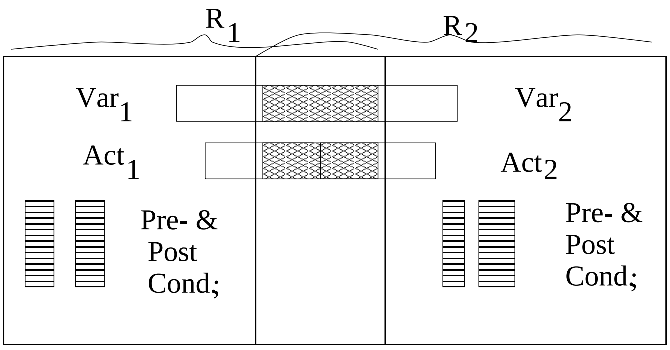


Figure 6.4: The system obtained by interconnecting two agents that can communicate.

We begin with a formal definition of a system. The relationships between systems that make possible communication will be discussed in Section 6.3.

In order to formally express the constraints on the possible combinations of values for the control variables, we need a language in which these are expressed.

Definition 6.1 *A* system $S = (\Sigma, X, \Gamma, M, A, C)$ *consists of:*

(1) A language $\mathcal{L}_S$ consisting of
(1a) a signature $\Sigma = (\mathsf{Sort}, O, P)$ (Sort is the set of sorts, O the set of operation symbols, and P the set of predicate symbols),
(1b) a (many-sorted) set of variables $X = \{X_s\}_{s \in \mathsf{Sort}}$,

(2) A set $\Gamma \subseteq \mathsf{Fma}_\Sigma(X)$ closed with respect to the semantical consequence relation[1] *$\models_M$ (the set of constraints of S),*

(3) A model M (structure of similarity type Σ),

(4) A set of actions A; for every $a \in A$ a set $X_a \subseteq X$ of variables on which the action a depends, and a transition relation Tr_a,

(5) A set C of constraints, expressed by boolean equations over $F_B(A)$ (the free boolean algebra generated by A) stating which actions can (or have to) be executed in parallel, and which cannot. (We impose that C contains all the possible boolean equations that can be deduced by C.)

Remarks:

(1) In what follows we will consider, for the sake of simplicity, only *finite* systems, i.e. systems in which the signature, the set of control variables and the set of actions are finite.

(2) In Definition 6.1 we fix a model M for the system S (corresponding e.g. to the real world) and as constraints we allow formulae in the many-sorted language of the system.

The formulae in $\mathsf{Fma}_\Sigma(X)$ are formed as explained in Section 3.3, Definition 3.54. The semantical consequence relation $\models_M$ is defined by $\Gamma \models_M \phi$ if and only if for every $s : X \to M$ with the property that if for every $\gamma \in \Gamma$, $s \models \gamma$ then $s \models \phi$ (where the relation $s \models \phi$ is defined as explained in Section 3.3, Definition 3.56). The closure of a set of Γ of formulae under the consequence relation $\models_M$ will be denoted by $\Gamma^\bullet$.

Concerning the variables, we point out that in this approach they are considered rather generators than simple variables as in first-order logic. Formulae of the type $\phi = \forall x \psi$ are allowed; such a formula is satisfied at a given state of s (assignment of values in M to the variables in X) if for every other state s' that agrees with s except possibly at x ϕ holds at s'. Thus, in what follows, $\forall x$ has the meaning "for every possible value of x in M".

(3) Note also that we assume that the constraints on actions are expressed by boolean equations over the free boolean algebra $F_B(A)$. In what follows we will assume that the constraints are of the form $a_1 = a_2$, with $a_1, a_2 \in A$ (expressing the fact that a_1 and a_2 have to be performed at the same time) or $a_1 \wedge a_2 = 0$ (expressing the fact that a_1 and a_2 are not allowed to be performed in parallel).

Given a system S, if not otherwise specified, we will refer to its signature, set of variables, constraints on the values of the variables, model, set of actions

[1] The relation $\models_M$ is defined by $\Gamma \models_M \phi$ if and only if for every assignment of values in M to the variables in X, $s : X \to M$, if $s \models \gamma$ for every $\gamma \in \Gamma$, then $s \models \phi$ (see also Remark (2))

and constraints on actions by Σ_S, X_S, Γ_S, M_S, A_S, C_S respectively. For every atomic action $a \in A_S$ the minimal set of variables a depends on will be denoted by X_a^S, and the transition relation associated with a will be denoted by Tr_a^S.

If a family of systems $\{S_i\}_{i \in I}$ is given, if not otherwise specified we will refer to the signature, set of variables, constraints on the values of the variables, model, set of actions and constraints on actions of S_i by Σ_i, X_i, Γ_i, M_i, A_i, C_i, for every $i \in I$. For every atomic action $a \in A_i$ the minimal set of variables a depends on will be denoted by X_a^i, and the transition relation associated with a will be denoted by $Tr_a^{S_i}$.

We will denote by $\mathsf{Term}(S)$ the algebra of terms of the system S, and by $\mathsf{Fma}(S)$ the formulae of S.

Definition 6.2 (States) *The* states *of a system* $S = (\Sigma, X, \Gamma, M, A, C)$ *are those interpretations* $s : X \to M$ *that satisfy all the formulae in* Γ.

The set of states of a system S is $St(S) = \{s : X \to M \mid s \models \Gamma\}$.

Definition 6.3 (Admissible Parallel Actions) *The set of* admissible actions *of the system* $S = (\Sigma, X, \Gamma, M, A, C)$ *will be the set* $Act(S) = \{f : A \to \{0, 1\} \mid$ *f satisfies* $C\}$.

As pointed out before, in what follows we will assume that all the constraints imposed on the actions can be expressed by boolean equations (in the boolean algebra freely generated by A) of the form $a_1 = a_2$ with $a_1, a_2 \in A$ or $a_1 \wedge a_2 = 0$.

An equation of the form $a_1 = a_2$ expresses the fact that a_1 and a_2 have to be performed in parallel. We assume that for every a_1, a_2 with $a_1 = a_2 \in C$ (or deducible from C) we have $X_{a_1} = X_{a_2}$ and $Tr_{a_1} = Tr_{a_2}$.

The set of admissible actions of the system $S = (\Sigma, X, \Gamma, M, A, C)$ will be the set $Act(S) = \{f : A \to \{0, 1\} \mid f \text{ satisfies } C\}$.

Let $f \in Act(S)$ be a parallel action. The set of variables on which f depends is $X_f = \bigcup_{a, f(a)=1} X_a$.

Without loss of generality we may identify the actions a_1 and a_2 if $a_1 = a_2$ is deducible from C (i.e. we consider that they are one and the same action).

The compatibility of the actions in an admissible parallel action can be expressed, depending on the situation, by one of the following two properties of the transitions (which are not equivalent):

(Gluing) If $f \in Act(S)$ and if $s \in St(S)$ such that for every $a \in A$ with $f(a) = 1$ there is a $s'_a \in St(S)_{|X_a}$ such that $(s_{|X_a}, s'_a) \in Tr_a$, then the new local states "agree on intersections", i.e. for every $x \in X_{a_1} \cap X_{a_2}$, $s'_{a_1}(x) = s'_{a_2}(x)$. Then we can associate a transition relation Tr_f to f, that shows how the state of the system changes after the action is performed, namely: $Tr_f^S \subseteq St(S)_{|X_f} \times St(S)_{|X_f}$, $Tr_f^S = \{(s_1, s_2) \mid (s_{1|X_a}, s_{2|X_a}) \in Tr_a$ for every a such that $f(a) = 1\}$.

The transition on the states of S induced by f is $Tr^S(f) = \{(s_1, s_2) \mid s_1, s_2 \in St(S), (s_{1|X_a}, s_{2|X_a}) \in Tr_a$ for every a such that $f(a) = 1$, and $s_1(x) = s_2(x)$ if $x \notin \bigcup_{a, f(a)=1} X_a\}$.

(Independence) Let $f \in Act(S)$ and $s_0 \in St(S)$. Let us identify all the elements $a_1, a_2 \in A$ with $a_1 = a_2 \in C$ and $f(a_1) = f(a_2) = 1$. After this identifications, let $f^{-1}(1) = \{a_1, \dots, a_n\}$. Assume that for any subset $\{a'_1, \dots, a'_m\}$ of $f^{-1}(1) = \{a_1, \dots, a_n\}$, if we have $s_0 \xrightarrow{a'_1} s_1 \xrightarrow{a'_2} s_2 \dots \xrightarrow{a'_m} s_m$ then for every permutation σ, we have $s_0 \xrightarrow{a'_{\sigma(1)}} s'_1 \xrightarrow{a'_{\sigma(2)}} s'_2 \dots \xrightarrow{a'_{\sigma(m)}} s_m$ (the final state is the same). Then we can associate a transition relation to f, that shows how the state of the system changes after the action is performed, namely: $Tr^S(f) \subseteq St(S) \times St(S)$, defined by $Tr^S(f) = \{(s_0, s_n) \mid$ there exist states $s_1, \dots, s_{n-1}$ s.t. $(s_{i-1|X_{a_i}}, s_{i|X_{a_i}}) \in Tr_{a_i}, \forall 1 \leq i \leq n\}$.

It is easy to see that if $(s_0, s_n) \in Tr^S(f)$ then for every $x \in X_S \backslash X_f$ $s_0(x) = s_n(x)$. Thus Tr^S_f can be defined by $Tr^S_f = \{(s_{1|X_f}, s_{2X_f}) \mid (s_1, s_2) \in Tr^S(f)\}$.

The property **(Gluing)** makes sense in situations when a parallel action $f : A \to \{0, 1\}$ is admissible if and only if the actions do not "consume" common resources. This happens for example if for every $a_1, a_2 \in A$ with $f(a_1) = f(a_2) = 1$ we either have $a_1 = a_2 \in C$ (i.e. a_1 and a_2 are to be executed at the same time) or X_{a_1} and X_{a_2} are disjoint. In this case, obviously, a parallel action $f : A \to \{0, 1\}$ can be applied at a state s if and only if its components can be applied locally, in the respective systems, at the corresponding restricted state. We might enforce this when defining a system by imposing $a_1 \wedge a_2 = 0$ whenever a_1 and a_2 do not *have* to be performed in parallel, and $X_{a_1} \cap X_{a_2} \neq \emptyset$.

The property **(Independence)** represents the way transitions of parallel actions are interpreted when the actions to be performed in parallel actually consume common resources. Moreover, it is specific to the situation when, after executing an action, the state reached is uniquely determined (i.e. in the case of deterministic actions). In this case, the fact that all the components of a given parallel action $f : A \to \{0, 1\}$ can be applied at a given state s_0 is a necessary condition for the action f to be applicable at state s_0, but in general it is not sufficient: additionally, one has to be sure that there are enough resources, such that all the actions can be performed.

Let S be a system and let $f \in Act(S)$. In what follows we will denote by $Tr(f)$ the set of transitions between states induced by f, i.e.

$$Tr(f) = \{(s_1, s_2) \mid s_1, s_2 \in St(S), (s_{1|X_f}, s_{2|X_f}) \in Tr_f, \text{ and } s_1(x) = s_2(x) \forall x \notin X_f\}.$$

If a is a single action we denote by $Tr(a)$ the set of transitions between states induced by a, i.e.

$$Tr(a) = \{(s_1, s_2) \mid s_1, s_2 \in St(S), (s_{1|X_a}, s_{2|X_a}) \in Tr_a, \text{ and } s_1(x) = s_2(x) \, \forall x \notin X_a\}.$$

6.3 The Category of Systems SYS

Systems often arise in relationship with other systems. The corresponding relationships between systems are expressed by morphisms. Obviously, there is some choice in how to define appropriate morphisms, depending on the extent of the relationship between systems we want to express.

For instance, a *category of systems* SYS can be defined, with systems as its objects. Intuitively, a morphism f in SYS from a system S_1 to a system S_2 consists of a "translation" of the language (resp. actions) of S_1 into the the language (resp. actions) of S_2 (i.e. a family of mappings $f_\Sigma : \Sigma_1 \to \Sigma_2$, $f_X : X_1 \to X_2$, $f_A : A_1 \to A_2$) that maps the constraints of S_1 to constraints of S_2.

Definition 6.4 *There is a morphism $f : S_1 \to S_2$ if and only if*

(M1) There are maps $f_X : X_1 \to X_2, f_A : A_1 \to A_2$, and a morphism of signatures $f_\Sigma : \Sigma_1 \to \Sigma_2$, i.e. a triple (f_S, f_O, f_P), where $f_S : \mathsf{Sort}_1 \to \mathsf{Sort}_2$, f_O is a $\mathsf{Sort}_1^* \times \mathsf{Sort}_1$-indexed family of maps on operation symbols, $f_O^{s_1 \dots s_n, s} : O_{s_1 \dots s_n, s} \to O'_{f_S(s_1) \dots f_S(s_n), f_S(s)}$, and f_P is a Sort^*-indexed family of maps on predicate symbols, $f_P^{s_1 \dots s_n} : P_{s_1 \dots s_n} \to P'_{f_S(s_1) \dots f_S(s_n)}$,

(M2) The model M_1 of S_1 is the restriction of M_2 to the signature Σ_1 via f_Σ (i.e. $M_1 = \mathsf{Str}(f_\Sigma)(M_2)$),

(M3) $f^\natural_{\mathsf{Fma}}(\Gamma_1) \subseteq \Gamma_2$,

(M4) $f^\natural_A(C_1) \subseteq C_2$,

where Str *is the functor from the category Str_{Σ_2} of Σ_2-structures to the category Str_{Σ_1} of Σ_1-structures induced by the morphism of signatures f_Σ (cf. Proposition 3.20), $f^\natural_{\mathsf{Fma}}$ is the unique morphism of Σ-structures from $\mathsf{Fma}(S_1)$ to $\mathsf{Fma}(S_2)$ induced by f_Σ and f_X, and $f^\natural_A$ is the unique extension to a morphism of Boolean algebras of f_A.*

Remark: Conditions $(M3)$ and $(M4)$ express the fact that for every $\phi \in \Gamma_1$ (resp. in C_1) the "translation" of ϕ to the language of S_2 via f_Σ, f_X (resp. f_A) is in Γ_2 (resp. in C_2).

Definition 6.5 (Local Morphism) *A morphism $f = (f_\Sigma, f_X, f_A) : S_1 \to S_2$ is a* local morphism *if it additionally satisfies*

(M5) If an action $a \in A_2$ depends on some variables in $f_X(X_1)$, then $a \in f_A(A_1)$, and $\bigcup_{f_A(b)=a} X^1_b = f_X^{-1}(X^2_a)$.

(M6) For every $a \in f_A(A_1)$ such that $f^{-1}(a) = \{b_1, \dots, b_n\}$, let $g : A_1 \to \{0, 1\}$ be defined by $g(a_1) = 1$ iff $f_A(a_1) = a$.

With this notation, for every $s_1, s_2 \in St(S_2)$ such that $(s_{1|X^2_a}, s_{2|X^2_a}) \in Tr^{S_2}_a$, we have $(s_1 \circ f_{X|X_g}, s_2 \circ f_{X|X_g}) \in Tr^{S_1}_g$.

It will be shown later that conditions $(M5)$ and $(M6)$ ensure that every transition in S_2 can be restricted to a valid transition in S_1.

Let $\mathsf{SYS_{lm}}$ be the category with systems as objects and local morphisms as arrows.

Lemma 6.1 *Let $f = (f_\Sigma, f_X, f_A) : S_1 \to S_2$ be a morphism, where $f_\Sigma : \Sigma_1 \to \Sigma_2$, $f_X : X_1 \to X_2$ and $f_A : A_1 \to A_2$. Then there is a map $St(f) : St(S_2) \to St(S_1)$.*

Proof: For every state $s : X_2 \to M_2$ of S_2, consider $s \circ f_X : X_1 \to M_2$. Note that for every $x \in (X_1)_s$, $f_X(x) \in (X_2)_{f_S(s)}$, hence $s(f_X(x)) \in (M_2)_{f_S(s)} = (M_{2|f_\Sigma})_s = (M_1)_s$. Therefore, $s \circ f_X : X_1 \to M_1$. We will show that if $s \models \Gamma_2$ then $s \circ f_X \models \Gamma_1$. Note first that by the considerations in Section 3.3, we have:

- There exists a unique morphism of O_1-algebras, $(f_{O,X})^\natural_{\mathsf{Term}} : T_{O_1}(X_1) \to T_{O_2}(X_2)_{|f_\Sigma}$ that extends $f_X : X_1 \to X_2$, $f_S : \mathsf{Sort}_1 \to \mathsf{Sort}_2$, and $f_O : O_1 \to O_2$.
- There exists a unique morphism of O_2-algebras, $s^\natural_{\mathsf{Term}} : T_{O_2}(X_2) \to M_2$, that extends s. This morphism induces a unique morphism of O_1-algebras, $s^\natural_{\mathsf{Term}|f_\Sigma} : T_{O_2}(X_2)_{|f_\Sigma} \to M_1$.
- There exists a unique morphism of O_1-algebras, $(s \circ f_X)^\natural_{\mathsf{Term}} : T_{O_1}(X_1) \to M_1$ that extends $s \circ f_X$.

It can be checked that $(s \circ f_X)^\natural_{\mathsf{Term}}$ and $s^\natural_{\mathsf{Term}|f_\Sigma} \circ (f_X)^\natural_{\mathsf{Term}}$ coincide on X_1, hence $(s \circ f_X)^\natural_{\mathsf{Term}} = s^\natural_{\mathsf{Term}|f_\Sigma} \circ (f_X)^\natural_{\mathsf{Term}}$. Similarly, by the universality property of the algebra of formulae it follows that $(s \circ f_X)^\natural_{\mathsf{Fma}} = s^\natural_{\mathsf{Fma}|f_\Sigma} \circ (f_\Sigma)^\natural_{\mathsf{Fma}}$. Let $\phi \in \Gamma_1$. Then $(s \circ f_\Sigma)^\natural_{\mathsf{Fma}}(\phi) = s^\natural_{\mathsf{Fma}|f_\Sigma}((f_\Sigma)^\natural_{\mathsf{Fma}}(\phi)) = s^\natural_{\mathsf{Fma}}((f_\Sigma)^\natural_{\mathsf{Fma}}(\phi))$. By $(M3)$, $(f_\Sigma)^\natural_{\mathsf{Fma}}(\phi) \in \Gamma_2$. Therefore, since $s \models \Gamma_2$, $s^\natural_{\mathsf{Fma}}((f_\Sigma)^\natural_{\mathsf{Fma}}(\phi)) = 1$, hence $s \circ f_X \models \phi$. $\square$

Lemma 6.2 *Let $f = (f_\Sigma, f_X, f_A) : S_1 \to S_2$ be a morphism, where $f_\Sigma : \Sigma_1 \to \Sigma_2$, $f_X : X_1 \to X_2$ and $f_A : A_1 \to A_2$. Then there is a map $Act(f) : Act(S_2) \to Act(S_1)$.*

Proof: Let $h \in Act(S_2)$ then $h : A_2 \to \{0,1\}$ and $h \models C_2$. Let $Act(f)(h) = h \circ f_A : A_1 \to \{0,1\}$. As in the case of states it is easy to show that:

- There is a unique morphism of Boolean algebras $(h \circ f_A)^\natural : F_B(A_1) \to \{0,1\}$ that extends $h \circ f_A$ (from the universality property of $F_B(A_1)$).
- There is a unique morphism of Boolean algebras $h^\natural : F_B(A_2) \to \{0,1\}$ that extends h (from the universality property of $F_B(A_2)$).
- There is a unique morphism of Boolean algebras ${f_A}^\natural : F_B(A_1) \to F_B(A_2)$ that extends f_A.

It is easy to see that $h^\natural \circ f_A{}^\natural$ and $(h \circ f_A)^\natural$ coincide on A_1, hence they are equal. Let $(e_1, e_1') \in C_1$. By $(M4)$, $(f_A^\natural(e_1), f_A^\natural(e_1')) \in C_2$. Since $h \models C_2$, it follows that $(h \circ f_A)^\natural(e_1) = h^\natural \circ f_A{}^\natural(e_1) = h^\natural(f_A{}^\natural(e_1)) = h^\natural(f_A{}^\natural(e_1')) = h^\natural \circ f_A{}^\natural(e_1') = (h \circ f_A)^\natural(e_1')$. Thus we proved that $(h \circ f_A) \models C_1$. □

Proposition 6.3 *$St : \mathsf{SYS}^{op} \to \mathsf{Sets}$ and $Act : \mathsf{SYS}^{op} \to \mathsf{Sets}$ are presheaves.*

Proof: Follows from Lemma 6.1 and Lemma 6.2. □

It follows that $St : \mathsf{SYS}_{\mathsf{lm}}{}^{op} \to \mathsf{Sets}$ and $Act : \mathsf{SYS}_{\mathsf{lm}}{}^{op} \to \mathsf{Sets}$ are also presheaves.

We now analyse the transitions induced by the admissible parallel actions.

Lemma 6.4 *Let $f : S_1 \to S_2$ be a local morphism. Let $a \in f_A(A_1)$ be such that $f^{-1}(a) = \{b_1, \dots, b_n\}$, and let $g : A_1 \to \{0,1\}$ be defined by $g(a_1) = 1$ iff $f_A(a_1) = a$.*

For every $s_1, s_2 \in St(S_2)$ with $(s_1, s_2) \in Tr^{S_2}(a)$, $(s_1 \circ f_X, s_2 \circ f_X) \in Tr^{S_1}(g)$.

Proof: Let $a \in f_A(A_1)$, and $s_1, s_2 \in St(S_2)$ with $(s_1, s_2) \in Tr^{S_2}(a)$. We want to show that $(s_1 \circ f_X, s_2 \circ f_X) \in Tr^{S_1}(g)$, i.e. $(s_1 \circ f_{X|X^1_g}, s_2 \circ f_{X|X^1_g}) \in Tr^{S_1}_g$ and for every $x \notin X^1_g$, $s_1 \circ f_X(x) = s_2 \circ f_X(x)$.

The fact that $(s_1 \circ f_{X|X^1_g}, s_2 \circ f_{X|X^1_g}) \in Tr^{S_1}_g$ follows by $(M6)$. Assume now that $s_1 \circ f_X(x) \neq s_2 \circ f_X(x)$ for some $x \notin X^1_g$. Then $f_X(x) \in X^2_a$, hence $x \in f_X^{-1}(X^2_a)$. But by $(M5)$, $f_X^{-1}(X^2_a) = \bigcup_{b, f(b)=a} X^1_b = X^1_g$. Thus, it follows that $x \in X^1_g$, contradiction. Hence, for every $x \notin X^1_g$, $s_1 \circ f_X(x) = s_2 \circ f_X(x)$. □

Proposition 6.5 *Assume that the transitions of parallel actions are composed using the property* **(Gluing)**. *Let $f : S_1 \to S_2$ be a local morphism in $\mathsf{SYS}_{\mathsf{lm}}$. Let $g \in Act(S_2)$, and let $(s_1, s_2) \in Tr^{S_2}_g$. Then $(s_1 \circ f_X, s_2 \circ f_X) \in Tr^{S_1}_{g \circ f_A}$.*

Proof: Let $f : S_1 \to S_2$ be a local morphism, $g \in Act(S_2)$ and $(s_1, s_2) \in Tr^{S_2}_g$. We want to show that $(s_1 \circ f_X, s_2 \circ f_X) \in Tr^{S_1}_{g \circ f_A}$, i.e. that

(a) $(s_1 \circ f_{X|X^1_b}, s_2 \circ f_{X|X^1_b}) \in Tr^{S_1}_b$ for every b such that $g(f_A(b)) = 1$,

(b) $s_1 \circ f_X(x) = s_2 \circ f_X(x)$ for every $x \in X_1 \backslash \bigcup_{b, g(f_A(b))=1} X^1_b$.

In order to prove (a), note that if $g(f_A(b)) = 1$, then $(s_{1|X^2_{f_A(b)}}, s_{1|X^2_{f_A(b)}}) \in Tr^{S_2}_{f_A(b)}$. Then, by $(M6)$ it follows that $(s_1 \circ f_{X|X^1_g}, s_1 \circ f_{X|X^1_g}) \in Tr^{S_1}_g$, where $g : A_1 \to \{0,1\}$ is defined by $g(c) = 1$ iff $f_A(c) = f_A(b)$. By the property **(Gluing)** it follows that for every c with $f_A(c) = f_A(b)$ we have $(s_1 \circ f_{X|X^1_c}, s_1 \circ f_{X|X^1_c}) \in Tr^{S_1}_c$. In particular, we have $(s_1 \circ f_{X|X^1_b}, s_1 \circ f_{X|X^1_b}) \in Tr^{S_1}_b$.

In order to prove (b), let x be such that $s_1 \circ f_X(x) \neq s_2 \circ f_X(x)$. Then $s_1(f_X(x)) \neq s_2(f_X(x))$, hence $f_X(x) \in X^2_a$ for some $a \in A_2$ with $g(a) = 1$. Therefore, $x \in f_X^{-1}(X^2_a) = \bigcup_{b, f_A(b)=a} X^1_b$. Note that for every b with

$f_A(b) = a$ we have also $g(f_A(b)) = 1$. Hence, $x \in f_X^{-1}(X_a^2) = \bigcup_{b, f_A(b)=a} X_b^1 \subseteq \bigcup_{b, g(f_A(b))=1} X_b^1$. This proves that for every $x \in X_1 \setminus \bigcup_{b, g(f_A(b))=1} X_b^1$ we have $s_1 \circ f_X(x) = s_2 \circ f_X(x)$. $\square$

Proposition 6.6 *Assume that the transitions of parallel actions are composed using the property* **(Independence)***. Let $f : S_1 \to S_2$ be a local morphism in $\mathsf{SYS}_{\mathsf{lm}}$. Let $g \in Act(S_2)$, and let $(s_1, s_2) \in Tr_g^{S_2}$. Then $(s_1 \circ f_X, s_2 \circ f_X) \in Tr_{g \circ f_A}^{S_1}$.*

Proof: Let $(s_1, s_2) \in Tr_g^{S_2}$. Assume that after identifying the elements $a_1, a_2 \in A_2$ with $a_1 = a_2 \in C$ and $g(a_1) = g(a_2) = 1$, we have $g^{-1}(1) = \{a_1, \dots, a_n\}$. By the property **(Independence)** we know that the final state in the sequence $s_1 = s'_0 \xrightarrow{a_1} s'_1 \xrightarrow{a_2} s'_2 \xrightarrow{a_3} \dots \xrightarrow{a_n} s'_n = s_2$ does not depend on the order in which the actions are executed.

For every $i \in \{1, \dots, n\}$, let g_i be the parallel action composed of those $b \in A_1$ with $g(b) = a_i$. Then by Lemma 6.4, $(s_{i-1} \circ f_X, s_i \circ f_X) \in Tr^{S_1}(g_i)$. Thus, $s_1 \circ f_X = s'_0 \circ f_X \xrightarrow{g_1} s'_1 \circ f_X \xrightarrow{g_2} s'_2 \circ f_X \xrightarrow{g_3} \dots \xrightarrow{g_n} s'_n \circ f_X = s_2 \circ f_X$. We know that the final state of a sequence containing all elements b in A_1 with $g(f_A(b)) = 1$ does not depend of the order in which they are executed. Therefore, $(s_1 \circ f_X, s_2 \circ f_X) \in Tr^{S_1}(g \circ f_A)$. $\square$

Let $\mathsf{PreSh}(\mathsf{SYS}_{\mathsf{lm}})$ be the category of presheaves over $\mathsf{SYS}_{\mathsf{lm}}$.

We now show that we can construct a natural transformation Tr from Act to $\Omega^{St \times St}$.

Lemma 6.7 *Let $S \in \mathsf{SYS}_{\mathsf{lm}}$. For $f \in Act(S)$, $g : S' \to S$ and $s_1, s_2 \in St(S')$, the set $Tr_S(f)(S')(g, s_1, s_2) = \{S'' \xrightarrow{h} S' \mid (s_1 \circ h_X, s_2 \circ h_X) \in Tr^{S''}(f \circ g_a \circ h_a)\}$ is a sieve on S'.*

Proof: Let S'' be an arbitrary element of $Tr_S(f)(S')(g, s_1, s_2)$ (i.e. $S'' \xrightarrow{h} S'$ such that $(s_1 \circ h_X, s_2 \circ h_X) \in Tr^{S''}(f \circ g_a \circ h_a)$). Let $\overline{S}''$ be such that $\overline{S}'' \xrightarrow{h'} S''$. Therefore, $\overline{S}'' \xrightarrow{h'} S'' \xrightarrow{h} S'$, By Lemma 6.4, $(s_1 \circ h_X \circ h'_X, s_2 \circ h_X \circ h'_X) \in Tr^{S''}(f \circ g_a \circ h_a \circ h'_a)$. Hence, $\overline{S}'' \xrightarrow{h \circ h'} S' \in Tr_S(f)(S')(g, s_1, s_2)$. $\square$

Lemma 6.8 *Let $Tr_S(f) : y(S) \times St \times St \to \Omega$ be defined, for every system S', by $Tr_S(f)(S') : \mathsf{Hom}(S', S) \times St(S') \times St(S') \to \Omega(S')$, where for every $g : S' \to S$ and every $s_1, s_2 \in St(S')$ $Tr_S(f)(S')(g, s_1, s_2) = \{S'' \xrightarrow{h} S' \mid (s_1 \circ h_X, s_2 \circ h_X) \in Tr^{S''}(f \circ g_a \circ h_a)\}$. Then $Tr_S(f)$ is a natural transformation.*

Proof: In order to show that $Tr_S(f) : y(S) \times St \times St \to \Omega$ is a natural transformation we have to show that for every two objects in $\mathsf{SYS}_{\mathsf{lm}}$ and every arrow between them the corresponding diagram is commutative.

Let $\overline{S}' \xrightarrow{u} S'$. We show that the following diagram is commutative:

$$\begin{tikzcd}
\mathsf{Hom}(S', S) \times St(S') \times St(S') \arrow[r, "Tr_S(f)(S')"] \arrow[d, "Hom(u) \times St(u) \times St(u)"'] & \Omega(S') \arrow[d, "\Omega(u)"] \\
\mathsf{Hom}(\overline{S}', S) \times St(\overline{S}') \times St(\overline{S}') \arrow[r, "Tr_S(f)(\overline{S}')"] & \Omega(\overline{S}')
\end{tikzcd} \tag{6.1}$$

Let $(g, s_1, s_2) \in \mathsf{Hom}(S', S) \times St(S') \times St(S')$. Then, on the one hand,

$$\Omega(u)(Tr_S(f)(S')(g, s_1, s_2)) = \{\overline{S}'' \xrightarrow{h} \overline{S}' \mid \overline{S}'' \xrightarrow{u \circ h} S' \in Tr_S(f)(S')(g, s_1, s_2)\} =$$

$$= \{\overline{S}'' \xrightarrow{h} \overline{S}' \mid (s_1 \circ u_X \circ h_X, s_2 \circ u_X \circ h_X) \in Tr^{\overline{S}''}(f \circ g_A \circ u_A \circ h_A)\}.$$

On the other hand, we have $Tr_S(f)(S')(g \circ u, s_1 \circ u_X, s_2 \circ u_X) = \{\overline{S}'' \xrightarrow{h} \overline{S}' \mid (s_1 \circ u_X \circ h_X, s_2 \circ u_X \circ h_X) \in Tr^{\overline{S}''}(f \circ g_A \circ u_A \circ h_A)\}$. Thus, the diagram commutes. □

Consequence 6.9 *For every system S and $f \in Act(S)$, $Tr_S(f) \in \Omega^{St \times St}(S)$.*

Proof: Follows from the definition of $\Omega^{St \times St}(S)$ in $\mathsf{PreSh}(\mathsf{SYS_{lm}})$. □

Lemma 6.10 *$Tr : Act \to \Omega^{St \times St}$ defined for every system S by $Tr_S : Act(S) \to \Omega^{St \times St}(S)$ is a natural transformation in* $\mathsf{PreSh}(\mathsf{SYS_{lm}})$.

Proof: We show that for every $\overline{S} \xrightarrow{\xi} S$ the following diagram commutes:

$$\begin{tikzcd} Act(S) \arrow[r, "Tr_S"] \arrow[d, "Act(\xi)"'] & \Omega^{St \times St}(S) \arrow[d, "\Omega^{St \times St}(\xi)"] \\ Act(\overline{S}) \arrow[r, "Tr_{\overline{S}}"] & \Omega^{St \times St}(\overline{S}) \end{tikzcd} \tag{6.2}$$

Let $f \in Act(S)$. Then $\Omega^{St \times St}(\xi)(Tr_S(f)) : y(\overline{S} \times St(\overline{S}) \times St(\overline{S})$ is defined for every $\overline{S}' \xrightarrow{u} \overline{S}$ and $s_1, s_2 \in St(\overline{S})$ by

$\Omega^{St \times St}(\xi)(Tr_S(f))(\overline{S}')(u, s_1, s_2) = Tr_S(f)(\overline{S}')(\xi \circ u, s_1, s_2) = \{\overline{S}'' \xrightarrow{h} \overline{S}' \mid (s_1 \circ h_X, s_2 \circ h_x) \in Tr^{\overline{S}''}(f \circ \xi_A \circ u_A \circ h_A)\} = Tr_{\overline{S}}(f \circ \xi_A)(\overline{S}')(u, s_1, s_2)$. □

It is also easy to see that $Tr : \mathsf{SYS_{lm}}^{op} \to \mathsf{Sets}$ defined by

$$Tr(S) = \{(f, s, s') \mid f \in Act(S), s, s' \in St(S), (s, s') \in Tr^S(f)\}$$

is a subpresheaf of $Act \times St \times St$ (follows from Proposition 6.5 resp. Proposition 6.6, depending on the rule applied for computing transitions of parallel actions).

We showed that St and Act are presheaves and Tr defines a natural transformation (or, alternatively, a subpresheaf of $Act \times St \times St$).

Until now we did not consider the interaction between systems, or the possibility of expressing that a system arises by interconnecting (in a certain way) a given family of systems.

Consider for example two systems that have a "common dictionary".

The situation in Figure 6.5 can be described in terms of category theory by the pushout of the diagram defined by the two systems S_1, S_2 together with their "dictionary" seen as a system S_{12} that is "translated" in both S_1 and S_2:

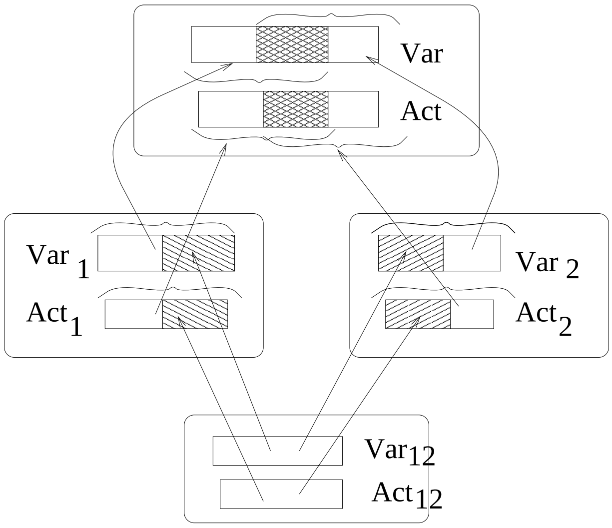


Figure 6.5: The system obtained by interconnecting two interacting systems.

$$\begin{tikzcd}
 & S & \\
S_1 \arrow[ur] & & S_2 \arrow[ul] \\
 & S_{12} \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Thus, interconnecting systems can be modeled by taking colimits (cf. [Gog92]).

A natural question arises: is it possible to define a covering relation on $\mathsf{SYS}_{\mathsf{Im}}$ that induces a Grothendieck topology on $\mathsf{SYS}_{\mathsf{Im}}$?

For the sake of simplicity we will not consider the general case here: we will restrict ourselves to the case where the morphisms are inclusions, i.e. the elements in a given family $\{S_i\}_{i \in I}$ of interconnected systems are seen as "parts" (subsystems) of the system S obtained by interconnecting them; as parts of S they are supposed to communicate via their "common subsystems".

A general theory that takes into account arbitrary morphisms between systems will be subject for future work.

Chapter 7

Categories of Systems with Inclusions as Morphisms

There are cases when systems are not able to "communicate" using "dictionaries" or "translations", as described in Section 6.3; important is that the systems have common subsystems by means of which the communication is done. In what follows, we will focus on this last aspect. We will first present some subcategories of the category SYS considered in Chapter 6, in which a morphism $f : S_1 \to S_2$ exists if S_1 is a subsystem of S_2 (possibly satisfying an additional "tightness" condition).

In this case, communication between two systems is assumed to take place only via the common subsystems. This situation arises naturally if the systems are assumed to be already "interconnected" and are regarded as parts of the system obtained by their interconnection.

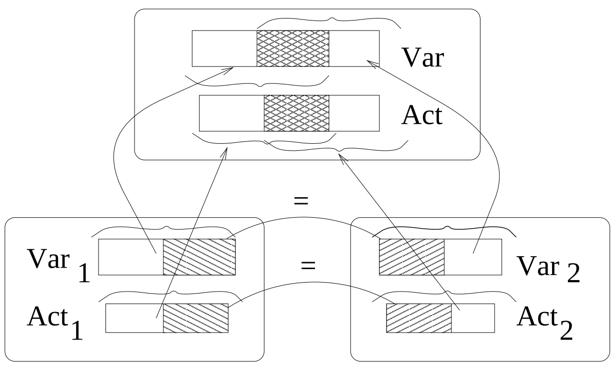


Figure 7.1: Interconnecting two systems with a common subsystem.

We begin by studying the category $\mathsf{SYS_i}$ that has systems as objects and inclusions as morphisms. Some categorical constructions in this category are illustrated and it is shown that two functors can be defined, one that associates with every system S its set of states $St(S)$, and one that associates with every system S its set of admissible parallel actions $Act(S)$. We show that these functors satisfy a "gluing" property with respect to colimits (interconnections) of systems.

The colimit (interconnection) of a family $\{S_i \mid i \in I\}$ of systems has as control variables the union X of the control variables X_i of the systems S_i, and as atomic actions the union A of the atomic actions A_i of the systems S_i.

The sets $X_i, i \in I$ can be regarded as open sets in a suitable topology on X expressing some appropriate notion of "neighborhood". Similarly, the sets $A_i, i \in I$ can be regarded as open sets in a suitable topology on A.

But when considering an interconnection of systems we would like to take into account these topological spaces (as well as the repartitioning of the constraints among systems) simultaneously, since the whole information characterizes a system.

In this case topology is not sufficient. Therefore we consider the category $\mathsf{SYS_i}$ and – in order to be able to express the notion of covering between systems – we define a Grothendieck topology on $\mathsf{SYS_i}$; it is proved that states and parallel actions can be modeled by sheaves with respect to this Grothendieck topology.

We would like to point out that – although $\mathsf{SYS_i}$ can be seen as a partially ordered set – it is in general not a lattice (it does not have a largest element, neither do all finite suprema (coproducts) nor all finite infima (products) exist in $\mathsf{SYS_i}$). Additionally, even in the case when the respective suprema and infima exist, no distributivity condition is fulfilled in general, as will be shown in Example 7.2. Therefore, the category $\mathsf{SYS_i}$ cannot be regarded as a locale (or, dually, as a frame).

However, the category $\mathsf{SYS_i}$ only captures the static aspect of systems. It can be the case that, if $S_1 \hookrightarrow S_2$ and we regard a transition in S_2 from the perspective of S_1, some variables in S_1 may change their values with no apparent cause. This happens when some actions in S_2, unknown in S_1, depend on variables of S_1. We call a subsystem $S_1 \hookrightarrow S_2$ in which all changements of the variables in S_1 can be explained *transition connected.*

We therefore consider $\mathsf{SYS_{il}}$, the subcategory of SYS having as objects systems and as morphisms so-called transition-connected inclusions.

We define a Grothendieck topology J on $\mathsf{SYS_{il}}$ and show that the presheaves St, Act (where for every system S, $St(S)$ is the set of states of S (i.e. possible configurations of values for the variables that control S), and $Act(S)$ the set of admissible parallel actions) are sheaves with respect to this Grothendieck topology. Moreover, we show that the transitions induced by admissible parallel actions define a natural transformation $Tr : Act \to \Omega^{St \times St}$ (in $\mathsf{Sh}(\mathsf{SYS_{il}}, \mathsf{J})$) (or, alternatively, a subsheaf of $Act \times St \times St$). We use these results for modeling the behavior of the systems. We start with the formalism developed by Goguen in [Gog92], and slightly modify Goguen's definition for the behavior of a system, explicitly indicating for every moment of time not only the state of the system but also the action performed at the given moment, and obtain a contravariant functor from the category of systems to the category of sheaves over time. We show that a contravariant functor from $\mathcal{T}$ to the category of sheaves over Sys

can also be defined.

We conclude by investigating behavior by traces of execution. Assume that all constraints between actions are of the form $a_i \wedge a_j = 0$ (meaning that $a_i \wedge a_j = 0$ cannot be performed in parallel). Then the functor that associates to every system S the partially commutative monoid $M(S)$ (obtained as a quotient of the free monoid freely generated by the set of actions of S by the congruence generated by $\{ab = ba \mid a, b \text{ independent}\}$) is a sheaf only under very restrictive conditions.

7.1 The Static Aspect: $\mathsf{SYS_i}$

For a first approximation to representing the states of a system and the possible parallel actions – in which only the "static aspects" are represented – no information about the transitions is needed.

We therefore give a simpler definition of a system, namely as a tuple

$$S = (\Sigma, X, \Gamma, M, A, C),$$

and ignore the description of the variables on which each action depends, as well as the description of transitions and the way transitions of admissible parallel actions are computed.

In this context we can define a notion of subsystem ("static", for the beginning).

Definition 7.1 *Let S_1 and S_2 be two systems. We say that S_1* is a subsystem *of S_2 (or S_2* is an extension *of S_1) if and only if $\Sigma_1 \subseteq \Sigma_2$, $X_1 \subseteq X_2$, $A_1 \subseteq A_2$, the constraints in Γ_1 (resp. C_1) are consequences of the constraints in Γ_2 (resp. C_2), and the model M_1 is the restriction of the model M_2 to the signature of S_1 ($M_1 = U^{\Sigma_2}_{\Sigma_1} M_2$).*

Remark: Since we assumed that the sets Γ_S and C_S of constraints in a system S are complete with respect to semantical consequence, it follows that the condition "the constraints in Γ_1 (resp. C_1) are consequences of the constraints in Γ_2 (resp. C_2)" in a subsystem in fact amounts to $\Gamma_1 \subseteq \Gamma_2$ (resp. $C_1 \subseteq C_2$).

Let $\mathsf{SYS_i}$ be the category having systems as objects, and a morphism from S_1 to S_2 if and only if S_2 is an extension of S_1 (i.e. all morphisms are inclusions – that is why we use the index i in the name of the category). We will briefly discuss the meaning of the standard limit and colimit constructions in the category $\mathsf{SYS_i}$.

7.1.1 Categorical Constructions in $\mathsf{SYS_i}$

1. Limits:

The *pullback* is the largest common subsystem of two systems that are known to be contained in a larger system.

$$\begin{tikzcd} S_1 \times_S S_2 \arrow[r] \arrow[d] & S_2 \arrow[d] \\ S_2 \arrow[r] & S \end{tikzcd} \tag{7.1}$$

Namely, if for $i = 1, 2$ $S_i = (\Sigma_i, X_i, \Gamma_i, M_i, A_i, C_i) \hookrightarrow S = (\Sigma, X, \Gamma, M, A, C)$, then $S_1 \times_S S_2 = (\Sigma_1 \cap \Sigma_2, X_1 \cap X_2, \Gamma_1 \cap \Gamma_2, M_{12}, A_1 \cap A_2, C_1 \cap C_2)$ where M_{12} is the restriction of M to the signature $\Sigma_1 \cap \Sigma_2$.

Pullbacks of this type always exist in $\mathsf{SYS_i}$.

The *product* $S_1 \times S_2$ of two systems S_1 and S_2 (if it exists) is a system with the following properties:

- $S_1 \times S_2$ is a subsystem of both S_1 and S_2,
- For every system S that is a subsystem of both S_1 and S_2, S is a subsystem of $S_1 \times S_2$,

i.e. $S_1 \times S_2$ is the "largest" common subsystem of S_1 and S_2. The product of two systems exists only if their models are compatible, in the sense that their restrictions to the common signature coincide.

$\mathsf{SYS_i}$ does not have a terminal object (i.e. there is no system that contains all the systems in $\mathsf{SYS_i}$).

2. Colimits:

Colimits play a special rôle in the study of systems. As already pointed out in Section 6.3 (cf. also [Gog92]), the system obtained by interconnecting a given family (diagram) of systems can be obtained computing the colimit of the given diagram.

A *coproduct* $S_1 \amalg S_2$ in SYS (the category with arbitrary morphisms) of a diagram consisting of only two systems S_1 and S_2 is the system obtained by taking the disjoint union of their languages, control variables, actions, respectively the closures under consequence of the unions of the corresponding constraints.

In $\mathsf{SYS_i}$ the morphisms are inclusions. Therefore, the coproduct of the diagram consisting of only two systems S_1 and S_2 is the system S such that

- S_1, S_2 are subsystems of S,
- For every system T such that S_1, S_2 are subsystems of T, S is a subsystem of T.

In other words, the coproduct in $\mathsf{SYS_i}$ of S_1 and S_2 is the smallest system that contains both S_1 and S_2. Note that the coproduct in $\mathsf{SYS_i}$ of the diagram consisting of S_1 and S_2 exists only if S_1 and S_2 are compatible, in the sense that $M_{1|\Sigma_1 \cap \Sigma_2} = M_{2|\Sigma_1 \cap \Sigma_2}$. In this case, the coproduct in $\mathsf{SYS_i}$ of S_1 and S_2 is the colimit in SYS of the diagram defined by $\{S_1, S_2, S_1 \times S_2\}$ (with the corresponding inclusions between them).

It is easy to see that if S_1 and S_2 are independent systems then $St(S_1 \amalg S_2) = St(S_1) \times St(S_2)$, and $Act(S_1 \amalg S_2) = Act(S_1) \times Act(S_2)$.

The *pushout* $S_1 \amalg_S S_2$ in SYS (see diagram 7.2) represents the system obtained by interconnecting S_1 and S_2 "gluing" them via their common subsystem S.

$$\begin{tikzcd} S \arrow[r] \arrow[d] & S_1 \arrow[d] \\ S_2 \arrow[r] & S_1 \amalg_S S_2 \end{tikzcd} \tag{7.2}$$

The *pushout* $S_1 \amalg_S S_2$ in $\mathsf{SYS_i}$ coincides with the coproduct of S_1 and S_2 in $\mathsf{SYS_i}$.

It is easy to see that if S_1 and S_2 have S as a largest common subsystem, then $St(S_1 \amalg_S S_2) = St(S_1) \times_{St(S)} St(S_2)$ and $Act(S_1 \amalg_S S_2) = Act(S_1) \times_{Act(S)} Act(S_2)$.

In what follows we will be interested in colimits in $\mathsf{SYS_i}$ of families of subsystems of a given system S. We show here how colimits of such families can be computed.

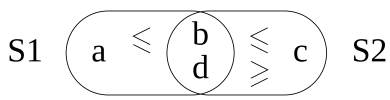
Lemma 7.1 *Let* $S = (\Sigma, X, M, \Gamma, A, C)$ *be a system and* $\{S_i \hookrightarrow S \mid i \in I\}$ *a family of subsystems of* S*, closed under taking subsystems, where for every* $i \in I$*,* $S_i = (\Sigma_i, X_i, M_i, \Gamma_i, A_i, C_i)$*. The colimit of this family of systems is* $\overline{S}$ *with* $\Sigma_{\overline{S}} = \bigcup_{i\in I} \Sigma_i$*,* $X_{\overline{S}} = \bigcup_{i\in I} X_i$*,* $M_{\overline{S}} = M_{|\bigcup_{i\in I}\Sigma_i}$*,* $\Gamma_{\overline{S}} = (\bigcup_{i\in I} \Gamma_i)^\bullet$*,* $A_{\overline{S}} = \bigcup_{i\in I} A_i$*,* $C_{\overline{S}} = (\bigcup_{i\in I} C_i)^\bullet$*.*

Proof: It is easy to see that $\overline{S}$ is a cocone. It remains to show that it satisfies the universality property of a colimit. Let T be another cocone, i.e. such that $S_i \hookrightarrow T$ for every $i \in I$. Then obviously $\bigcup_{i\in I} \Sigma_i \subseteq \Sigma_T$, $\bigcup_{i\in I} X_i \subseteq X_T$, $M_i = M_{T|\Sigma_i}$, $\bigcup_{i\in I} A_i \subseteq A_T$, and the constraints in $\bigcup_{i\in I} \Gamma_i$ (resp. $\bigcup_{i\in I} C_i$) are consequences of the constraints in Γ_T (resp. C_T), hence also their consequences are consequences of the constraints in Γ_T (resp. C_T). It follows that $\overline{S} \hookrightarrow T$. $\square$

Note that this colimit is equal to the colimit in SYS of the diagram obtained from the family $\{S_i \mid i \in I\}$ by closing it under all subsystems.

The *initial object* in $\mathsf{SYS_i}$ is the empty system, denoted by $\emptyset$ ($\Sigma_\emptyset = X_\emptyset = \Gamma_\emptyset = A_\emptyset = C_\emptyset = \emptyset$).

It is easy to see that a system may be subject to more constraints when interconnected with other systems than when considered independently, as can be seen from the following example.



Example 7.1 *Let $\Sigma = \{0, 1, \leq\}$ be a signature where 0 and 1 are 0-ary function symbols and $\leq$ is a binary predicate symbol. Let $M = (\{0, 1\}, \{0_M, 1_M, \leq_M\})$ be a Σ-structure where $\leq_M$ is an order relation and $0 \leq_M 1$. Consider the systems*

$$S_1 = (\Sigma, \{a, b, d\}, \{a \leq b\}, M, A_1, C_1),$$

$$S_2 = (\Sigma, \{b, c, d\}, \{b \leq c, c \leq d\}, M, A_2, C_2).$$

Let S the system obtained by interconnecting S_1 and S_2.

$$S = (\Sigma, \{a, b, c, d\}, \{a \leq b, b \leq c, c \leq d\}, M, A_1 \cup A_2, (C_1 \cup C_2)^\bullet).$$

Note that for every $s : \{a, b, c, d\} \to M$ such that $s \models \{a \leq b, b \leq c, c \leq d\}$ we have $s(a) \leq_M s(b) \leq_M s(c) \leq_M s(d)$, hence $s(a) \leq_M s(d)$. Thus, $a \leq d$ is a consequence of the constraints $\{a \leq b, b \leq c, c \leq d\}$.

This shows that for every state $s \in St(S)$, we have $s_{|S_1} \models a \leq d$. It follows therefore that for every s_1, s_2, with $s_1 \in St(S_1)$, and $s_2 \in St(S_2)$ and such that $s_{1|X_1 \cap X_2} = s_{2|X_1 \cap X_2}$, $s_1 \models a \leq d$.

Intuitively, this expresses the fact that in S_1 seen as a "part" of S, $a \leq d$ has to be satisfied (by interconnecting systems new constraints may arise).

7.1.2 A Grothendieck Topology on $\mathsf{SYS_i}$

Our goal now is to define a covering relation on the category $\mathsf{SYS_i}$. A first possible notion of "covering" (we will call it *quasi-covering*) is given below. We use the name quasi-covering in this case because it will be shown that it does not induce a Grothendieck topology. Later (in Definition 7.3) we will define a notion of cover that does define a Grothendieck topology.

Definition 7.2 *A* quasi-covering family *for a system $S = (\Sigma, X, \Gamma, M, A, C)$ is a family $\{S_i \hookrightarrow S \mid i \in I\}$ of subsystems of S, with the property that $\bigcup_i \Sigma_i = \Sigma$, $\bigcup_i X_i = X$, $\bigcup_i A_i = A$, and such that $\Gamma = (\bigcup_{i \in I} \Gamma_i)^\bullet$ and $C = (\bigcup_i C_i)^\bullet$.*

It is easy to see that the states and parallel actions satisfy a gluing condition on the quasi-covering families defined above.

Proposition 7.2 *Let $S = (\Sigma, X, \Gamma, M, A, C)$ and $\{S_j \hookrightarrow S \mid j \in J\}$ be a quasi-covering family for S, where for every $j \in J$, $S_j = (\Sigma_j, X_j, \Gamma_j, M_j, A_j, C_j)$. Let $\{s_j \mid j \in J\}$ be a matching family of elements $s_j \in St(S_j)$ (i.e. such that for every $j_1, j_2 \in J$, $s_{j_1 | X_{j_1} \cap X_{j_2}} = s_{j_2 | X_{j_1} \cap X_{j_2}}$). Then there exists a unique $s \in St(S)$ such that $s_{|X_j} = s_j$ for every $j \in J$.*

Proof: We can define $s : X \to M$, by $s(x) = s_j(x)$ if $x \in X_j$. Let $\phi \in \bigcup \Gamma_j$. Then ϕ contains exclusively symbols in $\mathcal{L}_j = (\Sigma_j, X_j)$ for some j, s satisfies ϕ because its restriction to X_j, namely s_j, does (we used Lemma 3.19). Therefore s satisfies all constraints in $\bigcup \Gamma_j$, hence also all their consequences, i.e. it satisfies all formulas in $\Gamma = (\bigcup \Gamma_j)^\bullet$; so $s \in St(S)$. It is easy to see (by the definition of s) that s is the unique element of $St(S)$ such that for every $i \in I$, $s_{|X_i} = s_i$. This proves that St is a sheaf. □

A similar result also holds for the parallel actions:

Proposition 7.3 *Let* $S = (\Sigma, X, \Gamma, M, A, C)$ *and* $\{S_j \hookrightarrow S \mid j \in J\}$ *be a cover of* S*, where for every* $j \in J$, $S_j = (\Sigma_j, X_j, \Gamma_j, M_j, A_j, C_j)$*. Let* $\{f_i \mid i \in I\}$ *be a family of parallel actions* $f_i \in Act(S_i)$ *such that for every* $i, j \in I$, $f_{i|A_i \cap A_j} = f_{j|A_i \cap A_j}$*. Then there exists a unique* $f \in Act(S)$ *such that* $f_{|A_i} = f_i$*.*

These gluing properties naturally raise the question whether the covering relation defines a Grothendieck topology such that St and Act are sheaves with respect to this Grothendieck topology.

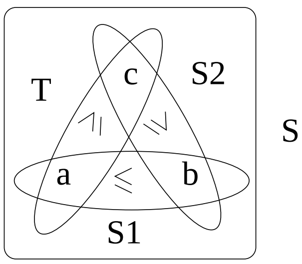
We therefore have to check whether the notion defined before satisfies the properties of a basis for a Grothendieck topology, cf. Definition 3.96, namely:

(1) $\{S\}$ is a quasi-cover for S,

(2) If $\{S_i \hookrightarrow S \mid i \in I\}$ is a quasi-cover for S then for any morphism $T \hookrightarrow S$ the family of pullbacks $\{S_i \times_S T \to T \mid \in I\}$ is a quasi-cover for T;

(3) If $\{S_i \hookrightarrow S \mid i \in I\}$ is a quasi-cover for S and if for each $i \in I$ one has a family $\{S_{ij} \to S_i \mid j \in I_i\}$ that is a quasi-cover for S_i, then the family $\{S_{ij} \hookrightarrow S \mid i \in I, j \in I_i\}$ is a quasi-cover for S.

Properties (1) and (3) are obviously satisfied. It is however easy to see that condition (2) (similar to a distributivity property) is not satisfied, as shown by the following example (for the sake of simplicity we assume that the set of actions is empty):



Example 7.2 *Let* $\Sigma = \{0, 1, \leq\}$ *be a signature where* 0 *and* 1 *are 0-ary function symbols and* $\leq$ *is a binary predicate symbol. Let* $M = (\{0,1\}, \{0_M, 1_M, \leq_M\})$ *be a* Σ*-structure where* $\leq_M$ *is an order relation and* $0 \leq_M 1$*. Consider the following systems:*

$$S_1 = (\Sigma, \{a, b\}, \{a \leq b\}, M, A, C),$$

$$S_2 = (\Sigma, \{b, c\}, \{b \leq c\}, M, A, C),$$

$$T = (\Sigma, \{a, c\}, \{a \leq c\}, M, A, C).$$

Let S *be the system obtained by interconnecting* S_1 *and* S_2*.*

$$S = (\Sigma, \{a, b, c\}, \{a \leq b, b \leq c, a \leq c\}, M, A, C).$$

The family $\{S_1, S_2, S_1 \cap S_2\}$ is a quasi-covering family for S, according to Definition 7.2. T is a subsystem of S, but $S_1 \cap T = \{\Sigma, \{a\}, \emptyset, M, A, C)$ and $S_2 \cap T = \{\Sigma, \{c\}, \emptyset, M, A, C)$, hence $\{S_1 \cap T, S_2 \cap T, S_1 \cap S_2 \cap T\}$ does not cover T.

This shows that the quasi-covering relation introduced in Definition 7.2 does not define a basis for a Grothendieck topology. Also, the induced "covering" relation J by sites,

$$R \in J(S) \text{ iff there exists a quasi-cover } T \text{ for } S \text{ such that } T \subseteq R$$

does not satisfy property (2) of a Grothendieck topology (consider in Example 7.2 the sieve generated by $\{S_1, S_2\}$).

This suggests that the obvious notion of quasi-cover defined above is not completely appropriate. We give a new definition of cover, trying to avoid situations like the one described in the example above.

Definition 7.3 (Covering Family) *Let $S = (\Sigma, X, \Gamma, M, A, C)$ be a system. A* covering family *for S is a family $\mathcal{S} = \{S_i \hookrightarrow S \mid i \in I\}$ of subsystems of S, with the following properties:*

(C1) $\mathcal{S}$ is a sieve (i.e. it is closed under subsystems),

(C2) S is a colimit of the diagram defined by the sieve $\mathcal{S}$,

(C3) For every $T \subseteq S$, T is the colimit of the diagram defined by the sieve $T \cap \mathcal{S} = \{S_i \in \mathcal{S} \mid S_i \hookrightarrow T\}$.

It is easy to see that, with the notation used in Example 7.2, the sieve $\mathcal{S}$ generated in $\mathsf{SYS_i}$ by $\{S_1, S_2, S_1 \cap S_2\}$ is not a covering family for S according to Definition 7.3 because T is a subsystem of S, but is not a colimit of the diagram defined by the sieve $\{S_i \in \mathcal{S} \mid S_i \hookrightarrow T\} = \{S_1 \cap T, S_2 \cap T, S_1 \cap S_2 \cap T\}$.

Proposition 7.4 *The function J assigning for every system S the set $J(S)$ of all covering families for S is a Grothendieck topology on $\mathsf{SYS_i}$.*

Proof: We show that the axioms required in the definition of a Grothendieck topology are satisfied:

(1) Let $\mathcal{S} = \{T \mid T \hookrightarrow S\}$ be the sieve of all subsystems of S. Since S is an element of $\mathcal{S}$, it follows immediately that S is a colimit of the diagram defined by $\mathcal{S}$, and since every $T \hookrightarrow S$ is an element of $T \cap \mathcal{S}$, it is the colimit of the diagram defined by the sieve $T \cap \mathcal{S}$. Thus $\mathcal{S}$ is a covering family for S.

(2) Let $\mathcal{S} = \{S_i \hookrightarrow S \mid i \in I\} \in J(S)$ and $T \hookrightarrow S$. We show that the sieve $T \cap \mathcal{S} = \{S_i \in \mathcal{S} \mid S_i \hookrightarrow T\}$ is a cover for T. It is obvious that it is a sieve and that its colimit (as a diagram) is T (by $(C3)$, since $\mathcal{S} \in J(S)$). Let $T' \hookrightarrow T$. By the fact that $T' \cap (T \cap \mathcal{S}) = T' \cap \mathcal{S}$, and since T' is in particular a subsystem

of S, it follows that T' is the colimit of the diagram defined by $T' \cap (T \cap \mathcal{S})$. Thus, $T \cap \mathcal{S} = \{S_i \in \mathcal{S} \mid S_i \hookrightarrow T\}$ is a cover for T.

(3) Let $\mathcal{S} = \{S_i \hookrightarrow S \mid i \in I\} \in J(S)$, and $\mathcal{R}$ an arbitrary sieve on S such that for every $S_i \in \mathcal{S}$, $S_i \cap \mathcal{R} \in J(S_i)$ (we assume that for $i \in I$, $S_i \cap \mathcal{R} = \{S_{ij} \mid j \in I_i\}$).

In order to show that $\mathcal{R} \in J(S)$ we have to prove that $(C1)$, $(C2)$ and $(C3)$ are fulfilled.

$(C1)$ is obviously true since $\mathcal{R}$ is a sieve.

In order to prove $(C2)$ note that the following holds:

$$\Sigma = \bigcup_{i \in I} \Sigma_i = \bigcup_{i \in I, j \in I_i} \Sigma_{ij}, X = \bigcup_{i \in I} X_i = \bigcup_{i \in I, j \in I_i} X_{ij},$$

$$M_{\varinjlim \mathcal{R}} = M_{S|\Sigma} = M_S, \Gamma = \left(\bigcup_{i \in I} \Gamma_i \right)^{\bullet} = \left(\bigcup_{i \in I, j \in I_i} \Gamma_{ij} \right)^{\bullet},$$

$$A = \bigcup_{i \in I} A_i = \bigcup_{i \in I, j \in I_i} A_{ij}, C = \left(\bigcup_{i \in I, j \in I_i} C_{ij} \right)^{\bullet}.$$

In order to prove $(C3)$, let $T \hookrightarrow S$ be a subsystem of S. We have to show that $T \cap \mathcal{R}$ has T as a colimit.

We show that in fact $T \cap \{S_{ij} \mid i \in I, j \in I_i\}$ has T as a colimit – and hence T is also the colimit of $T \cap \mathcal{R}$.

$T \cap \{S_{ij} \mid i \in I, j \in I_i\} = \{T \cap S_{ij} \mid i \in I, j \in I_i\}$. It is easy to see that, since $T \cap S_i \hookrightarrow S_i$ and $\{S_{ij} \mid j \in I_i\} \in J(S_i)$, the following holds:

$$\bigcup_{j \in I_i} (\Sigma_{ij} \cap \Sigma_T) = \bigcup_{j \in I_i} \Sigma_{ij} \cap (\Sigma_T \cap \Sigma_i) = \Sigma_T \cap \Sigma_i,$$

$$\bigcup_{j \in I_i} (X_{ij} \cap X_T) = X_T \cap X_i, \bigcup_{j \in I_i} (A_{ij} \cap A_T) = A_T \cap A_i,$$

$$\left(\bigcup_{j \in I_i} \Gamma_{ij} \cap \Gamma_T \right)^{\bullet} = \Gamma_T \cap \Gamma_i, \left(\bigcup_{j \in I_i} C_{ij} \cap C_T \right)^{\bullet} = C_T \cap C_i.$$

Since $\mathcal{S} = \{S_i \hookrightarrow S \mid i \in I\} \in J(S)$, T is the colimit of $T \cap \mathcal{S}$, i.e. $\Sigma_T = \bigcup_{i \in I} \Sigma_T \cap \Sigma_i$ (similarly for X_T and A_T), and the constraints in Γ_T (resp. C_T) are exactly the consequences of the constraints in $\bigcup_{i \in I} \Gamma_T \cap \Gamma_i$ (resp. $\bigcup_{i \in I} C_T \cap C_i$).

Therefore,

$$\Sigma_T = \bigcup_{i \in I} \Sigma_T \cap \Sigma_i = \bigcup_{i \in I} \bigcup_{j \in I_i} (\Sigma_{ij} \cap \Sigma_T), X_T = \bigcup_{i \in I} \bigcup_{j \in I_i} (X_{ij} \cap X_T),$$

$$\Gamma_T = \left(\bigcup_{i \in I} \Gamma_T \cap \Gamma_i \right)^{\bullet} = \left(\bigcup_{i \in I} \left(\bigcup_{j \in I_i} \Gamma_T \cap \Gamma_{ij} \right)^{\bullet} \right)^{\bullet} = \left(\bigcup_{i \in I} \bigcup_{j \in I_i} \Gamma_T \cap \Gamma_{ij} \right)^{\bullet},$$

$$M_T = M_{S|\Sigma_T} = M_{S|\bigcup \Sigma_{ij}},\ A_T = \bigcup_{i\in I}\bigcup_{j\in I_i}(A_{ij}\cap A_T),$$

$$C_T = \left(\bigcup_{i\in I} C_T \cap C_i\right)^{\bullet} = \left(\bigcup_{i\in I}\left(\bigcup_{j\in I_i} C_T \cap C_{ij}\right)^{\bullet}\right)^{\bullet} = \left(\bigcup_{i\in I}\bigcup_{j\in I_i} C_T \cap C_{ij}\right)^{\bullet}.$$

From this it follows that T is the colimit of the sieve $\{T \cap S_{ij} \mid i \in I, j \in I_i\}$. Since $S_{ij} \in \mathcal{R}$ for every $i \in I, j \in I_i$, this implies that T is the colimit of the sieve $T \cap \mathcal{R}$. Thus we proved that $\mathcal{R} \in J(S)$. □

Note that all the covering families according to Definition 7.3 are also quasi-covering families according to Definition 7.2. It is easy to see that in this case the gluing properties for *St* and *Act* are indeed sheaf conditions. Therefore, the following theorem holds.

Theorem 7.5 *The functors $St, Act : \mathsf{SYS_i}^{op} \to \mathsf{Sets}$ are sheaves with respect to the Grothendieck topology J.*

7.2 The Dynamic Aspect: $\mathsf{SYS_{il}}$

In order to also capture the "dynamic" aspect of systems, i.e. the way they evolve in time, we have to be able to represent the way the states of a system change when actions are executed. So, besides the language and actions, more information is needed in the definition of a system S. Namely, for every action we have to know which variables the action really depends upon, and the way these variables change after the action is performed.

Therefore, a system is represented as

$$S = (\Sigma, X, \Gamma, M, A, C, \{X_a\}_{a\in A}, \{Tr_a\}_{a\in A}),$$

where for every $a \in A$, X_a is the (minimal) set of all variables in X action a depends upon, and $Tr_a \subseteq \{(s_{1|X_a}, s_{2|X_a}) \mid s_1, s_2 \in St(S)\}$ shows how the values of these variables may change if a is performed.

It is easy to see that, given two systems $S_1 \hookrightarrow S_2$, if we have a transition in S_2 and regard it from the "perspective" of S_1, it may happen that in S_1 some variables change their values with no apparent cause. This is usually the case when some actions in S_2, that are unknown in S_1, depend on variables of S_1 (and change their values).

A subsystem $S_1 \hookrightarrow S_2$ in which all changes of the variables in S_1 can be explained "internally" will be called *transition connected.* Below we give the formal definition.

Definition 7.4 (Transition Connected Subsystems) *Let $S_1 \hookrightarrow S_2$ be a subsystem. We say that S_1 is a* transition-connected *subsystem of S_2 if*

(TC1) For every $a \in A_2$, if $X_a \cap X_1 \neq \emptyset$ then $a \in A_1$, and $X_a^1 = X_a^2 \cap X_1$,

(TC2) For every $a \in A_1$ and for every $s_1, s_2 \in St(S_2)$ if $(s_{1|X_a^2}, s_{2|X_a^2}) \in Tr_a^{S_2}$ then $(s_{1|X_a^1}, s_{2|X_a^1}) \in Tr_a^{S_1}$.

The next result shows that if $S_1 \subseteq S_2$ is transition-connected then valid transitions in S_2 restrict to valid transitions in S_1.

Lemma 7.6 *Let S_1 be a transition-connected subsystem of S_2, and let $a \in A_1$. Let $(s_1, s_2) \in Tr^{S_2}(a)$. Then $(s_{1|X_1}, s_{2|X_1}) \in Tr^{S_1}(a)$.*

Proof: Let $(s_1, s_2) \in Tr^{S_2}(a)$. Then $(s_{1|X_a^2}, s_{2|X_a^2}) \in Tr_a^{S_2}$, hence, since S_1 is a transition-connected subsystem of S_2, by $(TC2)$, $(s_{1|X_a^2 \cap X_1}, s_{2|X_a^2 \cap X_1}) \in Tr_a^{S_1}$. Let $x \in X_1$ be such that $s_1(x) \neq s_2(x)$. Then $x \in X_a^2 \cap X_1 = X_a^1$. Thus, for every $x \in X_1\ X_a$, $s_1(x) = s_2(x)$. This proves that $(s_{1|X_1}, s_{2|X_1}) \in Tr^{S_1}(a)$. □

We will now show the link between local morphisms of systems and transition-connected systems. We begin by defining the image of a system S via a (local) morphism of systems.

Definition 7.5 *Let $f : S_1 \to S_2$ be a local morphism. The image of S_1 by f is $f(S_1) = (f_\Sigma(\Sigma_1), f_X(X_1), (f^\natural_{\mathsf{Fma}}(\Gamma_1))^\bullet, M_{2|f_\Sigma}, f_A(A_1), (f^\natural_A(C_1))^\bullet, \{X_a^{f(S_1)}, Tr_a^{f(S_1)}\}_{a \in f_A(A_1)})$, where*

- *for all actions $a \in f_A(A_1)$, $X_a^{f(S_1)} = f_X(\bigcup_{f_A(b)=a} X_b^1)$, and*
- *for every $a \in f_A(A_1)$ and $s_1, s_2 \in St(f(S_1))$, let $g : A_1 \to \{0,1\}$ be defined by $g(b) = 1$ iff $f_A(b) = a$. Then $Tr_a^{f(S_1)} = \{(s_{1_{|X_a^{f(S_1)}}}, s_{2_{|X_a^{f(S_1)}}}) \mid (s_1 \circ f_{X|X_g}, s_2 \circ f_{X|X_g}) \in Tr_g^1\}$.*

Lemma 7.7 *Let $f : S_1 \to S_2$ be a local morphism. Then for every $a \in f_A(A_1)$, $X_a^{f(S_1)} = X_a^2 \cap f_X(X_1)$.*

Proof: By the definition of $f(S_1)$, for all actions $a \in f_A(A_1)$, $X_a^{f(S_1)} = f_X(\bigcup_{f_A(b)=a} X_b^1)$.

Since f is a local morphism, by $(M5)$ it follows that for every $a \in f_A(A_1)$, $\bigcup_{f_A(b)=a} X_b^1 = f_X^{-1}(X_a^2)$. It remains to show that $f_X(f_X^{-1}(X_a^2)) = X_a^2 \cap f_X(X_1)$.

First note that if $x \in f_X(f_X^{-1}(X_a^2))$, then $x = f_X(x_1)$ with $x_1 \in f_X^{-1}(X_a^2)$. Hence, $x = f_X(x_1) \in X_a^2 \cap f_X(X_1)$.

Let now $x \in X_a^2 \cap f_X(X_1)$. Then $x \in X_a^2$ and $x = f_X(x_1)$ with $x_1 \in X_1$. Therefore, $f_X(x_1) \in X_a^2$, hence, $x_1 \in f_X^{-1}(X_a^2)$. Thus, $x \in f_X(f_X^{-1}(X_a^2)$. □

Proposition 7.8 *The following holds:*

(1) Let S_1 be a transition-connected subsystem of S_2. Then the inclusion of S_1 in S_2 is a local morphism.

(2) Let $f : S_1 \to S_2$ be a local morphism. Then $f(S_1)$ is a transition-connected subsystem of S_2.

Proof: (1) Assume that S_1 is a transition-connected subsystem of S_2. Condition $(M5)$ in the definition of a local morphism follows from condition $(TC1)$, and condition $(M6)$ follows from condition $(TC2)$.
(2) Let $f : S_1 \to S_2$ be a local morphism. It is easy to see that $f(S_1)$ is a subsystem of S_2. We show that it is a transition-connected subsystem.

Let $a \in A_2$ be such that $X_a \cap f_X(X_1) \neq \emptyset$. Then, by $(M5)$, $a \in f_A(A_1)$.

Let $a \in f_A(A_1)$. Then by, Lemma 7.7, $X_a^{f(S_1)} = X_a^2 \cap f_X(X_1)$. Let $s_1, s_2 \in St(S_2)$ such that $(s_{1|X_a^2}, s_{2|X_a^2}) \in Tr_a^{S_2}$. Then, by $(M6)$, $(s_1 \circ f_{X|X_g}, s_2 \circ f_{X|X_g}) \in Tr_g^{S_1}$, where $g : A_1 \to \{0,1\}$ is defined by $g(a_1) = 1$ if and only if $f_A(a_1) = a$. Therefore, by the definition of $f(S_1)$, $(s_{1_{|X_a^{f(S_1)}}}, s_{2_{|X_a^{f(S_1)}}}) \in Tr_a^{f(S_1)}$. □

Remark 7.9 *Assume that S_1 is a transition-connected subsystem of S_2, and S_2 is a transition-connected subsystem of S_3. Then S_1 is a transition-connected subsystem of S_3.*

Proof: Assume that S_1 is a transition-connected subsystem of S_2, and S_2 is a transition-connected subsystem of S_3. Let $a \in A_3$ such that $X_a^3 \cap X_1 \neq \emptyset$. Since $X_1 \subseteq X_2$ it follows that $X_a^3 \cap X_2 \neq \emptyset$, and by the fact that S_2 is a transition-connected subsystem of S_3 we have $a \in A_2$ and $X_a^2 = X_a^3 \cap X_2$. Hence, $X_a^2 \cap X_1 \neq \emptyset$, and thus, since S_1 is a transition-connected subsystem of S_2 it follows that $a \in A_1$ and $X_a^1 = X_a^2 \cap X_1 = X_a^3 \cap X_2 \cap X_1 = X_a^3 \cap X_1$.

Let now $a \in A_1$ and $s_1, s_2 \in St(S_3)$ such that $(s_{1|X_a^3}, s_{2|X_a^3}) \in Tr_a^{S_3}$. Then $(s_{1|X_a^2}, s_{2|X_a^2}) \in Tr_a^{S_2}$, and hence, $(s_{1|X_a^1}, s_{2|X_a^1}) \in Tr_a^{S_1}$. This shows that S_1 is a transition-connected subsystem of S_3. □

Definition 7.6 $\mathsf{SYS}_{\mathsf{il}}$ *will denote the category having as objects systems and a morphism between S_1 and S_2 if and only if S_1 is a transition-connected subsystem of S_2 (i.e. all morphisms are transition-connected inclusions).*

7.2.1 Categorical Constructions in $\mathsf{SYS}_{\mathsf{il}}$

Proposition 7.10 *The category* $\mathsf{SYS}_{\mathsf{il}}$ *has pullbacks.*

Proof: Let S_1, S_2 be two transition-connected subsystems of S, as shown in Diagram 7.3.

$$\begin{tikzcd} S_{12} \arrow[r] \arrow[d] & S_1 \arrow[d] \\ S_2 \arrow[r] & S \end{tikzcd} \tag{7.3}$$

Then $M_1 = M_{|\Sigma_1}$, $M_2 = M_{|\Sigma_2}$; additionally, for every $a \in A_1$ (resp. in A_2), $X_a^1 = X_a^S \cap X_1$ (resp. $X_a^2 = X_a^S \cap X_2$). It follows therefore that for every $a \in A_1 \cap A_2$, $X_a^1 \cap X_2 = X_a^2 \cap X_1 = X_a^S \cap X_1 \cap X_2$.

Let $S_{12} = (\Sigma_1 \cap \Sigma_2, X_1 \cap X_2, \Gamma_1 \cap \Gamma_2, M_{S|\Sigma_1 \cap \Sigma_2}, A_1 \cap A_2, C_1 \cap C_2)$, and such that for every $a \in A_1 \cap A_2$, $X_a^{12} = X_a^1 \cap X_2 = X_a^2 \cap X_1 = X_a^S \cap X_1 \cap X_2$, and $Tr_a^{12} = \{(s_{1|X_a^{12}}, s_{2|X_a^{12}}) \mid s_1, s_2 \in St(S_1), (s_{1|X_a^1}, s_{2|X_a^1}) \in Tr_a^{S_1}\} \cup \{(s_{1|X_a^{12}}, s_{2|X_a^{12}}) \mid s_1, s_2 \in St(S_2), (s_{1|X_a^2}, s_{2|X_a^2}) \in Tr_a^{S_2}\}$.

It is easy to see that S_{12} is a transition-connected subsystem of both S_1 and S_2. As an example, we prove it for S_1.

Let $a \in A_1$ be such that $X_a^1 \cap X_1 \cap X_2 \neq \emptyset$. Then $X_a^1 \cap X_2 \neq \emptyset$, hence $X_a^S \cap X_2 \neq \emptyset$. Since $S_2 \hookrightarrow S$ is transition-connected it follows that $a \in A_2$. Thus, $a \in A_1 \cap A_2$.

If $a \in A_1 \cap A_2$, then $X_a^{12} = X_a^1 \cap X_1 \cap X_2 = X_a^1 \cap X_{12}$, and if $s_1, s_2 \in St(S_1)$ with $(s_{1|X_a^1}, s_{2|X_a^1}) \in Tr_a^{S_1}$, then by the definition of Tr_a^{12}, $(s_{1|X_a^{12}}, s_{2|X_a^{12}}) \in Tr_a^{12}$. This shows that S_{12} is a transition-connected subsystem of S_1. Analogously it can be shown that S_{12} is a transition-connected subsystem of S_2.

In order to check that S_{12} is indeed the pullback we show that it satisfies the universality property of a pullback: Let T be a transition-connected subsystem of both S_1 and S_2. We prove that T is a transition-connected subsystem of S_{12}.

It is easy to see that

$$\Sigma_T \subseteq \Sigma_1 \cap \Sigma_2, X_T \subseteq X_1 \cap X_2 = X_{12}, \Gamma_T \subseteq \Gamma_1 \cap \Gamma_2 = \Gamma_{12},$$

$$A_T \subseteq A_1 \cap A_2 = A_{12}, C_T \subseteq C_1 \cap C_2 = C_{12}.$$

Furthermore, we know that for every $a \in A_1$, if $X_a^1 \cap X_T \neq \emptyset$ then $a \in A_1 \cap A_T$ and $X_a^T = X_a^1 \cap X_T$ (and similarly for A_2). Hence, if $X_a^{12} \cap X_T \neq \emptyset$ then $a \in A_1 \cap A_2 \cap A_T = A_{12} \cap A_T$, and $X_a^T = X_a^1 \cap X_T = X_a^2 \cap X_T = X_a^1 \cap X_a^2 \cap X_T = X_a^{12} \cap X_T$.

We also know that if $s_1, s_2 \in St(S_i)$, such that $(s_{1|X_a^i}, s_{2|X_a^i}) \in Tr_a^{S_i}$, (for $i = 1$ or $i = 2$), then $(s_{1|X_a^T}, s_{2|X_a^T}) \in Tr_a^T$.

Let now $a \in A_T$ and $s_1, s_2 \in St(S_{12})$ such that $(s_{1|X_a^{12}}, s_{2|X_a^{12}}) \in Tr_a^{12}$. By the definition of Tr_a^{12}, there are either $\overline{s}_1, \overline{s}_2 \in St(S_1)$ with $(\overline{s}_{1|X_a^1}, \overline{s}_{2|X_a^1}) \in Tr_a^{S_1}$, or $\overline{s}_1, \overline{s}_2 \in St(S_2)$ with $(\overline{s}_{1|X_a^2}, \overline{s}_{2|X_a^2}) \in Tr_a^{S_2}$, such that $s_1 = \overline{s}_{1|X_{12}}, s_2 = \overline{s}_{2|X_{12}}$. In both situations it follows that $(\overline{s}_{1|X_a^T}, \overline{s}_{2|X_a^T}) \in Tr_a^T$. This proves that T is a transition-connected subsystem of S_{12}, and thus that S_{12} is the pullback of Diagram 7.3. □

Proposition 7.11 *Let* $S = (\Sigma, X, M, \Gamma, A, C)$ *be a system and* $\{S_i \hookrightarrow S \mid i \in I\}$ *a family of transition-connected subsystems of* S, *where for every* $i \in I$, $S_i = (\Sigma_i, X_i, M_i, \Gamma_i, A_i, C_i)$. *The colimit of this family in* $\mathsf{SYS}_{\mathsf{il}}$ *is the system* $\overline{S}$ *with* $\Sigma_{\overline{S}} = \bigcup_{i \in I} \Sigma_i$, $X_{\overline{S}} = \bigcup_{i \in I} X_i$, $M_{\overline{S}} = M_{|\bigcup_{i \in I} \Sigma_i}$, $\Gamma_{\overline{S}} = (\bigcup_{i \in I} \Gamma_i)^\bullet$, $A_{\overline{S}} = \bigcup_{i \in I} A_i$, $C_{\overline{S}} = (\bigcup_{i \in I} C_i)^\bullet$, *and where for every* $a \in \bigcup_{i \in I} A_i$ $X_a^{\overline{S}} = \bigcup_{a \in A_i} X_a^i$, *and* $Tr_a^{\overline{S}} = \{(s_{1|X_a^{\overline{S}}}, s_{2|X_a^{\overline{S}}}) \mid s_1, s_2 \in St(\overline{S})$, *and for every* $i \in I$ *with* $a \in A_i, (s_{1|X_a^i}, s_{2|X_a^i}) \in Tr_a^{S_i}\}$.

Proof: In order to see that for every $i \in I$, S_i is a transition-connected subsystem of $\overline{S}$, note that if $a \in A_{\overline{S}}$ depends on $x \in X_i$ for some $i \in I$ it follows that $a \in A_i$ (because $a \in A_{\overline{S}} \subseteq A$, using the fact that S_i is a transition-connected subsystem of S). Moreover, if $a \in A_i$ then $X_a^{\overline{S}} \cap X_i = \left(\bigcup_{a \in A_j} X_a^j\right) \cap X_i = X_a^i$ (in order to prove the last equality note that, on the one hand, it is obvious that $X_a^i \subseteq \left(\bigcup_{a \in A_j} X_a^j\right) \cap X_i$, and on the other hand, since for every $j \in I$, S_j is a transition-connected subsystem of S, it follows that if $a \in A_j$

then $X_a^j = X_a^S \cap X_j \subseteq X_a^S$; hence, $\bigcup_{a \in A_j} X_a^j \subseteq X_a^S$, and $\left(\bigcup_{a \in A_j} X_a^j\right) \cap X_i \subseteq X_a^S \cap X_i = X_a^i$).

Let $a \in A_i$ and $s_1, s_2 \in St(\overline{S})$ be such that $(s_{1_{|X_a^{\overline{S}}}}, s_{2_{|X_a^{\overline{S}}}}) \in Tr_a^{\overline{S}}$. By the definition of $Tr_a^{\overline{S}}$ it follows that $(s_{1|X_a^i}, s_{2|X_a^i}) \in Tr_a^{S_i}$. Thus, S_i is a transition-connected subsystem of $\overline{S}$ for every $i \in I$.

In order to show that $\overline{S}$ satisfies the universality property of a colimit, let T be such that for every $i \in I$, S_i is a transition-connected subsystem of T. Then $\bigcup_{i \in I} \Sigma_i \subseteq \Sigma_T$, $\bigcup_{i \in I} X_i \subseteq X_T$, $\bigcup_{i \in I} A_i \subseteq A_T$, and the constraints in $\bigcup_{i \in I} \Gamma_i$ (resp. $\bigcup_{i \in I} C_i$) are consequences of the constraints in Γ_T (resp. C_T).

We want to show that in this case $\overline{S}$ is a transition-connected subsystem of T.

Let $a \in A_T$ be such that $X_a^T \cap X_{\overline{S}} \neq \emptyset$. Then $X_a^T \cap X_{S_i} \neq \emptyset$ for some $i \in I$, and since S_i is a transition-connected subsystem of T, it follows that $a \in A_i \subseteq \bigcup_{i \in I} A_i = A_{\overline{S}}$. Moreover, $X_a^T \cap X_{\overline{S}} = X_a^T \cap (\bigcup_{i \in I} X_i) = \bigcup_{i \in I} \left(X_a^T \cap X_i\right) = \bigcup_{i \in I : a \in A_i} X_a^i = X_a^{\overline{S}}$.

Let $s_1, s_2 \in St(T)$, be such that $(s_{1|X_a^T}, s_{2|X_a^T}) \in Tr_a^T$. Then it follows that $(s_{1|X_a^i}, s_{2|X_a^i}) \in Tr_a^{S_i}$ for every $i \in I$ with $a \in A_i$, hence, by the definition of $Tr^{\overline{S}}$, $(s_{1_{|X_a^{\overline{S}}}}, s_{2_{|X_a^{\overline{S}}}}) \in Tr_a^{\overline{S}}$. Thus, $\overline{S}$ is a transition-connected subsystem of T. $\square$

7.2.2 A Grothendieck Topology on $\mathsf{SYS}_{\mathsf{il}}$

In order to define a covering relation on $\mathsf{SYS}_{\mathsf{il}}$ we have to request additionally that if a transition is "locally" defined on elements of a cover then from this "local" transitions a "global" transition can be constructed. In what follows, if not explicitely specified otherwise, all morphisms $S_1 \hookrightarrow S_2$ will be supposed to be transition-connected.

Definition 7.7 (Covering Family) *A family $\mathcal{S} = \{S_i \hookrightarrow S \mid i \in I\}$ of transition-connected subsystems of S is a* covering family *for S iff it has the following properties:*

(C1) $\mathcal{S}$ is a sieve (i.e. it is closed under all transition-connected subsystems),

(C2) S is a colimit in $\mathsf{SYS}_{\mathsf{il}}$ of the diagram defined by the sieve $\mathcal{S}$,

(C3) For every transition-connected subsystem $T \hookrightarrow S$, T is the colimit in $\mathsf{SYS}_{\mathsf{il}}$ of the diagram defined by the sieve $T \cap \mathcal{S} = \{S_i \in \mathcal{S} \mid s_i \hookrightarrow T\}$.

As in Proposition 7.4, it is easy to check that this notion of covering family induces a Grothendieck topology on $\mathsf{SYS}_{\mathsf{il}}$.

Proposition 7.12 *The following holds:*

(1) The function J assigning for every system S the set $J(S)$ of all covering families for S is a Grothendieck topology on $\mathsf{SYS}_{\mathsf{il}}$.

(2) *The functors* $St, Act : \mathsf{SYS_i}^{op} \to \mathsf{Sets}$ *are sheaves with respect to the Grothendieck topology* J.

Proof: (1) The proof closely follows the proof given in the case of $\mathsf{SYS_i}$. The first condition in the definition of a Grothendieck topology is obviously satisfied. The proof of the second condition is the same as in the case of $\mathsf{SYS_i}$. In order to see that the third condition is also satisfied, it only remains to check that if $\mathcal{S} = \{S_i \mid i \in I\}$ is a cover for S and $\mathcal{R}$ a sieve on S such that for every $i \in I$, $S_i \cap \mathcal{R} = \{S_{ij} \mid j \in J_i\}$ covers S_i then:

(i) for every $a \in A_S$, $X_a^S = \bigcup_{a \in A_{S_{ij}}} X_a^{S_{ij}}$, and

(ii) if $a \in A_s$ and $s_1, s_2 \in St(S)$ are such that $(s_{1|X_{ij} \cap X_a^S}, s_{2|X_{ij} \cap X_a^S}) \in Tr_a^{S_{ij}}$ for all S_{ij} with $a \in A_{S_{ij}}$, then $(s_{1|X_a^S}, s_{2|X_a^S}) \in Tr_a^S$.

For the first part, note that $X_a^S = \bigcup_{i, a \in A_{S_i}} X_a^{S_i} = \bigcup_{i, a \in A_{S_i}} \bigcup_{j, a \in A_{S_{ij}}} X_a^{S_{ij}} = \bigcup_{i,j, a \in A_{S_{ij}}} X_a^{S_{ij}}$.

For the second part, assume that $a \in A_s$ and $s_1, s_2 \in St(S)$ are such that $(s_{1|X_{ij} \cap X_a^S}, s_{2|X_{ij} \cap X_a^S}) \in Tr_a^{S_{ij}}$ for all S_{ij} with $a \in A_{S_{ij}}$. Note that for every i, j, $X_{ij} \cap X_a^S = X_s^{S_{ij}} = X_{ij} \cap X_a^{S_i}$. Since $\{S_{ij} \mid j \in I_i\} \in J(S_i)$, it follows that if $(s_{1|X_{ij} \cap X_a^S}, s_{2|X_{ij} \cap X_a^S}) \in Tr_a^{S_{ij}}$ for all $i \in I$ and $j \in J_i$ with $a \in A_{S_{ij}}$, then $(s_{1|X_i \cap X_a^S}, s_{2|X_i \cap X_a^S}) \in Tr_a^{S_i}$ for all $i \in I$ such that $a \in A_{S_i}$. From the fact that $\{S_i \mid i \in I\} \in J(S)$ it then follows $(s_{1|X_a^S}, s_{2|X_a^S}) \in Tr_a^S$.

(2) Follows immediately, since the covers in $\mathsf{SYS_{il}}$ are in particular covers in $\mathsf{SYS_i}$. □

7.2.3 Transitions within $\mathsf{SYS_{il}}$

We now study the relationships between transitions in the elements of a covering family for a system S and the transitions in S.

For atomic actions the following result holds.

Lemma 7.13 *Let* $\{S_i \mid i \in I\}$ *be a covering family for* S. *Let* $a \in A_S$, *and* $s_1, s_2 \in St(S)$ *be such that* $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(a)$ *for every* $i \in I$. *Then* $(s_1, s_2) \in Tr^S(a)$.

Proof: Let $s_1, s_2 \in St(S)$ be such that $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(a)$ for every $i \in I$. Then, for every $i \in I$ with $a \in A_i$, $(s_{1|X_i \cap X_a^S}, s_{2|X_i \cap X_a^S}) \in Tr_a^{S_i}$ and $s_1(x) = s_2(x)$ for all $x \in X_i \backslash X_a^S$. For every $i \in I$ with $a \notin A_i$, $s_{1|X_i} = s_{2|X_i}$.

Since for every $i \in I$ with $a \in A_i$, $(s_{1|X_i \cap X_a^S}, s_{2|X_i \cap X_a^S}) \in Tr_a^{S_i}$, it follows (by $(C2)$) that $(s_{1|X_a^S}, s_{2|X_a^S}) \in Tr_a^S$. If $x \in X_S \backslash X_a^S$ then $x \in X_i \backslash X_a^S$ for some $i \in I$, hence $s_1(x) = s_2(x)$. □

Proposition 7.14 shows that a similar gluing property also holds for parallel actions, if transitions of composed actions satisfy **(Gluing)**.

Proposition 7.14 *Assume that the transitions associated to admissible parallel actions satisfy* **(Gluing)**. *Let* $\mathcal{S} = \{S_i \mid i \in I\}$ *be a cover for* S *in* $\mathsf{SYS}_{\mathsf{il}}$. *Let* $s_1, s_2 \in St(S)$ *and* $f \in Act(S)$. *Assume that* $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(f_{|A_i})$ *for every* $i \in I$. *Then* $(s_1, s_2) \in Tr^S(f)$.

Proof: Let $s_1, s_2 \in St(S)$ and $f \in Act(S)$, and assume that $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(f_{|A_i})$ for every $i \in I$.

Let $a \in A_S$ with $f(a) = 1$. Then $(s_{1|X_a^S \cap X_i}, s_{2|X_a^S \cap X_i}) \in Tr_a^{S_i}$ for all $i \in I$ with $a \in A_i$. Hence, by the property of a covering family, $(s_{1|X_a^S}, s_{2|X_a^S}) \in Tr_a^S$. For $x \notin \bigcup_{a, f(a)=1} X_a^S$ we know that $s_{1|X_i}(x) = s_{2|X_i}(x)$ if $x \in X_i$, hence $s_1(x) = s_2(x)$. Thus, by **(Gluing)**, $(s_1, s_2) \in Tr^S(f)$. □

If we assume that the actions may consume common resources and are deterministic, and the transitions of parallel actions are obtained according to the **(Independence)**, then a similar results holds, but under stronger conditions.

Lemma 7.15 *Let* $\{S_i \mid i \in I\}$ *be a covering family for* S. *Assume that the actions in* S *and* $\{S_i \mid i \in I\}$ *are deterministic (the final state is uniquely determined by the initial state in case an action is applied).*

Let $a \in A_S$ *and let* $s \in St(S)$ *be such that for every* $i \in I$ *there exists a state* $s_i \in St(S_i)$ *such that* $(s_{|X_i}, s_i) \in Tr^{S_i}(a)$ *(i.e. if* $a \notin A_i$ *then* $s_{|X_i} = s_i$, *otherwise* $(s_{|X_i \cap X_a^S}, s_{i X_i \cap X_a^S}) \in Tr_a^{S_i}$ *and* $s(x) = s_i(x)$ *if* $x \in X_i \backslash X_a^S$*). Then there is a unique state* $\overline{s} \in St(S)$ *such that for all* $i \in I$, $\overline{s}_{|X_i} = s_i$ *and* $(s, \overline{s}) \in Tr^S(a)$.

Proof: By the determinism of actions and by the fact that $\{S_i \mid i \in I\}$ is a sieve (hence, closed under pullbacks ("intersections") of systems), it follows that for every $i, j \in I$, $s_{i|X_i \cap X_j} = s_{j|X_i \cap X_j}$. Therefore there is a unique $\overline{s} \in St(S)$ such that $\overline{s}_{|X_i} = s_i$. It follows that $(s_{|X_i}, \overline{s}_{|X_i}) \in Tr^i(a)$ for every $i \in I$, hence by Lemma 7.13, $(s, \overline{s}) \in Tr(a)$. □

Remark: Note that in proving the previous lemma we used the fact that the covering family $\mathcal{S}$ for S has the property that for every $S_1, S_2 \in \mathcal{S}$, the pullback $S_1 \times_S S_2$ is a transition-connected subsystem of both S_1 and S_2, hence is in $\mathcal{S}$. This is true in $\mathsf{SYS}_{\mathsf{il}}$, but might not be true in some of its subcategories, if they are not closed under pullbacks. In Section 8 we will show that a similar result holds for a certain subcategory of $\mathsf{SYS}_{\mathsf{il}}$ which is not closed under pullbacks.

The next proposition shows that a gluing condition holds, similar to that of Proposition 7.14, under the assumption that the transitions of parallel actions satisfy **(Independence)**. Since one of the conditions in this case is that for every $a \in A_2$, $f^{-1}(a)$ is finite, we assume, for the sake of simplicity, that the set of atomic actions of the system S is *finite*. Since we decided to consider only finite systems, this is not a limitation.

Proposition 7.16 *Let* $\mathcal{S} = \{S_1, \ldots, S_n\}$ *be a covering family for* S *in* $\mathsf{SYS}_{\mathsf{il}}$. *Assume that the transitions associated to admissible parallel actions obey* **(Independence)**. *Let* $s_1, s_2 \in St(S)$ *and* $f \in Act(S)$. *Assume that* $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(f_{|A_i})$ *for every* $i \in I$. *Then* $(s_1, s_2) \in Tr^S(f)$.

Proof: Assume that we have identified all elements $a_1, a_2 \in A_S$ with $a_1 = a_2 \in C_S$ and $f(a_1) = f(a_2) = 1$, and after this identification, $f^{-1}(1) = \{a_1, \dots, a_n\}$. We proceed by induction on the number n of elements in $f^{-1}(1)$ after this identification.

If $n = 1$ then f consists of only one action, and the property is true by Lemma 7.13.

Assume that the property is true for every admissible action g such that (after identifying all elements $a_1, a_2 \in A_S$ with $a_1 = a_2 \in C_S$ and $g(a_1) = g(a_2) = 1$), the set $g^{-1}(1)$ has $n-1$ elements.

Let $f \in Act(S)$ with $f^{-1}(1) = \{a_1, \dots, a_n\}$. Since all possible relations between the actions in A_S are of the form $a \wedge a' = 0$, it follows that also the parallel action $g : A \to \{0, 1\}$ with $g(a) = 1$ iff $a \in \{a_2, \dots, a_n\}$ is admissible, i.e. $g \in Act(S)$, and so are all restrictions of g to elements in the covering family $g_{|A_i}$, for any $i \in I$.

We know that $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(f_{|A_i})$ for every $i \in I$. By **(Independence)** it follows that for every $i \in I$, $f_{|A_i}$ can be applied at $s_{1|X_i}$ in the system S_i, and the final state does not depend on the order in which the actions are applied. Therefore we can assume that a_1 is applied first in all systems S_i with $a_1 \in A_i$.

Hence, for every $i \in I$ there exists a state $s_i \in St(S_i)$ such that $(s_{1|X_i}, s_i) \in Tr^{S_i}(a_1)$ if $a_1 \in A_i$, or such that $s_{1|X_i} = s_i$ if $a_1 \notin A_i$. It also follows that $g_{|A_i}$ can be applied at s_i for every $i \in I$ (note that the final state does not depend on the order in which the actions are applied), i.e. that $(s_i, s_{2|X_i}) \in Tr^S(g_{|A_i})$ for all $i \in I$.

By Lemma 7.15 and by the determinism of parallel actions guaranteed by **(Independence)** it follows that there is a unique state $\overline{s} \in St(S)$ such that $\overline{s}_{|X_i} = s_i$ and $(s_1, \overline{s}) \in Tr^S(a)$. Moreover, for every $i \in I$, $(\overline{s}_{|X_i}, s_{2|X_i}) \in Tr^{S_i}(g_{|A_i})$. Therefore, by the induction hypothesis, $(\overline{s}, s_2) \in Tr^S(g)$. This proves that $(s_1, s_2) \in Tr^S(f)$. □

Note again that, since in the proof we used Lemma 7.15, Proposition 7.16 only holds if all covering families are closed under intersections (pullbacks). Therefore, it may not remain valid in subcategories in $\mathsf{SYS}_{\mathsf{il}}$ that are not closed under pullbacks. In Section 8 we will show that for a particular such subcategory, $\mathsf{Sys}(\mathsf{InSys})$, the result is still true.

In what follows we assume that the transitions of parallel actions are computed by **(Gluing)** or resp. **(Independence)** and show that transitions can be expressed by natural transformations between the sheaves Act and $\Omega^{St \times St}$ over the site $(\mathsf{SYS}_{\mathsf{il}}, J)$ or resp. $(\mathsf{SYS}^{\mathsf{f}}_{\mathsf{il}}, J)$, where $\mathsf{SYS}^{\mathsf{f}}_{\mathsf{il}}$ is the full subcategory of $\mathsf{SYS}_{\mathsf{il}}$ having as objects *finite* systems.

In what follows, $\mathsf{SYS}^{*}_{\mathsf{il}}$ will denote either $\mathsf{SYS}_{\mathsf{il}}$, if the transitions of parallel actions are computed by **(Gluing)**, or $(\mathsf{SYS}^{\mathsf{f}}_{\mathsf{il}}, J)$ in case the transitions of parallel actions obey **(Independence)**.

Lemma 7.17 *Let* $S \in \mathsf{SYS}^{*}_{\mathsf{il}}$. *Let* $f \in Act(S)$ *be an admissible parallel action,* $g : S' \hookrightarrow S$ *a transition-connected subsystem, and* $s_1, s_2 \in St(S')$. *Then the* *set* $Tr_S(f)(S')(g, s_1, s_2) = \{S'' \overset{h}{\hookrightarrow} S' \mid (s_{1|X''}, s_{2|X''}) \in Tr^{S''}(f \circ g_A \circ h_A)\}$ *is a closed sieve on* S'.

Proof: Let $S'' \overset{h}{\hookrightarrow} S' \in Tr_S(f)(S')(g, s_1, s_2)$, and let $\overline{S}'' \overset{h'}{\hookrightarrow} S'' \overset{h}{\hookrightarrow} S'$. We want to show that $\overline{S}'' \overset{h'}{\hookrightarrow} S'' \overset{h}{\hookrightarrow} S' \in Tr_S(f)(S')(g, s_1, s_2)$.

Since $S'' \overset{h}{\hookrightarrow} S' \in Tr_S(f)(S')(g, s_1, s_2)$ it follows that $(s_{1|X''}, s_{2|X''}) \in Tr^{S''}(f \circ g_A \circ h_A)$. Therefore, by Proposition 6.5 resp. Proposition 6.6, $(s_{1|X_{\overline{S}''}}, s_{2|X_{\overline{S}''}}) \in Tr^{\overline{S}''}(f \circ g_A \circ (h_A \circ h'_A))$. Hence, $\overline{S}'' \overset{h \circ h'}{\hookrightarrow} S' \in Tr_S(f)(S')(g, s_1, s_2)$. This proves that $Tr_S(f)(S')(g, s_1, s_2)$ is a sieve on S'.

In order to prove that it is a closed sieve, let $\overline{S} \overset{h}{\hookrightarrow} S$ and assume that $\{\overline{S}_i \overset{h_i}{\hookrightarrow} \overline{S} \overset{h}{\hookrightarrow} S \mid i \in I\}$ covers $\overline{S}$, where for every $i \in I$, $(s_{1|X_{\overline{S}_i}}, s_{2|X_{\overline{S}_i}}) \in Tr^{\overline{S}_i}(f \circ g_A \circ h_A \circ h_{iA})\}$. By Proposition 7.16 it then follows that $(s_{1|X_{\overline{S}}}, s_{2|X_{\overline{S}}}) \in Tr^{\overline{S}}(f \circ g_A \circ h_A)$, i.e. $\overline{S} \in Tr_S(f)(S')(g, s_1, s_2)$. Hence, $Tr_S(f)(S')(g, s_1, s_2)$ is a closed sieve. □

A proof analogous to the one given in Lemma 6.8 leads to the following result.

Lemma 7.18 $Tr_S(f) : y(S) \times St \times St \to \Omega$, *defined, for every system* S', *by* $Tr_S(f)(S') : \mathsf{Hom}_{\mathsf{SYS}^*_{\mathsf{il}}}(S', S) \times St(S') \times St(S') \to \Omega(S')$, *where for every* $g : S' \to S$ *and* $s_1, s_2 \in St(S')$, $Tr_S(f)(S')(g, s_1, s_2) = \{S'' \overset{h}{\to} S' \mid (s_1 \circ h_X, s_2 \circ h_X) \in Tr^{S''}(f \circ g_a \circ h_a)\}$, *is a natural transformation.*

Consequence 7.19 *For every system* S *and* $f \in Act(S)$, $Tr_S(f) \in \Omega^{St \times St}(S)$.

Proof: $\Omega^{St \times St}$ is defined in $\mathsf{Sh}(\mathsf{SYS}^*_{\mathsf{il}})$ as follows: for every $S \in \mathsf{SYS}^*_{\mathsf{il}}$, $\Omega^{St \times St}(S) = \{\tau : y(S) \times St \times St \to \Omega \mid \tau \text{ natural transformation}\}$.

In Lemma 7.18 we showed that $Tr_S(f) : y(S) \times St \times St \to \Omega$ is a natural transformation. Hence, $Tr_S(f) \in \Omega^{St \times St}(S)$. □

Proposition 7.20 $Tr : Act \to \Omega^{St \times St}$ *defined for every system* S *by* $Tr_S : Act(S) \to \Omega^{St \times St}(S)$ *is a natural transformation in* $\mathsf{Sh}(\mathsf{SYS}^*_{\mathsf{il}})$.

Proof: We have to show that for every $i : S_1 \hookrightarrow S_2$, where S_1 is a transition-connected subsystem of S_2, the following diagram is commutative:

$$\begin{tikzcd} Act(S_2) \arrow[r, "Tr_{S_2}"] \arrow[d, "Act(i)"'] & \Omega^{St \times St}(S_2) \arrow[d, "\Omega^{St \times St}(i)"] \\ Act(S_1) \arrow[r, "Tr_{S_1}"] & \Omega^{St \times St}(S_1) \end{tikzcd} \tag{7.4}$$

Let $f \in Act(S_2)$. Then $\Omega^{St \times St}(i)(Tr_{S_2}(f)) : y(S_1) \times St \times St \to \Omega$ is defined for every $g : \overline{S}_1 \hookrightarrow S_1$ and $s_1, s_2 \in \overline{S}_1$ by

$\Omega^{St \times St}(i)(Tr_{S_2}(f))(\overline{S}_1)(g, s_1, s_2) = Tr_{S_2}(f)(\overline{S}_1)(i \circ g, s_1, s_2) = \{S'' \overset{h}{\hookrightarrow} \overline{S}_1 \mid (s_{1|X''}, s_{2|X''}) \in Tr^{S''}(f \circ (i_A \circ g_A) \circ h_A)\}$.

On the other hand, $Tr_{S_1}(f_{|A_1})$ is defined for every $g : \overline{S}_1 \hookrightarrow S_1$ and $s_1, s_2 \in \overline{S}_1$ by

$$Tr_{S_1}(f_{|A_1})(\overline{S}_1) = \{S'' \overset{h}{\hookrightarrow} \overline{S}_1 \mid (s_{1|X''}, s_{2|X''}) \in Tr^{S''}((f \circ i_A) \circ g_A \circ h_A)\}.$$

This proves that the diagram commutes. □

It is also easy to see that $Tr : \mathsf{SYS}_{\mathrm{il}}^{*op} \to \mathsf{Sets}$ defined by

$$Tr(S) = \{(f, s, s') \mid f \in Act(S), s, s' \in St(S), (s, s') \in Tr^S(f)\}$$

is a subsheaf of $Act \times St \times St$ (a short direct proof of this fact can be given: from Proposition 6.5 resp. Proposition 6.6 it follows that it is a subpresheaf, and from Proposition 7.14 resp. Proposition 7.16 it follows that it is a sheaf).

7.2.4 Temporal Behavior of Systems in $\mathsf{SYS}_{\mathrm{il}}$

The starting point of our approach to dealing with temporal behavior is the formalism developed by J. Goguen in [Gog92]. He starts with the assumption that every system can be described by a set of attributes X, each attribute $x \in X$ having a prescribed set of values V_x. In what follows we assume that time is considered to be discrete. In this particular situation, in [Gog92] the behavior of a given system S in time is modeled by a functor $F : \mathcal{T}^{op} \to \mathsf{Sets}$, where $\mathcal{T}$ is the basis for the topology on $\mathbb{N}$ consisting of all the sets $\{0, 1, \dots, n\}, n \in \mathbb{N}$. Intuitively, for every open set $U = \{0, 1, \dots, n\}$, $F(U)$ represents the "observations" in the interval of time U. Formally, the functor F is defined on objects by $F(U) = \{h : U \to \prod_{p \in P} V_p \mid K(h)\}$ where $K(h)$ represents a set of conditions that have to be satisfied by h — usually some prescribed rules indicating how the states of the system can change, reflecting the pre- and postconditions of the relevant actions. It is defined on morphisms by $F(\iota_V^U)(h) = h_{|V}$ for every $\iota_V^U : V \hookrightarrow U$ and every $h \in F(U)$. In order to study the behavior of a system consisting of several subsystems the intercommunication between the subsystems is taken into account. A system is seen as a diagram of subsystems, where the morphisms represent inheritance. Goguen shows that the behavior of the system can be described by $F(U) = \{\{h_i \mid i \in I\} \mid h_i \in F_i(U) \text{ and if } \phi_e : S_i \to S_{i'} \text{ then } \phi_e(h_i) = h_{i'}\}$, where for every $i \in I$, F_i is a sheaf that describes the behavior of the system S_i. Therefore, the behavior of a system is the limit of the behaviors of its subsystems (for details see [Gog92]).

In what follows we will develop the idea of representing behavior in time of systems by sheaves over time. We modify the definition for the behavior of a system slightly, by also taking into account the actions that are performed at every step. We will assume that all actions need one unit of time. In future work the more realistic case where actions can have different durations will be considered.

Definition 7.8 *Let S be a system in $\mathsf{SYS}_{\mathrm{il}}$. The* behavior *of S is a functor $B_S : \mathcal{T}^{op} \to \mathsf{Sets}$ defined for every $U \in \mathcal{T}$ by $B_S(U) = \{h : U \to St(S) \times Act(S) \mid K(h, U)\}$, where $K(h, U)$ can be expressed by* for every n, if $n, n+1 \in U$ and $h(n) = (s, f)$, $h(n+1) = (s', f')$ then $(s, s') \in Tr(f)$,

and for every $\iota : V \subseteq U$, $B_S(\iota) : B_S(U) \to B_S(V)$ *is the restriction to* V.

Thanks to the particular form of the open sets of $\mathcal{T}$ (all $\{0, 1, \ldots, n\}$ for some n), it can easily be shown that B_S is a sheaf.

Let $\iota : S_1 \hookrightarrow S_2$ in $\mathsf{SYS_{il}}$. We define $\rho^{S_2}_{S_1} : B_{S_2} \to B_{S_1}$ by $\rho^{S_2}_{S_1}(U) : B_{S_2}(U) \to B_{S_1}(U)$ for every $U \in \mathcal{T}$, where for every $h : U \to St(S_2) \times Act(S_2)$, $\rho^{S_2}_{S_1}(U)(h) = \langle St(\iota), Act(\iota)\rangle \circ h : U \to St(S_1) \times Act(S_1)$ (with $St(\iota)(s) = s_{|X_1}$ for every $s \in St(S_2)$ and $Act(\iota)(f) = f_{|A_1}$). In what follows, for every $U \in \mathcal{T}$ and every $h \in B_{S_1}(U)$, we will abbreviate $\rho^{S_2}_{S_1}(U)(h)$ by $h_{|S_1}$.

($\mathsf{SYS^*_{il}}$ will denote either $\mathsf{SYS_{il}}$, if the transitions of parallel actions are computed by **(Gluing)**, or $(\mathsf{SYS^f_{il}}, J)$ in case the transitions of parallel actions obey **(Independence)**.)

Lemma 7.21 *Let* $U \in \mathcal{T}$ *be arbitrary but fixed. Let* $B'_U : \mathsf{SYS^{*op}_{il}} \to \mathsf{Sets}$ *be defined for every object* $S \in \mathsf{SYS^*_{il}}$ *by* $B'_U(S) = B_S(U)$ *and for every morphism* $\iota : S_1 \hookrightarrow S_2$ *by* $B'_U(\iota) = \rho^{S_2}_{S_1} : B_{S_2}(U) \to B_{S_1}(U)$. *Then* B'_U *is a sheaf.*

Proof: Let $S \in \mathsf{SYS^*_{il}}$ and $\{S_j \hookrightarrow S \mid j \in J\}$ be a cover of S. Let $\{h_j\}_{j \in J}$ be such that for every $j \in J$, $h_j \in B_{S_j}(U)$ is a matching family, i.e. such that for every $j_1, j_2 \in J$, $h_{j_1 | S_{j_1} \times_S S_{j_2}} = h_{j_1 | S_{j_1} \times_S S_{j_2}}$. We show that there is a unique $h \in B_S(U)$ such that $h_{|S_j} = h_j$ for every $j \in J$.

From the definition of the behavior of a subsystem, for all $j \in J$ and every $t \in U$, $h_j(t) = (s^t_j, f^t_j)$, where $s^t_j \in St(S_j)$ and $f^t_j \in Act(S_j)$, and if $t, t+1 \in U$ then $(s^t_j, s^{t+1}_j) \in Tr(f^t_j)$.

The family $\{h_j \mid j \in J\}$ is compatible; therefore for every $t \in U$ and every $j_1, j_2 \in J$, $s^t_{j_1 | X_1 \cap X_2} = s^t_{j_2 | X_1 \cap X_2}$ and $f^t_{j_2 | A_1 \cap A_2} = f^t_{j_2 | A_1 \cap A_2}$. Since St and Act are sheaves, it follows that for every $t \in U$ there is a unique $s^t \in St(S)$ such that $s^t_{|X_j} = s^t_j$ for every $j \in J$, and a unique $f^t \in Act(S)$ such that $f^t_{|A_j} = f^t_j$ for every $j \in J$. Define $h : U \to St(S) \times Act(S)$ by $h(t) = (s^t, f^t)$ for every $t \in U$. Note that if $t, t+1 \in U$, then $(s^t_j, s^{t+1}_j) \in Tr(f^t_j)$ for every $j \in J$, hence $(s^t_{|X_j}, s^{t+1}_{|X_j}) \in Tr(f^t_{|A_j})$ for every $j \in J$. By Proposition 7.14 or Proposition 7.16 (depending on the rule which is applied for the computation of transitions of parallel actions), $(s^t, s^{t+1}) \in Tr(f^t)$. It follows that h satisfies also the conditions $K(f)$. Then $h \in B(S)$ and for every $j \in J$, $h_{|S_j} = h_j$. □

Proposition 7.22 *Let* $B : \mathsf{SYS^{*op}_{il}} \to \mathsf{Sh}(\mathcal{T})$ *be defined for every object* S *of* $\mathsf{SYS^*_{il}}$ *by* $B(S) = B_S : \mathcal{T}^{op} \to \mathsf{Sets}$, *and for every morphism* $\iota : S_1 \hookrightarrow S_2$ *by* $B(i) = \rho^{S_2}_{S_1} : B(S_2) \to B(S_1)$, *and let* $B' : \mathcal{T}^{op} \to \mathsf{Sh}(\mathsf{SYS^*_{il}})$ *be defined for every* $U \in \mathcal{T}$ *by* $B'(U) : \mathsf{SYS^{*op}_{il}} \to \mathsf{Sets}$, $B'(U)(S) = B_S(U)$. *Then* B *and* B' *are functors.*

Proof: By the fact that all morphisms in $\mathsf{SYS^*_{il}}$ are transition-connected inclusions and from Proposition 6.5 resp. 6.6 (depending of the rule for computing transitions of parallel actions) it follows that if $S_1 \hookrightarrow S_2$ is a morphism in $\mathsf{SYS}^*_{\mathsf{il}}$, then for every $h \in B_{S_2}(U)$, $\rho^{S_2}_{S_1}(h) \in B_{S_1}(U)$, i.e. that $\rho^{S_2}_{S_1}$ is well-defined. It is easy to see that $\rho^{S_2}_{S_1} : B_{S_2} \to B_{S_1}$ is a natural transformation. Let $V \subseteq U$, and let i^U_V be the inclusion of V in U. Then the following diagram commutes:

$$\begin{tikzcd} B_{S_2}(U) \arrow[r, "\rho^{S_2}_{S_1}(U)"] \arrow[d, "B_{S_2}(i^U_V)"'] & B_{S_1}(U) \arrow[d, "B_{S_1}(i^U_V)"] \\ B_{S_2}(V) \arrow[r, "\rho^{S_2}_{S_1}(V)"'] & B_{S_1}(V) \end{tikzcd} \tag{7.5}$$

Hence, B is a functor. In order to show that B' is a functor, note first that from Lemma 7.21 it follows that B' is well-defined on objects. Let $i : V \hookrightarrow U$ be the inclusion between the open sets $U, V \in \mathcal{T}$. Let us define $B'(i) : B'(U) \to B'(V)$ by $B'(i)(S) : B_S(U) \to B_S(V)$ by $B'(i)(S)(h) = B_S(i^U_V)(h) = h_{|V}$ for every $h : U \to St(S) \times Act(S) \in B_S(U)$. $B'(i)$ is a natural transformation between the sheaves $B'(U)$ and $B'(V)$. This follows from the commutativity of diagram 7.5. □

Proposition 7.22 suggests that it might be possible to define behavior as a sheaf $\mathsf{Bhv} : (\mathsf{SYS}^*_{\mathsf{il}} \times \mathcal{T})^{op} \to \mathsf{Sets}$ for a suitable Grothendieck topology on the product category $\mathsf{SYS}^*_{\mathsf{il}} \times \mathcal{T}$. One possible notion of covering is presented in what follows.

Definition 7.9 *A* covering family *for* $(S, U) \in |\mathsf{SYS}^*_{\mathsf{il}} \times \mathcal{T}|$ *is a family that contains a family of the form* $\{S_i \mid i \in I\} \times \{U\}$ *where* $\{S_i \mid i \in I\}$ *is a cover for* S *in* J.

Lemma 7.23 *The map that associates with every system* (S, U) *the family* $K(S, T)$ *of covering families for* (S, U) *in the sense of Definition 7.9, is a basis for a Grothendieck topology on* $\mathsf{SYS}^*_{\mathsf{il}} \times \mathcal{T}$.

Proof: (1) The first property from the definition of a basis for a Grothendieck topology is satisfied, since $\{(S, U)\}$ is a cover for (S, U) according to Definition 7.9.

(2) Let $(S', U') \hookrightarrow (S, U)$, and let $\mathcal{S} \in K(S)$ contain $\{S_i \mid i \in I\} \times \{U\}$. Then $\mathcal{S} \cap (S', U')$ contains $(\mathcal{S} \cap S') \times \{U \cap U'\}$. Therefore it is a cover for (S', U'), since $\mathcal{S} \cap S'$ is a cover of S'.

(3) Let $\mathcal{S} \in K(S, U)$. Assume that $\mathcal{S}$ contains the family $\{S_i \mid i \in I\} \times \{U\}$. For every $i \in I$, let $\mathcal{S}_i \in K(S_i, U)$. Assume that for every $i \in I$, $\mathcal{S}_i$ contains the family $(\{S_{ij} \mid j \in I_i\} \times \{U\})$.

Therefore, $\{T \mid T \in \mathcal{S}_i, i \in I\}$ contains $\{S_{ij} \mid i \in I, j \in I_i\} \times \{U\}$. We know that $\{S_{ij} \mid i \in I, j \in I_i\}$ is a cover for S; it therefore follows that $\{T \mid T \in \mathcal{S}_i, i \in I\}$ covers (S, U). □

Let J_{ST} be the Grothendieck topology generated by the basis K.

Proposition 7.24 *The functor* $B : (\mathsf{SYS}^*_{\mathsf{il}} \times \mathcal{T})^{op} \to \mathsf{Sets}$ *defined by* $B(S,U) = B_S(U)$ *is a sheaf with respect to the Grothendieck topology* J_{ST}*.*

Proof: Let (S,U) be an object in $\mathsf{SYS}^*_{\mathsf{il}} \times \mathcal{T}$ and let $\mathcal{S} = \{(S_i, U_i) \mid i \in I\} \in J_{ST}(S,U)$. Let $(h_i)_{i \in I}$ be such that for every $i \in I$, $h_i : U_i \to St(S_i) \times Act(S_i)$, and for every $i, j \in I$, the restriction to $U_i \cap U_j$ of $h_{i|S_i \times_S S_j}$ is equal to the restriction to $U_i \cap U_j$ of $h_{j|S_i \times_S S_j}$. We will show that there is a unique $h : U \to St(S) \times Act(S)$ such that for every $i \in I$, $h_{|(S_i,U_i)} = h_i$.

Since $\mathcal{S} \in J_{ST}$, it follows that it contains a family of the form $(\{S_k \mid k \in K\} \times \{U\})$ where $\{S_i \mid i \in I\}$ is a cover for S in J. We will denote the index in $\mathcal{S}$ of (S_k, U), $k \in K$, by i_k.

Since $(h_i)_{i \in I}$ is compatible, it follows that, in particular, its subfamily $(h_{i_k})_{k \in K}$ is compatible. Therefore, there is a unique $h : U \to St(S) \times St(S)$ such that $h_{S_{i_k}} = h_{i_k}$. From the compatibility of the family $(h_i)_{i \in I}$ it follows that $h_{|(S_i,U_i)} = h_i$ for every $i \in I$. □

Proposition 7.22 also suggests that the theory can be developed in two further directions:

(1) Regard the category of systems as an internal category in $\mathsf{Sh}(\mathcal{T})$. This might offer some generalizations to the study of systems that vary in time, and is left as a topic for future research.

(2) Regard $\mathcal{T}$ as an internal category in $\mathsf{Sh}(\mathsf{SYS}^*_{\mathsf{il}}, \mathsf{J})$. That is, regard every time interval U in $\mathcal{T}$ as a sheaf in $\mathsf{Sh}(\mathsf{SYS}^*_{\mathsf{il}}, \mathsf{J})$. This can easily be done if for every $U \in \mathcal{T}$ we consider the sheaf $\mathbb{U} : \mathsf{SYS}^{*op}_{\mathsf{il}} \to \mathsf{Sets}$ obtained by sheafification from the constant presheaf $\mathcal{U} : \mathsf{SYS}^{*op}_{\mathsf{il}} \to \mathsf{Sets}$, defined on objects by $\mathcal{U}(S) = U$, and for every $h : S_1 \hookrightarrow S_2$ by $\mathcal{U}(h) = Id_U$.

   This approach will be analyzed in Section 8.5.2 (in the particular case when only those systems are considered that arise by interconnecting a given family of systems), and used in order to express temporal properties of systems.

   We will show that the representation of time intervals as sheaves over $(\mathsf{SYS}^*_{\mathsf{il}}, J)$ in a natural way allows to have "different time cycles" for independent systems.

Now we focus on simpler models for the behavior of systems, namely those that only take the *actions* performed into account, and ignore the *states*.

7.2.5 Models for the Behavior of Systems: Monoids and Languages

In what follows we will assume that all the constraints on actions are of the form $a_i \wedge a_j = 0$ (the constraints state which actions are dependent and cannot be performed in parallel).

In the model proposed in Section 7.2.4, the behavior of a system was described by a functor $B_S : \mathcal{T}^{op} \to \mathsf{Sets}$, defined for every $U \in \mathcal{T}$ by $B_S(U) = \{h : U \to St(S) \times Act(S) \mid K(h, U)\}$, where $K(h, U)$ can be expressed by

> *for every n, if $n, n+1 \in U$ and $h(n) = (s, f)$, then $h(n+1) = (s', f')$ such that $(s, s') \in Tr(f)$,*

and for every $\iota : V \subseteq U$, $B_S(\iota) : B_S(U) \to B_S(V)$ is the restriction to V.

If we ignore the states of the system, then for every system S we can express the behavior of S by

$L_S = \{f_1 \dots f_n \mid n \in \mathbb{N}, f_i \in Act(S) : \forall i \in \{1, \dots, n\}, \exists h \in B_S(\{1, \dots, n\}), \exists s_i \in St(S)$, s.t. $\forall i \in \{1, \dots, n\}, h(i) = (s_i, f_i)\}$.

The elements of L_S are strings of elements in $Act(S)$.

For every system S, let $Act(S)^*$ be the free monoid freely generated by $Act(S)$, where the empty action $0 : A_S \to \{0, 1\}$, $0(a) = 0$ for every $a \in A_S$ is identified with the identity ε.

Let $S_1 \hookrightarrow S_2$ be a transition-connected subsystem. The restriction map $Act(S_2) \to Act(S_1)$ extends in a canonical way to a morphism of monoids,

$$Act(S_2)^* \to Act(S_1)^*.$$

Note that in general, if arbitrary constraints on actions are allowed, this canonical morphism of monoids is not necessarily surjective: since in S_2 more constraints may exist, not every parallel action that is allowed in S_1 is also allowed in S_2.

Lemma 7.25 *Assume that all the constraints on actions are of the form $a_i \wedge a_j = 0$. Let $S_1 \hookrightarrow S_2$. Then the following affirmations are equivalent:*

(1) For every action $f \in Act(S_1)$ there exists an action $\overline{f} \in Act(S_2)$ such that $\overline{f}_{|A_1} = f$.

(2) $C_1 = C_2 \cap F_B^2(A_1)$.

Proof: (1) $\Rightarrow$ (2) : Assume $C_1 \neq C_2 \cap F_B^2(A_1)$, i.e. there is a constraint $a_i \wedge a_j = 0 \in C_2$, with $a_i, a_j \in A_1$, but $a_i \wedge a_j = 0 \notin C_1$.

Let $f : A_1 \to \{0, 1\}$ such that $f(a_i) = f(a_j) = 1$. Then there is no $\overline{f} \in Act(S_2)$ such that $\overline{f}_{|A_1} = f_2$.

(2) $\Rightarrow$ (1) : Assume $C_1 = C_2 \cap F_B^2(A_1)$. Let $f \in Act(S_1)$. Let $\overline{f} : A_2 \to \{0, 1\}$ be defined by $\overline{f}(a) = f(a)$ for every $a \in A_1$ and $\overline{f}(a) = 0$ for every $a \in A_2 \backslash A_1$.

Let $a_i \wedge a_j = 0$ be a constraint in C_2. If $a_i, a_j \in A_1$, then $a_i \wedge a_j = 0 \in C_1$, hence $\overline{f}(a_i) \wedge \overline{f}(a_j) = f(a_i) \wedge f(a_j) = 0$.

If at least one of $a_i, a_j \notin A_1$ (say a_i) then $\overline{f}(a_i) = 0$, hence $\overline{f}(a_i) \wedge \overline{f}(a_j) = 0$. This proves that $\overline{f} \models C_2$. □

Let now S be a system and $\{S_i \mid i \in I\}$ a cover of S. For every $i \in I$, let $p_i : Act(S)^* \to Act(S_i)^*$ be the canonical morphism of monoids that extends the restriction map, and let $\theta_i = ker(p_i)$. Let $\theta = \bigwedge_{i \in I} \theta_i$.

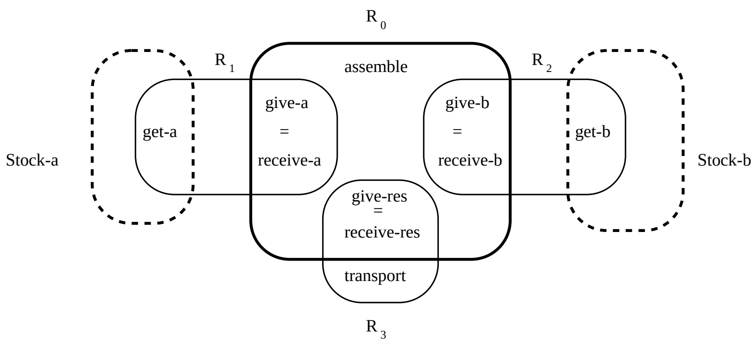


Figure 7.2: A simple robotics scenario

For every $i, j \in I$ such that $S_i \hookrightarrow S_j$ let $p_i^j : Act(S_j)^* \to Act(S_i)^*$ be the unique morphism of monoids that extends the restriction function from $Act(S_j)$ to $Act(S_j)$. Let $\mathbb{D}$ be the diagram defined by $\{Act(S_i) \mid i \in I\}$ with morphisms p_i^j for every $S_i \hookrightarrow S_j$. Then the limit of this diagram is

$$\varprojlim{}_{\mathbb{D}} Act(S_i)^* = \{(w_i)_{i \in I} \mid p_i^j(w_j) = w_i \text{ for every } i, j \in I \text{ s.t. } S_i \hookrightarrow S_j\}.$$

Then the canonical morphism $p : Act(S)^* \to \varprojlim{}_{\mathbb{D}} Act(S_i)^*$, defined by $p(f_1 \dots f_n) = (f_{1|A_i} \dots f_{n|A_i})_{i \in I}$, is such that $ker(p) = \theta$.

Remark: Note that the morphism $p : Act(S)^* \to \varprojlim Act(S_i)^*$ is not necessarily injective (in general $\theta \neq \Delta$), as can be easily seen from the following example (in what follows we will denote by a the parallel action consisting only of a):

Example 7.3 *Consider the example given in Section 6.1 and described in Figure 7.2. Let S the system covered by the sieve generated by $S_1, S_2, S_1 \cap S_2$. Let $f, g \in Act(S)^*$, $f =$ bring-a $\cdot$ bring-b, $g =$ bring-b $\cdot$ bring-a. Then $f_{|S_1} = g_{|S_1} =$ bring-a, $f_{|S_2} = g_{|S_2} =$ bring-b, and $f_{|S_1 \cap S_2} = g_{|S_1 \cap S_2} = \varepsilon$. Similarly, for any other system S' in the sieve generated by $S_1, S_2, S_1 \cap S_2$ we have $f_{|S'} = g_{|S'} = \varepsilon$. However, $f \neq g$.*

Remark: Note that the morphism $p : Act(S)^* \to \varprojlim_{i \in I} Act(S_i)$ is not necessarily surjective. There may be compatible families (even if we only consider singleton parallel actions) of sequences of actions that cannot be "glued together" to a sequence of actions on $Act(S)$, as can be seen in the following example:

Example 7.4 *Let S_1, S_2, S_3 be three systems all having the same language, the same constraints on variables and the same model for the variables, such that*

$$A_{S_1} = \{a, b\}, C_{S_1} = \{a \wedge b = 0\},$$

$$A_{S_2} = \{b, c\}, C_{S_2} = \{b \wedge c = 0\},$$

$$A_{S_3} = \{a, c\}, C_{S_3} = \{a \wedge c = 0\}.$$

Let S be the system obtained by interconnecting S_1, S_2, S_3. Then

$$A_S = \{a, b, c\}, C_S = \{a \wedge b = 0, b \wedge c = 0, a \wedge c = 0\}.$$

Consider $w_1 = ab \in Act(S_1)^*$, $w_2 = bc \in Act(S_2)^*$, $w_3 = ca \in Act(S_3)^*$.

It is easy to see that $p^1_{12}(w_1) = p^2_{12}(w_2) = b$, $p^2_{23}(w_2) = p^3_{23}(w_3) = c$, $p^1_{13}(w_1) = p^3_{13}(w_3) = a$, but there is no $w \in Act(S)^*$ such that $w_{|S_i} = w_i, i = 1, 2, 3$.

It follows that the (injective) map α,

$$Act(S)^*/\theta \overset{\alpha}{\hookrightarrow} \varprojlim Act(S_i)^* \hookrightarrow \prod_{i \in I} Act(S_i)^*.$$

is not necessarily surjective. This shows that the functor $Act^* : \mathsf{SYS}_{\mathsf{il}}{}^{op} \rightarrow \mathsf{Sets}$ defined by $Act^*(S) = Act(S)^*$, is in general not a sheaf: neither the existence nor the uniqueness of a "global" sequence of actions that extends a compatible family of "local" sequences of actions is guaranteed.

The phenomenon illustrated by Example 7.4 — namely the fact that there may exist "compatible" families of "local" sequences of actions that cannot be glued together to a "global" sequence of actions — has been studied in the (less general) context of asynchronous models for computations. In what follows we briefly present some results concerning modeling behavior in the asynchronous case, by using partially commutative monoids, and then make the link between our approach and such models. For the basic definitions cf. Section 4.3.1; for details see also [Die90].

Definition 7.10 *Let S be a system, with set of actions A_S and set of constraints C_S. Assume that all the constraints in C_S are of the type $a_i \wedge a_j = 0$.*

(1) Let $D_S \subseteq A_S \times A_S$ be defined by $(a, b) \in D_S$ iff $a \wedge b = 0 \in C_S$. The dependence alphabet $(A_S, D_S \cup \Delta_{A_S})$, denoted $D(S)$, is the dependence alphabet of S.

(2) The partially commutative monoid $M(D(S))$, denoted $M(S)$, is the partially commutative monoid of S.

(3) The graph (A_S, D_S), denoted $G(S)$, is the dependence graph of S.

Let $S_1 \hookrightarrow S_2$ in $\mathsf{SYS}_{\mathsf{il}}$. Then $A_{S_1} \subseteq A_{S_2}$ and $D_{S_1} \subseteq D_{S_2}$. By Theorem 4.21, there is a unique canonical projection, $p^2_1 : M(S_2) \rightarrow M(S_1)$ (which is surjective). The canonical projection p^2_1 is the unique morphism of (free) partially commutative monoids that extends the map $h : A_{S_2} \rightarrow A_{S_1}$ defined by $h(a) = \begin{cases} a & \text{if } a \in A_{S_1}, \\ \varepsilon & \text{if } a \notin A_{S_1}. \end{cases}$

Let S be a system and $\mathcal{S} = \{S_i \mid i \in I\}$ a covering family for S. For every $i \in I$, there is a canonical projection $p_i : M(S) \to M(S_i)$ (which is surjective). Moreover, if $S_i \hookrightarrow S_j$, then we denote the canonical projection by $p_i^j : M(S_j) \to M(S_i)$, and if $S_i, S_j \in \mathcal{S}$, then $p_{ij}^j : M(S_j) \to M(S_i \cap S_j)$, and $p_{ij}^i : M(S_i) \to M(S_i \cap S_j)$ are the canonical mappings.

Proposition 7.26 *Let S a system and $\mathcal{S} = \{S_i \mid i \in I\}$ a covering family for S.*

(1) If the covering $\mathcal{S}$ is finite, then there is a canonical embedding

$$i : M(S) \to \{(m_i)_{i \in I} \mid m_i \in M(S_i), \forall i \in I \text{ and } p_{ij}^i(m_i) = p_{ij}^j(m_j), \forall i, j \in I\}.$$

The canonical embedding is an isomorphism if and only if every chordless cycle in the dependence graph G_S of S is a cycle in a subgraph G_{S_i} for some $i \in I$.

(2) If the covering $\mathcal{S}$ is infinite, and if for every $a \in A_S$ there are at most finitely many $i \in I$ such that $a \in A_{S_i}$, then there is an injective morphism $M(S) \to \bigoplus_{i \in I} M(S_i)$, where $\bigoplus M(S_i) = \{(w_i)_{i \in I} \mid w_i \in M(S_i), w_i = \varepsilon$ almost everywhere$\}$ is the weak product of the family $\{M(G_{S_i})\}_{S_i \in \mathsf{In\text{-}Sys}}$.

Proof: (1) The first part of the affirmation follows from Consequence 4.22. The second part follows from Theorem 4.23 (see also [Die90], [MP86]).

(2) The affirmation follows from the comments following Consequence 4.22 (see also [Die90]). □

Remark: Intuitively, affirmation (2) can be explained by the fact that the traces have to be of finite length: if infinitely many systems are working in parallel this has to be "controlled" by requiring that a family $(m_j)_{j \in J}$ has almost all its components equal to ε.

(Note also that weak products are special cases of global sections of sheaves of algebras (namely of sheaves over the co-finite topology on the index set) [KC79].)

It follows that the presheaf $M : \mathsf{SYS}_{\mathsf{il}} \to \mathsf{FPCM}$ satisfies a gluing property for those *finite covers* $\mathcal{S} = \{S_j \mid j \in J\}$ of an object S that have the additional property that every chordless cycle in the dependence graph of S is a cycle in the dependence graph of S_j for some $j \in J$.

Chapter 8

Interconnecting a Given Family of Interacting Systems

In concrete applications we usually are only interested in some subcategory of $\mathsf{SYS}_{\mathsf{il}}$, having as objects those systems relevant for the given application (in the example in Section 6.1 the relevant systems are $S_0, \dots, S_3$ together with their common subsystems, and the systems obtained by interconnecting them).

In what follows we will assume a family InSys of interacting systems given. To enforce the compatibility of models on common sorts, we may assume that all these systems are subsystems of a finite "universal system" S_U. We assume that they are *transition-connected subsystems* of S_U. We further assume that the family InSys is closed under intersections (pullbacks in $\mathsf{SYS}_{\mathsf{il}}$ as subsystems of S_U) i.e. it contains all those subsystems by means of which intercommunication is done. The elements of InSys are the "building blocks" from whose interconnection larger systems arise.

We can regard a system obtained by interconnecting the elements of the family InSys either as a system on its own, or as the set of all elements of InSys by whose interaction it arises (i.e. as a downwards-closed subset of InSys). We will analyze both these approaches, and then the relationship between them.

First we study the category $\mathsf{Sys}(\mathsf{InSys})$ that has as objects all the systems that can be obtained by interconnecting elements in InSys.

Note that, as in the case of $\mathsf{SYS}_{\mathsf{i}}$ and $\mathsf{SYS}_{\mathsf{il}}$, although $\mathsf{Sys}(\mathsf{InSys})$ is a partially ordered set, it is in general not a lattice (meets may not exist) and even if the corresponding meets and joins exist the distributivity law is not always satisfied. Thus, $\mathsf{Sys}(\mathsf{InSys})$ is not a locale.

We define a Grothendieck topology J on $\mathsf{Sys}(\mathsf{InSys})$ and show that in this case admissible states and parallel actions define sheaves with respect to the Grothendieck topology J. Transitions are also analyzed: we show that also in this case they define a natural transformation between Act and $\Omega^{St \times St}$.

Often it is useful to regard systems that arise when interconnecting elements in InSys as "diagrams", or equivalently, as subsets of InSys closed under all possible subsystems. Let $\Omega_1(\mathsf{InSys})$ consist of all families of elements of InSys which are closed under subsystems. We can regard it as a partially ordered set with respect to set inclusion. Alternatively, if we assume given a coverage relation C on the elements in InSys we can instead consider the free frame generated by (InSys, C). If no system in InSys can be covered by other systems this frame is exactly $\Omega_1(\mathsf{InSys})\backslash\emptyset$. In what follows we will denote by InSys^* the set $\mathsf{InSys}\backslash\emptyset$ and by $\Omega(\mathsf{InSys}^*)$ the Heyting algebra of all downwards-closed subsets of InSys^*. ($\Omega(\mathsf{InSys}^*)$ is isomorphic to $\Omega_1(\mathsf{InSys})\backslash\emptyset$.)

States and parallel actions can be defined component-wise; it can be easily seen that they define again sheaves (over the topological space $(\mathsf{InSys}^*, \Omega(\mathsf{InSys}^*))$. Behavior can also be analyzed in this case: this can be done as in the case of $\mathsf{SYS}_{\mathsf{il}}$. We also analyze the behavior given by traces of executions. Our approach is different from the approach presented in [MP86]: we apply a theorem by Davey [Dav73] on the existence of a sheaf of algebras having as fibers quotients of a given algebra, and obtain a result similar to the one given in [MP86], but for partially commutative monoids instead of monoids. It shows that there is a sheaf of monoids having as fibers the free partially commutative monoids associated to the systems in InSys, but that only under very restrictive conditions the monoid of global sections is isomorphic to the free partially commutative monoids associated to the system obtained by interconnecting the elements in InSys.

In order to study the link between the categories $\mathsf{Sys}(\mathsf{InSys})$ and $\Omega(\mathsf{InSys})$, we show that there is an adjunction $\phi : \mathsf{Sys}(\mathsf{InSys}) \to \Omega(\mathsf{InSys})$ and $\pi : \Omega(\mathsf{InSys}) \to \mathsf{Sys}(\mathsf{InSys})$ such that the functor ϕ preserves the covering relation. This adjoint pair induces a geometric morphism $f : \mathsf{Sh}(\mathsf{InSys}) \to \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$.

In order to express properties about systems we need a language in which to formulate them, and an interpretation in different categories. Since many such properties are statements about states, actions and transitions, and we showed that they can be expressed by sheaves in both cases considered above, it seems natural to define a language and interpret it in $\mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$ and $\mathsf{Sh}(\mathsf{InSys})$.

Therefore, we use geometric logic, having as goal to analyze the links between properties of the systems in InSys and those of the system obtained by interconnecting these systems. We show how certain sorts (denoting e.g. states, actions, pairs of states, sets of states) and function symbols (e.g. for representing transitions) can be interpreted in the toposes analyzed above. If we consider the topos $\mathsf{Sh}(\mathsf{InSys})$, we note that the stalk functors (inverse image functors) preserve the validity of so-called coherent axioms; since they are also collectively faithful they also reflect it. On the other hand, the global section functor is an example of direct image functor (it preserves limits), hence it preserves the validity of so-called cartesian axioms. If the topological space $\Omega(\mathsf{InSys}^*)$ is compact and totally disconnected (i.e. if there are finitely many independent systems) then the restrictions about uniqueness for the existentially quantified variables are not necessary anymore.

These considerations help in deciding which properties are inherited by the system obtained by interconnecting a family of given systems. Several examples are provided: determinism, deadlock freedom, fairness of execution, as well as a discussion of general properties of the behavior of systems in time.

In this chapter, if not explicitly specified otherwise, $S_1 \hookrightarrow S_2$ will denote a transition-connected morphism.

8.1 The Category of Systems Obtained by Interconnecting Elements of InSys, Sys(InSys)

Definition 8.1 (Interconnection) *Let* $\{S_i \mid i \in I\}$ *be a family of elements in* InSys*. The* system obtained by their interconnection *is the colimit of the family* $\{S_i \mid i \in I\}$*, computed in* $\mathsf{SYS_{il}}$*.*

Definition 8.2 *The category* Sys(InSys) *has as objects all systems that can be obtained by interconnecting elements in* InSys*, and a morphism from* S_1 *to* S_2 *if and only if* S_1 *is a transition-connected subsystem of* S_2*.*

It is easy to see that as a subcategory of $\mathsf{SYS_{il}}$, Sys(InSys) is closed under colimits (colimits of colimits of elements in In-Sys are again colimits of elements in In-Sys), but it is in general not closed under pullbacks: it can happen that the pullback in $\mathsf{SYS_{il}}$ of two systems that are colimits of elements in InSys is not the colimit of elements in InSys, as the following example will show.

Example 8.1 *Let* $\Sigma = \{0, 1, \leq\}$ *be a signature where* 0 *and* 1 *are 0-ary function symbols and* $\leq$ *is a binary predicate symbol. Let* $M = (\{0,1\}, \{0_M, 1_M, \leq_M\})$ *be a* Σ*-structure where* $\leq_M$ *is an order relation and* $0 \leq_M 1$*. For the sake of simplicity we assume that in what follows the set* A *of atomic actions and the set* C *of constraints are empty.*

Assume that InSys *consists of the systems:*

$$S_1 = (\Sigma, \{a, b\}, \{a \leq b\}, M, A, C),$$

$$S_2 = (\Sigma, \{b, c\}, \{b \leq c\}, M, A, C),$$

$$T = (\Sigma, \{a, c, e, f\}, \{a \leq c, e \leq f\}, M, A, C),$$

together with their intersections. The family InSys *is represented in Figure 8.1*

Let S *be the system obtained by interconnecting* S_1 *and* S_2*.*

$$S = (\Sigma, \{a, b, c\}, \{a \leq b, b \leq c, a \leq c\}, M, A, C)$$

is the colimit of the diagram $\{S_1, S_2, S_1 \cap S_2\}$ *in* $\mathsf{SYS_{il}}$*, hence* $S \in$ Sys(InSys)*.*

Consider the pullback $S \cap T$ *of* S *and* T *(as subsystems of the systems obtained by interconnecting all elements in* InSys*) in* $\mathsf{SYS_{il}}$*,*

$$S \cap T = (\Sigma, \{a, c\}, \{a \leq c\}, M, A, C).$$

It is easy to see that $S \cap T$ *is not a colimit of elements in* InSys*.*

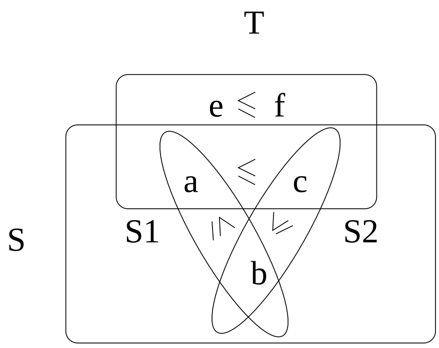


Figure 8.1:

The covering families are defined as in $\mathsf{SYS}_{||}$.

Definition 8.3 *Let S be a system in $\mathsf{Sys}(\mathsf{InSys})$. A family $\mathcal{S} = \{S_i \mid i \in I\}$ of transition-connected subsystems of S, in $\mathsf{Sys}(\mathsf{InSys})$ is a covering family for S if and only if*

(C1) $\mathcal{S}$ is a sieve in $\mathsf{Sys}(\mathsf{InSys})$,

(C2) S is a colimit of the diagram defined by $\mathcal{S}$,

(C3) For every transition connected subsystem T of S in $\mathsf{Sys}(\mathsf{InSys})$,
$T \cap \mathcal{S} = \{S_i \in \mathcal{S} \mid S_i$ transition-connected subsystem of $T\}$
has T as colimit.

Example 8.2 *We give two examples:*

(1) Consider the family InSys introduced in Example 8.1, and represented in Figure 8.1. It is easy to see that $\{S_1, S_2, S_1 \cap S_2, S_1 \cap S_T, S_2 \cap S_T\}$ is a cover of S.

(2) Assume that InSys consists of the systems:

$$S_1 = (\Sigma, \{a, b\}, \{a \leq b\}, M),$$

$$S_2 = (\Sigma, \{b, c\}, \{b \leq c\}, M),$$

$$T = (\Sigma, \{a, c\}, \{a \leq c\}, M),$$

together with their intersections (where Σ, M, A, C are as in Example 8.1). The family InSys is represented in Figure 8.2.

Let S be the system obtained by interconnecting S_1 and S_2, i.e.

$$S = (\Sigma, \{a, b, c\}, \{a \leq b, b \leq c, a \leq c\}, M, A, C)$$

is the colimit of the diagram $\{S_1, S_2, S_1 \cap S_2\}$

It is easy to see that $\{S_1, S_2, S_1 \cap S_2\}$ is not a covering family for S because in particular $T \hookrightarrow S$ is a transition-connected subsystem, but $\{T \cap S_1, T \cap S_2, T \cap S_1 \cap S_2\}$ does not have T as a colimit.

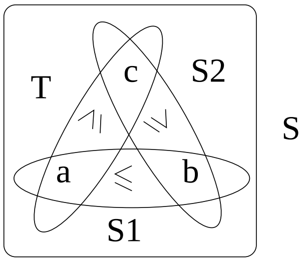


Figure 8.2:

Proposition 8.1 *The function J assigning for every system S the set $J(S)$ of all covering families for S is a Grothendieck topology on* Sys(InSys).

Proof: The proof closely follows the proof given in the case of $\mathsf{SYS}_{\mathsf{il}}$. □

Lemma 8.2 *Let S be an object in* Sys(InSys). *Then the family of all transition-connected subsystems of S contained in* InSys, $\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$ *is a covering family for S.*

Proof: S is an object of Sys(InSys), hence $S = \varinjlim\{R_k \mid k \in K\}$, where $R_k \in \mathsf{InSys}$ for every $k \in K$. The set $\{R_k \mid k \in K\}$ is contained in the set $\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$, hence by the universality property of the colimit it follows that $S = \varinjlim\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$.

Let $T \hookrightarrow S$ be a subsystem of S in Sys(InSys). Then $T = \varinjlim\{T_l \mid l \in L\}$, with $T_l \in \mathsf{InSys}$ for every $l \in L$. Since $T_l \hookrightarrow T$ and $T \hookrightarrow S$, it follows that $\{T_l \mid l \in L\} \subseteq \{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$, hence $\{T_l \mid l \in L\} \subseteq \{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow T\}$. Thus, by the universality property of the colimit it follows that T is the colimit of $\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow T\}$.

This shows that all conditions in the definition of a cover are fulfilled, and that $\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$ is a covering family for S. □

Proposition 8.3 *Assume that in* InSys *no system is a colimit of other subsystems. Let S be a system in* Sys(InSys). *Then every covering family of S contains all the elements of* InSys *that are subsystems of S.*

Proof: Let $\mathcal{S} = \{S_i \mid i \in I\}$ be a cover of S. Let $T \in \mathsf{InSys}$ be a transition-connected subsystem of S. Then, by the definition of a covering family it follows that T is covered by $\{S_i \mid S_i \in \mathcal{S}, S_i \hookrightarrow T\}$.

We know that for every $i \in I$, S_i is covered by the family $\mathcal{S}(S_i) = \{S_{ij} \in \mathsf{InSys} \mid S_{ij} \hookrightarrow S_i\}$. Hence, the family $\{S_{ij} \mid S_{ij} \in \mathcal{S}(S_i), S_i \hookrightarrow T\}$ covers T. It follows that $T = S_{ij}$ for some $S_{ij} \in \mathcal{S}(S_i)$, where $S_i \hookrightarrow T$. Thus, we have $T \hookrightarrow S_i$, hence, $T = S_i$ for some $i \in I$. □

Definition 8.4 *A* finest cover *is a cover none of whose elements can be further decomposed via a cover (except the trivial one).*

Consequence 8.4 *Assume that in* InSys *no system is a colimit of other systems. Then every system* S *in* Sys(InSys) *has a finest cover, namely* $\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$.

Until now, all categories taken into consideration had pullbacks. Hence, when checking whether a functor is a sheaf it sufficed to use Proposition 3.32.

Since **Sys(InSys)** does not have pullbacks, we will need to use the general definition of a sheaf, as given in Definition 3.98 in what follows, namely:

A presheaf F over **Sys(InSys)** is a sheaf if and only if for every object S in **Sys(InSys)** and each cover $\mathcal{S}$ of S in **Sys(InSys)**, the following diagram is an equalizer:

$$F(C) \xrightarrow{e} \prod_{S_i \hookrightarrow S \in \mathcal{S}} F(S_i) \overset{p}{\underset{a}{\rightrightarrows}} \prod_{\substack{T \hookrightarrow S_j, \\ S_j \hookrightarrow S \in \mathcal{S}}} F(T). \tag{8.1}$$

Lemma 8.5 *The functors* $St : \mathsf{Sys(InSys)}^{op} \to \mathsf{Sets}$ *and* $Act : \mathsf{Sys(InSys)}^{op} \to \mathsf{Sets}$ *defined by* $St(S) = \{s : X_S \to M_S \mid s \models \Gamma_S\}$ *and* $Act(S) = \{f : A_S \to \{0,1\} \mid f \models C_S\}$ *are sheaves.*

Proof: In order to prove that for every cover $\mathcal{S} = \{S_i \mid i \in I\}$ of S, $St(S)$ is the coequalizer from diagram 8.1 (with F correspondingly replaced by St), we have to show that every family $\{s_i\}_{i \in I}$, such that for every $i \in I$, $s_i \in St(S_i)$, and for every $S_j \hookrightarrow S_i$, $s_{i|X_j} = s_j$ can be amalgamated to a unique $s \in St(S)$.

Let $\{s_i\}_{i \in I}$ be such that for every $i \in I$, $s_i \in St(S_i)$, and for every $i, j \in I$ with $S_j \subseteq S_i$, $s_{i|X_j} = s_j$. We first show that for every $S_{i_1}, S_{i_2} \in \mathcal{S}$ there exists a system $S_j \in \mathcal{S}$ such that $X_j = X_{i_1} \cap X_{i_2}$.

Let $S_{i_1}, S_{i_2} \in \mathcal{S}$ be arbitrary but fixed. $S_{i_1} = \varinjlim\{T^1_j \mid j \in J_1\}$, $S_{i_2} = \varinjlim\{T^2_k \mid k \in J_2\}$, where T^1_j, T^2_k are elements of InSys.

By the fact that InSys is closed under intersections and from the universality property of colimits it follows that $T = \varinjlim\{T^1_j \cap T^2_k \mid j \in J_1, k \in J_2\}$ is a transition-connected subsystem of both S_{i_1} and S_{i_2}. Moreover, T is a system in **Sys(InSys)**, so, since $\mathcal{S}$ is a sieve, T is in $\mathcal{S}$. Thus, $T = S_j$ for some $j \in I$.

Therefore, $s_{i_1|X_j} = s_j = s_{i_2|X_j}$. It is easy to see that $X_j = X_{i_1} \cap X_{i_2}$. Thus, we proved that for every $i_1, i_2 \in I$, $s_{i_1|X_{i_1} \cap X_{i_2}} = s_{i_2|X_{i_1} \cap X_{i_2}}$.

Then it follows immediately that there is a unique $s : X \to M$ such that $s_{|X_i} = s_i$ for all $i \in I$. Since for every $i \in I$, $s_i \models \Gamma_i$, it follows that $s \models (\bigcup_{i \in I} \Gamma_i)^\bullet = \Gamma_S$.

The fact that Act is a sheaf can be proved analogously. □

8.1.1 Transitions within Sys(InSys)

We now study the properties of transitions in **Sys(InSys)**.

The fact that for every arrow $S_1 \hookrightarrow S_2$ in Sys(InSys) valid transitions in S_2 restrict to valid transitions in S_1 is a consequence of Proposition 6.5, Proposition 6.6 and Proposition 7.8.

Concerning the relationship between transitions in the elements of a covering family for a system S and the transitions in S, note first that the proof of Proposition 7.14 also holds in this case. Hence, if the transitions associated to admissible parallel actions satisfy **(Gluing)** and $\mathcal{S} = \{S_i \mid i \in I\}$ is a cover for S in Sys(InSys) then for every $s_1, s_2 \in St(S)$ and $f \in Act(S)$, $(s_{1|X_i}, s_{2|X_i}) \in Tr^{S_i}(f_{|A_i})$ for every $i \in I$ implies $(s_1, s_2) \in Tr^S(f)$.

If we assume that the actions may consume common resources and are deterministic, and the transitions of parallel actions are obtained according to the rule **(Independence)**, then it turnes out that Proposition 7.14 also holds for Sys(InSys), although this category is not closed under pullbacks.

Proposition 8.6 *Let $\{S_i \mid i \in I\}$ be a covering family for S. Assume that the actions in S and $\{S_i \mid i \in I\}$ are deterministic (the final state is uniquely determined by the initial state in case an action is applied).*

Let $a \in A_S$ and let $s \in St_1(S)$ such that for every $i \in I$ there exists a state $s_i \in St_1(S_i)$ such that $(s_{|X_i}, s_i) \in Tr^{S_i}(a)$ (i.e. if $a \notin A_i$ then $s_{|X_i} = s_i$, otherwise $(s_{|X_i \cap X_a}, s_{i X_i \cap X_a}) \in Tr_a^{S_i}$ and $s(x) = s_i(x)$ if $x \in X_i \backslash X_a$). Then there is a unique state $\overline{s} \in St_1(S)$ such that for all $i \in I$, $\overline{s}_{|X_i} = s_i$ and $(s, \overline{s}) \in Tr^S(a)$.

Proof: From the hypothesis we know that for every $i \in I$ there exists a state $s_i \in St_1(S_i)$ such that $(s_{|X_i}, s_i) \in Tr^{S_i}(a)$. By the determinism of actions it follows that the family $\{s_i\}_{i \in I}$ is such that for every $i \in I$, $s_i \in St(S_i)$ and for every $S_j \hookrightarrow S_i$, $s_{i|X_j} = s_j$ (from the fact that if $S_j \hookrightarrow S_i$, $(s_{|X_j}, s_{i|X_j}) \in Tr^{S_j}(a)$, and from the determinism of a in S_j).

By Proposition 8.5, this family can be amalgamated to a unique $\overline{s} \in St(S)$. It follows that $(s_{|X_i}, \overline{s}_{|X_i}) \in Tr^{S_i}(a)$ for every $i \in I$, hence by Lemma 7.13, $(s, \overline{s}) \in Tr^S(a)$. □

All the other results from Section 7.2.3 also hold for the category Sys(InSys). Thus also in this case transitions define a natural transformation between sheaves in Sh(Sys(InSys), J), $Tr : Act \to \Omega^{St \times St}$.

As in Section 7.2.3, note that $Tr : \mathsf{Sys(InSys)}^{op} \to \mathsf{Sets}$ defined by

$$Tr(S) = \{(f, s, s') \mid f \in Act(S), s, s' \in St(S), (s, s') \in Tr^S(f)\}$$

is a subsheaf of $Act \times St \times St$.

8.2 The Category of Downwards-closed Subsets of InSys Ω(InSys)

It is often useful to regard systems that arise when interconnecting elements in InSys as "diagrams", or equivalently, as subsets of InSys closed under all possible subsystems. We now show that, if we assume that the elements in InSys are independent then the family of those "diagrams" of elements in InSys consisting of subsets of $\mathsf{InSys}\backslash\emptyset$ closed under all possible subsystems is the free frame freely generated by InSys together with the constraint that the empty family of systems covers the empty system.

We assumed that the "building blocks" form a meet-semilattice InSys. Let C be a coverage relation on InSys, i.e. a function assigning to each $S \in \mathsf{InSys}$ a set $C(S)$ of subsets of $\downarrow S$, called *coverings* of S, with the following "meet-stability" property:

$$\text{If } \mathcal{R} \in C(S) \text{ then for all } T \hookrightarrow S, T \in \mathsf{InSys}, \{R \cap T \mid R \in \mathcal{R}\} \in C(T).$$

Let $\mathsf{F_F}(\mathsf{InSys}, \mathsf{C})$ be the free frame freely generated by the meet semilattice InSys with the coverage relation C. It is known ([Joh82], pp.57-59) that $\mathsf{F_F}(\mathsf{InSys}, \mathsf{C})$ is isomorphic (as a frame) with the frame of all C-ideals of InSys (a C-ideal of InSys is an order-ideal I of the meet-semilattice InSys that satisfies

$$\text{If } \exists \mathcal{R} \in C(S) \text{ such that } \mathcal{R} \subseteq I \text{ then } S \in I.)$$

In what follows we assume that the systems in InSys are independent, in the sense that no (non-empty) element in InSys is the colimit of other elements of InSys. (However, the empty system $\emptyset$ is the colimit of the empty family of systems.) This defines a covering relation C on InSys. Let $\mathsf{F_F}(\mathsf{InSys}, \mathsf{C})$ be the free frame freely generated by InSys together with the covering relation C. We know that $\mathsf{F_F}(\mathsf{InSys}, \mathsf{C})$ is isomorphic to the set of all C-ideals of InSys. It is easy to see that an order-ideal I of InSys is a C-ideal if and only if it contains the empty set.

Therefore, the free frame freely generated by InSys together with the covering relation C is the family of those order-ideals that contain the empty system.

Let $\Omega_1(\mathsf{InSys})$ consist of all families of elements of InSys which are closed under subsystems. We can regard it as a partially ordered set with respect to set inclusion. It is easy to see that there is an isomorphism of frames between $\mathsf{F_F}(\mathsf{InSys}, \mathsf{C})$ and $\Omega_1(\mathsf{InSys})\backslash\emptyset$. But $\Omega_1(\mathsf{InSys})\backslash\emptyset$ is isomorphic (as a frame) with the set of downwards-closed subsets of InSys that do not contain $\emptyset$.

Therefore, in what follows we will consider the set $\Omega(\mathsf{InSys})$ of those downwards-closed subsets of InSys that do not contain $\emptyset$ (it is easy to see that this is indeed a topology on $\mathsf{InSys}\backslash\emptyset$).

We will denote by $\mathsf{Sh}(\mathsf{InSys})$ the category of sheaves over $\mathsf{InSys}\backslash\emptyset$ with the topology $\Omega(\mathsf{InSys})$.

States and parallel actions of such "diagrams" of systems can be expressed component-wise.

Let $\overline{St} : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ be defined on objects by

$$\overline{St}(U) = \{(s_i)_{S_i \in U} \mid s_i \in St(S_i), \text{ and if } S_i \hookrightarrow S_j \text{ then } s_i = s_{j|X_i}\},$$

and such that for $\iota : U_1 \subseteq U_2$, $\overline{St}(\iota) : \overline{St}(U_2) \to \overline{St}(U_1)$ is defined by $\overline{St}(\iota)((s_i)_{S_i \in U_2}) = (s_i)_{S_i \in U_1}$.

Let $\overline{Act} : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ be defined on objects by

$$\overline{Act}(U) = \{(f_i)_{S_i \in U} \mid f_i \in Act(S_i), \text{ and if } S_i \hookrightarrow S_j \text{ then } f_i = f_{j|A_i}\},$$

and such that for $\iota : U_1 \subseteq U_2$, $\overline{Act}(\iota) : \overline{Act}(U_2) \to \overline{Act}(U_1)$ is defined by $\overline{Act}(\iota)((f_i)_{S_i \in U_2}) = (f_i)_{S_i \in U_1}$.

Proposition 8.7 *$\overline{St}$ and $\overline{Act}$ are objects in $\mathsf{Sh}(\mathsf{InSys})$ (as a topological space) with the topology consisting of all downwards-closed sets.*

Proof: Obvious. □

In what follows, for every $S_i \in \mathsf{InSys}$ we will denote by $\downarrow S_i$ the set of all elements in InSys that are contained in S.

Lemma 8.8 *Let $F : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ be a sheaf. Then for every $S_i \in \mathsf{InSys}$, the stalk of F at S_i, F_{S_i} is in bijective correspondence with $F(\downarrow S_i)$.*

Proof: We know that the stalk of F at S_i, $F_{S_i} = \varinjlim_{S_i \in U} F(U)$. It is easy to see that, if U is a downwards-closed set with $S_i \in U$, then $\downarrow S_i \subseteq U$. It is easy to show that $F(\downarrow S_i)$ satisfies the universality property of the colimit of the diagram $\{F(U) \mid S_i \in U\}$ (with the appropriate morphisms $F(U) \stackrel{\rho^U_V}{\to} F(V)$ for every $V \subseteq U$). This shows that there is a bijection between F_{S_i} and $F(\downarrow S_i)$. □

Consequence 8.9 *For every $S_i \in \mathsf{InSys}$, the stalk of $\overline{St}$ at S_i, $\overline{St}_{S_i}$ is in bijective correspondence with $\overline{St}(\downarrow S_i)$ and the stalk of $\overline{Act}$ at S_i, $\overline{Act}_{S_i}$ is in bijective correspondence with $\overline{Act}(\downarrow S_i)$.*

Proof: Follows from Lemma 8.8, since $\overline{St}$ and $\overline{Act}$ are sheaves. □

Proposition 8.10 *Let $U \in \Omega(\mathsf{InSys})$ be a downwards-closed set, and let S be the colimit of the diagram defined by U. There is a bijective correspondence between $St(S)$ and $\overline{St}(U)$, and a bijective correspondence between $Act(S)$ and $\overline{Act}(U)$.*

Proof: The colimit of the diagram defined by $U = \{S_i \mid i \in I\}$ is the system S, with $\Sigma_S = \bigcup_{i\in I} \Sigma_i, X_S = \bigcup_{i \in I} X_i, M_S = M_{U|\Sigma_S}, \Gamma_S = (\bigcup_{i\in I} \Gamma_i)^\bullet, A_S = \bigcup_{i \in I} A_i, C_S = (\bigcup_{i\in I} C_i)^\bullet$.

Let $s \in St(S)$. Then $s : \bigcup_{i \in I} X_i \to M$, hence for every $i \in I$, $s_i = s_{|X_i} : X_i \to M_i \in St(S_i)$. Define $f(s) = (s_i)_{S_i \in U}$. Then $f(s) \in \overline{St}(U)$.

Conversely, for every $(s_i)_{S_i \in U}$ such that $s_i \in St(S_i)$ and if $S_i \hookrightarrow S_j$, $s_i = s_{j|X_j}$, there exists a unique $s \in St(S)$ such that $s_{|X_i} = s_i$, for every $i \in I$ (follows from the fact that InSys is closed under intersections). This proves the bijectivity of f. □

Consequence 8.11 *For every $S_i \in \mathsf{InSys}$ there are bijective correspondences between the stalk $\overline{St}_{S_i}$ at S_i and $St(S_i)$, and between the stalk $\overline{Act}_{S_i}$ at S_i and $Act(S_i)$.*

8.2.1 Transitions within $\Omega(\mathsf{InSys})$

We now study the transitions between systems. Recall that for every system S_i in InSys and every action $a \in A_i$, the transition defined by a in S_i is denoted by $Tr^{S_i}(a)$. For every $f \in \overline{Act}(S_i)$, we denote the transition associated to f in S_i by $Tr^{S_i}(f)$.
For every $U \in \Omega(\mathsf{InSys})$ and every $f = (f_i)_{S_i \in U} \in \overline{Act}(U)$ we will denote by $\overline{Tr}^U(f)$ the set $\{(s, s') \mid s = (s_i)_{i \in U}, s' = (s'_i)_{i \in U} \in \overline{St}(S_i)$ and $\forall i \in U, (s_i, s'_i) \in Tr^{S_i}(f_i)\}$.

We want to show that, like in the category $\mathsf{SYS}_{\mathsf{il}}$, transitions can be modeled by a natural transformation $\overline{Tr} : \overline{Act} \to \Omega^{\overline{St} \times \overline{St}}$. We proceed as in Section 7.2.3:

For every $U \in \Omega(\mathsf{InSys})$ and every $f = (f_i)_{S_i \in U} \in \overline{Act}(U)$, let $\overline{Tr}_U(f) : y(U) \times \overline{St} \times \overline{St} \to \Omega$ be defined for every $U' \subseteq U$ and every $s, s' \in \overline{St}(U')$ by

$$\overline{Tr}_U(f)(U')(s, s') = \{U'' \subseteq U \mid \forall S_i \in U'', (s_i, s'_i) \in Tr^{S_i}(f_i)\}.$$

Lemma 8.12 *For every $U, U' \in \Omega(\mathsf{InSys})$ with $U' \subseteq U$, every $f = (f_i)_{S_i \in U} \in \overline{Act}(U)$, and every $s = (s_i)_{i \in U'}, s' = (s'_i)_{i \in U'} \in \overline{St}(U')$, $\overline{Tr}_U(f)(U')(s, s')$ is a closed sieve.*

Proof: It is easy to see that $\overline{Tr}_U(f)(U')(s, s')$ is a sieve. We prove that it is closed.

Let $\{U''_k\}_{k \in K}$ be a cover for U''. Assume that for every $k \in K$, $U''_k \in \overline{Tr}_U(f)(U')(s, s')$, i.e. for every $S_i \in U''_k$, $(s_i, s'_i) \in \overline{Tr}^{S_i}(f_i)$. Let $S_i \in U''$. Then $S_i \in U''_k$ for some $k \in K$, hence $(s_i, s'_i) \in \overline{Tr}^{S_i}(f_i)$. This proves that $U'' \in \overline{Tr}_U(f)(U')$. Thus, $\overline{Tr}_U(f)(U')$ is a closed sieve. □

Lemma 8.13 *Let $f = (f_i)_{S_i \in U} \in \overline{Act}(U)$. Then $\overline{Tr}_U(f) : y(U) \times \overline{St} \times \overline{St} \to \Omega$, defined for every $U' \in \Omega(\mathsf{InSys})$ and every $s = (s_i)_{i \in U'}, s' = (s'_i)_{i \in U'} \in \overline{St}(U')$ by $\overline{Tr}_U(f)(U')(s, s') = \{U'' \subseteq U' \mid \forall S_i \in U'', (s_i, s'_i) \in Tr^{S_i}(f_i)\}$, is a natural transformation.*

Proof: It is easy to see that for every $U_1 \subseteq U_2 \subseteq U'$ the following diagram is commutative:

$$\begin{tikzcd}
\overline{St}(U_2) \times \overline{St}(U_2) \arrow[r, "\overline{Tr}_U(f)(U_2)"] \arrow[d, "\overline{St}(i) \times \overline{St}(i)"'] & \Omega(U_2) \arrow[d, "\Omega(i)"] \\
\overline{St}(U_1) \times \overline{St}(U_1) \arrow[r, "\overline{Tr}_U(f)(U_1)"'] & \Omega(U_1)
\end{tikzcd} \tag{8.2}$$

Indeed, for every $(s, s') \in \overline{St}(U_2) \times \overline{St}(U_2)$ such that $s = (s_i)_{i \in U_2}$ and $s' = (s'_i)_{i \in U_2}$, we have on the one hand

$$\Omega(i)(\overline{Tr}_U(f)(U_2)(s, s')) = \{U'' \subseteq U_1 \mid U'' \subseteq U_1 \subseteq U_2 \in \overline{Tr}_U(f)(U_2)(s, s')\} =$$

$$= \{U'' \subseteq U_1 \mid (s_i, s'_i) \in \overline{Tr}^{S_i}(f_i), \forall i \in U''\},$$

and on the other hand, $\overline{Tr}_U(f)(U_1)(s_{|U_1}, s'_{|U_1}) = \{U'' \subseteq U_1 \mid (s_i, s'_i) \in \overline{Tr}^{S_i}(f_i), \forall i \in U''\}$. This proves that the diagram commutes. □

Consequence 8.14 *For every $U \in \Omega(\mathsf{InSys})$ and $f \in \overline{Act}(S)$, $\overline{Tr}_U(f) \in \Omega^{\overline{St}\times\overline{St}}(U)$.*

Proposition 8.15 *$\overline{Tr} : \overline{Act} \to \Omega^{\overline{St}\times\overline{St}}$ defined for every $U \in \Omega(\mathsf{InSys})$ by $\overline{Tr}_U : \overline{Act}(U) \to \Omega^{\overline{St}\times\overline{St}}(U)$ is a natural transformation in $\mathsf{Sh}(\mathsf{InSys})$.*

Proof: We show that for every $U_1 \subseteq U_2 \in \Omega(\mathsf{InSys})$ the following diagram commutes:

$$\begin{tikzcd}
\overline{Act}(U_2) \arrow[r, "\overline{Tr}_{U_2}"] \arrow[d, "\overline{Act}(i)"'] & \Omega^{\overline{St}\times\overline{St}}(U_2) \arrow[d, "\Omega^{\overline{St}\times\overline{St}}(i)"] \\
\overline{Act}(U_1) \arrow[r, "\overline{Tr}_{U_1}"] & \Omega^{\overline{St}\times\overline{St}}(U_1)
\end{tikzcd} \tag{8.3}$$

Let $f = (f_i)_{i\in U_2} \in \overline{Act}(U_2)$. Then $\Omega^{\overline{St}\times\overline{St}}(i)(\overline{Tr}_{U_2}(f))$ is defined for every $U' \subseteq U_1$ and $s, s' \in \overline{St}(U')$ by $\Omega^{\overline{St}\times\overline{St}}(i)(\overline{Tr}_{U_2}(f))(U')(s,s') = \{U'' \subseteq U' \mid (s_{|U''}, s'_{|U''}) \in \overline{Tr}_{U_2}(f_{|U''}\} = \overline{Tr}_{U_2}(f_{|U_1})(U')(s,s')$. □

Also, $\overline{Tr} : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ defined by

$$\overline{Tr}(U) = \{(f, s, s') \mid f = (f_i)_{S_i\in U} \in \overline{Act}(U), s = (s_i)_{S_i\in U},$$

$$s' = (s'_i)_{S_i\in U} \in \overline{St}(U), (s_i, s'_i) \in Tr^{S_i}(f_i), \forall S_i \in U\}$$

is a subsheaf of $\overline{Act} \times \overline{Act} \times \overline{Act}$.

8.3 Temporal Behavior in Sys(InSys) and Ω(InSys)

For the sake of simplicity, in what follows Sys will denote one of the categories Sys(InSys) or Ω(InSys). The objects of the category (either systems of downwards-closed subsets of InSys) will be denoted by $S, S_1, S_2, \ldots, S_i, \ldots$.

On both these categories we have a suitable notion of covering, that induces Grothendieck topologies. By St, Act, and Tr we will denote either the sheaves St, Act resp. the natural transformation Tr if Sys = Sys(InSys) or $\overline{St}$, $\overline{Act}$, and $\overline{Tr}$ in the case when Sys = Ω(InSys).

Definition 8.5 *Let S be a system in* Sys. *The behavior of S is a functor $B_S : \mathcal{T}^{op} \to \mathsf{Sets}$ defined for every $T \in \mathcal{T}$ by $B_S(T) = \{h : T \to St(S) \times Act(S) \mid K(h,T)\}$, where $K(h,T)$ can be expressed by*

> for every n, if $n, n+1 \in T$ and $h(n) = (s, f)$, $h(n+1) = (s', f')$ then $(s, s') \in Tr(f)$,

and for every $\iota : T_1 \subseteq T_2$, $B_S(\iota) : B_S(T_2) \to B_S(T_1)$ is the restriction to T_1.

Thanks to the particular form of the open sets of $\mathcal{T}$ (all $\{0, 1, \ldots, n\}$ for some n), it can easily be shown that B_S is a sheaf.

Let $\iota : S_1 \hookrightarrow S_2$ in Sys. We define $\rho^{S_2}_{S_1} : B_{S_2} \to B_{S_1}$ by $\rho^{S_2}_{S_1}(T) : B_{S_2}(T) \to B_{S_1}(T)$ for every $T \in \mathcal{T}$, where for every $h : T \to St(S_2) \times Act(S_2)$, $\rho^{S_2}_{S_1}(T)(h) = \langle St(\iota), Act(\iota)\rangle \circ h : T \to St(S_1) \times Act(S_1)$ (with $St(\iota)(s) = s_{|X_1}$ for every $s \in St(S_2)$ and $Act(\iota)(f) = f_{|A_1}$). In what follows, for every $T \in \mathcal{T}$ and every $h \in B_{S_1}(T)$, we will abbreviate $\rho^{S_2}_{S_1}(T)(h)$ by $h_{|S_1}$.

Lemma 8.16 *Let $T \in \mathcal{T}$ be arbitrary but fixed. Let $B'_T : \mathsf{Sys}^{op} \to \mathsf{Sets}$ be defined for every object $S \in \mathsf{Sys}$ by $B'_T(S) = B_S(T)$ and for every morphism $\iota : S_1 \hookrightarrow S_2$ by $B'_T(\iota) = \rho^{S_2}_{S_1} : B_{S_2}(T) \to B_{S_1}(T)$. Then B'_T is a sheaf.*

Proof: Let $S \in \mathsf{Sys}$ and $\{S_j \hookrightarrow S \mid j \in J\}$ be a cover of S. Let $\{h_j \mid j \in J\}$, $h_j \in B_{S_j}(T)$ be a matching family. We show that there is a unique $h \in B_S(T)$ such that $h_{|S_j} = h_j$ for every $j \in J$.

From the definition of the behavior of a subsystem, we know that for all $j \in J$ and every $t \in T$, $h_j(t) = (s^t_j, f^t_j)$, where $s^t_j \in St(S_j)$ and $f^t_j \in Act(S_j)$, and if $t, t+1 \in T$ then $(s^t_j, s^{t+1}_j) \in Tr^{S_j}(f^t_j)$.

Since $\{h_j \mid j \in J\}$ is a matching family, the families $\{s^t_j \mid j \in J\}$ and $\{f^t_j \mid j \in J\}$ are matching families for every $t \in T$.

Since St and Act are sheaves, it follows that for every $t \in T$ there is a unique $s^t \in St(S)$ such that $s^t_{|S_j} = s^t_j$ for every $j \in J$, and a unique $f^t \in Act(S)$ such that $f^t_{|S_j} = f^t_j$ for every $j \in J$. Define $h : T \to St(S) \times Act(S)$ by $h(t) = (s^t, f^t)$ for $t \in T$. Note that if $t, t+1 \in T$, then $(s^t_j, s^{t+1}_j) \in Tr^{S_j}(f^t_j)$ for every $j \in J$, hence $(s^t_{|S_j}, s^{t+1}_{|S_j}) \in Tr^{S_j}(f^t_{|S_j})$ for every $j \in J$. By Proposition 7.14 adapted to $\mathsf{Sys(InSys)}$ or Proposition 8.6 (depending on the rule which is applied for the computation of transitions of parallel actions) if $\mathsf{Sys} = \mathsf{Sys(InSys)}$, resp. by the definition of $\overline{Tr}$, if $\mathsf{Sys} = \Omega(\mathsf{InSys})$, we know that $(s^t, s^{t+1}) \in Tr^S(f^t)$. It follows that also h satisfies the conditions $K(f)$. Then $h \in B(S)$ and $h_{|S_j} = h_j$. $\square$

Proposition 8.17 *Let $B : \mathsf{Sys}^{op} \to \mathsf{Sh}(\mathcal{T})$ be defined for every object S of Sys by $B(S) = B_S : \mathcal{T}^{op} \to \mathsf{Sets}$, and for every morphism $\iota : S_1 \hookrightarrow S_2$ by $B(i) = \rho^{S_2}_{S_1} : B(S_2) \to B(S_1)$, and let $B' : \mathcal{T}^{op} \to \mathsf{Sh(Sys)}$ be defined for every $T \in \mathcal{T}$ by $B'(T) : \mathsf{Sys}^{op} \to \mathsf{Sets}$, $B'(T)(S) = B_S(T)$, and for every $T_2 \overset{\iota}{\subseteq} T_1$ by $B'(\iota) : B'(T_1) \to B'(T_2)$, where for every system S, $B'(\iota)_S = B_S(\iota) : B_S(T_1) \to B_S(T_2)$. Then B and B' are functors.*

Proof: We know that in both $\mathsf{Sys(InSys)}$ and $\Omega(InSys)$ for every arrow $S_1 \hookrightarrow S_2$, transitions in S_2 restrict to transitions in S_1. It follows that if $S_1 \hookrightarrow S_2$ is a morphism in Sys, then for every $h \in B_{S_2}(T)$, $\rho^{S_2}_{S_1}(h) \in B_{S_1}(T)$, i.e. that $\rho^{S_2}_{S_1}$ is well-defined. It is easy to see that $\rho^{S_2}_{S_1} : B_{S_2} \to B_{S_1}$ is a natural transformation. Let $T_1 \subseteq T_2$, and let $i^{T_2}_{T_1}$ be the inclusion of T_1 in T_2. Then the following diagram commutes:

$$\begin{tikzcd} B_{S_2}(T_2) \arrow[r, "\rho_{S_1}^{S_2}(T_2)"] \arrow[d, "B_{S_2}(i_{T_1}^{T_2})"'] & B_{S_1}(T_2) \arrow[d, "B_{S_1}(i_{T_1}^{T_2})"] \\ B_{S_2}(T_1) \arrow[r, "\rho_{S_1}^{S_2}(T_1)"'] & B_{S_1}(T_1) \end{tikzcd} \tag{8.4}$$

Hence, B is a functor. In order to show that B' is a functor, note first that from Lemma 8.16 it follows that B' is well-defined on objects. Let $i : T_1 \hookrightarrow T_2$ be the inclusion between the open sets $T_1, T_2 \in \mathcal{T}$. Let us define $B'(i) : B'(T_2) \to B'(T_1)$ by $B'(i)(S) : B_S(T_2) \to B_S(T_1)$ by $B'(i)(S)(h) = B_S(i)(h) = h_{|T_1}$ for every $h : T_2 \to St(S) \times Act(S) \in B_S(T_2)$. $B'(i)$ is a natural transformation between the sheaves $B'(T_2)$ and $B'(T_1)$. This follows also from the commutativity of diagram 8.4. $\square$

Similar functors, B_{St}, B'_{St}, resp. B_{Act}, B'_{Act} can be defined, in which only the information about states (resp. actions) is encoded. Natural transformations can be defined: e.g. for every object T of $\mathcal{T}$, $\pi_1^T : B'(T) \to B'_{St}(T)$ and $\pi_2^T : B'(T) \to B'_{Act}(T)$ defined for every system S by $\pi_1^T(S)(h) = \pi_1 \circ h : T \to St(S)$ and $\pi_2^T(S)(h) = \pi_2 \circ h : T \to Act(S)$.

8.4 Models for Behavior in $\Omega(\mathsf{InSys})$: Traces

If we ignore the states of the system, then we can express the behavior of any diagram of systems $V \in \Omega(\mathsf{InSys})$ by

$$L_S = \{f_1 \dots f_n \mid n \in \mathbb{N}, f_i \in \overline{Act}(V) \forall i \in \{1, \dots, n\}, \exists h \in B_S(\{1, \dots, n\}),$$

$$\exists s_i \in \overline{St}(S), \text{ such that } \forall i \in \{1, \dots, n\}, h(i) = (s_i, f_i)\}.$$

The elements of L_S are strings of elements in $\overline{Act}(V)$.

In what follows we will assume that all constraints on actions are of the form $a_i \wedge a_j = 0$ (the constraints state which actions are interdependent and therefore cannot be performed in parallel).

Consider the family $\{Act(S_i)^* \mid S_i \in \mathsf{InSys}\}$, where for every $S_i \in \mathsf{InSys}$, $Act(S_i)^*$ is the monoid freely generated by $Act(S_i)$. For every $S_i, S_j \in \mathsf{InSys}$ such that $S_i \hookrightarrow S_j$, let $\rho_{S_i}^{S_j} : Act(S_j) \to Act(S_i)$ be the restriction to S_i. The restriction extends in a canonical way to a homomorphism of monoids, $p_i^j : Act(S_j)^* \to Act(S_i)^*$.

Let $M = \{(w_i)_{S_i \in \mathsf{InSys}} \mid w_i \in Act(S_i)^* \text{ and } \forall S_i \hookrightarrow S_j, p_i^j(w_j) = w_i\}$.

Lemma 8.18 *M is the limit of the diagram $\{Act(S_i)^* \mid S_i \in \mathsf{InSys}\}$ (with the appropriate morphisms p_i^j for every $S_i \hookrightarrow S_j$).*

Proof: It is easy to see that M defines a cone and satisfies the universality property of a limit. $\square$

For every $V \in \Omega(\mathsf{InSys})$ let $M(V)$ be defined by $M(V) = \{(w_i)_{S_i \in V} \mid w_i \in Act(S_i)^* \text{ and } w_{i|S_j} = w_j \text{ for every } S_j \hookrightarrow S_i\}$. $M(V)$ is the limit of the diagram $\{Act(S_i)^* \mid S_i \in V\}$ (with the corresponding morphisms).

Proposition 8.19 *Let* $M : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ *be defined on objects by* $M(U) = \{(w_i)_{S_i \in V} \mid w_i \in Act(S_i)^*$ *and* $w_{i|S_j} = w_j$ *for every* $S_j \hookrightarrow S_i\}$, *and for every* $\iota : U_1 \subseteq U_2$ *by* $M(\iota) : M(U_2) \to M(U_1)$ *defined for every* $(w_i)_{S_i \in U_2}$ *by* $M(\iota)((w_i)_{S_i \in U_2}) = (w_i)_{S_i \in U_1}$. *Then* M *is a sheaf of monoids.*

Proof: Let $U \in \Omega(\mathsf{InSys})$ and $\{U_k \mid k \in K\}$ be a cover for U. Let $\{w_k\}_{k \in K}$ be a family of elements, such that for every $k \in K$, $w_k = (w_k^i)_{S_i \in U_k}$ and for every $k_1, k_2 \in K$, if $S_i \in U_{k_1} \cap U_{k_2}$ then $w_{k_1}^i = w_{k_2}^i$.

Let $w = (w_i)_{Si \in U}$ be defined as follows: for every $S_i \in U$, $S_i \in U_k$ for some k. Then w_i is defined to be w_k^i. It is easy to see that w_i is well-defined, because of the compatibility of the family $\{w_k\}_{k \in K}$. It is clear that $p^U_{U_k}(w) = w_k$ for every $k \in K$. The uniqueness of w follows easily from the fact that for every $w' = (w'_i)_{S_i \in U}$ such that $p^U_{U_k}(w') = w_k$ for every $k \in K$ we have $w'_i = w_i^k$ for every $S_i \in U_k$. $\square$

However, note that in general $M(U)$ is not in bijective correspondence with the monoid $Act(S)^*$, where S is the colimit of the diagram defined by U.

In what follows we will focus on "one-element" actions, i.e. take into account models for *asynchronous systems.* Instead of using the method described in [MP86], we will use the results on sheaves of algebras presented in Section 4.1.1. This method seems to be more natural, since intuitively, when studying a system of interacting systems we start by studying the "fibers", not the open sets. We will deduce (in a slightly more general form) results similar to those given in [MP86].

Let S be the colimit of the diagram defined by InSys. Let (A_S, D_S) be the dependence alphabet of S, and $M(S) = M(A_S, D_S)$. For every $S_i \in \mathsf{InSys}$, $S_i \neq \emptyset$, let $M(S_i) = M(A_i, D_i)$ be the partially commutative monoid associated with the dependence alphabet of S_i. We know that $A_S = \bigcup_{S_i \in \mathsf{InSys}} A_i$ and $D_S = \bigcup_{S_i \in \mathsf{InSys}} D_i$. By Theorem 4.21, for every $S_i \in \mathsf{InSys}$ there is a canonical projection $p_i : M(S) \to M(S_i)$ (which is surjective)[1]. Moreover, if $S_j \hookrightarrow S_i$, then we denote the canonical projection by $p^i_j : M(S_i) \to M(S_j)$, and if $S_i, S_j \in \mathcal{S}$, then $p^j_{ij} : M(S_j) \to M(S_i \cap S_j)$, and $p^i_{ij} : M(S_i) \to M(S_i \cap S_j)$ are the canonical mappings. Note that by Theorem 4.21 all homomorphisms $p^i_j : M(S_i) \to M(S_j)$ and $p^i_{ij} : M(S_i) \to M(S_i \cap S_j)$ are surjective.

[1]The canonical projection p_i is the unique morphism of (free) partially commutative monoids that extends the map $h_i : A \to A_i$ defined by $h_i(a) = \begin{cases} a & \text{if } a \in A_i \\ \varepsilon & \text{if } a \notin A_i \end{cases}$.

For every $S_i \in \mathsf{InSys}$ let $\theta_i = ker(p_i)$. Then $M(S_i) \simeq M(S)/\theta_i$.

From the definition of the canonical projections p_i it follows that for every $S_j \hookrightarrow S_i$, the following diagram is commutative:

$$\begin{tikzcd} M(S_i) \arrow[dd, "p^i_j"'] & \\ & M(S) \arrow[ul, "p_i"'] \arrow[dl, "p_j"] \\ M(S_j) & \end{tikzcd} \tag{8.5}$$

Lemma 8.20 $\Omega(\mathsf{InSys})$ *is a S-topology (cf. Definition 4.2).*

Proof: We have to show that for every $m_1, m_2 \in M(S)$, if $p_i(m_1) = p_i(m_2)$ then there exists an open neighborhood U of S_i in $\Omega(\mathsf{InSys})$ such that for every $S_j \in U$, $p_j(m_1) = p_j(m_2)$.

Let $m_1, m_2 \in M(S)$. Assume that $p_i(m_1) = p_i(m_2)$. Let $U = \downarrow S_i$ be the downwards-closed subset of $\mathsf{InSys} \backslash \emptyset$ generated by S_i. It is an open set, and from the commutativity of Diagram 8.5, $p_j(m_1) = p^i_j(p_i(m_1)) = p^i_j(p_i(m_2)) = p_j(m_2)$ for every $S_j \in U$. □

Let (F, f, InSys) be defined by $F = \coprod_{S_i \in \mathsf{InSys}} M(S_i)$, and $f : F \to \mathsf{InSys}$ be the natural projection. Assume that a subbasis for the topology on F is $\mathcal{SB} = \{[m](U) \mid U \in \Omega(\mathsf{InSys}), m \in M(S)\}$, where $[m](U) = \{p_i(m) \mid S_i \in U\}$. Since $\Omega(\mathsf{InSys})$ is an S-topology, by Theorem 4.3 and Corollary 4.4 it follows that:

(1) (F, f, InSys) is a sheaf space of algebras,

(2) The stalk at $S_i \in \mathsf{InSys}$ is isomorphic to $M(S_i)$,

(3) In $M(S) \xrightarrow{\alpha} \Gamma(\mathsf{InSys}, F) \leq \prod_{S_i \in \mathsf{InSys}} M(S_i) \xrightarrow{\pi_i} M(S_i)$:

 (3.i) $\pi_i \circ \alpha$ is an epimorphism,

 (3.ii) $M(S)$ is a subdirect product of the family $\{M(S_i)\}_{S_i \in \mathsf{InSys}}$ if and only if α is a monomorphism.

Lemma 8.21 *Let* $s : \mathsf{InSys} \to \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ *be such that* $s(S_i) \in M(S_i)$ *for every* $S_i \in \mathsf{InSys}$. *Let* $m \in M(S)$ *and* $U \in \Omega(\mathsf{InSys})$. *Then* $S_i \in s^{-1}([m](U))$ *if and only if* $S_i \in U$ *and* $s(S_i) = p_i(m)$.

Proof: Note that $s^{-1}([m](U)) = \{S_i \in \mathsf{InSys} \mid s(S_i) \in [m](U)\} = \{S_i \in \mathsf{InSys} \mid s(S_i) \in \{p_j(m) \mid S_j \in U\}\}$.

For proving the direct implication assume that $S_i \in s^{-1}([m](U))$. Then $s(S_i) = p_j(m)$ for some $S_j \in U$. Since $f \circ s(S_i) = S_i$, it follows that $S_i = f(s(S_i)) = f(p_j(m)) = S_j$, hence $S_i \in U$ and $s(S_i) = p_i(m)$.

To prove the converse, assume that $S_i \in U$ and $s(S_i) = p_i(m)$. Then $s(S_i) \in \{p_j(m) \mid S_j \in U\}$, hence $S_i \in s^{-1}([m](U))$.

This shows that $S_i \in s^{-1}([m](U))$ if and only if $S_i \in U$ and $s(S_i) = p_i(m)$. □

Lemma 8.22 *Let τ be the topology on $F = \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ generated by $\mathcal{SB}$ as a subbasis. Then $s : \mathsf{InSys} \to \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ such that for every $S_i \in \mathsf{InSys}$, $s(S_i) \in M(S_i)$ is continuous if and only if for every $S_i, S_j \in \mathsf{InSys}$ such that $S_j \hookrightarrow S_i$, $p^i_j(s(S_i)) = s(S_j)$.*

Proof: Since $\mathcal{SB}$ is a subbasis for the topology on $F = \coprod_{S_i \in \mathsf{InSys}} M(S_i)$, $s : \mathsf{InSys}^* \to \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ is continuous if and only if for every $[m](U) \in \mathcal{SB}$, $s^{-1}([m](U)) \in \Omega(\mathsf{InSys})$.

In order to prove the direct implication, assume that $s : \mathsf{InSys} \to \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ is continuous. Let $S_i, S_j \in \mathsf{InSys}$ be such that $S_j \hookrightarrow S_i$. We prove that $p^i_j(s(S_i)) = s(S_j)$. Let $U = \downarrow S_i \in \Omega(\mathsf{InSys})$ and let $m \in M(S)$ be such that $p_i(m) = s(S_i)$ (the existence of m is ensured by the fact that $p_i : M(S) \to M(S_i)$ is surjective).

From the continuity of s we know that $s^{-1}([m](\downarrow S_i)) \in \Omega(\mathsf{InSys})$. Obviously, $S_i \in s^{-1}([m](\downarrow S_i))$. Therefore, since $S_j \hookrightarrow S_i$, $S_j \in s^{-1}([m](\downarrow S_i))$, hence, by Lemma 8.21, $s(S_j) = p_j(m)$. Therefore, $s(S_j) = p_j(m) = p^i_j(p_i(m)) = p^i_j(s(S_i))$.

Conversely, assume that for every $S_i, S_j \in \mathsf{InSys}$ such that $S_j \hookrightarrow S_i$ it holds that $p^i_j(s(S_i)) = s(S_j)$. We prove that s is continuous.

Let $[m](U) \in \mathcal{SB}$, where $m \in M(S)$ and $U \in \Omega(\mathsf{InSys})$. We prove that $s^{-1}([m](U)) \in \Omega(\mathsf{InSys})$. Let $S_i \in s^{-1}([m](U))$. Then $S_i \in U$ and $s(S_i) = p_i(m)$. Let $S_j \hookrightarrow S_i$. Then $S_j \in U$ and by the hypothesis, $s(S_j) = p^i_j(s(S_i)) = p^i_j(p_i(m)) = p_j(m)$. Thus, $S_j \in s^{-1}([m](U))$. Therefore $s^{-1}([m](U)) \in \Omega(\mathsf{InSys})$. □

Lemma 8.23 *The set $\Gamma(\mathsf{InSys}, F)$ of global sections of F has the form*

$$\Gamma(\mathsf{InSys}, F) = \{(m_i)_{S_i \in \mathsf{InSys}} \mid m_i \in M(S_i) \text{ and } \forall S_j \hookrightarrow S_i \in \mathsf{InSys}, p^i_j(m_i) = m_j\}.$$

Proof: $\Gamma(\mathsf{InSys}, F) = \{s : \mathsf{InSys} \to \coprod_{S_i \in \mathsf{InSys}} M(S_i) \mid s \text{ continuous and } s(S_i) \in M(S_i), \forall S_i \in \mathsf{InSys}\}$. So, the elements of $\Gamma(\mathsf{InSys}, F)$ are indexed elements $(s(S_i))_{S_i \in \mathsf{InSys}}$. We show by double inclusion that $\Gamma(\mathsf{InSys}, F) = \{(m_i)_{S_i \in \mathsf{InSys}} \mid \forall S_i \in \mathsf{InSys}, m_i \in M(S_i) \text{ and } \forall S_j \hookrightarrow S_i \in \mathsf{InSys}, p^i_j(m_i) = m_j\}$.

Let $s : \mathsf{InSys} \to \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ be continuous. Then by Lemma 8.22 it follows that for every $S_j \hookrightarrow S_i \in \mathsf{InSys}$, $p^i_j(s(S_i)) = s(S_j)$.

Let now $(m_i)_{S_i \in \mathsf{InSys}}$ be such that for every $S_i \in \mathsf{InSys}, m_i \in M(S_i)$ and if $S_j \hookrightarrow S_i \in \mathsf{InSys}$ then $p^i_j(m_i) = m_j$. Define $s : \mathsf{InSys} \to \coprod_{S_i \in \mathsf{InSys}} M(S_i)$ by $s(S_i) = m_i$ for every $S_i \in \mathsf{InSys}$. Hence, for every $S_j \hookrightarrow S_i \in \mathsf{InSys}$, $p^i_j(s(S_i)) = s(S_j)$, and by Lemma 8.22 it follows that s is continuous. □

Therefore, the following result holds:

Proposition 8.24 *Let (F, f, InSys) be defined as above. Then (F, f, InSys) is a sheaf of algebras, and the stalk at $S_i \in \mathsf{InSys}$ is isomorphic to $M(S_i)$. The set of global sections is*

$$\Gamma(\mathsf{InSys}, F) = \{(m_i)_{S_i \in \mathsf{InSys}} \mid m_i \in M(S_i) \text{ and } \forall S_j \hookrightarrow S_i \in \mathsf{InSys}, p^i_j(m_i) = m_j\}.$$

Additionally the following holds:

(1) *If* InSys *is finite, then* $M(S) \hookrightarrow \Gamma(\mathsf{InSys}, F) \leq \prod_{S_i \in \mathsf{InSys}} M(S_i) \xrightarrow{\pi_i} M(S_i)$ *is a subdirect product.*

The embedding $M(S) \hookrightarrow \Gamma(\mathsf{InSys}, F)$ *is an isomorphism if and only if every chordless cycle in the dependence graph* G_S *of* S *is a cycle in a subgraph* G_{S_i} *for some* $S_i \in \mathsf{InSys}$.

(2) *If* InSys *is infinite, and if for every* $a \in A_S$ *there are at most finitely many* $S_i \in \mathsf{InSys}$ *such that* $a \in A_i$*, then there is an injective morphism* $M(S) \to \bigoplus_{i \in I} M(S_i)$*, where* $\bigoplus M(S_i) = \{(w_i)_{i \in I} \mid w_i \in M(S_i), w_i = \varepsilon \text{ almost everywhere}\}$ *is the weak product of the family* $\{M(G_{S_i})\}_{S_i \in \mathsf{In\text{-}Sys}}$.

Proof: The form of the set of global sections of F follows from Lemma 8.23. The next results follow from Corollary 4.22 and the subsequent comments. □

8.5 Some More Remarks Concerning $\Omega(\mathsf{InSys})$

We make some comments about the topology $\Omega(\mathsf{InSys})$ and about an internal representation of time in $\mathsf{Sh}(\mathsf{InSys})$.

8.5.1 Properties of the Topology $\Omega(\mathsf{InSys})$

We make some remarks about the topological space $(\mathsf{InSys}\backslash\emptyset, \Omega(\mathsf{InSys}))$.

It is easy to see that this space is separated if and only if for every two elements $S_1, S_2 \in \mathsf{InSys}$, their largest common transition-connected subsystem $S_1 \cap S_2$ is the empty system.

Moreover, it is easy to see that if this is the case then $(\mathsf{InSys}, \Omega(\mathsf{InSys}))$ is totally disconnected.

The space $(\mathsf{InSys}\backslash\emptyset, \Omega(\mathsf{InSys}))$ is Hausdorff compact if InSys is finite, and all elements in InSys are independent, in the sense that they do not have common subsystems.

Proposition 8.25 *Assume that* InSys *is finite and all the elements in* InSys *are independent (they have no common (nonempty) subsystems). Then the sheaf* $M : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ *has the property that for every* $U \in \Omega(\mathsf{InSys})$*, if* S *is the colimit of the diagram defined by* U*, and* $M(S)$ *is the trace language associated with* S*, then* $M(S)$ *is isomorphic to the set of global sections of* M.

8.5.2 Internal Representation of Time in $\mathsf{Sh}(\mathsf{InSys})$

The remarks in Section 7.2.4 suggest that, in order to reason about the evolution in time of systems, it may be useful to express time internally in the category $\mathsf{Sh}(\mathsf{InSys})$ (or in $\mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$).

In what follows we assume that the time is discrete, hence it can be modeled by the set $\mathbb{N}$ of natural numbers. Let $\mathcal{N} : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ be the constant presheaf defined by $\mathcal{N}(U) = \mathbb{N}$, for every $U \in \Omega(\mathsf{InSys})$, and for every $U \overset{i}{\subseteq} V$, $\mathcal{N}(i) : \mathcal{N}(V) \to \mathcal{N}(U)$ is the identity of $\mathbb{N}$. It is easy to see that $\mathcal{N}$ is not necessarily a sheaf. Let $\mathbb{N} : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ be the sheafification of $\mathcal{N}$.

The sheafification $\mathbb{N}$ of $\mathcal{N}$ (denoted $(\mathcal{N}^+)^+$ in [MLM92] p.130) can be constructed as follows:

(1) Construct $\mathcal{N}^+ : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$, defined by $\mathcal{N}^+(U) = \mathbb{N}$ if $U \neq \emptyset$ and $\mathcal{N}^+(\emptyset) = 1$ (for the empty cover there is exactly one matching family, namely the empty one).

(2) Construct $(\mathcal{N}^+)^+ : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$. An element of $(\mathcal{N}^+)^+(U)$ is an equivalence class of sets of elements $i_j \in \mathcal{N}(U_j)$ for some open covering $\{U_j \mid j \in J\}$ of U, which match ($i_{j_1} = i_{j_2}$) whenever the overlap $U_{j_1} \cap U_{j_2}$ is nonempty. Thus, these elements "glue" together to give a function $i : U \to \mathbb{N}$, with the property that every point of U has some open neighborhood on which the function is constant.

Therefore, for every $U \in \Omega(\mathsf{InSys})$, $\mathbb{N}(U) = \{i : U \to \mathbb{N} \mid f \text{ locally constant}^2\}$. It also follows that there are arrows (in $\mathsf{Sh(InSys)}$), $1 \xrightarrow{0} \mathbb{N} \xrightarrow{s} \mathbb{N}$. The sheaf $\mathbb{N}$ is the natural number object in $\mathsf{Sh(InSys)}$. (For details on the construction of the associated sheaf functor for a given presheaf we refer to [MLM92] pp.128-134.)

Assume that all the systems in InSys are independent from each other (i.e. for every $S_1, S_2 \in \mathsf{InSys}$, $S_1 \cap S_2 = \emptyset$). Therefore, $\Omega(\mathsf{InSys})$ is the set of all subsets of $\mathsf{InSys} \backslash \emptyset$, i.e. the discrete topology. Let $U \in \Omega(\mathsf{InSys})$. In this case any function $f : U \to \mathbb{N}$ is locally constant.

From the above remarks it follows that the internal representation of time in $\mathsf{Sh(InSys)}$ by the sheaf $\mathbb{N}$ enables us to model the fact that *independent systems may have independent clocks.*

8.6 Relationship between Sys(InSys) and Ω(InSys)

We can define a functor $\phi : \mathsf{Sys(InSys)} \to \Omega(\mathsf{InSys})$, that associates to every system S of $\mathsf{Sys(InSys)}$ the family of all elements of InSys which are transition-connected subsystems of S. It is easy to see that if $S_1 \hookrightarrow S_2$ are systems in $\mathsf{Sys(InSys)}$, then every member of $\phi(S_1)$ is also a transition-connected subsystem of S_2, hence it is contained in $\phi(S_2)$. So ϕ is order-preserving and hence a functor.

We can also define a functor $\pi : \Omega(\mathsf{InSys}) \to \mathsf{Sys(InSys)}$ that associates to every downwards-closed family U of elements from InSys its colimit (the system

[2] $f : U \to X$ is locally constant if for every $x \in U$ there is an open neighborhood $U_1 \subseteq U$ of x on which f is constant.

obtained by the interconnection of the elements in U). It is easy to see that π is order preserving.

Proposition 8.26 *The functors* ϕ : Sys(InSys) $\to \Omega$(InSys) *and* $\pi : \Omega$(InSys) $\to$ Sys(InSys) *define an adjoint pair.*

Proof: It is easy to see that for every system S, $\pi(\phi(S)) = S$, hence $\phi(\pi(\phi(S))) = \phi(S)$ and for every $U \in \Omega(\mathsf{InSys})$ $\pi(\phi(\pi(U))) = \pi(U)$. Moreover, for every set $U \in \Omega(\mathsf{InSys})$ $U \subseteq \phi(\pi(U))$. Therefore it follows that there are two natural transformations, $\eta : id_{\Omega(\mathsf{InSys})} \to \phi\pi$ and $\epsilon : \pi\phi \to id_{\mathsf{Sys(InSys)}}$, which satisfy the following two triangular identities (which state that ϕ is a right adjoint of π):

$$
\begin{tikzcd}
\pi \arrow[d, "\pi\eta"'] \arrow[dr, "id_\pi"] & \\
\pi\phi\pi \arrow[r, "\epsilon\pi"'] & \pi
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\phi \arrow[r, "\eta\phi"] \arrow[dr, "id_\phi"'] & \phi\pi\phi \arrow[d, "\phi\epsilon"] \\
& \phi
\end{tikzcd}
$$

□

It is easy to see that Ω(InSys) is a Heyting algebra, whereas Sys(InSys) is a partially ordered set (in which meets, when they exist, do not in general distribute over joins).

Moreover, being a right adjoint, the functor ϕ preserves limits (if they exist), but in general it does not preserve the colimits (joins). Similarly, being a left adjoint, π preserves colimits.

Proposition 8.27 *The functor* ϕ *preserves the covering relation.*

Proof: Let $\{S_i \mid i \in I\}$ be a covering family for S. For every $i \in I$, let $U_i = \phi(S_i) = \{S_{ij} \in \mathsf{InSys} \mid S_{ij} \hookrightarrow S_i\}$, and let $U = \phi(S) = \{T_k \in \mathsf{InSys} \mid T_k \hookrightarrow S\}$. We prove that $U = \bigcup_{i \in I} U_i$.

It is easy to see that $\bigcup_{i \in I} U_i \subseteq U$, since for every $i \in I$, if $S_{ij} \hookrightarrow S_i$ then $S_{ij} \hookrightarrow S$.

Let $T_k \in \mathsf{InSys}$ be such that $T_k \hookrightarrow S$. By Lemma 8.2 we know that for every $i \in I$, $\phi(S_i) = \{S_{ij} \in \mathsf{InSys} \mid S_{ij} \hookrightarrow S_i\}$ is a cover for S_i. Moreover, since $\{S_i \mid i \in I\}$ is a covering family for S, we know that $\{S_{ij} \mid i \in I, j \in J_i\}$ is a covering family for S. Therefore, since $T_k \hookrightarrow S$, it follows that T_k is a colimit of $\{T_k \cap S_{ij} \mid i \in I, j \in J_i\}$.

Since InSys is closed under intersection, and we assumed that no element in InSys is the colimit of other (different) elements in InSys, it follows that $T_k = T_k \cap S_{ij}$ for some $i \in I$ and $j \in J_i$. Hence, $T_k \subseteq S_i$ for some $i \in I$, i.e. $T_k \in \phi(S_i)$ for some $i \in I$. □

Remark: In general π does not map covers to covers, as can be seen from the following example.

Example 8.3 *Consider the situation described in Example 8.2 (2), and represented in Figure 8.3 where* InSys *is supposed to be* $\{S_1, S_2, T, S_1 \cap S_2, T \cap S_1, T \cap$

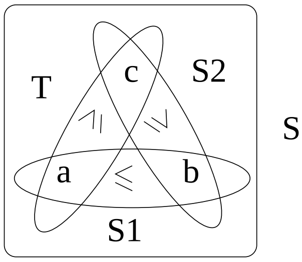


Figure 8.3:

$S_2\}$. *Let* $U = \{S_1, S_2, S_1 \cap S_2, T \cap S_1, T \cap S_2\}$, $U_1 = \{S_1, S_1 \cap S_2, T \cap S_1\}$, $U_2 = \{S_2, S_1 \cap S_2, T \cap S_2\}$, $U_{12} = \{S_1 \cap S_2\}$, $U_{1T} = \{T \cap S_1\}$, $U_{2T} = \{T \cap S_2\}$.

Then $U, U_1, U_2, U_{12}, U_{1T}, U_{2T}$ *are downwards-closed sets, and* $U = U_1 \cup U_2 \cup U_{12} \cup U_{1T} \cup U_{2T}$. *However,* $\pi(U_1) = S_1$, $\pi(U_2) = S_2$, $\pi(U_{12}) = S_1 \cap S_2$, $\pi(U_{1T}) = S_1 \cap S_T$, $\pi(U_{2T}) = S_2 \cap S_T$, *and* $\pi(U) = S$, *but* $\{S_1, S_2, S_1 \cap S_2, T \cap S_1, T \cap S_2\}$ *is not a covering family for* S.

So far we proved that there is an adjoint pair

$$\phi : \mathsf{Sys}(\mathsf{InSys}) \to \Omega(\mathsf{InSys}) \qquad\qquad \pi : \Omega(\mathsf{InSys}) \to \mathsf{Sys}(\mathsf{InSys})$$

where ϕ is right adjoint of π, and additionally that ϕ preserves covers. In order to point out the link between the sheaves over the two sites, we use the following theorem.

Theorem 8.28 (cf. [MLM92] p.412) *Let* (C, J) *and* (D, K) *be sites, and let* $\pi : \mathsf{D} \to \mathsf{C}$ *and* $\phi : \mathsf{C} \to \mathsf{D}$ *be functors such that* π *is left adjoint to* ϕ. *If* ϕ *preserves covers, then there is an induced geometric morphism* $f : \mathsf{Sh}(\mathsf{D}, \mathsf{K}) \to \mathsf{Sh}(\mathsf{C}, \mathsf{J})$, *with inverse and direct image functors described, for sheaves* F *on* (C, J) *and* G *on* (D, K) *by* $f^*(F) = a(F \circ \pi)$ *and* $f_*(G) = G \circ \phi$.

Consequence 8.29 *The adjoint pair* $\phi : \mathsf{Sys}(\mathsf{InSys}) \to \Omega(\mathsf{InSys})$, $\pi : \Omega(\mathsf{InSys}) \to \mathsf{Sys}(\mathsf{InSys})$ *induces a geometric morphism* $f : \mathsf{Sh}(\mathsf{InSys}) \to \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$, *with inverse and direct image functors described, for sheaves* F *on* $(\mathsf{Sys}(\mathsf{InSys}), J)$ *and* G *on* $\Omega(\mathsf{InSys})$, *by*

$$f^*(F) = a(F \circ \pi) : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets} \qquad\qquad f_*(G) = G \circ \phi : \mathsf{Sys}(\mathsf{InSys})^{op} \to \mathsf{Sets}.$$

Remark: For every sheaf F on $(\mathsf{Sys}(\mathsf{InSys}), J)$, $f^*(F)$ is the sheafification of $F \circ \pi$, where $(F \circ \pi)(U) = F(S)$, with $S = \varinjlim \{S_i \mid S_i \in U\}$. For every sheaf G on $\Omega(\mathsf{InSys})$, and every object S of $\mathsf{Sys}(\mathsf{InSys})$, $f_*(G)(S) = G(\phi(S)) = G(\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\})$.

In particular, for $St, Act \in \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$, note that $St \circ \pi$ and $Act \circ \pi$ are again sheaves. Hence, $f^*(St), f^*(Act) : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets}$ are defined by

$$f^*(St)(U) = St(\varinjlim \{S_i \mid i \in U\}),$$

$$f^*(Act)(U) = Act(\varinjlim\{S_i \mid i \in U\}).$$

Consider now $\overline{St}, \overline{Act} \in \mathsf{Sh}(\mathsf{InSys})$. $f_*(\overline{St}), f_*(\overline{Act}) : \mathsf{Sys}(\mathsf{InSys})^{op} \to \mathsf{Sets}$ are defined by

$$f_*(\overline{St})(S) = \overline{St}(\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}) =$$
$$= \{(s_i)_{\substack{S_i \in \mathsf{InSys}\\ S_i \hookrightarrow S}} \mid s_i \in St(S_i) \forall S_i \in \mathsf{InSys} \text{ and } \forall S_i \hookrightarrow S_j, s_{j|X_i} = s_i\},$$
$$f_*(\overline{Act})(S) = \overline{Act}(\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}) =$$
$$= \{(f_i)_{\substack{S_i \in \mathsf{InSys}\\ S_i \hookrightarrow S}} \mid f_i \in Act(S_i) \forall S_i \in \mathsf{InSys} \text{ and } \forall S_i \hookrightarrow S_j, s_{j|X_i} = s_i\}.$$

Concerning behavior, note that for the sheaf of monoids $M : \Omega(\mathsf{InSys})^{op} \to \mathsf{Mon}$, its image via f_*, $f_*(M) : \mathsf{Sys}(\mathsf{InSys}) \to \mathsf{Sets}$ is defined by

$$f_*(M)(S) = \{(m_i)_{\substack{S_i \in \mathsf{InSys}\\ S_i \hookrightarrow S}} \mid m_i \in M(S_i) \text{ and } \forall S_i \hookrightarrow S_j, m_{j|S_i} = m_i\},$$

and by the remarks made in Section 8.4, it is in general different from $M(S)$.

8.7 Geometric Logic, Preservation of Axioms

We showed so far that for both categories considered in this section, namely $\mathsf{Sys}(\mathsf{InSys})$ and $\Omega(\mathsf{InSys})$, states and parallel actions can be expressed by sheaves St, Act (resp. $\overline{St}, \overline{Act}$), and transitions by a subsheaf Tr of $Act \times St \times St$ (resp. a subsheaf $\overline{Tr}$ of $\overline{Act} \times \overline{St} \times \overline{St}$). Additionally, we showed that behavior over a fixed interval of time T can be described by a sheaf, B_T. Similarly, the set of observations of states of the system over the interval T can be described by a sheaf, B_T^{St}, and the set of observations of actions of the system over the interval T can be described by a sheaf, B_T^{Act}. There exist natural transformations $\pi_{St} : B_T \to B_T^{St}$ and $\pi_{Act} : B_T \to B_T^{Act}$. Also, we can single out the empty action ε, which can be considered a constant of sort Act, $\varepsilon : 1 \to Act$.

Some properties about systems can be expressed in terms of states, actions or behavior in time. In order to be able to express these properties in a uniform way we will now introduce a many-sorted language $\mathcal{L}$, having among its sorts St (for states), Act (for actions), $Time$ for time, and B_{Time}, B_{Time}^{St} resp. B_{Time}^{Act} for the behavior in time, relations as $=_X \subseteq X \times X$ for every sort X and $Tr \subseteq Act \times St \times St$, etc.

Let $\mathcal{E} = \mathsf{Sh}(\mathsf{InSys})$, $\mathcal{F} = \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$. The above mentioned sorts, function and relation symbols of the language $\mathcal{L}$ can be interpreted in both $\mathcal{E}$ and $\mathcal{F}$ as will be showed in what follows.

On the other hand, for every system S we know that $St(S)$ represents the set of states of S, $Act(S)$ the set of states of S, etc. Thus, for every system we obtain a concrete interpretation in Sets of the language $\mathcal{L}$.

We have therefore the following possible interpretations of the language $\mathcal{L}$ in the topoi $\mathcal{E} = \mathsf{Sh}(\mathsf{InSys})$, $\mathcal{F} = \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$, and Sets:

	$\mathcal{E} = \mathsf{Sh(InSys)}$	$\mathcal{F} = \mathsf{Sh(Sys(InSys), J)}$	Sets
X	X^M	X^N	X^{M_S}
St	St	$\overline{St}$	$St(S)$
Act	Act	$\overline{Act}$	$Act(S)$
$Time$	$\mathbb{N}_{\mathcal{E}}$	$\mathbb{N}_{\mathcal{F}}$	$\mathbb{N}$
B_{Time}	$B_{\mathbb{N}\mathcal{E}}$	$B_{\mathbb{N}\mathcal{F}}$	$B_{\mathbb{N}}(S)$
$\varepsilon : 1 \to Act$	$\varepsilon_{\mathcal{E}}$	$\varepsilon_{\mathcal{F}}$	$\varepsilon \in Act(S)$
$=_X \subseteq X \times X$	$=_{\mathcal{E}}$	$=_{\mathcal{F}}$	$=$
$Tr \subseteq Act \times St \times St$	Tr	$\overline{Tr}$	Tr^S

We recall that if M is an interpretation of a language in a topos $\mathcal{E}$, then the interpretations of the formulae are defined as follows:

If $\phi = t_1(x_1, \ldots, x_n) = t_2(x_1, \ldots, x_n)$, then the subobject $\{(x_1, \ldots, x_n) \mid t_1 = t_2\}$ is the equalizer of the arrows t_1^M and t_2^M,

$$\{(x_1, \ldots, x_n) \mid t_1 = t_2\} \longrightarrow X_1^M \times \ldots \times X_2^M \rightrightarrows Y^M$$

If $\phi = R(t_1, \ldots, t_k)$ for a relation symbol R and terms t_i of sort Y_i (each with free variables among $x_1, \ldots, x_n$ with sorts $X_1, \ldots, X_n$), then $\{(x_1, \ldots, x_n) \mid R(t_1, \ldots, t_k)\}^M$ is the pullback of the subobject R^M (the interpretation of R) along $\langle t_1, \ldots, t_k \rangle$:

$$\begin{tikzcd}
\{(x_1, \ldots, x_n) \mid R(t_1, \ldots, t_k)\}^M \arrow[r] \arrow[d] & R^M \arrow[d] \\
X_1^M \times \ldots \times X_n^M \arrow[r, "\langle t_1, \ldots, t_k \rangle"] & Y_1^M \times \ldots \times Y_k^M
\end{tikzcd} \tag{8.6}$$

(Note that it may happen in particular cases that some of the terms (say t_i) does not involve all the variables $x_1, \ldots, x_n$, and so defines a morphism $X_{i_1}^M \times \ldots \times X_{i_p}^M \to Y_i^M$ for some $1 < i_1 < i_2 < \ldots < i_p < n$; if so, we simply compose with the appropriate product projection $X_1^M \times \ldots \times X_n^M \to X_{i_1}^M \times \ldots \times X_{i_p}^M$.)

Let T be a theory in $\mathcal{L}$. M is a model for T if all the axioms of T are valid in M.

Example 8.4 Assume that $s_0 : 1 \to St$ is a constant of sort St. Let $\mathsf{Deadlock}_{\mathsf{s_0}}$ be the formula that expresses that s_0 is a deadlock state.

$$\mathsf{Deadlock}_{\mathsf{s_0}} = (\forall a : Act)(\forall s : St)(Tr(a, s_0, s) \Rightarrow (a = \varepsilon) \wedge (s = s_0)).$$

Let M be the interpretation of $\mathcal{L}$ in $\mathcal{E}$ described above. $\mathsf{Deadlock}_{\mathsf{s_0}}$ is true in M if $\{(a, s) \mid Tr(a, s_0, s)\}^M$ is a subobject of $\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M$ in $\mathcal{E}$.

Let $s_0^M : 1 \to \overline{St}$ be an interpretation of s_0. We briefly explain how $\{(a, s) \mid Tr(a, s_0, s)\}^M$ and $\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M$ are constructed.

(1) $\{(a,s) \mid Tr(a,s_0,s)\}^M$ is the pullback of Tr^M along the terms $t_1(a,s) = a$, $t_2(a,s) = s_0$ and $t_3(a,s) = s$. The interpretation in M of $t_1 : Act \times St \to Act$ is the first projection $t_1^M : \overline{Act} \times \overline{St} \to \overline{Act}$. Similarly, $t_3^M : \overline{Act} \times \overline{St} \to \overline{St}$ is the second projection and $t_2^M : \overline{Act} \times \overline{St} \to \overline{St}$ is the composition $\overline{Act} \times \overline{St} \xrightarrow{!} 1 \xrightarrow{s_0^M} St$.

$$\begin{tikzcd}
\{(a,s) \mid Tr(a,s_0,s)\}^M \arrow[r] \arrow[d] & \overline{Tr} = \{(a,s',s) \mid \overline{Tr}(a,s',s)\} \arrow[d] \\
\overline{Act} \times \overline{St} \arrow[r, "\langle t_1^M, t_2^M, t_3^M \rangle"] & \overline{Act} \times \overline{St} \times \overline{St}
\end{tikzcd}$$

(2) $\{(a,s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M$ is the pullback in $\mathcal{E}$ of $\{(a,s) \mid a = \varepsilon\}^M$ and $\{(a,s) \mid s = s_0\}^M$. $\{(a,s) \mid a = \varepsilon\}^M$ is the equalizer

$$\begin{tikzcd}
\{(a,s) \mid a = \varepsilon\}^M \arrow[r] & \overline{Act} \times \overline{St} \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\varepsilon"'] & \overline{Act}
\end{tikzcd}$$

where $\pi_1 : \overline{Act} \times \overline{St} \to \overline{Act}$ is the first projection, whereas $\varepsilon : \overline{Act} \times \overline{St} \to \overline{Act}$ is the composition $\overline{Act} \times \overline{St} \xrightarrow{!} 1 \xrightarrow{\varepsilon^M} \overline{Act}$.

Similarly, $\{(a,s) \mid s = s_0\}^M$ is the equalizer

$$\begin{tikzcd}
\{(a,s) \mid s = s_0\}^M \arrow[r] & \overline{Act} \times \overline{St} \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "s_0"'] & \overline{St}
\end{tikzcd}$$

where $\pi_2 : \overline{Act} \times \overline{St} \to \overline{St}$ is the second projection, whereas $s_0 : \overline{Act} \times \overline{St} \to \overline{Act}$ is the composition $\overline{Act} \times \overline{St} \xrightarrow{!} 1 \xrightarrow{s_0^M} \overline{St}$.

Example 8.5 Consider the formula that expresses determinism:

$$\mathsf{Determ} = (\forall s, s', s'' : St)(\forall a : Act)((Tr(a,s,s') \wedge Tr(a,s,s'')) \Rightarrow (s' = s'')).$$

The formula Determ is valid in M if $\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M$ is a subobject in $\mathcal{E}$ of $\{(a,s,s',s'') \mid s' = s''\}^M$.

(1) $\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M$ is the pullback in $\mathcal{E}$ of $\{(a,s,s',s'') \mid Tr(a,s,s')\}^M$ and $\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M$.

(2) $\{(a,s,s',s'') \mid s' = s''\}^M$ is the equalizer

$$\begin{tikzcd}
\{(a,s,s',s'') \mid s' = s''\}^M \arrow[r] & \overline{Act} \times \overline{St} \times \overline{St} \times \overline{St} \arrow[r, shift left, "\pi_3"] \arrow[r, shift right, "\pi_4"'] & \overline{St}
\end{tikzcd}$$

where π_3 is the third projection and π_4 is the fourth projection.

We know that if $\mathcal{E}_1$ and $\mathcal{E}_2$ are topoi and $f : \mathcal{E}_2 \to \mathcal{E}_1$ a geometric morphism, then the inverse image functor f^* yields for every interpretation M of $\mathcal{L}$ in $\mathcal{E}_1$ an interpretation f^*M of $\mathcal{L}$ in $\mathcal{E}_2$, such that

(1) $X^{f^*M} = f^*(X^M)$ for every sort X,

(2) $R^{f^*M} = f^*(R^M) \subseteq X_1^{f^*M} \times \ldots \times X_n^{f^*M}$ for every relation symbol $R \subseteq X_1 \times \ldots \times X_n$,

(3) Let g be a function symbol of arity $X_1 \times \ldots \times X_n \to Y$. The interpretation of g is translated similarly:

$$\begin{tikzcd} f^*(X_1^M \times \ldots \times X_n^M) \arrow[r, "f^*(g^M)"] \arrow[d, equal] & f^*(Y^M) \arrow[d, equal] \\ X_1^{f^*M} \times \ldots \times X_n^{f^*M} \arrow[r, "g^{f^M}"] & Y^{f^*M} \end{tikzcd}$$

In what follows we will use these facts in order to study the link between models in different topoi.

8.7.1 The Stalk Functors: Preservation Properties

For every $S_i \in \mathsf{InSys}$ let $f_i : \{*\} \to \mathsf{InSys}$ defined by $f_i(*) = S_i$. The corresponding inverse image functor $f_i^* : \mathsf{Sh(InSys)} \to \mathsf{Sets}$ is the functor that associates to every sheaf $F \in \mathsf{Sh(InSys)}$ the stalk at S_i, F_{S_i}, which by Lemma 8.8 is isomorphic to $F(\downarrow S_i)$. Therefore, for all $S_i \in \mathsf{InSys}$ the functor $f_i^* : \mathsf{Sh(InSys)} \to \mathsf{Sets}$ preserves the validity of coherent axioms. The stalk functors are collectively faithful, hence they also reflect the validity of coherent axioms (cf. also [Joh82], p.178).

This shows that the topos $\mathsf{Sh(InSys)}$ satisfies a coherent axiom ϕ if and only if each stalk satisfies ϕ. For example, we can say that $\mathsf{Sh(InSys)}$ has (internally) property ϕ if and only if S_i has property ϕ for every system $S_i \in \mathsf{Sh(InSys)}$.

We now point out how coherent axioms in the language $\mathcal{L}$ are translated by the stalk functions:

For every sort F of $\mathcal{L}$, let $F^M \in \mathsf{Sh(InSys)}$ be its interpretation in $\mathsf{Sh(InSys)}$. Then, the corresponding sort in Sets induced by f_i^* is $f_i^*(F^M) = F^M_{S_i}$, the stalk at S_i of the sheaf F^M. Similarly, for every function symbol $f : F_1 \to F_2$ in $\mathcal{L}$, let $f^M : F_1^M \to F_2^M$ be its interpretation in $\mathsf{Sh(InSys)}$. Then, the corresponding function in Sets induced by f_i^* is $f_i^*(f^M) : f_i^*(F_1^M) \to f_i^*(F_2^M)$, i.e. the image of f^M by the stalk functor, $f^M_{S_i} : {F_1^M}_{S_i} \to {F_2^M}_{S_i}$. The translation of relation symbols is done in a similar way.

Example 8.6 Consider the formula $\mathsf{Deadlock_{s_0}}$ analyzed in Example 8.4, namely:

$$\mathsf{Deadlock_{s_0}} = (\forall a : Act)(\forall s : St)(Tr(a, s_0, s) \Rightarrow (a = \varepsilon) \wedge (s = s_0)).$$

Let M be the interpretation of $\mathcal{L}$ in $\mathcal{E}$. Assume that $\mathsf{Deadlock_{s_0}}$ is true in M i.e. $\{(a, s) \mid Tr(a, s_0, s)\}^M$ is a subobject of $\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M$ in $\mathcal{E}$.

By the remarks above we know that $\{(a,s) \mid Tr(a,s_0,s)\}^M$ is a subobject of $\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M$ in $\mathcal{E}$ if and only if $f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M)$ is a subobject of $f_i^*(\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M)$ in Sets for every $S_i \in \mathsf{InSys}$.

We now analyze what is the form of $f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M)$ resp. $f_i^*(\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M)$.

(1) Since $\{(a,s) \mid Tr(a,s_0,s)\}^M$ is the pullback of Tr^M along the terms $t_1(a,s)=a$, $t_2(a,s)=s_0$ and $t_3(a,s)=s$:

$$\begin{tikzcd}
\{(a,s) \mid Tr(a,s_0,s)\}^M \arrow[r] \arrow[d] & \overline{Tr} = \{(a,s',s) \mid \overline{Tr}(a,s',s)\} \arrow[d] \\
\overline{Act} \times \overline{St} \arrow[r] & \overline{Act} \times \overline{St} \times \overline{St}
\end{tikzcd}$$

and f_i^* commutes with pullbacks it follows that $f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M)$ is the pullback (in Sets):

$$\begin{tikzcd}
f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M) \arrow[r] \arrow[d] & f_i^*(\overline{Tr}) = f_i^*(\{(a,s',s) \mid \overline{Tr}(a,s',s)\}) \arrow[d] \\
f_i^*(\overline{Act}) \times f_i^*(\overline{St}) \arrow[r] & f_i^*(\overline{Act}) \times f_i^*(\overline{St}) \times f_i^*(\overline{St}).
\end{tikzcd}$$

Taking into account that $f_i^*(\overline{Act})$, $f_i^*(\overline{St})$ resp. $f_i^*(\overline{Tr})$ are in bijection with $Act(S_i)$, $St(S_i)$ and $Tr(S_i) = \{(f,s,s') \mid (s,s') \in Tr^{S_i}(f)\}$ it follows that $f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M)$ is the pullback (in Sets):

$$\begin{tikzcd}
f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M) \arrow[r] \arrow[d] & Tr(S_i) \arrow[d] \\
Act(S_i) \times St(S_i) \arrow[r] & Act(S_i) \times St(S_i) \times St(S_i)
\end{tikzcd}$$

Thus, $f_i^*(\{(a,s) \mid Tr(a,s_0,s)\}^M) = \{(a,s) \mid Tr(a,s_0,s)\}^{M_{S_i}}$.

(2) $\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M$ is the pullback in $\mathcal{E}$ of $\{(a,s) \mid a=\varepsilon\}^M$ and $\{(a,s) \mid s=s_0\}^M$, where $\{(a,s) \mid a=\varepsilon\}^M$ is the equalizer

$$\begin{tikzcd}
\{(a,s) \mid a=\varepsilon\}^M \arrow[r] & \overline{Act} \times \overline{St} \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\varepsilon"'] & \overline{Act}
\end{tikzcd}$$

and $\{(a,s) \mid s=s_0\}^M$ is the equalizer

$$\begin{tikzcd}
\{(a,s) \mid s=s_0\}^M \arrow[r] & \overline{Act} \times \overline{St} \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "s_0"'] & \overline{St}.
\end{tikzcd}$$

Since f_i^* preserves equalizers, it follows that

$$\begin{tikzcd}
f_i^*(\{(a,s) \mid a=\varepsilon\}^M) \arrow[r] & Act(S_i) \times St(S_i) \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\varepsilon"'] & Act(S_i)
\end{tikzcd}$$

is an equalizer in **Sets**, and

$$f_i^*(\{(a,s) \mid s = s_0\}^M) \longrightarrow Act(S_i) \times St(S_i) \underset{s_0}{\overset{\pi_1}{\rightrightarrows}} St(S_i)$$

is an equalizer in **Sets**. Thus, $f_i^*(\{(a,s) \mid a = \varepsilon\}^M) = \{(\varepsilon, s) \mid s \in St(S_i)\}$ and $f_i^*(\{(a,s) \mid s = s_0\}^M) = \{(a, s_0(S_i)) \mid a \in Act(S_i)\}$.

Moreover, $f_i^*(\{(a,s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M)$ is the pullback in **Sets** of $f_i^*(\{(a,s) \mid a = \varepsilon\}^M)$ and $f_i^*(\{(a,s) \mid s = s_0\}^M)$. Thus, $f_i^*(\{(a,s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M) = \{(\varepsilon, s_0(S_i))\}$.

Thus, $\mathsf{Deadlock}_{s_0}$ is valid in the interpretation M in $\mathcal{E}$ is and only if for every $S_i \in \mathsf{InSys}$ $(s_0(S_i), s') \in Tr^{S_i}(a)$ implies $a = \varepsilon$ and $s' = s_0(S_i)$.

Example 8.7 Now consider Example 8.5. Let

$$\mathsf{Determ} = (\forall s, s', s'' : St)(\forall a : Act)((Tr(a,s,s') \wedge Tr(a,s,s'')) \Rightarrow (s' = s''))$$

be the formula that expresses determinism. The formula Determ is valid in M if $\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M$ is a subobject in $\mathcal{E}$ of $\{(a,s,s',s'') \mid s = s'\}^M$.

For every $S_i \in \mathsf{InSys}$ we explain how $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M)$ and $f_i^*(\{(a,s,s',s'') \mid s = s'\}^M)$ can be computed.

(1) $\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M$ is the pullback in $\mathcal{E}$ of $\{(a,s,s',s'') \mid Tr(a,s,s')\}^M$ and $\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M$. These are computed as equalizers in $\mathcal{E}$. Therefore, $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M)$ is the pullback in **Sets** of $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s')\}^M)$ and $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M)$.

It is easy to see that $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s')\}^M) = \{(a,s,s',s'') \mid (s,s') \in Tr^{S_i}(a)\}$, and $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M) = \{(a,s,s',s'') \mid (s,s'') \in Tr^{S_i}(a)\}$. Hence, $f_i^*(\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M) = \{(a,s,s',s'') \mid (s,s') \in Tr^{S_i}(a) \text{ and } (s,s'') \in Tr^{S_i}(a)\}$.

(2) $\{(a,s,s',s'') \mid s' = s''\}^M$ is the equalizer

$$\{(a,s,s',s'') \mid s' = s''\}^M \longrightarrow \overline{Act} \times \overline{St} \times \overline{St} \times \overline{St} \underset{\pi_4}{\overset{\pi_3}{\rightrightarrows}} \overline{St}$$

Therefore, $f_i^*(\{(a,s,s',s'') \mid s' = s''\}^M)$ is the equalizer

$$f_i^*(\{(a,s,s',s'') \mid s' = s''\}^M) \longrightarrow Act(S_i) \times St(S_i) \times St(S_i) \times St(S_i) \underset{\pi_4}{\overset{\pi_3}{\rightrightarrows}} St(S_i)$$

in **Sets**, i.e. $f_i^*(\{(a,s,s',s'') \mid s' = s''\}^M) = \{(a,s,s',s'') \mid a \in Act(S_i), s, s', s'' \in St(S_i), s' = s''\}$.

Thus, Determ is valid in interpretation M in $\mathcal{E}$ if and only if for every $S_i \in \mathsf{InSys}$, for every $a \in Act(S_i)$ and every $s, s', s'' \in St(S_i)$, $(s,s') \in Tr(a)$ and $(s,s'') \in Tr(a)$ implies $s' = s''$.

8.7.2 The Global Section Functor: Preservation Properties

Consider now the unique map $g : \mathsf{InSys} \to \{*\}$. The corresponding direct image functor, $g_* : \mathsf{Sh(InSys)} \to \mathsf{Sets}$ is the global section functor $g_*(F) = F(\mathsf{InSys})$ for every sheaf $F \in \mathsf{Sh(InSys)}$.

Direct image functors preserve limits but in general they do not preserve unions or images, hence they do not preserve the validity of coherent axioms.

Following Johnstone [Joh82], we define a formula to be *cartesian relative to a a given theory* $\mathbb{T}$ if it is constructed from atomic formulae using only conjunction and existential quantification over "$\mathbb{T}$-provably unique" variables (i.e. variables whose values, in any model of $\mathbb{T}$, are uniquely determined by the values of the remaining free variables). Similarly, one defines cartesian axioms (relative to a given theory $\mathbb{T}$) as axioms of the form $(\forall x)(\phi(x) \Rightarrow \psi(x))$ where the formulae ϕ and ψ are cartesian relatively to $\mathbb{T}$. We say that a theory is cartesian if its axioms can be ordered such that each is cartesian relative to those which precede it. Then it follows that models of cartesian theories are preserved by direct image functors (in particular by global section functors).

For every sort F of $\mathcal{L}$, let $F^M \in \mathsf{Sh(InSys)}$ be its interpretation in $\mathsf{Sh(InSys)}$. Then, the corresponding sort in Sets induced by g_* is $g(F^M) = \Gamma(\mathsf{InSys}, F^M)$, the set of global sections of F^M.

For every function symbol $f : F_1 \to F_2$ in $\mathcal{L}$, let $f^M : F_1^M \to F_2^M$ be its interpretation in $\mathsf{Sh(InSys)}$. Then, the corresponding function in Sets induced by g is $g_*(f^M) : \Gamma(\mathsf{InSys}, F_1^M) \to \Gamma(\mathsf{InSys}, F_2^M)$. The translation of relation symbols is done in a similar way.

Example 8.8 Consider the formula $\mathsf{Deadlock}_{\mathsf{s_0}}$ analyzed in Example 8.4, namely:

$$\mathsf{Deadlock}_{\mathsf{s_0}} = (\forall a : Act)(\forall s : St)(Tr(a, s_0, s) \Rightarrow (a = \varepsilon) \wedge (s = s_0)).$$

Assume that $\mathsf{Deadlock}_{\mathsf{s_0}}$ is true in M i.e. $\{(a, s) \mid Tr(a, s_0, s)\}^M$ is a subobject of $\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M$ in $\mathcal{E}$.

We know that if $\{(a, s) \mid Tr(a, s_0, s)\}^M$ is a subobject of $\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M$ in $\mathcal{E}$ then $g_*(\{(a, s) \mid Tr(a, s_0, s)\}^M)$ is a subobject of $g_*(\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M)$ in Sets.

We now analyze the form of $g_*(\{(a, s) \mid Tr(a, s_0, s)\}^M)$ and $g_*(\{(a, s) \mid (a = \varepsilon) \wedge (s = s_0)\}^M)$.

(1) $\{(a, s) \mid Tr(a, s_0, s)\}^M$ is the pullback of Tr^M along the terms $t_1(a, s) = a$, $t_2(a, s) = s_0$ and $t_3(a, s) = s$

$$\begin{array}{ccc}
\{(a, s) \mid Tr(a, s_0, s)\}^M & \longrightarrow & \overline{Tr} = \{(a, s', s) \mid \overline{Tr}(a, s', s)\} \\
\downarrow & & \downarrow \\
\overline{Act} \times \overline{St} & \xrightarrow{\langle t_1^M, t_2^M, t_3^M \rangle} & \overline{Act} \times \overline{St} \times \overline{St}.
\end{array}$$

Since g_* commutes with pullbacks,

$$\begin{tikzcd} g_*(\{(a,s) \mid Tr(a,s_0,s)\}^M) \arrow[r] \arrow[d] & g_*(\overline{Tr}) = g_*(\{(a,s',s) \mid \overline{Tr}(a,s',s)\}) \arrow[d] \\ g_*(\overline{Act}) \times g_*(\overline{St}) \arrow[r] & g_*(\overline{Act}) \times g_*(\overline{St}) \times g_*(\overline{St}). \end{tikzcd}$$

is an equalizer in **Sets**. Let S be the system obtained by interconnecting all elements of **InSys**. It is easy to see that up to a bijection, $g_*(\overline{Act}) = Act(S)$, $g_*(\overline{St}) = St(S)$, $g_*(\overline{Tr}) = Tr(S)$. Hence,

$$\begin{tikzcd} g_*(\{(a,s) \mid Tr(a,s_0,s)\}^M) \arrow[r] \arrow[d] & Tr(S) \arrow[d] \\ Act(S) \times St(S) \arrow[r] & Act(S) \times St(S) \times St(S) \end{tikzcd}$$

is an equalizer in **Sets**. Thus, $g_*(\{(a,s) \mid Tr(a,s_0,s)\}^M) = \{(a,s) \in Act(S) \times St(S) \mid (s_0,s) \in Tr^S(a)\}$.

(2) $\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M$ is the pullback in $\mathcal{E}$ of $\{(a,s) \mid a=\varepsilon\}^M$ and $\{(a,s) \mid s=s_0\}^M$, where $\{(a,s) \mid a=\varepsilon\}^M$ is the equalizer

$$\begin{tikzcd} \{(a,s) \mid a=\varepsilon\}^M \arrow[r] & \overline{Act} \times \overline{St} \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\varepsilon"'] & \overline{Act} \end{tikzcd}$$

and $\{(a,s) \mid s=s_0\}^M$ is the equalizer

$$\begin{tikzcd} \{(a,s) \mid s=s_0\}^M \arrow[r] & \overline{Act} \times \overline{St} \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "s_0"'] & \overline{St}. \end{tikzcd}$$

Since g_* preserves equalizers, it follows that

$$\begin{tikzcd} g_*(\{(a,s) \mid a=\varepsilon\}^M) \arrow[r] & Act(S) \times St(S) \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\varepsilon"'] & Act(S). \end{tikzcd}$$

is an equalizer in **Sets**, and

$$\begin{tikzcd} g_*(\{(a,s) \mid s=s_0\}^M) \arrow[r] & Act(S) \times St(S) \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "s_0"'] & St(S) \end{tikzcd}$$

is an equalizer in **Sets**. Thus, $g_*(\{(a,s) \mid a=\varepsilon\}^M) = \{(\varepsilon,s) \mid s \in St(S)\}$ and $g_*(\{(a,s) \mid s=s_0\}^M) = \{(a,s_0(S)) \mid a \in Act(S)\}$. Moreover, $g_*(\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M)$ is the pullback in **Sets** of $g_*(\{(a,s) \mid a=\varepsilon\}^M)$ and $g_*(\{(a,s) \mid s=s_0\}^M)$. Thus, $g_*(\{(a,s) \mid (a=\varepsilon) \wedge (s=s_0)\}^M) = \{(\varepsilon, \Gamma(\mathsf{InSys}, s_0))\}$.

Thus, if $\mathsf{Deadlock}_{s_0}$ is valid in the interpretation M in $\mathcal{E}$ then for every $a \in Act(S)$ and $s' \in St(S)$, $(s_0(S), s') \in Tr^S(a)$ implies $a=\varepsilon$ and $s' = \Gamma(\mathsf{InSys}, s_0)$.

Example 8.9 Now consider Example 8.5. Let

$$\mathsf{Determ} = (\forall s, s', s'' : St)(\forall a : Act)((Tr(a,s,s') \wedge Tr(a,s,s'')) \Rightarrow (s' = s''))$$

be the formula that expresses determinism. The formula Determ is valid in M if $\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M$ is a subobject in $\mathcal{E}$ of $\{(a,s,s',s'') \mid s' = s''\}^M$.

(1) $\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M$ is the pullback in $\mathcal{E}$ of $\{(a,s,s',s'') \mid Tr(a,s,s')\}^M$ and $\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M$. These are computed as equalizers in $\mathcal{E}$. Therefore, $g_*(\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M)$ is the pullback in Sets of $g_*(\{(a,s,s',s'') \mid Tr(a,s,s')\}^M)$ and $g_*(\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M)$. It is easy to see that $g_*(\{(a,s,s',s'') \mid Tr(a,s,s')\}^M) = \{(a,s,s',s'') \mid (s,s') \in Tr^S(a)\}$, and $g_*(\{(a,s,s',s'') \mid Tr(a,s,s'')\}^M) = \{(a,s,s',s'') \mid (s,s'') \in Tr^S(a)\}$. Hence, $g_*(\{(a,s,s',s'') \mid Tr(a,s,s') \wedge Tr(a,s,s'')\}^M) = \{(a,s,s',s'') \mid (s,s') \in Tr^S(a)$ and $(s,s'') \in Tr^S(a)\}$.

(2) $\{(a,s,s',s'') \mid s' = s''\}^M$ is the equalizer

$$\{(a,s,s',s'') \mid s' = s''\}^M \longrightarrow \overline{Act} \times \overline{St} \times \overline{St} \times \overline{St} \overset{\pi_3}{\underset{\pi_4}{\rightrightarrows}} \overline{St}$$

Therefore, $g_*(\{(a,s,s',s'') \mid s' = s''\}^M)$ is the equalizer

$$g_*(\{(a,s,s',s'') \mid s' = s''\}^M) \longrightarrow Act(S) \times St(S) \times St(S) \times St(S) \overset{\pi_3}{\underset{\pi_4}{\rightrightarrows}} St(S)$$

in Sets, i.e. $g_*(\{(a,s,s',s'') \mid s' = s''\}^M) = \{(a,s,s',s'') \mid a \in Act(S), s, s', s'' \in St(S), s' = s''\}$.

Thus, if Determ is valid in interpretation M in $\mathcal{E}$ then for every $a \in Act(S)$ and every $s, s', s'' \in St(S)$, $(s,s') \in Tr(a)$ and $(s,s'') \in Tr(a)$ implies $s' = s''$.

8.7.3 Relationship between Sh(InSys) and Sh(Sys(InSys), J)

In Proposition 8.26 we showed that there is a geometric morphism $f : \mathsf{Sh(InSys)} \to \mathsf{Sh(Sys(InSys), J)}$, with inverse and direct image functors

$$f^* : \mathsf{Sh(Sys(InSys), J)} \to \mathsf{Sh(InSys)} \text{ and } f_* : \mathsf{Sh(InSys)} \to \mathsf{Sh(Sys(InSys), J)}$$

described, for sheaves F on $(\mathsf{Sys(InSys)}, J)$ and G on $\Omega(\mathsf{InSys})$, by

$$f^*(F) = a(F \circ \pi) : \Omega(\mathsf{InSys})^{op} \to \mathsf{Sets} \qquad f_*(G) = G \circ \phi : \mathsf{Sys(InSys)}^{op} \to \mathsf{Sets}.$$

f^* preserves the validity of geometric axioms, and f_* the validity of coherent axioms.

Let S be the colimit in $\mathsf{SYS_{il}}$ of all elements of InSys.

Let **1** be the category with only one object, namely $*$, and only one arrow, namely the identity arrow.

Let $\phi_S : \mathbf{1} \to \mathsf{Sys(InSys)}$ defined by $\phi(*) = S$, and $\phi(id_*) = id_S$, and let $\psi : \mathsf{Sys(InSys)} \to \mathbf{1}$ be defined for every $T \in \mathsf{Sys(InSys)}$ by $\psi(T) = *$ and for every $h : T_1 \hookrightarrow T_2$ by $\psi(h) = id_*$.

Then there exist natural transformations $\eta : id_{\mathsf{Sys(InSys)}} \to \phi_S\psi$ and $\varepsilon : \psi\phi_S \to id_{\mathbf{1}}$.

For every system T in $\mathsf{Sys(InSys)}$, $\eta_T : T \hookrightarrow S$ is the canonical inclusion, and for $*$ in $\varepsilon_* : * \to *$ is the identity.

Moreover, the following diagrams commute:

$$\begin{tikzcd} \psi \arrow[d, "\psi\eta"'] \arrow[dr, "id_\psi"] & \\ \psi\phi_S\psi \arrow[r, "\varepsilon\psi"'] & \psi \end{tikzcd} \qquad \begin{tikzcd} \phi_S \arrow[r, "\eta\phi_S"] \arrow[dr, "id_{\phi_S}"'] & \phi_S\psi\phi_S \arrow[d, "\phi_S\varepsilon"] \\ & \phi_S \end{tikzcd} \tag{8.7}$$

(for every object T of $\mathsf{Sys(InSys)}$, $\psi(\phi_S(\psi(T))) = * = \psi(T)$ and $\phi_S(\psi(\phi_S(*))) = S = \phi_S(*)$).

Thus, ψ is left adjoint to ϕ_S. Let J_* be a covering relation on $\mathbf{1}$ defined by $J(*) = \{\{*\}\}$. It is easy to see that it induces a Grothendieck topology on $\mathbf{1}$. It is obvious that ϕ_S preserves covers,

Therefore, by Theorem 8.28, ψ and ϕ_S induce a geometric morphism

$$h_S : \mathsf{Sh(Sys(InSys), J)} \to \mathsf{Sh}(\mathbf{1}, \mathsf{J}_*)$$

with inverse and direct image functors described, for sheaves F on $\mathsf{Sh}(\mathbf{1}, \mathsf{J}_*)$ and G on $\mathsf{Sh(Sys(InSys), J)}$ by $h_S^*(F) = a(F \circ \psi)$ and $h_{S_*}(G) = G \circ \phi$.

Since $\mathsf{Sh}(\mathbf{1}, \mathsf{J}_*) = \mathsf{Sets}$, the direct image functor $h_{S_*} : \mathsf{Sh(Sys(InSys), J)} \to \mathsf{Sets}$ is defined by $h_{S_*}(F) = F(S)$.

8.7.4 Concluding Remarks

In conclusion, we have the following (direct and inverse geometric functions):

$$\begin{tikzcd} & & & \mathsf{Sets} \\ \mathsf{Sets} & \mathsf{Sh(Sys(InSys), J)} \arrow[l, "h_{S_*}"'] \arrow[r, shift left, "f^*"] & \mathsf{Sh(InSys)} \arrow[l, shift left, "f_*"] \arrow[ur, "f_i^*"] \arrow[dr, "g_*"'] & \\ & & & \mathsf{Sets} \end{tikzcd}$$

A geometric formula ϕ, holds (internally) in $\mathsf{Sh(InSys)}$ if and only if it holds in S_i for every $S_i \in \mathsf{InSys}$.

If ϕ is a cartesian axiom (with respect to a cartesian theory $\mathbb{T}$) and it holds (internally) in $\mathsf{Sh(InSys)}$, then it holds in S, where S is the system obtained by interconnecting all elements in InSys.

If ϕ is a cartesian axiom (with respect to a cartesian theory $\mathbb{T}$) and it holds (internally) in $\mathsf{Sh(InSys)}$, then it also holds (internally) in $\mathsf{Sh(Sys(InSys), J)}$.

If ϕ is a cartesian axiom (with respect to a cartesian theory $\mathbb{T}$) and it holds (internally) in $\mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}), \mathsf{J})$, then it holds also in S, where S is the system obtained by interconnecting all elements in InSys.

If additionally all the elements in InSys are independent (do not have common subsystems) (we assumed that there are finitely many elements in InSys) then the global section functor g_* also preserves image factorization, hence it preserves the truth of all formulas that contain only conjunction and existential quantification.

Example 8.10 *Let ϕ_1 be the formula that expresses determinism:*

$$\phi_1 = (\forall s : St)(\forall a : Act)(\forall s' : St)(\forall s'' : St)$$
$$\left[(Tr(a, s, s') \wedge Tr(a, s, s'')) \Rightarrow s' = s'' \right].$$

ϕ_1 is a cartesian axiom. The theory presented above and the Examples 8.7 and 8.9 show the following:

If ϕ_1 is true in all systems in InSys *(i.e. if for every $S_i \in$* InSys*, $(s, s') \in Tr^{S_i}(a)$ and $(s, s'') \in Tr^{S_i}(a)$ implies $s' = s''$) then ϕ_1 is true (internally) in* $\mathsf{Sh}(\mathsf{InSys})$*, and it is also preserved by the global section functor. Moreover, it is also true (internally) in $\mathcal{F} = \mathsf{Sh}(\mathsf{Sys}(\mathsf{InSys}))$ and its truth is preserved by h_{S_*}.*

Example 8.11 *We analyze the formula that expresses deadlock freedom:*

$$\phi_2 = (\forall s : St)(\exists a : Act)(\exists s' : St)(Tr(a, s, s') \wedge (s \neq s')).$$

It can be seen that this formula is not a geometric axiom (it contains the negation sign), hence it is not necessarily preserved by direct and inverse geometric morphisms, in particular it is not preserved by the global section functor, nor by the stalk functors.

Therefore, we cannot infer anything about the system obtained by interconnecting a family of systems, if we know that the components are deadlock free. Intuitively, it is easy to see that it is possible that all the systems in InSys *can be deadlock free while in the system obtained by interconnecting them deadlock may occur, because the corresponding actions a_i that can be "locally" executed at a given state s (or the final states s'_i) may not form a matching family.*

Example 8.12 *Consider now the formula expressing fairness of execution:*

$$\phi_3 = (\forall h : B'(\mathbb{N}))(\forall a : Act)(\forall i : \mathbb{N})$$
$$((\exists s : St)Tr(a, \pi_{St}(h(i)), s) \Rightarrow (\exists j)(j \geq i \ \wedge \ \pi_{Act}(h(j)) = a))$$

ϕ_3 is a geometric axiom. However it is not a coherent axiom. Hence, it is not preserved by the global section functor.

Assume that the systems in InSys *are independent, in the sense that they do not have common nonempty subsystems. In this case ϕ_3 is preserved by the global section functor.*

Of course, many other properties of systems can be formulated in the language $\mathcal{L}$ as in the above examples. We will now present some classes of properties of systems taken from [Krö87] and will explain under which conditions these classes of properties can be expressed in $\mathcal{L}$, and use the theoretical results established so far for explaining the way these properties are preserved by interconnection.

Example 8.13 *Taking a more general viewpoint, we now consider translations in the language of sheaves of classes of properties of programs adapted from [Krö87]. Let $h : B'(\mathbb{N})$ be a possible behavior in time in* $\mathsf{Sh}(\mathsf{InSys})$.

(a) Safety (or invariance) properties: *Safety properties are properties of the form $(\forall j)(P(h(0)) \Rightarrow Q(h(j)))$, where P and Q are formulae in $\mathcal{L}$. Examples of such properties are:*

(a1) Partial Correctness: $(\forall j)(P(h(0)) \wedge \mathsf{Final}(h(j)) \Rightarrow Q(h(j)))$,

(a2) Global Invariance of Q: $(\forall j)(P(h(0)) \Rightarrow Q(h(j))$.

(b) Liveness (or eventuality) properties: *Liveness properties are properties of the form $P(h(0)) \Rightarrow (\exists j)(Q(h(j)))$. Examples of such properties are:*

(b1) Total correctness and termination: $P(h(0)) \Rightarrow (\exists j)(\mathsf{Final}(h(j)) \wedge Q(h(j)))$,

(b2) Accessibility: $(h(0) = s_0) \Rightarrow (\exists j)(h(j) = s_f)$.

(c) Precedence properties: *Precedence properties are properties of the form*

$$(\forall j)(P(h(0)) \wedge A(h(j))) \Rightarrow Q(h(j)).$$

(a1) *Partial Correctness:* We make the following assumptions:

(1) the final states form a subsheaf St_f of St (this happens for example if in the definition of a system final states are specified by additional constraints, and in defining the covering relation this information is also used). Let $i : St_f \to St$ be the inclusion.

In this case the property $\mathsf{Final}(s)$ can be expressed by $(\exists s' : St_f)(i(s') = s)$. Since St_f is a subobject of St, it is easy to see that if such a state exists, it is unique.

(2) the properties P and Q can be expressed in the language $\mathcal{L}$, and can be interpreted in Sets (for every system S_i in InSys expressing the corresponding property of S_i) as well as in $\mathcal{E}$ and $\mathcal{F}$.

Then, the formula that describes partial correctness is preserved under inverse image functors if in the definitions of P and Q appear only conjunction, disjunction and existential quantification.

It is preserved by both direct and inverse functors if in the definitions of P and Q occur only conjunctions and the existential quantifier (requiring uniqueness; existential quantification without requiring uniqueness is allowed if all systems in InSys are independent).

(a2) *Global Invariance of Q:* Assume that the properties P and Q can be expressed in the language $\mathcal{L}$, and can be interpreted in Sets (for every system S_i in InSys expressing the corresponding property of S_i) as well as in $\mathcal{E}$ and $\mathcal{F}$.

The formula that describes the fact that Q is a global invariant is preserved under inverse image functors if in the definitions of P and Q appear only conjunction, disjunction and existential quantification. It is preserved by both direct and inverse functors if in the definitions of P and Q only conjunctions and the existential quantifier (requiring uniqueness; existential quantification without requiring uniqueness is allowed if all systems in InSys are independent).

(b1) *Total Correctness and Termination:* We make the following assumptions:

(1) the final states form a subsheaf St_f of St,

(2) the properties P and Q can be expressed in the language $\mathcal{L}$, and can be interpreted in Sets (for every system S_i in InSys expressing the corresponding property of S_i) as well as in $\mathcal{E}$ and $\mathcal{F}$.

It was shown that under these assumptions $\mathsf{Final}(s)$ can be expressed by $(\exists s' : St_f)(i(s') = s)$. Therefore, the formula describing total correctness and termination is a geometric formula. It is therefore preserved under inverse image functors if in the definitions of P and Q appear only conjunction, disjunction and existential quantification. It is preserved by both direct and inverse functors if in the definitions of P and Q only conjunctions and the existential quantifier (requiring uniqueness; existential quantification without requiring uniqueness is allowed if all systems in InSys are independent).

(b2) *Accessibility:* The formula describing the accessibility of s_f from s_0 in h, is a geometric axiom, hence it is preserved by inverse image functors.

If the existence of j is provably unique (with respect to some cartesian theory $\mathbb{T}$), or if all systems in InSys are independent, then the formula is also preserved by direct image functors.

(c) *Precedence properties:* Assume that the properties P and Q can be expressed in the language $\mathcal{L}$.

Formulas describing precedence properties are preserved under inverse image functors if in the definitions of P, A, and Q appear only conjunction, disjunction and existential quantification. They are preserved by both direct and inverse functors if in the definitions of P, A, and Q only conjunctions and the existential quantifier (requiring uniqueness; existential quantification without requiring uniqueness is allowed if all systems in InSys are independent).

Chapter 9

Applications

This chapter contains applications of the theoretical results in the previous chapters. We present several problems that occur in distributed systems. The solutions to some of these problems will be presented as algorithms.

- given a set of actions decide using only "local information" if they can be performed in parallel,
- "parallelize" a sequence of atomic actions,
- study whether plans elaborated "locally" can be "glued together" to a global plan,
- study the link between properties of component parts and properties of the system obtained by interconnecting them.

We would like to point out that many of the problems that appear in modeling distributed systems can be formulated as particular cases of "divide-and-conquer" problems: the domain of the problem is "split" (e.g. by finding a cover for a given system), then "local subproblems" are solved locally and finally the local solutions are combined (if possible) to a global solution.

Finally, we will present a case study, part of our joint research together with J. Pfalzgraf and K. Stokkermans, towards possible uses of fibered models for describing robotics scenarios in which space- and time-dependent formulae appear.

9.1 Checking Whether a Set of Atomic Actions can be Performed in Parallel in a Distributed System

Let a family InSys of interacting systems be given, satisfying the conditions imposed in Chapter 8. We assume that no element in InSys is the colimit of other elements of InSys. $\mathsf{Sys}(\mathsf{InSys})$ denotes the category of all systems obtained by interconnecting elements in InSys. In Chapter 8 we studied the properties of $\mathsf{Sys}(\mathsf{InSys})$. We showed that a Grothendieck topology J can be defined on

$\mathsf{Sys(InSys)}$ such that a family $\mathcal{S}$ is a covering family for a system S if and only if $\mathcal{S}$ contains all the elements of InSys that are transition-connected subsystems of S. Therefore, a finest[1] cover exists for every object S in $\mathsf{Sys(InSys)}$, namely $\{S_i \in \mathsf{InSys} \mid S_i \hookrightarrow S\}$.

Description of the problem:

Given: a system S in $\mathsf{Sys(InSys)}$, a state s_0 of S and a set of actions $f \subseteq A_U$.

Task: Decide whether all the actions in f can be executed in parallel in state s_0 in S, and if so compute the final state.

In order to make the problem computationally tractable, we make the additional assumption that InSys is finite (and all its elements are finite systems). In this situation all objects in the site $(\mathsf{Sys(InSys)}, J)$ are finite, and a finest cover exists for each object and is finite.

Usually, a system S is specified by giving a cover $\mathcal{S} = \{S_i \mid i \in I\}$ for S by elements of InSys. If the system S is complex (i.e. is obtained by interconnecting many components), it may be quite time consuming to actually construct S and then "globally" decide if the action f can be performed in S. Many of the systems that compose S may not even be involved in executing the action.

A "modular approach" seems to be more appropriate. Namely, we do not construct S, but analyze the components in $\mathcal{S}$ one by one, and see if the corresponding restrictions of f are admissible parallel actions in these components. If this is the case, we obtain a compatible family of elements $\{f_i \mid i \in I\}$, where $f_i = f_{|S_i}$ and $f_i \in Act(S_i)$, that has a unique amalgamation, namely $f_{|S} \in Act(S)$.

If additionally $f \subseteq A_S$, then it follows that $f \in Act(S)$.

The final state can then be computed from the local final states by applying Proposition 7.14 (if transitions of parallel actions are obtained according to rule **(Gluing)**) or Proposition 8.6 (if all the actions are deterministic and transitions of parallel actions are obtained according to rule **(Independence)**).

The data can be represented as follows:

Data Structure:

- InSys: a finite set of finite systems.
- Every $S_i \in \mathsf{InSys}$ will be represented by
  – X_i, the set of control variables (ordered list);
  – A_i, the set of atomic actions (ordered list);
  – C_i, the set of constraints on atomic actions;
    of the form $a_i \wedge a_j = 0$ or $a_i = a_j$ (ordered list of constraints);
  – $St(S_i)$ the set of states, where every state $s \in St(S_i)$ is represented as an ordered list
  $$((x_1, v_1)(x_2, v_2) \ldots (x_n, v_n));$$
  – $Tr(a) \subseteq St(S_i) \times St(S_i)$, the set of transitions induced by a, for every $a \in A_i$;

[1] A *finest cover* is a cover none of whose elements can be further decomposed via a cover.

- Every element S of $(\mathsf{Sys}(\mathsf{InSys}), J)$ will be represented by its finest cover.

Distributing Parallel Actions in Distributed Systems

Given :

A finite family of finite systems InSys,

a system S in $\mathsf{Sys}(\mathsf{InSys})$ (a finest cover $\mathcal{S}$ for S),

an initial state s_0 of S,

a parallel action $f = \{f_1, \dots, f_n\}$.

Task : Decide whether f can be performed at state s_0 in S. If so, determine the active subsystems of S, and the final state.

Sketch of the Algorithm :

[Step0] *Decompose S*

Let $\mathcal{S} = \{S_j \mid j \in J\}$ be the finest cover for S in $(\mathsf{Sys}(\mathsf{InSys}), J)$.

[Step1] *Check if f is contained in A_S*

[Step1a] *Solve Subproblems*
For every $i \in \{1, \dots, n\}, j \in J$ test if $f_i \in A_{S_j}$.
Let $\{S_{i_1}, \dots, S_{i_k}\}$ be the set of those systems in the cover that contain at least one action in f.
For every $i \in \{i_1, \dots, i_k\}$ let $f_{|S_i}$ be the set of all actions in f contained in A_{S_i}.

[Step1b] *Compose*
If there exists some action in f that is not contained in A_{S_j} for any j then *Return*(f not known in S).
Otherwise f is known in S; goto [Step2].

[Step2] *Check if $f = \{f_1, \dots, f_n\} \in Act(S)$*

[Step2a] *Solve Subproblems*
For every $j \in J$

if $f_{|S_j} \notin Act(S_j)$ then *Return*($f \notin Act(S)$).
if $f_{|S_j} \in Act(S_j)$ and cannot be applied in state $s_{0|X_j}$ then *Return*(f cannot be applied at s_0).
if $f_{|S_j}$ can be applied in state $s_{0|X_j}$ let s_j be the state of S_j reached after performing $f_{|S_j}$.

[Step2b] *Compose*
Use the sheaf property of *Act* to deduce that $f \in Act(S)$.
Use the sheaf property of *St* to glue the family of locally computed states $\{s_j\}_{j \in J}$ together. The amalgamation of this family is in this case the final state.

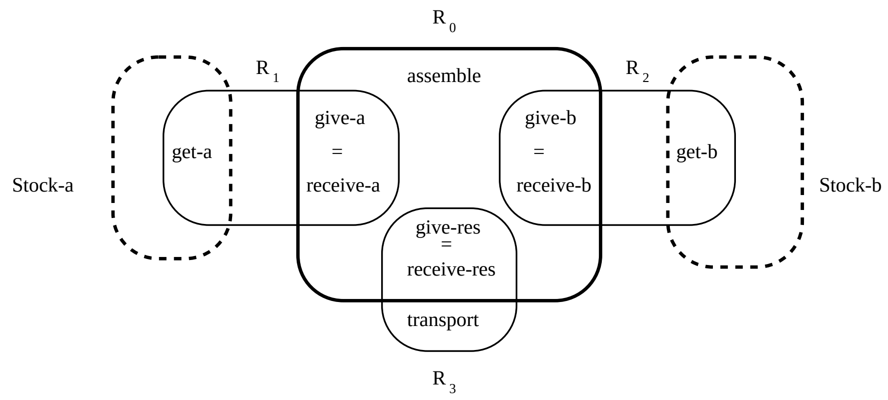


Figure 9.1: Actions

Example 9.1 Consider the situation described in Section 6.1. Let R_0, R_1, R_2, R_3 be the four robots, and let S_0, S_1, S_2, S_3 be the corresponding systems.

We have:

$$\mathsf{InSys} = \{S_0, S_1, S_2, S_3\} \cup \{\bigcap_{i \in I} S_i \mid I \subseteq \{0, 1, 2, 3\}\}.$$

Input:
(1) S, the system obtained by interconnecting S_0, S_1, S_2, S_3,
(2) s_0 a state of S, and
(3) $f = \{\mathsf{bring\text{-}a}, \mathsf{bring\text{-}b}\}$.

Task:
Decide whether f can be performed at s_0 in S, and if so, which are the subsystems of S that are active and what is the final state.

We can proceed as follows:

The actions in f are bring-a and bring-b. bring-a is known in the systems S_1 and S_0; bring-b is known in the systems S_2 and S_0. We know that the parallel action { bring-a, bring-b } is allowed in S_0.

If bring-a can be executed at $s_{0|S_1}$ in S_1 and bring-b can be executed at $s_{0|S_2}$ in S_2, and bring-a || bring-b can be executed at $s_{0|S_0}$ in S_0, then by the properties of transitions in Sys(InSys) (cf. Section 8.1.1) it follows that $f = \{$ bring-a, bring-b $\}$ can be executed at s_0 in S. Additionally, it can be seen that the subsystems (in InSys) of S that have a rôle in performing f are S_0, S_1, S_2.

We can compute the transition (in S) associated with this action by computing (locally) the transitions associated with f, and then gluing the final states together.

Example 9.2 Consider now $f = \{$ bring-a, give-res $\}$.

The actions in f are bring-a and give-res. bring-a is known in S_0 and S_1 and give-res is known in S_0 and S_3.

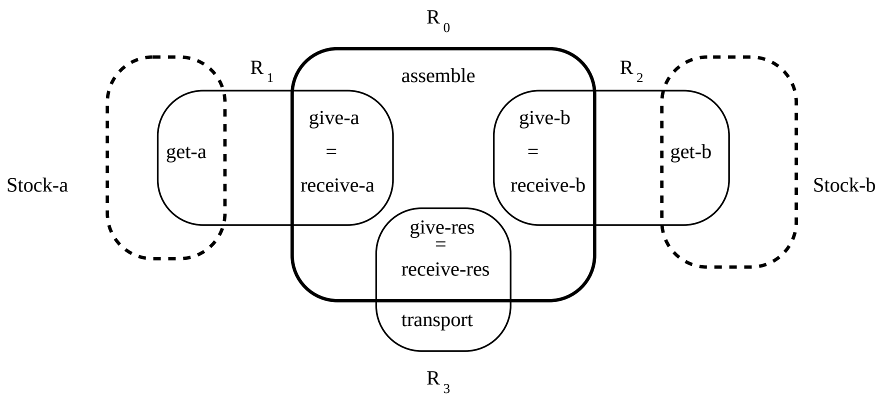


Figure 9.2: Actions

Hence, they are both known in S_0, but in S_0 we have the additional constraints $\mathsf{bring\text{-}a} \wedge \mathsf{receive\text{-}res} = 0$ and $\mathsf{receive\text{-}res} = \mathsf{give\text{-}res}$, so they cannot be performed in parallel in S_0. It follows that they cannot be performed in parallel in S.

9.2 Parallelizing Global Plans

Plans in complex systems can be described as successions of actions. We illustrate the notion, referring again to the example described in Section 6.1.

Consider the following problem:

Given: Initial state:

3 pieces of type a are in $\mathsf{Stock_a}$, and
2 pieces of type b are in $\mathsf{Stock_b}$.
Nothing on the assembly bench.
No robot holds a piece.

(i.e. $s_a = 3, s_b = 2, h_a = h_b = 0, p_a = p_b = p_r = 0$).

Task: Assemble two pieces of type r and move them to the stock.

A possible plan is the following:

get-a, get-b, bring-a, bring-b, assemble, give-res, transport,
get-a, get-b, bring-a, bring-b, assemble, give-res, transport.

Assume that the following constraints on actions are imposed:

R_0 :

$\mathsf{assemble} \wedge \mathsf{bring\text{-}a} = 0$, $\mathsf{assemble} \wedge \mathsf{bring\text{-}b} = 0$, $\mathsf{assemble} \wedge \mathsf{give\text{-}res} = 0$,
$\mathsf{bring\text{-}a} \wedge \mathsf{give\text{-}res} = 0$, $\mathsf{bring\text{-}b} \wedge \mathsf{give\text{-}res} = 0$,
$\mathsf{bring\text{-}a} = \mathsf{receive\text{-}a} = \mathsf{give\text{-}a}$, $\mathsf{give\text{-}res} = \mathsf{receive\text{-}res}$,

R_1 :
get-a $\wedge$ bring-a $= 0$, bring-a $=$ give-a $=$ receive-a,

R_2 :
get-a $\wedge$ bring-a $= 0$, bring-a $=$ give-a $=$ receive-a, get-b $\wedge$ bring-b $= 0$, bring-b $=$ give-b $=$ receive-b,

R_3 :
give-res $\wedge$ transport $= 0$, give-res $=$ receive-res.

Therefore, the plan given before can for instance be parallelized as follows:

[get-a || get-b], [bring-a || bring-b], [assemble || get-a || get-b], give-res, [transport || bring-a || bring-b], assemble, give-res, transport.

At this level of generality we assume that all actions need the same amount of time. The parallelization of sequences of actions (considered "plans") when all actions need the same amount of time, taking into account which actions are independent can be done for instance by computing the Foata normal form (for the definition see Section 4.3.1). For an algorithm for computing the Foata normal form we refer to [Die90] p.30. In what follows we just give the main idea of the algorithm:

Let S be a system and let $G(S) = (A_S, D(S))$ be the associated dependency graph. Let $\{G_1, \ldots, G_k\}$ be a covering of $G(S)$ by cliques (i.e. complete subgraphs), $G_i = (A_i, A_i \times A_i)$ for every $i \in \{1, \ldots, k\}$. By Theorem 4.22 there exists a canonical embedding $M(S) \hookrightarrow \prod_{i=1}^{k} A_i^*$.

The algorithm in [Die90] has an input a sequence w of atomic actions in $A_S = \bigcup_{i=1}^{n} A_i$. The algorithm can be schematically represented as follows:

Data structure

- Every system S will be represented by its dependence graph $(A, D(S))$, (A : list, $D(S)$: list of pairs of elements in A),
- Every clique S_i is represented by its set of vertices A_i (list),
- A word $w \in A^*$ is represented as a list.

Foata Normal Form

Given :

A system $S = (A, D(S))$ and a finite covering of its dependency graph by cliques $\{S_i \mid i \in I\}$,

$w \in A^*$

Task : Find elementary steps $F_1, \ldots F_m$ with $[w] = [F_1] \ldots [F_m]$ in $M(S)$ such that for every $i \in \{1, \ldots m\}$ $a, b \in F_i$ implies $(a, b) \notin D(S)$ and for every $i \in \{2, \ldots m\}$ and every $a \in F_i$ there exists a $b \in F_{i-1}$ such that $(a, b) \in D(S)$.

Sketch of the Algorithm :

S := 1,

while $w \neq \varepsilon$ do:

$F := min(w)$,
$S := S \cdot F$,
$w := F^{-1}w$

endwhile

end.

If $w = w_1w_2$ in $M(S)$ then $w_1^{-1}w$ is by definition w_2. $min(w)$ provides the set of the "minimal" elements of w, i.e. the set of F of atomic actions a with the property that a is the first symbol in $w_{|A_i}$ for every i such that $a \in A_i$.

Finding Minimal Elements (min(w))

Given: A system S and a covering of its dependency graph by cliques,
$w \in M(S)$.

Task : Find the set F of atomic actions a with the property that a is the first symbol in w_{A_i} for every i such that $a \in A_i$.

Sketch of the Algorithm min(w) :

$b := true$,

$F := \emptyset$,

$I := \{i \in \{1, \ldots, k\} \mid w_{|A_i} \neq \varepsilon\}$,

while $I \neq \emptyset$ do

choose $i \in I$, $I := I \backslash \{i\}$,
$a := first(w_{|A_i}), J := \{i \mid a \in A_i\} \backslash \{i\}$,
while $J \neq \emptyset$ and b do

choose $j \in J$,
if $j \in I$ and $a = first(w_{|A_j})$ then $I := I \backslash \{i\}, J := J \backslash \{j\}$,
else $b := false$

if b then $F := F \cup \{a\}$,
else $b := true$

endwhile,

return F

end.

Restrict a word to a subalphabet ($w_{|A_i}$)

Given: A word $w \in A^*$ and $A_i \subseteq A$,

Task : Find the restriction $w_i = w_{|A_i} \in A_i^*$ of w to A_i.

Sketch of the Algorithm :

$w_i := nil$,

while $w \neq nil$ do

$a := first(w)$,
$w := rest(w)$,
if $a \in A_i$ then $w_i := concatenate(w_i, (a))$,

endwhile end.

The parallelization in the example above can be obtained choosing the covering by cliques of $D(S)$ as shown in Figure 9.3.

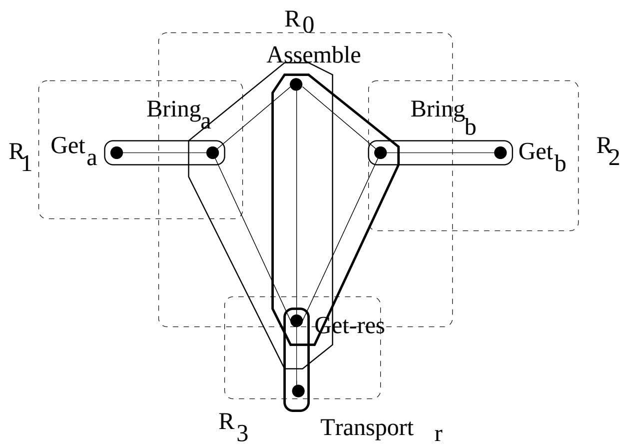


Figure 9.3: Covering of the Dependence Graph by Cliques

9.3 Putting Together Local Plans

Another problem that often arises in decentralized planning is that of *putting together compatible local plans* to a global plan.

Consider the following example:

Given: Initial state:
3 pieces of type a are in $\mathsf{Stock_a}$, and
2 pieces of type b are in $\mathsf{Stock_b}$.
Nothing on the assembly bench.
The robots hold no pieces.

Task: Assemble one piece of type r and move it to the stock.

We assume that every agent knows the problem and elaborates a "local" plan for solving it. For instance, assume that the local plans are as follows:

R0: bring-a, bring-b, assemble, give-res,

R1: get-a, bring-a,

R2: get-b, bring-b,

R3: give-res, transport.

It is easy to see that these plans are compatible, in the sense that for every two systems, if we "delete" actions that do not belong to both of them, we obtain the same sequences of actions.

These plans can be "glued together" to a global plan, for instance to:

get-a, get-b, bring-a, bring-b, assemble, give-res, transport.

This global plan can be further parallelized as discussed before, for example to:

[get-a || get-b], [bring-a || bring-b], assemble, give-res, transport.

Theorem 4.23 (based on the results in [CM85], [MP86] and [Die90]) shows that it is always possible to glue together compatible local plans in the above example.

The theorem states that for every finite system S obtained by interconnecting the family $\{S_1, \ldots, S_n\}$, if $G = \bigcup_{i=1}^n G(S_i)$ then the following assertions are equivalent:

(1) Every compatible family of local plans $\{w_1, \ldots, w_n\}$, where for every i, $w_i \in A_i^*$, can be *glued* to a global plan $w \in (\bigcup_{i=1}^n A_i)^*$.

(2) Every chordless cycle in the graph G is a cycle in a subgraph $G(S_i)$ for some $i \in \{1, \ldots, n\}$.

In the example described above the condition is satisfied, as can be seen from the dependence graph of the system represented in Figure 9.4.

From the proof of Theorem 4.23 given in [MP86] (for the case when $G(S_i)$ is complete for every $S_i \in \mathsf{InSys}$) it is easy to deduce a way of constructing $w \in A^* = (\bigcup_{i=1}^n A_i)^*$ such that $w_{|A_i} = w_i$ for every matching family $\{w_1, \ldots, w_n\}$, where $w_i \in A_i^*$ for every $i \in \{1, \ldots, n\}$.

In what follows we present a method that, given a a matching family of elements $w_i \in M(S_i)$, provides a way of computing the element $w \in M(S)$ with the property $w_{|A_i} = w_i$, under the assumption that every chordless cycle in the graph G is a cycle in a subgraph $G(S_i)$ for some $i \in \{1, \ldots, n\}$.

We first recall that for every system S every trace $w = a_1 \ldots a_n \in M(S)$ can be represented by a directed graph (that has a set of vertices $\{v_1, \ldots, v_n\}$, each

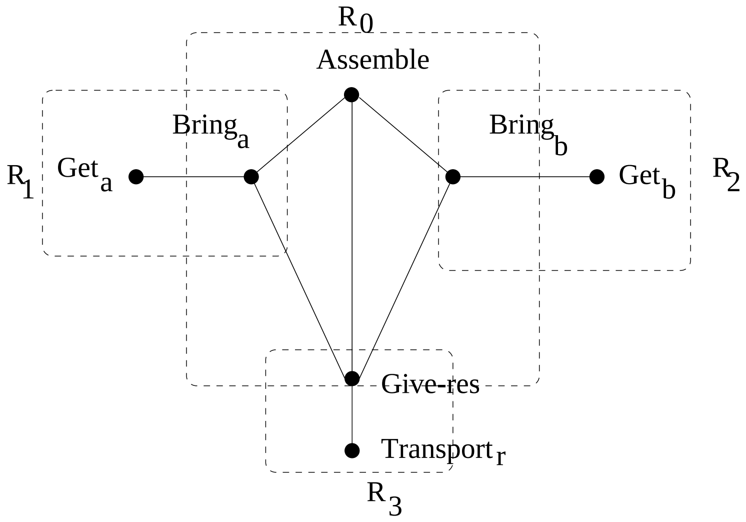


Figure 9.4: Dependence Graph

vertex v_i being labelled with the value a_i, and for every $i < j$ there is a directed edge from v_i to v_j if and only if $(x_i, x_j) \in D(S)$). In what follows, in order to distinguish different occurrences of the same symbol a we will represent these occurrences by $(a,1),(a,2),\ldots,(a,k)$ (where (a,k) occurs before (a,l) if and only if $k < l$). For two occurrences (a,k) and (b,l) at vertex v_i respectively v_j we say that (a,k) occurs before (b,l) if there is a directed path from v_i to v_j. It is easy to see that for every two occurrences (a,k) and (b,l) in w either (a,k) occurs before (b,l) or (b,l) occurs before (a,k), but not both.

Lemma 9.1 *Let $\{w_1,\ldots,w_n\}$ be a matching family of elements $w_i \in A_i^*$.*

(1) Let $a \in A$. If a occurs in some $w_i \in A_i^*$ then it occurs in every $w_j \in A_j^*$ such that $a \in A_j$.

For every w_i with $a \in A_i$, a has a finite number of occurrences in w_i. We represent these occurrences by $(a,1),(a,2),\ldots,(a,k)$ (where (a,k) occurs before (a,l) if and only if $k < l$).

(2) If $a \in A_j$ for some other S_j the number of occurrences of a in w_i and w_j is the same.

Let π be the relation on the set of occurrences defined by $(a,k)\pi(b,l)$ if and only if $a,b \in A_i$ for some i, and (a,k) occurs before (b,l) in w_i.

(3) Assume that every chordless cycle in the graph G is a cycle in a subgraph $G(S_i)$ for some $i \in \{1,\ldots,n\}$. Then there is an occurrence (a,k) that has no π-predecessor.

Proof: (1) and (2) follow immediately from the compatibility of the family $\{w_1, \ldots, w_n\}$.

(3) Consider the directed graph (V, E, λ) obtained by taking the union of the directed graphs $\{G(w_i) \mid i \in I\}$ corresponding to the family $\{w_i \mid i \in I\}$, by identifying the vertices labeled with the same occurrence symbol. Such a union exists because the family $\{w_i \mid i \in I\}$ is a matching family. Note that for every $i \in I$, $G(w_i)$ does not contain any directed cycle.

Assume that every occurrence (a, k) has a π-predecessor. In this case it is easy to see that the graph (V, E, λ) contains (directed) cycles. Let $(x_1, \ldots, x_n)$ be a directed cycle of minimal length. It is easy to see that the assumption that every chordless cycle is a cycle in a subgraph $G(S_i)$ for some $i \in \{1, \ldots, n\}$ implies that $n \geq 3$. (If $(x, y) \in E$ then $(y, x) \notin E$; otherwise both "$\lambda(x)$ precedes $\lambda(y)$" and "$\lambda(y)$ precedes $\lambda(x)$" hold in some w_i, which is absurd.)

We show that in this case the actions in the labels $\lambda(x_1), \ldots, \lambda(x_n)$ are all different. Assume that for $i \neq j$ we have $\lambda(x_i) = (a, k_i)$ and $\lambda(x_j) = (a, k_j)$. Without loss of generality we can assume that i, j are the minimal indices with this property, $i < j$ and $i - 1 \neq j \pmod{n}$. Let $\lambda(x_{i-1}) = (b, l)$. Since there is an edge between x_{i-1} and x_i, it follows that $(b, a) \in D(S_m)$ for some $m \in I$. Thus, we have either $(x_{i-1}, x_j) \in E$ or $(x_j, x_{i-1}) \in E$. In the first case we found a shorter cycle, namely $(x_1, \ldots, x_{i-1}, x_j, \ldots, x_n)$, contradiction. In the second case, $(x_j, x_{i-1}, x_i, \ldots, x_{j-1})$ is a shorter cycle if $j + 1 \neq i - 1 \pmod{n}$. If $j + 1 = i - 1 \pmod{n}$, we distinguish again several cases:

Case 1: $i + 1 = j \pmod{n}$. In this case we have

$$\begin{tikzcd}
x_{i-1} \arrow[rr] & & x_i \arrow[dl] \\
& x_j \arrow[ul] &
\end{tikzcd}$$

This is impossible since on the one hand it follows that (a, k_j) precedes (b, j) in S_m, which precedes (a, k_i), and on the other hand, (a, k_i) precedes (a, k_j).

Case 2: $i + 1 \neq j \pmod{n}$. Assume that $\lambda(x_{i+1}) = (c, r)$. From the fact that $(x_i, x_{i+1}) \in E$ it follows that $(a, c) \in D(S_p)$ for some $p \in I$, hence either $(x_{i+1}, x_j) \in E$ or $(x_j, x_{i+1}) \in E$.

If $(x_{i+1}, x_j) \in E$ we obtain on the one hand that (a, k_j) precedes (a, k_i) in w_m and on the other hand that (a, k_i) precedes (a, k_j) in w_p. Contradiction.

If $(x_j, x_{i+1}) \in E$ we can find a shorter cycle, namely $(x_{i+1}, x_{i+2}, \ldots, x_j\}$ if $i + 1 < j \leq n$ or $\{x_{i+1}, x_{i+2}, \ldots, x_n, x_1, \ldots, x_{j-1}\}$ if n is between $i + 1$ and j. Contradiction.

Thus, all the labels $\lambda(x_1), \ldots, \lambda(x_n)$ are different, and $\{\lambda(x_1), \ldots, \lambda(x_n)\}$ is a chordless cycle in the dependence graph $G(S)$, hence, it is contained in the dependence graph $G(S_i)$ for some $i \in I$. This impossible because the restriction to S_i of the graph (V, E, λ) is the graph $G(w_i)$ which does not contain cycles. $\square$

Lemma 9.2 *Let $\{S_1, \ldots, S_n\}$ be a cover of S. Assume that every chordless cycle in the graph G is a cycle in a subgraph $G(S_i)$ for some $i \in \{1, \ldots, n\}$. Let*

$\{w_1, \ldots, w_n\}$ where for every $i \in \{1, \ldots, n\}$, $w_i \in M(S_i)$ be a matching family. Then there exists $w \in M(S)$ such that for every $i \in \{1, \ldots, n\}$, $w_{|S_i} = w_i$.

Proof: We proceed by induction on the number of occurrences in the family $\{w_1, \ldots, w_n\}$.

If $w_1 = w_2 = \ldots = w_n = \varepsilon$ then $w = \varepsilon$ has the required property.

Assume that at least one element of $\{w_1, \ldots, w_n\}$ is different from ε. Let m be the number of occurrences in $\{w_1, \ldots, w_n\}$. We assume that the property is true for every family $\{v_1, \ldots, v_n\}$ with at most $m-1$ occurrences.

Determine the set M of occurrences (a, k) that have no π-predecessor. Then for every S_i, $w_i = [M_{|S_i}]v_i$ (where $[M_{|S_i}] = \prod\{a \mid (a, k) \in M$ and $a \in A_i\})$. It is easy to see that $\{v_1, \ldots, v_n\}$ is a compatible family of elements $v_i \in M(S_i)$.

There are strictly less occurrences of atomic actions in $\{v_1, \ldots, v_n\}$ than in $\{w_1, \ldots, w_n\}$. Then, by the induction hypothesis there exists a $v \in M(S)$ such that for every $i \in \{1, \ldots, n\}$, $v_{|A_i} = v_i$. It is easy to see that $w = [M]v$ has the property that for every $i \in \{1, \ldots, n\}$, $w_{|A_i} = w_i$. □

The "amalgamation" of the family $\{w_1, \ldots, w_n\}$ can then be recursively constructed as follows:

Given: A family of systems $S_i = (A_i, D(S_i))$, satisfying the condition in (3),

A matching family $\{w_1, \ldots, w_n\}$, $w_i \in M(S_i)$.

Task: Find $w \in M(S)$ such that for every $i \in \{1, \ldots, n\}$, $w_{|A_i} = w_i$.

Sketch of the Algorithm (glue$(w_1, \ldots, w_n)$)

If $w_1 = \ldots = w_n = \varepsilon$ then $w := \varepsilon$, exit.

Determine the relation π;

Determine the set M of occurrences (a, k) that have no π-predecessor;

For every $i \in \{1, \ldots, n\}$

determine $v_i \in M(S_i)$ such that $w_i = [M_{|S_i}]v_i$;

$v := glue(v_1, \ldots, v_n)$;

$w := [M]v$

end.

We conclude with a classical example in which it can be seen that there are situations when compatible local plans cannot be glued together to a global plan.

Example 9.3 (cf. [MP86, Gog92]) *Let n philosophers sitting around a circular table the center of which always contains a plate of food. This food must be eaten seated at the table with one fork in each hand. The table has n forks, one between each two adjacent chairs. Philosophers can be seen as "agents" that*

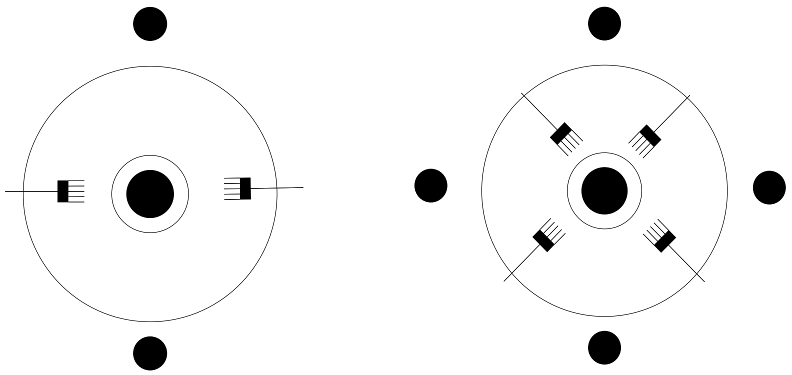


Figure 9.5:

have as goal to eat. For the sake of simplicity we assume that a philosopher eats iff he has a fork in each hand, and that this process takes one unit of time.

The system (in the case $n = 2$) can be described as follows:

The individuals considered are $\mathsf{Philosopher}_1, \mathsf{Philosopher}_2, \mathsf{Fork}_1, \mathsf{Fork}_2$.

(1) $\mathsf{Philosopher}_i$*,* $i = 1, 2$*, can execute two actions:* $\mathsf{take\text{-}fork}^i_1$, $\mathsf{take\text{-}fork}^i_2$.

(2) Fork_i*,* $i = 1, 2$*, can execute two actions:* $\mathsf{to\text{-}philosopher}^i_1$, $\mathsf{to\text{-}philosopher}^i_2$.

We assume that $\mathsf{take\text{-}fork}^i_j$ *if and only if* $\mathsf{to\text{-}philosopher}^j_i$*, hence they can be identified.*

Consider the following local plans:

Philosopher$_1$: $\mathsf{take\text{-}fork}^1_1$, $\mathsf{take\text{-}fork}^1_2$

Philosopher$_2$: $\mathsf{take\text{-}fork}^2_2$, $\mathsf{take\text{-}fork}^2_1$

Fork$_1$: $\mathsf{to\text{-}philosopher}^1_2$ $(= \mathsf{take\text{-}fork}^2_1)$, $\mathsf{to\text{-}philosopher}^1_1$ $(= \mathsf{take\text{-}fork}^1_1)$

Fork$_2$: $\mathsf{to\text{-}philosopher}^2_1$ $(= \mathsf{take\text{-}fork}^1_2)$, $\mathsf{to\text{-}philosopher}^2_2$ $(= \mathsf{take\text{-}fork}^2_2)$.

It can be easily seen that these global plans cannot be glued to a global plan. The reason is that in this case the dependence graph, represented in Figure 9.6, contains a cycle that is not contained in the dependence graph of any of its composing subsystems.

9.4 Properties of the Interconnection of a Family of Systems

From the remarks in Section 8.7 we know that those properties that can be expressed in terms of states, actions, transitions, possibly involving behavior over an interval of time, are "inherited" by the system obtained by interconnecting

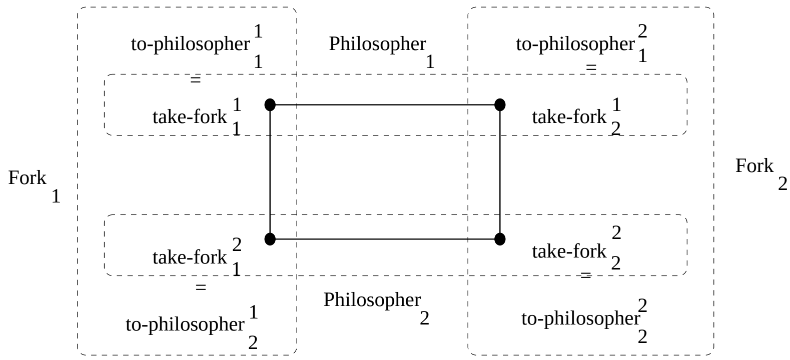


Figure 9.6: Dining Philosophers (n = 2), Dependence Graph

a given family of systems if these properties can be expressed by cartesian sets of axioms (i.e. sets of axioms that can be ordered in such a way that each is cartesian relative to those which precede it).

Therefore a "modular" approach to checking whether a system S satisfies such axioms could be formulated as follows:

Given :

A finite family of finite systems InSys,

A cartesian set of axioms $\mathbb{T}$ involving information about states, actions, transitions, behavior over time ($\mathbb{N}$).

Task : Decide whether the system S obtained by interconnecting the elements satisfies the axioms in $\mathbb{T}$.

Sketch of the Algorithm :

[Step1] *Solve subproblems*

For every system $S_i \in$ InSys check whether the axioms in $\mathbb{T}$ hold in S_i.

[Step2] *Compose*

If for all system in InSys the axioms in $\mathbb{T}$ hold, *Return*($\mathbb{T}$ holds in S).

Note that if there exists one system in InSys where one axiom in $\mathbb{T}$ does not hold, we do not know whether $\mathbb{T}$ holds in S or not.

9.5 Description of a Time and Space Dependent Scenario

We now present a scenario in which time- and space-dependency arise in a natural way. This is part of joint work with J. Pfalzgraf and K. Stokkermans in which fibered structures are used for modeling cooperating agents cf. [PSS96b].

The following scenario is based on the original sample scenario of [Pfa91]. However, it is significantly extended in that it now reflects a generic way to deal with time and space dependency.

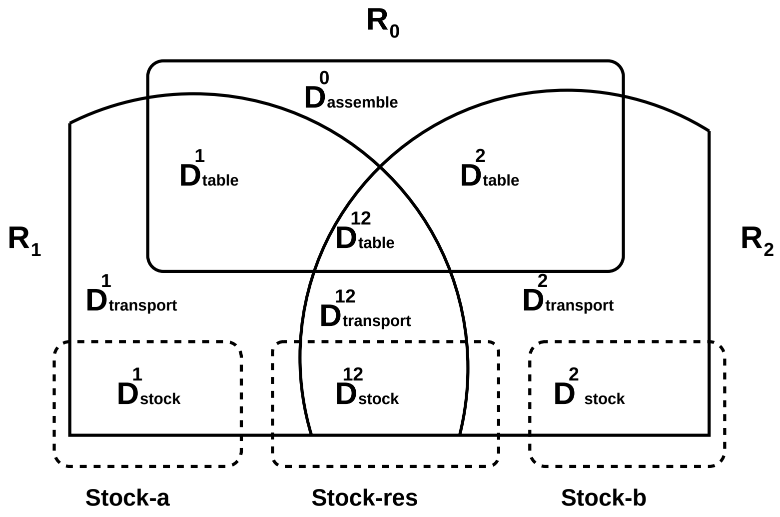


Figure 9.7: Partition of the Workspaces

Mixing space and time dependency, formally we have in mind the product of space and time, as follows. Let X denote the space (physically, this can be the coordinatized ground floor in a working hall, for example) and T the entire set of time parameters (possibly a union of time intervals or clock cycles corresponding to individual agents). Then time constraints arise in cooperation of agents (synchronization constraints) and this can depend on that part of the working cell of an agent where he cooperates with others. Thus, if $D \subset X$ denotes such a subdomain and if $\pi_1 : X \times T \to X$ is the first projection map, then in $\pi_1^{-1}(D)$ we are modeling the time dependent processes, including time constraints arising over subdomains D where cooperation of different agents takes place.

The following scenario is taken from [PSS96b] and contains three robots, R_0, R_1, and R_2. They perform an assembly task similar to those described in [PSS95, PSS96a].

- R_0 receives a work piece a and a work piece b and performs an assembly task. The work piece r obtained from assembling a and b is placed on the assembly bench, at a part reachable for both R_1 and R_2 (so in the intersection of all three workspaces).

- R_1 furnishes pieces of type a. He checks whether there are pieces of type a left in stock, and whether a piece of type a or an r resulting from assembling a and b is placed on the assembly bench of R_0. If there are pieces of type a in stock, and if no a or r are placed on the table, R_1 brings a piece of type a to R_0 (and places it in the intersection of their workspaces).

- R_2 furnishes pieces of type b. He checks whether there are pieces of type b left in stock, and whether a b or an r is placed on the table. If there are pieces of type b in stock, and no b or r is on the table, R_2 brings a piece of type b to R_0, placing it in the intersection of their workspaces.

- After R_0 has assembled a and b, either R_1 or R_2 transports the result r from the table to the final storage; under certain circumstances, the two robots R_1 and R_2 will have to cooperate to transport the piece r though; this will be described below.

Since the agents are performing complex and different tasks (taking pieces, transporting them, assembling) and are able to cooperate, we can assume that their working space can be partitioned in different subdomains where specific tasks are performed. A partition of the workspaces for the example described above is illustrated in Figure 9.7.

Moreover we will assume that every agent knows "where" in space he is (at least in which domain) and that at the moment when a given agent begins to execute an action A, an internal clock cycle begins. Moreover, at this moment the agent knows the time t_A needed for accomplishing A. If this limit time is reached with the action not accomplished, a signal to a control device C is activated (meaning that something is going wrong). C will then correct the situation.

In the scenario described here, we assume that in the different regions of their workspace the robots can do specific actions and can have access to certain information (in the form of values for control variables). This is described in Table 9.8 (where A is the action being performed and t_A is a maximal duration, assumed to be sufficient for performing A).

One possible development of the scenario is the following:

Initial State: The agents R_1 and R_2 are at the positions $x_1 \in {D^1}_{\mathsf{transport}}$ (resp. $x_2 \in {D^2}_{\mathsf{transport}}$). There are no pieces of type a in stock and there is at least one piece of type b in stock. No piece is on the table.

Goal: Transport the piece r obtained by assembling a piece of type a and one of type b in the stock of pieces of type r.

A Possible Plan (note that the logical state spaces of agents – control variables, activation values for actions – vary depending on the moment of time and the position in space of the agents).

(1) Agent R_1 moves to the stock (in ${D^1}_{\mathsf{stock}}$). His internal clock is set to 0. He reaches the stock at moment t_s (the normal time necessary for reaching the stock), and attempts to take a piece from the stock. The internal clock is set to 0 again. Since no piece of type a is left in stock, after the time t_a (sufficient in normal circumstances for taking a piece of type a from the stock) the action is still not accomplished. Therefore at

Agent	Domain	Time (int. clock: $[0, t_A)$)	Variables	Actions
R_1	D^1_{stock}	$0 \leq t < t_A$	s_a, x_1, t_1, h_a	take-a, move-to-$D^1_{\text{transport}}$
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	$D^1_{\text{transport}}$	$0 \leq t < t_A$	x_1, t_1, h_a	move-to-D^1_{table}, move-to-D^1_{stock}
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	D^1_{table}	$0 \leq t < t_A$	p_a, p_r, x_1, t_1, h_a	give-a, move-to-$D^1_{\text{transport}}$
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
R_2	D^2_{stock}	$0 \leq t < t_A$	s_b, x_2, t_2, h_b	take-b, move-to-$D^2_{\text{transport}}$
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	$D^2_{\text{transport}}$	$0 \leq t < t_A$	x_2, t_2, h_b	move-to-D^2_{table}, move-to-D^2_{stock}
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	D^2_{table}	$0 \leq t < t_A$	p_b, p_r, x_2, t_2, h_b	give-b, move-to-$D^2_{\text{transport}}$
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
R_1, R_2	D^{12}_{table}	$0 \leq t < t_A$	$p_r, x_1, t_1, h_a, x_2, t_2, h_b$	take-res
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
R_0	D^0_{assemble}	$0 \leq t < t_A$	p_a, p_b, p_r, x_0, t_0	assemble, move-to-D^1_{table}, move-to-D^2_{table}, move-res-to-D^{12}_{table}
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	D^1_{table}	$0 \leq t < t_A$	p_a, p_r, x_0, t_0	receive-a, move-a-to-D^0_{assemble}
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	D^2_{table}	$0 \leq t < t_A$	p_b, p_r, x_0, t_0	receive-b, move-b-to-D^0_{assemble}
		$t = t_A$	in addition: alarm-C(A)	call-C(A)
	D^{12}_{table}	$0 \leq t < t_A$	p_r, x_0, t_0	put-res, move-to-D^0_{assemble}
		$t = t_A$	in addition: alarm-C(A)	call-C(A)

Figure 9.8: Description of Space and Time Dependencies

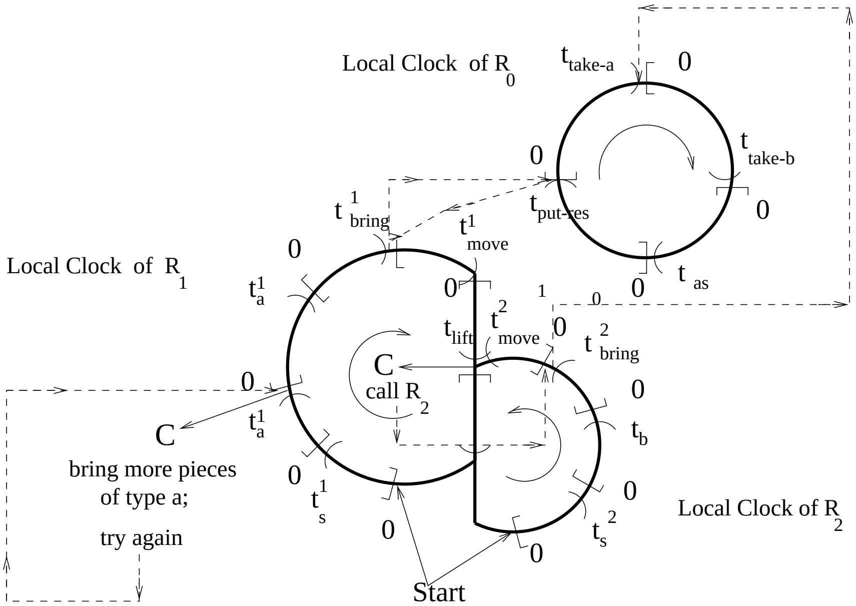


Figure 9.9: Time Cycles

moment t_a a signal is activated, by which R_1 asks the control device C for help. C takes care that the stock is provided with more pieces of type a, and R_1 can eventually perform the action of picking a piece. He brings it to the assembly bench and puts it there.

(2) Agent R_2 does an analogous task for pieces of type b. Since there still are pieces of type b in stock, the intervention of C is not necessary in this case.

(3) A piece of type a and a piece of type b are on the assembly bench. Robot R_0 starts to assemble the pieces. At the beginning of the process the internal clock of R_0 is set to 0. Assume that the assembly process is not successful, in the sense that after the time t_{as}, normally sufficient for assembling the pieces, they are still not assembled. Then at the moment t_{as} a signal is activated, by which R_0 asks for help from C.

C discovers which of the pieces causes the problem and gives the corresponding agent the command to bring another piece of that type. After this is provided and the assembly process is completed, R_0 puts the result on the assembly bench.

(4) The result has to be transported to a stock with pieces of type r (by one or both agents). R_1 moves to the table (region D^{12}_{table}, which is accessible to all the robots) and tries to lift r. Its internal clock is set to 0. If after the time t_r (sufficient for lifting r) the action has not yet been accomplished, R_1 activates a signal which is sent to C. The control device then calls R_2 to help lifting and transporting the piece.

The cooperation between R_1 and R_2 is only possible in the domains D^{12}_{table} and $D^{12}_{\mathsf{transport}}$.

The robot R_2 comes to help R_1. Together they lift the piece r and transport it to the stock. While R_1 and R_2 work together, their internal clocks must be synchronized.

The behavior in time of the system is illustrated in Figure 9.9. For the sake of simplicity, the error situation described in (3) above is omitted. The local clock-cycles are denoted by intervals of the form $[0, t_A)$, where t_A is a maximal duration, assumed to be sufficient for executing a given action A. We used this type of notation in order to emphasize the fact that at the moment t_A the state space changes: if the action is not yet accomplished a signal is switched on, and the robot asks for help from the control device C.

The way these local time-cycles synchronize and build one time-cycle for the whole system is represented in Figure 9.10.

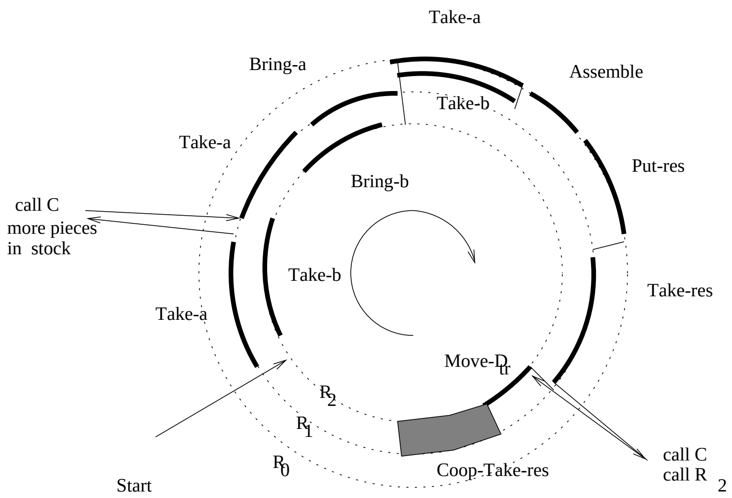


Figure 9.10: Synchronization

Chapter 10

Conclusions and Plans of Future Work

In this thesis we pointed out some applications of fibered structures in computer science, namely in solving algorithmic problems in algebra and logic, as well as in modeling interacting systems. We will give a summary here, and indicate some directions for future research.

10.1 Applications for Solving Algorithmic Problems in Universal Algebra and Logic

In Chapter 5 we presented representation theorems in universal algebra – such as sheaf representation and Priestley type representation theorems – and their applications in solving algorithmic problems in algebra or logic.

We started with a motivating example, namely that of the SHn-logics, because the idea of the approach to automated theorem proving presented here came to us when studying SHn-logics.

We presented two representation theorems for the algebraic models of SHn-logics, namely a sheaf representation theorem, which is an easy consequence of a theorem due to Werner about the representability of algebras in discriminator varieties as sheaves of algebras over boolean spaces, and a duality theorem derived from the Priestley duality for distributive lattices with 0 and 1.

The common feature of these representation theorems is that they state that every element of a given SHn-algebra A can be regarded as a continuous map $s : I \to \coprod_{i\in I} A_i$, with the property that for every $i \in I$, $s(i) \in A_i$, and which additionally might preserve some relations defined on I and $\coprod_{i\in I} A_i$. Note that also the notion of Logical Fiberings (cf. [Pfa91]) is based on the idea of decomposing logics by defining a "base space" and "fibers".

In what concerns sheaf representation and applications, we specialized the theorems on the existence of most general unifiers for discriminator varieties due to Burris (cf. [Bur92]) to the variety SHn, and obtained a method of generating most general unifiers in this variety. We also showed that, given a discriminator variety $\mathcal{V}$, Löwenheim's theorem on finding the reproductive solutions for *systems* of Boolean equations can be extended to a theorem that shows how "most general solutions" can be found for systems of equations in $\mathcal{V}$.

We showed next that the Priestley representation can be used in order to give a procedure for automated theorem proving for some classes of logics.

We noticed that the dual spaces associated to SHn algebras are in particular special SHn-frames (as defined in [IO96]), where the relation is a partial order.

We showed that the propositional SHn-logic is sound and complete with respect to the Kripke model with $2(n-1)$ elements $D(S_{n^2})$.

We gave a transformation procedure to clause form and a refutation procedure based on negative hyperresolution.

Moreover, we extended the ideas illustrated for the case of SHn-logics to more general classes of logics, namely those logics that are sound and complete with respect to a class of algebras with a distributive lattice reduct such that the variety is generated by a single finite algebra and such that the Priestley duality for distributive lattices extends to a dual equivalence between the given variety and a suitable category of Priestley spaces endowed with operations and relations. The relationships between algebraic and relational models were studied, and a possibility of defining frames and models (as done in [IO96] for the case of SHn-logics) was discussed. A natural way of deducing the definition of the satisfiability relation in such frames – starting from the well-known satisfiability notion with respect to an algebra – was presented. Then an automated theorem proving procedure was given in this general case. For the sake of simplicity, this procedure was first presented for the propositional case, then it was extended to certain classes of first-order logics.

Note that analyzing the proofs in Chapter 5 we noticed that the condition imposed on the logic $\mathcal{L}$ (namely that it is sound and complete with respect to a variety $\mathcal{V}$ of algebras with an underlying distributive lattice structure, such that $\mathcal{V}$ is generated by one finite algebra A and the Priestley duality extends to a dual equivalence between $\mathcal{V}$ and a category $\mathcal{V}\mathsf{Sp}$ of Priestley spaces with operators) can be relaxed.

In all these proofs we only use the fact that the logic $\mathcal{L}$ is sound and complete with respect to a finite Kripke-style frame (finite set endowed e.g. with an order relation and with additional relations associated to the operations in the logic). A formal treatment of this fact will be subject of future work. In the thesis we decided to impose the additional, more restrictive, condition concerning the existence of the dual equivalence between the class of algebraic models of the logic $\mathcal{L}$ and a suitable category of Priestley spaces with operations because it furnishes an intuitive description of how such a Kripke frame can be constructed.

We finished by giving some examples of logics where this procedure can be applied (P_{mn}-logics and $SHKn$-logics).

At the end of Chapter 5 we presented an implementation in Prolog of the procedure and tested it on several examples. Then, a comparison with other existing approaches was made.

10.1.1 Plans of Future Work

We conclude the description of our research in this area by sketching some directions of future work.

Let us analyze the two representation theorems for SHn-algebras described in Section 5.1 more closely:

	Sheaf Representation Theorem	**Priestley Duality Theorem**
Index set:	Maximal congruences w.r.t. the signature $\{\vee, \wedge, \neg, \Rightarrow, \sim, s_1, \dots, s_{n-1}\}$ (also ∇)	Maximal congruences w.r.t. the signature $\{0, 1, \vee, \wedge\}$
Subbasis for topology	$E(a, b)$, $D(a, b)$	$X_a = E(x, 1)$, $X \backslash X_a = E(x, 0) = D(x, 1)$
Order:	discrete	defined pointwise
Fibers:	Subalgebras of S_{n^2} (all simple SHn-algebras) and the one-point algebra	$\{0, 1\}$ (all simple 0,1-distributive lattices)

This suggests that other types of representations – "intermediate" between the sheaf representation and the Priestley representation – might be possible. The rôle of such a representation would be to "separate" different types of operators, including some of them in the fibers (if they are easy to manipulate) and expressing the other ones as relations on the base space (if they are harder to manipulate).

It still has to be seen whether such representations would bring any advantages, for example in improving the efficiency or parallelizing automated theorem proving procedures.

10.2 Modeling Cooperating Agents

The second direction of work presented in the thesis is the use of sheaf theory for modeling systems that are obtained by interconnecting interacting agents.

Starting from a motivating example we proposed a definition of systems as well as a definition of morphisms between systems. Two categories of systems, SYS and $\mathsf{SYS}_{\mathsf{lm}}$, were defined, both having as objects systems, with the morphisms reflecting different types of relationships between systems: the morphisms in SYS can be seen as "translations" from the language of one system to the language of another system that preserve constraints and satisfy a compatibility condition with respect to the models, whereas the morphisms in $\mathsf{SYS}_{\mathsf{lm}}$ additionally satisfy a certain "tightness" condition with respect to transitions. We showed that two contravariant functors St and Act – expressing states and parallel actions – from these categories of systems to Sets can be defined. In the category $\mathsf{SYS_{lm}}$ – considering St and Act as functors from $\mathsf{SYS_{lm}}$ to Sets – we showed that transitions define a natural transformation $Tr : Act \to \Omega^{St \times St}$ (morphism in the category of presheaves).

We then continued by considering a category $\mathsf{SYS_i}$ having only inclusions between systems as morphisms, and a category $\mathsf{SYS_{il}}$ having so-called transition-connected inclusions as morphisms (this was necessary in order to express the fact that there is a morphism from S_1 to S_2 if S_1 is a subsystem of S_2 and transitions in S_2 "restrict" to transitions in S_1).

Notions of covering were defined in both $\mathsf{SYS_i}$ and $\mathsf{SYS_{il}}$ and it was shown that these covering relations induce Grothendieck topologies on both these categories.

States, actions and transitions were also analyzed in this context. It turned out that states and parallel actions define sheaves (with respect to the above mentioned Grothendieck topologies) in both $\mathsf{SYS_i}$ and $\mathsf{SYS_{il}}$, and that the transitions define a natural transformation $Tr : Act \to \Omega^{St \times St}$ in the category of sheaves over $\mathsf{SYS_{il}}$ (in some cases a certain finiteness condition was required) with the corresponding Grothendieck topology.

Temporal behavior of systems in $\mathsf{SYS_{il}}$ was also analyzed – the starting point is the approach of Goguen [Gog92]. We took into account both the state and the action executed at a given moment of time, and showed that two types of "gluing" conditions hold: one with respect to the basis of observation over time and the other with respect to the structure of the system.

Also the possibility of modeling the behavior of a system by monoids and languages was considered.

In concrete applications we usually are only interested in some subcategory of $\mathsf{SYS_{il}}$, having as objects those systems relevant for the given application. Therefore, we continued by considering the category of those systems obtained by interconnecting a given family InSys of interacting systems, all contained in a given finite system S_U (to enforce compatibility of models as well as finiteness) and which was assumed to be closed under all subsystems by means of which communication can be done. A system obtained by interconnecting the elements of the family InSys can be regarded either as a system on its own, or as the set of all elements of InSys by whose interaction it arises (a downwards-closed subset of InSys).

We showed that on both categories defined this way, $\mathsf{Sys(InSys)}$ resp. $\Omega(\mathsf{InSys})$, suitable Grothendieck topologies can be defined, expressing the way systems arise from smaller subsystems. We showed that in both these approaches one can define notions as *states* and *parallel actions*, and showed that these define sheaves St and Act, resp. $\overline{St}$ and $\overline{Act}$, with respect to the corresponding Grothendieck topologies. Moreover, transitions can be expressed in both cases by natural transformations $Tr : Act \to \Omega^{St \times St}$, resp. $\overline{Tr} : \overline{Act} \to \Omega^{\overline{St} \times \overline{St}}$ (or alternatively as a subsheaf of $Act \times St \times St$ resp. $\overline{Act} \times \overline{St} \times \overline{St}$). The link between the categories $\mathsf{Sys(InSys)}$ and $\Omega(\mathsf{InSys})$ was also investigated: we showed that an adjunction between these two categories exists; additionally, the right adjoint preserves covers, which implies that a geometric morphism between the category of sheaves over $\mathsf{Sys(InSys)}$ and the category of sheaves over $\Omega(\mathsf{InSys})$

(with the corresponding Grothendieck topologies) can be established.

Also in this case we studied the behavior in time of systems – we showed that both gluing properties (with respect to time and with respect to the structure of the systems) established for the case SYS_i also hold for the categories $\mathsf{Sys}(\mathsf{InSys})$ and $\Omega(\mathsf{InSys})$.

All these results were used in the last chapter, where – by using classical results in sheaf theory and geometric logic – we investigated the links between the properties of the elements of InSys and the system obtained by interconnecting them. The theoretical considerations were illustrated on three examples: deadlock freedom, determinism, and fairness of execution.

10.2.1 Prospects of Future Work

There are many directions of future work in modeling cooperating agents. We briefly refer to them, without entering into too much detail.

First of all, in the approach presented here we assumed that the execution time for all actions is taking one unit of time. We would like to analyze the more complex and realistic approach in which for every action a duration for its accomplishment is given. The problem of representing different durations for the actions is not trivial. The representation of the behavior in time by monoids and languages turns out to be more difficult if we consider different durations for the actions. One possible solution could be to use results from timed automata theory [AD94]. This might also have applications in scheduling.

Another direction of future work is towards generalizing the notion of "observation in time", in the sense of using a "path category" as done in [Win96, CW96] instead of the basis for observation in time used here.

We would like to obtain a better understanding of the links between Priestley duality, Kripke models and the approach to the study of concurrency described in the thesis. The main observation that suggests that there is such a relationship is the following: Let S be a system with a set X of variables that can only take the values 0 (false) and 1 (true). Assume that the relationships between these variables can be described by a set Γ of Boolean equations in variables from X. Then a state of the system S is an assignment $s : X \to \{0, 1\}$ that satisfies the set Γ of Boolean equations. Note however that there is a bijection between the set

$$\{s : X \to \{0, 1\} \mid s \models \Gamma\}$$

and the set

$$\{s : F_B(X) \to \{0, 1\} \mid s \text{ Boolean algebra homomorphism and } s \models \Gamma\},$$

i.e. with the set

$$\{s \mid s : F_B(X)/ =_\Gamma \to \{0, 1\} \text{ Boolean algebra homomorphism}\},$$

where $F_B(X)$ is the free Boolean algebra generated by X and $=_\Gamma$ is the congruence relation on $F_B(X)$ generated by the set Γ of Boolean equations.

Note that the set $\{s \mid s : F_B(X)/=_\Gamma \to \{0,1\}$ Boolean algebra homomorphism$\}$ is the Stone space of the Boolean algebra $F_B(X)/=_\Gamma$.

In [JNW94, CW96] a parallel is drawn between so-called P-open maps (with respect to a path category) and bounded morphisms of Kripke models. In the same line of research, we would like to investigate whether the set of states can define (in a certain way) a Kripke model for a suitable logic expressing the properties of the actions.

In this thesis we defined a general notion of morphisms, but when considering interconnections of systems we only considered inclusions between systems as morphisms. We would like to continue this research in two directions:

(1) Consider categories that have even more special types of inclusions as morphisms (e.g. conservative extensions, definitional extensions).

(2) Study the possibility of defining a Grothendieck topology on the category of systems SYS or on $\mathsf{SYS}_{\mathsf{Im}}$ (i.e. with arbitrary morphisms).

Finally, we point out one more direction for future research. In this thesis we showed that transitions define natural transformations between sheaves, $Tr : Act \to \Omega^{St \times St}$. In this way, a "generic transition system" can be associated to a given category of systems, where both states and actions are sheaves, and such that the transitions can be expressed by natural transformations. We would like to apply the results of Adámek and Trnková [AT90] on defining automata in a category to the concrete category of sheaves over a category of systems with a suitable Grothendieck topology (this category has the properties required in order to apply the theory from [AT90]), and see how general constructions as for instance minimal realization can be carried out.

Bibliography

[AB70] R. Anderson and W. Bledsoe. A linear format for resolution with merging and a new technique for establishing completeness. *Journal of the ACM*, 17:525–534, 1970.

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AHS90] J. Adámek, H. Herrlich, and G. Strecker. *Abstract and Concrete Categories*. John Wiley & Sons, Inc., 1990.

[AK48] R.F. Ahrens and I. Kaplanski. Topological representation of algebras. *Trans. AMS*, 63:457–481, 1948.

[AN93] H. Andréka and I. Németi. General algebraic logic: A perspective on “what is logic”. Lecture Notes for the TEMPUS Summer School, July 11-17, Budapest, November 1993.

[ANS94] H. Andréka, I. Németi, and I. Sain. Universal algebraic basics for algebraic logics. Lecture Notes for the TEMPUS Summer School, July 11-17, Budapest, 1994.

[ANSK94] H. Andréka, I. Németi, I. Sain, and Á. Kurucz. Applying algebraic logic: A general methodology. Lecture Notes for the TEMPUS Summer School, July 11-17, Budapest, July 1994.

[AP94] M.E. Adams and H.A. Priestley. Equational bases for varieties of Ockham algebras. *Algebra Universalis*, 32(3):368–397, 1994.

[AT90] J. Adámek and V. Trnková. *Automata and Algebras in Categories*. Kluwer Academic Publishers, 1990.

[Baa92] M. Baaz. Automatisches Beweisen für endlichwertige Logiken. *Mitt. Math. Ges. Hamburg*, 12:1141–1155, 1992.

[Bar74] J. Barwise. Axioms for abstract model theory. *Ann. Math. Logic*, 7:221–265, 1974.

[Bet55] E.W. Beth. Semantic entailment and formal derivability. *Medelingen der Koninklijke Nederlandse Akademie van Wetenschappen*, 18(13):309–342, 1955.

[Bet59] E.W. Beth. *The Foundations of Mathematics.* North-Holland, Amsterdam, 1959.

[Bet86] E.W. Beth. Semantic entailment and formal derivability. In K. Berka and Kreiser L., editors, *Logik-Texte. Kommentierte Auswahl zur Geschichte der modernen Logik*, pages 262–266. Akademie Verlag, Berlin, 1986.

[BF85] J. Barwise and S. Feferman, editors. *Model-Theoretic Logics.* Springer-Verlag, 1985.

[BF92] M. Baaz and C.G. Fermüller. Resolution for many-valued logics. In A. Voronkov, editor, *Proc. Logic Programming and Automated Reasoning LPAR'92*, LNAI 624, 1992.

[BF95] M. Baaz and C.G. Fermüller. Resolution-based theorem proving for many-valued logics. *Journal of Symbolic Computation*, 19:353–391, 1995.

[BHK90] J. Bergstra, J. Heering, and P. Klint. Module algebra. *Journal of the ACM*, 37(2):335–372, 1990.

[BP90] G. Bordalo and H.A. Priestley. Relative Ockham lattices: Their order theoretic and algebraic characterisation. *Glasgow Math. Journal*, 32:47–66, 1990.

[BP94] G. Bordalo and H.A. Priestley. Series-parallel posets and relative Ockham lattices. *Order*, 11:281–305, 1994.

[BS81] S. Burris and H.P. Sankappanavar. *A Course in Universal Algebra.* Graduate Texts in Mathematics. Springer Verlag, 1981.

[Buc65] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Universität Innsbruck, 1965.

[Buc83] B. Buchberger. A critical-pair/completion algorithm for finitely generated ideals in rings. In *Proc. Symposium Rekursive Kombinatorik, Münster, May 23-28,*, LNCS 171, pages 137–161, 1983.

[Buc85] B. Buchberger. Basic features and development of the critical-pair/completion procedure. In Jean-Pierre Jouannaud, editor, *Proc. First Int. Conf. RTA, LNCS 202*, pages 1–45, Dijon, 1985. Springer Verlag.

[Bur92] S. Burris. Discriminator varieties and symbolic computation. *Journal of Symbolic Computation*, 13:175–207, 1992.

[BW79] S. Burris and H. Werner. Sheaf constructions and their elementary properties. *Transactions of the AMS*, 248(2):269–309, March 1979.

[Car50] H. Cartan. Idéaux et modules de fonctions analytiques de variables complexes. *Bulletin de la Société Mathématique de France*, 78:29–64, 1950.

[Car87] W.A. Carnielli. Systematization of finite many-valued logics through the method of tableaux. *Journal of Symbolic Logic*, 52(2):473–493, 1987.

[Car91] W.A. Carnielli. On sequents and tableaux for many-valued logics. *Journal of Non-Classical Logics*, 8(1):59–76, 1991.

[Cas91] R. Casley. *On the Specification of Concurrent Systems.* PhD thesis, Stanford University, January 1991.

[CDH93] R. Caferra, S. Demri, and M. Herment. A framework for the transfer of proofs, lemmas and strategies from classical to non classical logics. *Studia Logica*, 52:197–232, 1993.

[CF69] P. Cartier and D. Foata. *Problèmes combinatoires de commutation et réarangements.* LNM 85. Springer Verlag, 1969.

[CF77] W.H. Cornish and P.R. Fowler. Coproducts of De Morgan algebras. *Bull. Australian Math. Soc.*, 16:1–12, 1977.

[CHZ91] R. Caferra, M. Herment, and N. Zabel. User-oriented theorem proving with the ATINF graphic proof editor. In P. Jorrand and J. Kelemen, editors, *Proceedings of Fundamentals of Artificial Inteligence – JELIA'90*, LNAI 535. Springer Verlag, 1991.

[Cig70] R. Cignoli. Moisil algebras. *Notas de Lógica Matemática*, 27, 1970. Univ. Nac del Sur, Bahía.

[Cig72] R. Cignoli. Representation of Łukasiewicz and Post algebras by continuous functions. *Algebra Universalis*, XXIV:127–138, 1972.

[Cig82] R. Cignoli. Proper n-valued Łukasiewicz algebras as s-algebras of Łukasiewicz n-valued propositional calculi. *Studia Logica*, 41:3–16, 1982.

[Cig91] R. Cignoli. Quantifiers on distributive lattices. *Discrete Mathematics*, 96:183–197, 1991.

[Cîr95] C. Cîrstea. A distributed semantics for FOOPS, PRG-TR-20-95. Technical report, Oxford University Computing Laboratory, 1995.

[CL73] C. Chang and R.C. Lee. *Symbolic Logic and Mechanical Theorem Proving.* Academic Press, 1973.

[CLP91] R. Cignoli, S. Lafalce, and A. Petrovich. Remarks on Priestley duality for distributive lattices. *Order*, 8:299–315, 1991.

[CM85] R. Cori and Y. Métivier. Recognizable subsets of some partially abelian monoids. *Theoretical Computer Science*, 35:241–254, 1985.

[Com71] S.D. Comer. Representations by algebras of sections over Boolean spaces. *Pacific J. Math.*, 38:29–38, 1971.

[Cos90] M. M. do C. Costa. *Characterization of Modal [Action] Logic*. PhD thesis, Imperial College, London, 1990.

[CRAB91] J. Chazarain, A. Riscos, J. A. Alonso, and E. Briales. Multi-valued logic and Gröbner bases with applications to modal logic. *Journal of Symbolic Computation*, 11-12:181–191, 1991.

[Cre91] R. Crew. *Metric Process Models*. PhD thesis, Stanford University, December 1991.

[CW96] G.L. Cattani and G. Winskel. Presheaf models for concurrency. Proceedings of CSL'96, to appear, 1996.

[CZ90a] R. Caferra and N. Zabel. An application to many-valued logic to decide propositional S_5 formulae: a strategy designed for a parametrized tableaux-based prover. In *Proceedings AIMSA'90, Artificial Inteligence – Methodology Systems Applications*, pages 23–32, 1990.

[CZ90b] R. Caferra and N. Zabel. Extending resolution for model construction. In J. van Eick, editor, *Proceedings of Logics in AI – JELIA'90*, LNAI 478, pages 153–169. Springer Verlag, 1990.

[CZ92] R. Caferra and N. Zabel. A method for simultaneous search for refutations and models by equational constraint solving. *Journal of Symbolic Computation*, 13:613–641, 1992.

[Cze82] J. Czelakowski. Logical matrices and the amalgamation property. *Studia Logica*, XLI(4):329–342, 1982.

[Dav72] B.A. Davey. m-Stone lattices. *Can. J. Math.*, 24:1027–1032, 1972.

[Dav73] B.A. Davey. Sheaf spaces and sheaves of universal algebras. *Math. Zeitschrift*, 134:275–290, 1973.

[DGS91] R. Diaconescu, J. Goguen, and P. Stefaneas. Logical support for modularity. In *Proceedings of the Workshop on Logical Frameworks*, Edinburgh, May 1991.

[DH66] J. Dauns and K.H. Hofmann. The representation of biregular rings by sheaves. *Math. Zeitschrift*, 91:103–123, 1966.

[Dia96] R. Diaconescu. Category-based moodulerization for equational logic programming. *Acta Informatica*, 33(5):477–510, 1996.

[Die90] V. Diekert. Combinatorics on Traces. In *LNCS 454*. Springer Verlag, 1990.

[DP90] B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990.

[DPSS91] F. Dargam, J. Pfalzgraf, K. Stokkermans, and V. Stahl. Towards a toolkit for benckmark scenarios in robot multi-tasking. Technical Report 91-45, RISC-Linz, J. Kepler University, Linz, Austria, December 1991.

[Ebb85] H.D. Ebbinghaus. Extended logics: The general framework. In J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, pages 25–76. Springer-Verlag, 1985.

[Ede92] E. Eder. *Relative Complexities of First Order Calculi.* Wiesbaden: Vieweg, Brauschweig, 1992.

[Fia96] J.L. Fiadeiro. On the emergence of properties in component-based systems. In M. Wirsing and M. Nivat, editors, *Proceedings AMAST'96*, LNCS 1101, pages 421–443. Springer-Verlag, 1996.

[Fil80] A. Filipoiu. Representation of Lukasiewicz algebras by means of ordered Stone spaces. *Discrete Mathematics*, 30:111–116, 1980.

[Fis86] W. Fischer. Über erkennbare und rationale Mengen in freien partiell kommutativen Monoiden. Technical Report FBI-HH-B-121/86, Fachbereich Informatik der Universität Hamburg, 1986. (Diplomarbeit 1985).

[Fos53a] A.L. Foster. Generalized Boolean theory of universal algebras I. *Math. Z.*, 58:306–336, 1953.

[Fos53b] A.L. Foster. Generalized Boolean theory of universal algebras II. *Math. Z.*, 59:191–199, 1953.

[FS79] M.P. Fourman and D.S. Scott. Sheaves and Logic. In M. Fourman, editor, *Durham Proceedings (1977). Applications of Sheaves*, LNM 753, pages 302–401. Springer Verlag, 1979.

[Gab92] D.M. Gabbay. Fibred semantics and the weaving of logics. MEDLAR II Workshop, Garmisch-Partenkirchen, January, 24-26, 1992.

[Gab94] D. Gabbay. *LDS - Labelled Deductive Systems. Vol 1 - Foundations.* Technical Report MPI-I-94-223; (Part I: Oxford University Press 1995), 1994.

[Gab96] D. Gabbay. An overview of fibered semantics and the combination of logics. In F. Baader and K.U. Schulz, editors, *Frontiers of Combining Systems*, Applied Logics Series 3, pages 1–56. Kluwer Academic Publishers, 1996.

[GB85] J.A. Goguen and R.M. Burstall. Institutions: Abstact model theory for computer science. Technical Report CSLI-85-30, Center for the Study of Language and Information, August 1985.

[Gis88] J.L. Gischer. The equational theory of pomsets. *Theoretical Computer Science*, 61:199–224, 1988.

[God58] R. Godement. *Topologie Algébrique et Théorie des Faisceaux*. Hermann, Paris, 1958.

[Gog75] J.A. Goguen. Objects. *International Journal of General Systems*, 1:237–243, 1975.

[Gog92] J.A. Goguen. Sheaf semantics for concurrent interacting objects. *Mathematical Structures in Computer Science*, 11:159–191, 1992.

[Gog96] J. Goguen. *Theorem Proving and Algebra*. Preliminary version available via the world wide web at http://www-cse.ucsd.edu/users/goguen/pubs/books.html, 1996.

[Gol81] M. Goldberg. Distributive Ockham algebras: Free algebras and injectivity. *Bull. Austral. Math. Soc.*, 24:161–203, 1981.

[Gol84] R. Goldblatt. *Topoi. The Categorial Analysis of Logic*, volume 98 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, Elsevier Publishers, Amsterdam, 2nd, revised edition, 1984.

[Gol89] R. Goldblatt. Varieties of complex algebras. *Annals of Pure and Applied Logic*, 44(3):153–301, 1989.

[Gou95] E. Goubault. Schedulers as abstract interpretations of higher-dimensional automata. Available via the world wide web at http://Boole.Stanford.EDU:80/pub/, 1995.

[Gra79] J.W. Gray. Fragments of the history of sheaf theory. In M. Fourman, editor, *Durham Proceedings (1977). Applications of Sheaves*, LNM 753, pages 1–79. Springer Verlag, 1979.

[Gup94] V. Gupta. *Chu Spaces: A Model of Concurrency*. PhD thesis, Stanford University, August 1994.

[Häh90] R. Hähnle. Towards and efficient tableau proving procedure for multiple-valued logics. In *Proceedings of Workshop on Computer Science Logic, Heidelberg*, LNCS 533, pages 248–260. Springer Verlag, Berlin, 1990.

[Häh91] R. Hähnle. Uniform notation of tableaux rules for multiple-valued logics. In *Proceedings of International Symposium of Multiple-valued Logics, Victoria*, pages 238–245, Los Alamitos. CA., 1991. IEEE Press.

[Häh93] R. Hähnle. *Automated Theorem Proving in Multiple-Valued Logics*, volume 10 of *International Series of Monographs on Computer Science*. Oxford University Press, 1993.

[Häh94] R. Hähnle. Short conjunctive normal forms in finitely valued logics. *Journal of Logic and Computation*, 4(6):905–927, 1994.

[Häh96a] R. Hähnle. Commodious axiomatization of quantifiers in many-valued logics. Proc. International Symposium on Multiple-Valued Logics, ISMVL'96, Santiago de Compostela, Spain, 1996.

[Häh96b] R. Hähnle. Exploiting data dependencies in many-valued logics. *Journal of Applied Non-Classical Logics*, 6(1):49–69, 1996.

[Har84] D. Harel. Dynamic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume II: Extensions of Classical Logic, chapter II.10, pages 497–604. D. Reidel Publ. Comp., 1984.

[HD83] J. Hsiang and N. Dershowitz. Rewrite methods for clausal and non-clausal theorem proving. In J. Díaz, editor, *Proc. Int. Conf. on Automata, Language and Programming, LNCS 154*, pages 331–346, Barcelona, 1983. Springer Verlag.

[Hin55] K.J.J. Hintikka. Form and content in quantification theory. *Acta Philosophica Fennica*, 8:7–55, 1955.

[Hir56] F. Hirzebruch. *Topological Methods in Algebraic Geometry.* Springer Verlag, 3rd edition, 1956.

[HKLR92] J. Hsiang, H. Kirchner, P. Lescanne, and M. Rusinowitch. The term rewriting approach to automated theorem proving. *Journal of Logic Programming*, 14:71–99, 1992.

[Hof72] K.H. Hofmann. Representations of algebras by continuous sections. *Bulletin of the AMS*, 78:291–373, 1972.

[HS79] H. Herrlich and G.E. Strecker. *Category Theory: An Introduction.* Heldermann Verlag Berlin, 1979.

[Hsi87] J. Hsiang. Rewrite method for theorem proving in first order logic with equality. *Journal of Symbolic Computation*, 3:133–151, 1987.

[Hue80] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.

[IO96] L. Iturrioz and E. Orlowska. A Kripke-style and relational semantics for logics based on Łukasiewicz algebras. Conference in honour of J. Łukasiewicz, Dublin, 1996.

[Ior84] A. Iorgulescu. $(1+\theta)$*-valued Łukasiewicz-Moisil algebras with negation (Romanian).* PhD thesis, University of Bucharest, Romania, 1984.

[Itu82] L. Iturrioz. Modal operators on symmetrical Heyting algebras. In T. Traczyk, editor, *Universal Algebra and Applications*, volume 9 of *Banach Center Publications*, pages 289–303. PWN-Polish Scientific Publishers, 1982.

[Itu83] L. Iturrioz. Symmetrical Heyting algebras with operators. *Zeitschrift f. math. Logik und Grundlagen d. Mathematik*, 29:33–70, 1983.

[JK86] J.P. Jouannaud and H. Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, 15:1155–1194, 1986.

[JNW94] A. Joyal, M. Nielsen, and G. Winskel. Bisimulation from open maps. Technical Report BRICS RS-94-7, BRICS, Department of Computer Science, University of Aarhus, May, 1994.

[Joh82] P. Johnstone. *Stone Spaces*. Cambridge Studies in Advanced Mathematics 3. Cambridge University Press, 1982.

[JT51] B. Jónsson and A. Tarski. Boolean algebras with operators, Part I. *American Journal of Mathematics*, 73:891–939, 1951.

[JT52] B. Jónsson and A. Tarski. Boolean algebras with operators, Part II. *Americal Journal of Mathematics*, 74:127–162, 1952.

[KB67] D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–298, Oxford, 1967. Pergamon Press. Appeared 1970.

[KC79] P.H. Krauss and D.M. Clark. Global subdirect products. *Memoirs of the AMS*, 17(210):1–109, January 1979.

[Kei70] K. Keimel. Darstellungen von Halbgruppen und universellen Algebren durch Schnitte in Garben, bireguläre Halbgruppen. *Math. Nachr.*, 45:81–96, 1970.

[Kei71] K. Keimel. The representation of lattice-ordered groups and rings by sections in sheaves. *LNM 248*, 248, 1971.

[KN85] D. Kapur and P. Narendran. An equational approach to theorem proving in first order predicate calculus. In *Proceedings of the Int. Joint Conf. on Artificial Intelligence*, Los Anngeles, 1985.

[Kri63] S. A. Kripke. Semantical analysis of modal logic I. Normal modal propositional calculi. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

[Krö87] F. Kröger. *Temporal Logic of Programs*, volume 8 of *EATCS Monographs on Theoretical Computer Science*. Springer Verlag, 1987.

[Lem66a] E.J. Lemmon. Algebraic semantics for modal logics I. *The Journal of Symbolic Logic*, 31(1):46–65, 1966.

[Lem66b] E.J. Lemmon. Algebraic semantics for modal logics II. *The Journal of Symbolic Logic*, 31(2):191–218, 1966.

[Ler45] J. Leray. Sur la forme des espaces topologiques et sur les points fixes des représentations. *J. Math. Pures Appl.*, 9:95–248, 1945.

[Lil93] J. Lilius. A sheaf semantics for Petri nets. Technical Report A23, Dept. of Computer Science, Helsinki University of Technology, 1993.

[Mal94] G. Malcolm. Interconnections of object specifications. In R. Wieringa and R. Feenstra, editors, *Working Papers of the International Workshop on Information Systems – Correctness and Reusability*, 1994. Appeared as internal report IR-357 of the Vrije Universiteit Amsterdam.

[Mar90] N.G. Martinez. The Priestley duality for Wajsberg algebras. *Studia Logica*, 49:31–46, 1990.

[Mas64] S.Ju. Maslov. An inverse method of establishing deducability in the classical predicate calculus (translated). *Soviet Math.-Doklady*, 5:1420–1424, 1964.

[Maz77] A. Mazurkiewicz. Concurrent program schemes and their interpretations. Technical Report DAIMI Rep. PB 78, Aarhus University, 1977.

[Mes89] J. Meseguer. General logics. In *Proc. Logic Coll.'87* (North-Holland, Amsterdam), 1989.

[Mil80] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *LNCS*. Springer Verlag, 1980.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice-Hall, 1989.

[Min90] G. Mints. Gentzen-type systems and resolution rules, part i: Propositional logic. In *Proceedings of COLOG-88, Tallin*, LNCS 417, pages 198–231. Springer, Berlin, 1990.

[ML71] S. Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics.* Springer Verlag, 1971.

[MLM92] S. Mac Lane and I. Moerdijk. *Sheaves in Geometry and Logic.* Universitext. Springer Verlag, 1992.

[MMT87] R.N. McKenzie, G.F. McNulty, and W.F. Taylor. *Algebras, Lattices, Varieties. Vol. I.* The Wadsworth & Brooks/Cole Advanced Books and Software, 1987.

[Moi63] Gr.C. Moisil. Le algebra di Lukasiewicz. *Analele Universităţii Bucureşti, Seria Acta Logica*, 6:97–135, 1963.

[Moi65] Gr.C. Moisil. Algebre Lukasiewicziene. In: Incercări vechi şi noi de logică neclasică, Ed. Ştiinţifică, Bucureşti, 1965.

[MOM91] N. Martí-Oliet and J. Meseguer. From Petri nets to linear logic through categories: A survey. *International Journal of Foundations of Computer Science*, 2(4):297–399, 1991.

[Mon76] J.D. Monk. *Mathematical Logic*. Graduate Texts in Mathematics. Springer Verlag, 1976.

[Mor76] C.G. Morgan. A resolution principle for a class of many-valued logics. *Logique et Analyse*, 19(74-76):311–339, 1976.

[Mos48] A. Mostowski. Proofs of non-deducibility in intuitionistic functional calculus. *The Journal of Symbolic Logic*, 13, 1948.

[MP86] L. Monteiro and F. Pereira. A sheaf theoretic model for concurrency. *Proc. Logic in Computer Science (LICS'86)*, 1986.

[Ném94] I. Németi. Algebraizations of quantifier logics, an introductory overview. Lecture Notes for the TEMPUS Summer School, July 11-17, Budapest, June 1994.

[Ohl93] H.J. Ohlbach. Translation methods for non-classical logics - an overview. *Bulletin of the IGPL*, 1(1):69–89, 1993.

[Orl78] E. Orlowska. The resolution principle for ω^+-valued logic. *Fundamenta Informaticae*, II:1–15, 1978.

[Orl79] E. Orlowska. Resolution systems and their applications I. *Fundamenta Informaticae*, 3:235–268, 1979.

[Orl80] E. Orlowska. Resolution systems and their applications II. *Fundamenta Informaticae*, 3:333–362, 1980.

[Pet62a] C.A. Petri. Fundamentals of a theory of asynchronous information flow. In *Proc. IFIP Congress 62*, pages 368–390. North-Holland, Amsterdam, 1962.

[Pet62b] C.A. Petri. *Kommunication mit Automaten.* PhD thesis, Scriften des Institutes für Instrumentelle Mathematik, 1962.

[Pfa91] J. Pfalzgraf. Logical fiberings and polycontextural systems. In P. Jorrand and J. Kelemen, editors, *Proc. Fundamentals of Artificial Intelligence Research*, volume 535 of *LNCS (subseries LNAI)*, pages 170–184. Springer Verlag, 1991.

[Pfa93] J. Pfalzgraf. On mathematical modeling in robotics. In J. Calmet and J.A. Campbell, editors, *AI and Symbolic Mathematical Computing. Proceedings AISMC-1*, volume 737 of *LNCS*, pages 116–132. Springer Verlag, 1993.

[Pfa96] J. Pfalzgraf. On geometric and topological reasoning in robotics. *Annals of Mathematics and AI, special issue on AI and Symbolic Mathematical Computing,* to appear, 1996.

[Pie91] B.C. Pierce. *Basic Category Theory for Computer Scientists.* The MIT Press Cambridge, Massachusetts and London, England, 1991.

[Plo93] G.D. Plotkin. Notes on event structures and Chu. Draft Report, available at http://Boole.Stanford.EDU:80/pub/gdp2.ps.gz, August 1993.

[Pra82] V. Pratt. On the composition of processes. In *Proceedings of the Ninth Annual ACM Symposium on Principles of Programming Languages*, January 1982.

[Pra86] V.R. Pratt. Modeling concurrency with partial orders. *Int. J. of Parallel Programming*, 15(1):33–71, February 1986.

[Pra91] V.R. Pratt. Modeling concurrency with geometry. In *Proc. 18th Ann. ACM Symposium on Principles of Programming Languages*, pages 311–322, January 1991.

[Pra93] V.R. Pratt. The second calculus of binary relations. In *MFCS'93, Gdańsk*, pages 142–155, Poland, 1993.

[Pra94] V.R. Pratt. Chu spaces: Complementarity and uncertainty in rational mechanics. Course notes, TEMPUS summer school, Budapest, 35pp, 1994.

[Pri70] H.A. Priestley. Representation of distributive lattices by means of ordered Stone spaces. *Bull. London Math Soc.*, 2:186–190, 1970.

[Pri72] H.A. Priestley. Ordered topological spaces and the representation of distributive lattices. *Proc. London Math. Soc.*, 3:507–530, 1972.

[PS81] G.E. Peterson and M.E. Stickel. Complete sets of reductions for some equational theories. *Journal of the ACM*, 28(2):233–264, 1981.

[PS92] J. Pfalzgraf and K. Stokkermans. Scenario construction continued and extended with a view to test and enhancement of reasoning methods. Technical Report 92-27, RISC-Linz, J. Kepler University, Linz, Austria, May 1992.

[PS95] J. Pfalzgraf and K. Stokkermans. On robotics scenarios and modeling with fibered structures. In J. Pfalzgraf and D. Wang, editors, Springer Series Texts and Monographs in Symbolic Computation, Automated Practical Reasoning: Algebraic Approaches, pages 53–80. Springer Verlag, 1995.

[PSS95] J. Pfalzgraf, U. Sigmund, and K. Stokkermans. Modeling cooperative agents scenarios by deductive planning methods and logical fiberings. In J. Calmet and J.A. Campbell, editors, *2nd Workshop on Artificial Intelligence and Symbolic Mathematical Computing*, volume 958 of *LNCS*, pages 167–190. Springer-Verlag, 1995.

[PSS96a] J. Pfalzgraf, U. Sigmund, and K. Stokkermans. Towards a general approach for modeling actions and change in cooperating agents scenarios. *IGPL (Journal of the Interest Group in Pure and Applied Logics)*, 4(3):445–472, 1996.

[PSS96b] J. Pfalzgraf, V. Sofronie, and K. Stokkermans. A fibered approach to modeling space-time dependent cooperating agents scenarios. FAPR'96 (Workshop "Reasoning about Actions and Planning in Complex Environments"), 1996.

[PSS96c] J. Pfalzgraf, V. Sofronie, and K. Stokkermans. On a semantics for cooperative agents scenarios. In R. Trappl, editor, *Cybernetics and Systems '96, Volume 1, Proceedings of the Thirteenth European Meeting on Cybernetics and Systems Research*, pages 201–206. Austrian Society for Cybernetic Studies, 1996.

[PSSS95] J. Pfalzgraf, U.C. Sigmund, V. Sofronie, and K. Stokkermans. MEDLAR II: Third Year Deliverable Task V.2: Towards a Cooperating Robots Demonstrator. Technical Report 95-49.0, RISC-Linz, J. Kepler University, Linz, Austria, Europe, September 1995.

[Ras74] H. Rasiowa. *Am Algebraic Approach to Non-classical Logics.* North-Holland Publishing Company, Amsterdam, 1974.

[Rei85] W. Reisig. *Petri Nets: An Introduction.* Springer Verlag, 1985.

[Rob65] J.A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.

[Rob79] J.A. Robinson. *Logic: Form and Function. The Mechanization of Deductive Reasoning.* Edinburgh University Press, 1979.

[Rud74] S. Rudeanu. *Boolean Functions and Equations.* North-Holland Publ. Comp., 1974.

[Sai88] I. Sain. Beth's and Craig's properties via epimorphisms and amalgamation in algebraic logic. In C.H. Bergman, R.D. Maddux, and D.L. Pigozzi, editors, *Algebraic Logic and Univeral Algebra in Computer Science, Proc. Conf. Ames, USA, LNCS 425*, 1988.

[Sco73] D.S. Scott. Background to formalization. In Hugues Leblanc, editor, *Truth, Syntax and Modality. Proceedings of the Temple University Conference on Alternative Semantics*, volume 68 of *Studies in Logic and the Foundations of Mathematics*, pages 244–273, 1973.

[Sic67] C. Sicoe. Notă asupra algebrelor Lukasievicziene polivalente. *Studii şi Cercetări Matematice*, 19:1203–1207, 1967.

[Smu68] R. Smullyan. *First-Order Logic.* Springer Verlag, New York, 1968.

[Sof88] V. Sofronie. Modal algebras and rewriting algorithms. Specialization Thesis, University of Bucharest (in romanian), 1988.

[Sof89] V. Sofronie. Formula-handling computer solution of boolean equations. I. Ring equations. *Bull. of the EATCS*, 37:181–186, 1989.

[Sof96] V. Sofronie. Towards a sheaf theoretic approach to cooperating agents scenarios. In J. Calmet, J.A. Campbell, and J. Pfalzgraf, editors, *Proceedings of Artificial Intelligence and Symbolic Mathematical Computation, International Conference, AISMC-3, Steyr*, LNCS 1138, pages 289–304. Springer-Verlag, 1996.

[Sti93] S. Stifter. Gröbner bases in non-associative reduction structures. Technical Report 93-70, RISC-Linz, Johannes Kepler University, Linz, Austria, 1993.

[Sto95] K. Stokkermans. *A Categorical Framework and Calculus for Critical-Pair Completion.* PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University, Linz, 1995.

[Suc74] W. Suchoń. La méthode de Smullyan de construire le calcul n-valent de Lukasiewicz avec implication and négation. *Reports on Mathematical Logic, Universities Cracow and Katowicze*, pages 37–42, 1974.

[Sur84] S.J. Surma. An algorithm for axiomatizing every finite logic. In E. Lusk and R. Overbeek, editors, *Computer Science and Multiple-Valued Logics*, pages 143–149. North-Holland, Amsterdam, 1984.

[Tak75] G. Takeuti. *Proof Theory.* North-Holland Publishing Company - Amsterdam, 1975.

[Tar56] A. Tarski. On some fundamental concepts in metamathematics. in *Logic, Semantics, Metamathematics*, pages 30-37, Oxford U.P., 1956.

[Tra77] T. Traczyk. Representation theorems for Post algebras. *Acta Polytechnica, Prace ČVUT v PRAZE*, 13:57–63, 1977.

[Urq79] A. Urquhart. Distributive lattices with a dual homomorphic operation. *Studia Logica*, 38(2):201–209, 1979.

[Wer75] H. Werner. Algebraic Representation and Model Theoretic Properties of Algebras with the Ternary Discriminator. Preprint Nr. 237 Technische Hochschule Darmstadt; Habilitationsschrift, November 1975.

[Win84] F. Winkler. *The Church-Rosser Property in Computer Algebra and Special Theorem Proving: An Investigation of Critical-Pair/Completion Algorithms.* PhD thesis, Institut für Mathematik der Technisch-Naturwissenschaftlichen Fakultät der Johannes Kepler Universität Linz, 1984.

[Win96] G. Winskel. A presheaf semantics of value-passing proceses. In Montanari and Sassone, editors, *Concurrency Theory: 7th International Conference, CONCUR '96 Proceedings*, LNCS 1119, pages 98–114, 1996.

[WN93] G. Winskel and M. Nielsen. *Models for Concurrency.* TEMPUS Summer School 1993, Algebraic and Categorical Methods in Computer Science, Brno, June 28 - July 3 1993.

[Zar56] O. Zariski. Scientific report of the second summer institute; III, Algebraic Sheaf Theory. *Bulletin of the AMS*, 62:117–141, 1956.

Index