JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis

Solving and Proving in Equational Theories with Sequence Variables and Flexible Arity Symbols

DISSERTATION

zur Erlangung des akademischen Grades

DOKTOR DER TECHNISCHEN WISSENSCHAFTEN

Angefertigt am *Institut für Symbolisches Rechnen*

Betreuung:

Erster Begutachter: o. Univ.-Prof. Dr. DDr. h.c Bruno Buchberger
Zweiter Begutachter: a. Univ.-Prof. Dr. Josef Küng

Eingereicht von:

Teimuraz Kutsia, Dipl.-Math.

Linz, May 2002

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Teimuraz Kutsia
Hagenberg, 12. Mai 2002

Abstract

We study equational theories whose alphabet includes individual and sequence variables, constants and fixed and flexible arity function symbols. Sequence variables are variables which can be instantiated by an arbitrary finite (possibly empty) sequence of terms. Flexible arity function symbols can take arbitrary finite (possibly empty) sequence of arguments. In this thesis we

- describe a Birkhoff-style calculus for pure equational logic with sequence variables and flexible arity symbols and prove its soundness and completeness;
- design unification procedures for free, flat, restricted flat and orderless equational theories with sequence variables and flexible arity symbols, prove decidability of the unification problem in these theories and minimality and completeness of the unification procedures;
- design a minimal and complete unification procedure for an extension of the free theory with patterns;
- define ground total reduction orderings for terms and pattern-terms with sequence variables and flexible arity symbols;
- show that the basic equational superposition with ordering and equational constraints is a refutationally complete proving method for the free theory with sequence variables and flexible arity symbols;
- show that for a special class of equations in the free theory with sequence variables and flexible arity symbols, unfailing completion can be used as a refutationally complete proving method.

The unification procedure for the free theory is implemented in the mathematical software system *Theorema*.

Zusammenfassung

Wir untersuchen Gleichungstheorien mit individuelle Variablen, Sequenzvariablen, Konstanten und Funktionensymbolen mit fester oder flexibler Stelligkeit. Sequenzvariablen sind Variablen die durch eine beliebige endliche (möglicherweise leere) Folge von Termen instantiert werden können. Funktionensymbole mit flexibler Stelligkeit können eine beliebige endliche Anzahl von Argumenten (möglicherweise auch kein Argument) haben. In dieser Doktorarbeit

- beschreiben wir einen Birkhoff-ähnlichen Kalkül für reine Gleichheitslogik mit Sequenzvariablen und Symbolen flexibler Stelligkeit und beweisen seine Korrektheit und Vollständigkeit;

– entwickeln wir Unifikations-Prozeduren für freie, flache, eingeschränkt flache und ordnungsfreie Gleichungstheorien mit Sequenzvariablen und Symbolen flexibler Stelligkeit. Weiterhin beweisen wir, dass das Unifikationsproblem in diesen Theorien entscheidbar ist und zeigen, dass die Unifikations-Prozeduren vollständig und minimal sind;
– entwickeln wir eine minimale und vollständige Unifikations-Prozedur für eine Erweiterung freier Theorien mit "Pattern-Termen";
– definieren wir eine Reduktionsordnung, welche total auf Grundtermen ist, für Terme und "Pattern-Terme" in denen Sequenzvariablen und Symbole mit flexibler Stelligkeit vorkommen können;
– zeigen wir für freie Theorien mit Sequenzvariablen und Symbolen flexibler Stelligkeit, dass die "basic superposition" bezüglich Ordnungs- und Gleichungs-Constraints eine widerlegungsvollständige Beweismethode ist;
– beweisen wir, dass für eine spezielle Klasse von Gleichungen einer freien Theorie mit Sequenzvariablen und Symbolen flexibler Stelligkeit, "unfailing completion" als eine widerlegungsvollständige Beweismethode verwendet werden kann.

Die Unifikations-Prozedur für die freie Theorie ist in dem mathematischen Software-System *Theorema* implementiert.

Acknowledgements

First of all, I would like to thank my supervisor Bruno Buchberger. One of his many ideas allowed this work to begin and I have been very fortunate to be one of his students. His creative influence, patient guidance, inspiration, encouragement and helpful suggestions have played a major role in the fact this work has resulted in a PhD thesis. I greatly appreciate the way he combined, on the one hand, giving advices and assistance, and on the other hand, giving much freedom in the research. It was a very important combination for me.

Bruno was not only the best supervisor but also a person who always backed me in difficult situations. His deep understanding of both research and people helped me through many times. Thanks!

I am grateful to Franz Lichtenberger for all his support, encouragement, and for being very friendly and kind. He was an excellent manager who always provided an outstanding working environment and a stimulating atmosphere.

I would like to thank the past and present members of the *Theorema* group: Adrian Craciun, Daniela Dupré, Claudio Dupré, Tudor Jebelean, Boris Konev, Felix Kossak, Franz Kriftner, Gabor Kusper, Koji Nakagawa, Florina Piroi, Nikolaj Popov, Judit Robu, Markus Rosenkranz, Christian Vogt and Wolfgang Windsteiger. It has been a great fun to work in this group. Thanks!

I am grateful to the RISC members who were responsible for creating a pleasant environment during the three and a half years I spent here. This is a long list. I refer to `http://www.risc.uni-linz.ac.at/institute/people/`.

Special thanks go to Mircea Marin for many helpful comments and discussions, to Peter Paule and Axel Riese for the Omega package and to Carsten Schneider for helping with LaTeX.

The family of Mayrs, my landlords for almost three years, introduced me to Austrian hospitality. I want to thank them for that.

There are several other people who contributed to my development, both on a scientific and a personal level. First of all, I want to mention Shalva Pkhakadze, Khimuri Rukhaia, Jemal Antidze, Kote Pkhakadze and Vakhtang Pkhakadze, who introduced me into the fascinating field of computational logic. Gela Chankvetadze was an excellent programmer and a very good friend who could create a joyful atmosphere around him. Zurab Khasidashvili was always ready to give a helping hand. I was very lucky to meet Andrei Voronkov, who gave invaluable support to me and to our research group in Georgia in the last six years, and who encouraged me to come here. I am very grateful to all of them.

Finally, I want to express my gratitude to my family. Nino gave me all her love and understanding. I am amazed of her strength to be able to look after our two little daughters – Anna and Elene – almost single-handedly for the last two years, and to study and do research at the same time. I feel very sorry that I could not help much for that time, and I promise that it will never happen again. Without their support and patience this work would never have been completed. They persuaded me into "finishing the damn thing finally". Especially Elene was not satisfied with my average writing speed of four sentences per day and used every opportunity to contribute to the writing process (see, for instance, the last paragraph at page 19).

I am grateful to my parents for their moral support and encouragement.

Work on the thesis was supported by Research Institute for Symbolic Computation (RISC), Austrian Science Foundation (FWF) under the Project SFB F1302 and by Software Competence Center Hagenberg (SCCH) under the projects MathSoft and ForMI.

Table of Contents

1. Introduction

We study equational theories whose alphabet includes individual and sequence variables, constants and fixed and flexible arity function symbols. The subject of this research was proposed by Bruno Buchberger in [Buc96] and in a couple of personal discussions [Buc01].

We refer to a theory with individual and sequence variables, constants and fixed and flexible arity function symbols shortly as a theory with sequence variables and flexible arity symbols, underlining the importance of these two constructs. Sequence variables are variables which can be instantiated by an arbitrary finite (possibly empty) sequence of terms. Flexible arity function symbols can take arbitrary finite (possibly empty) sequence of arguments. In the literature the symbols with similar property are also referred to as "variable arity", "variadic" or "multiple arity" symbols.

Motivation

Languages with sequence variables and variable arity symbols have been used in various areas. Here we enumerate some of them:

- Knowledge management – Knowledge Interchange Format KIF [GF92] and its version SKIF [HM01] are extensions of first order language with (among other constructs) individual and sequence variables and variable arity function symbols. KIF is used to interchange knowledge among disparate computer systems. Another example of using sequence variables and variable arity symbols in knowledge systems is Ontolingua [FFR97] – a tool which provides a distributed collaborative environment to browse, create, edit, modify, and use ontologies.
- Databases – sequences and sequence variables provide flexibility in data representation and manipulation for genome or text databases, where much of the data has an inherently sequential structure. Numerous formalisms involving sequences and sequence variables, like Sequence Logic [GW92], Alignment Logic [GNU94], Sequence Datalog [MB95], String Calculus [GW00], [BLSS01], have been developed for this field.
- Rewriting – variable arity symbols used in rewriting usually come from flattening terms with associative top function symbol. Sequences and sequence variables (sometimes called also patterns), which are used together

with variable arity symbols, make the syntax more flexible and expressive, and increase the performance of a rewriting system (see [WB01], [Ham97]).
- Programming languages – variable arity symbols are supported by many of them. The programming language of Mathematica [Wol99] is one of such examples, which uses the full expressive power of sequence variables as well. A relation of Mathematica programming language and rewrite rule languages, and the role of sequence variables in this relation is discussed in [Buc96].
- Theorem proving – the Epilog package [Gen95] can be used in programs that manipulate information encoded in Standard Information Format (SIF) – a subset of KIF [GF92] language, containing sequence variables and variable arity symbols. Among the other routines, Epilog includes pattern matchers of various sorts, and an inference procedure based on model elimination.

These applications involve (and in some cases, essentially depend on) solving equations with sequence variables and variable arity symbols. The most used solving technique is matching. However, for some applications, like theorem proving or completion, more powerful solving techniques (unification, for instance) are needed.

The problem whether Knuth-Bendix completion procedure [KB67] can be extended to handle term rewriting systems with function symbols of variable arity, sequences and sequence variables (patterns) is stated as an open problem in [WB01]. The primary reason why it is an open problem is the absence of appropriate unification algorithm.

Particular instances of unification with sequence variables and flexible arity symbols are word equations [AP90], [Jaf90], [Sch93], equations over free semigroups [Mak77], equations over lists of atoms with concatenation [Col90], pattern matching.

Our Contribution

In this thesis we study in details solving and proving issues in theories with sequence variables and flexible arity symbols and

1. describe a Birkhoff-style calculus for pure equational logic with sequence variables and flexible arity symbols and prove its soundness and completeness;
2. design unification procedures for free, flat, restricted flat[1] and orderless equational theories with sequence variables and flexible arity symbols, prove decidability of the unification problem in these theories and minimality and completeness of the unification procedures;

[1] The difference between flat and restricted flat theories is in flattening terms like $f(\ldots, f(), \ldots)$: in the flat theory this term is equal to $f(\ldots, \ldots)$, while in the restricted flat theory it is not. Restricted flatness allows to flatten terms with non-empty argument lists only.

3. design a minimal and complete unification procedure for an extension of the free theory with patterns. Terms in this extension are called pattern-terms;
4. define ground total reduction orderings for terms and pattern-terms with sequence variables and flexible arity symbols;
5. show that the basic equational superposition with ordering and equational constraints [NR92a], [NR92b] is a refutationally complete proving method for the free theory with sequence variables and flexible arity symbols;
6. show that for a special class of free equations with sequence variables and flexible arity symbols unfailing completion [BDP89] can be used as a refutationally complete proving method.

Sequence variables and pattern-terms can be seen as particular examples of the pattern construct of [WB01]. Therefore, the unification procedure in the free theory with sequence variables and flexible arity symbols, and its extension to pattern-terms provide a solution to the open problem of designing unification procedure for these constructs ([WB01]).

The type of free, flat, restricted flat and orderless unification with sequence variables and flexible arity symbols in the Siekmann unification hierarchy [Sie78] is infinitary: for any unification problem there exists a minimal complete set of unifiers which is infinite for some problems. In the flat theory even matching is infinitary. However, in all theories special cases can be identified when the procedure terminates. The most important terminating case is the free unification problem in which all occurrences of sequence variables happen only in the last argument positions, like, for instance, in the unification problem $f(a, g(b, \overline{x}), \overline{x}) \stackrel{?}{=}_{\emptyset} f(x, g(y, h(\overline{y}), c), d, \overline{z})$. Such cases cover many useful applications and have finite minimal complete set of unifiers.

It should be mentioned that in the theorem proving context, quantification over sequence variables naturally introduces flexible arity symbols and constructs that we call patterns. For instance, Skolemizing the expression $\forall \overline{x} \exists y \Phi[\overline{x}, y]$, where $\overline{x}$ is a sequence variable, y is an individual variable and $\Phi[\overline{x}, y]$ is a formula which depends on $\overline{x}$ and y, introduces a flexible arity Skolem function f: $\forall \overline{x} \Phi[\overline{x}, f(\overline{x})]$. On the other hand, Skolemizing the expression $\forall x \exists \overline{y} \Phi[x, \overline{y}]$ introduces a pattern $h_{1,n(x)}(x)$, which can be seen as an abbreviation of a sequence of terms $h_1(x), \ldots, h_{n(x)}(x)$ of unknown length, where $h_1, \ldots, h_{n(x)}$ are Skolem functions. Therefore, unification for terms involving such patterns will play an important role in proving methods for theories where quantification over sequence variables is allowed. The pattern-unification procedure designed in this thesis solves unification problems that involve such pattern-terms.

In this thesis we use the term "flexible arity" instead of " variable arity", as it was suggested by Buchberger [Buc01], mainly because of the following reason: variable arity symbols, as they are understood in most of the cases in theorem proving or rewriting, are flattened associative symbols, i.e. flat symbols which take at least two arguments, while flexible arity symbols can have zero or one argument as well and are not necessarily flat. Non-flatness is one of main differences between unification with sequence variables and flexible arity symbols and associative unification: the unification problem $f(x, f(y, z)) \stackrel{?}{=} f(f(a, b), c)$, with the variables x, y, z and constants a, b, c, has no unifier, if f has a flexible arity, but admits a unifier $\{x \leftarrow a, y \leftarrow b, z \leftarrow c\}$ for an associative f. Even when f is a flat flexible arity symbol the problem would not be equivalent to A-unification: the substitution $\{x \leftarrow f(a), y \leftarrow f(b, c), z \leftarrow f()\}$ is a unifier for a flat f, but not for an associative f.

We define a reduction ordering $\succ_{mposv}$ for terms with sequence variables and flexible arity symbols, which is total on ground terms. It coincides with the lexicographic extension of multiset path ordering [Der82] on terms without sequence variables.

Similarly, we define a reduction ordering $\succ_{mpop}$ for pattern-terms with sequence variables and flexible arity symbols, which is total on ground terms. It coincides with $\succ_{mposv}$ on terms without patterns.

Using these two essential ingredients – minimal complete unification procedure and ground total reduction ordering – we show that the basic superposition with constraints is a refutationally complete theorem proving method for the free theory with sequence variables and flexible arity symbols (compare [NR95], [Rub95]).

In the special case of the free theory, where sequence variables appear only as the last arguments of terms, unfailing completion [BDP89] can be used to build a refutationally complete prover.

We have implemented the unification procedure (without the decision algorithm) as a Mathematica package and incorporated it into the Theorema system [BDJ$^+$00b], which aims at extending computer algebra systems by facilities for supporting mathematical proving. Currently the package is used in the Theorema Equational Prover. It makes Theorema probably the only system being able to handle equations which involve sequence variables and flexible arity symbols. The package also enhances Mathematica solving capabilities, considering unification as a solving method. We used the package, for instance, to find matches for S-polynomials in non-commutative Gröbner Bases algorithm [Mor85].

Structure of the Thesis

The thesis is organized as follows:

In Chapter 2 syntax and semantics of equational theories with sequence variables and flexible arity symbols are defined. A Birkhoff-style inference system is described and soundness and completeness of the calculus is proved.

In Chapter 3 a unification procedure in the free theory with sequence variables and flexible arity symbols is designed. Decidability of a free unification problem and soundness and completeness of the procedure are proved. Sufficient criteria for termination of the procedure are established. The theory is extended with pattern-terms and a unification procedure for the extension is designed. For a given unification problem, the pattern unification procedure enumerates a set of substitution/constraint pairs which constitutes the minimal complete set of solutions of the problem.

Chapter 4 starts with the definition of the reduction orderings $\succ_{mposv}$ and $\succ_{mpop}$. Further, refutation completeness of basic superposition with constraints is shown for the free theory with sequence variables and flexible arity symbols, and refutation completeness of unfailing completion is proved for a special case of this theory.

Equational unification in flat, restricted flat and orderless theories is studied in Chapter 5. Decidability of unification in these theories and minimal complete unification procedures are described.

The last chapter of the thesis is Conclusion.

2. Equational Theories with Sequence Variables and Flexible Arity Symbols

In this chapter syntax and semantics of equational theories with sequence variables and flexible arity symbols are described. Birkhoff-style calculus is given and its soundness and completeness is proved.

2.1 Syntax

The set of individual variables $\mathcal{IV}$ is a denumerable set of words consisting of an English letter and subsequent letters or digits, starting with x, y or z. The set of sequence variables $\mathcal{SV}$ is a denumerable set of words with overline, consisting of an English letter and subsequent letters or digits, starting with x, y or z.

Definition 2.1.1 (Constants). $(\mathcal{CONST}, \mathcal{FFIX}, \mathcal{FFLEX}, \doteq, \mathcal{AR})$ *is domain of constants iff*

- $\mathcal{CONST}$ *is a set of symbols ("the set of object constants"),*
- $\mathcal{FFIX}$ *is a set of symbols ("the set of function constants of fixed arity"),*
- $\mathcal{FFLEX}$ *is a set of symbols ("the set of function constants of flexible arity"),*
- $\doteq$ *is the "equality relation constant",*
- $\mathcal{AR} : (\mathcal{FFIX} \cup \{\doteq\}) \to \mathbb{N}$ *("the arity function"),*
- $\mathcal{AR}(\doteq) = 2$*,*
- $\mathcal{CONST}, \mathcal{FFIX}, \mathcal{FFLEX}, \{\doteq\}$ *are pairwise disjoint and disjoint from* $\mathcal{IV}$ *and* $\mathcal{SV}$*.*

Let now $\mathcal{V}$ stand for $(\mathcal{IV}, \mathcal{SV})$ (variables), $\mathcal{C}$ – for $(\mathcal{CONST}, \mathcal{FFIX}, \mathcal{FFLEX}, \doteq)$ (a domain of constants) and $\mathcal{P}$ – for $\{(,),,\}$ ("parentheses and comma"). We define terms and equations over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$.

Definition 2.1.2 (Term). *The set of terms (over* $(\mathcal{V}, \mathcal{C}, \mathcal{P})$*) is the smallest set of strings over* $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ *that satisfies the following conditions:*

- *If* $v \in \mathcal{IV} \cup \mathcal{SV}$ *then* v *is a term.*
- *If* $c \in \mathcal{CONST}$ *then* c *is a term.*

– *If $f \in \mathcal{FFIX}$, $\mathcal{AR}(f) = n$, $n \geq 0$ and $t_1, \dots, t_n$ are terms such that for all $1 \leq i \leq n$, $t_i \notin \mathcal{SV}$, then $f(t_1, \dots, t_n)$ is a term. f is called the head of $f(t_1, \dots, t_n)$.*
– *If $f \in \mathcal{FFLEX}$ and $t_1, \dots, t_n$ $(n \geq 0)$ are terms, then $f(t_1, \dots, t_n)$ is a term. f is called the head of $f(t_1, \dots, t_n)$.*

By the head of a constant we understand the constant itself.

Definition 2.1.3 (Equation). *The set of equations (over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$) is the smallest set of strings over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ that satisfies the following condition:*

– *If t_1 and t_2 are terms over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ such that $t_1 \notin \mathcal{SV}$ and $t_2 \notin \mathcal{SV}$, then $\doteq(t_1, t_2)$ is an equation over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$. $\doteq$ is called the head of the equation $\doteq(t_1, t_2)$.*

For equations we will use infix notation and write $t_1 \doteq t_2$ for $\doteq(t_1, t_2)$. If not otherwise stated, we will use x, y and z as metavariables ranging over individual variables, $\overline{x}$, $\overline{y}$ and $\overline{z}$ – metavariables over sequence variables, v and u – over (individual or sequence) variables, c – over object constants, f, g and h – over (fixed or flexible arity) function symbols, s and t – over terms, q – over equations. We might use indices with these symbols as well. The symbol '$=$' is used to denote equality on the meta level.

Let $\mathcal{L}$ be an equational language with the alphabet $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ and terms and equations defined as above. The set of terms of $\mathcal{L}$ is denoted by $\mathcal{T}(\mathcal{L})$ and the set of equations of $\mathcal{L}$ is denoted by $\mathcal{E}(\mathcal{L})$.

Let $t_1, \dots, t_n$ $(n \geq 0)$ be a sequence of terms. Then we denote by

– $ivars(t_1, \dots, t_n)$ – the set of all individual variables occurring in the sequence $t_1, \dots, t_n$.
– $svars(t_1, \dots, t_n)$ – the set of all sequence variables in $t_1, \dots, t_n$.
– $vars(t_1, \dots, t_n) = ivars(t_1, \dots, t_n) \cup svars(t_1, \dots, t_n)$ – the set of all variables in $t_1, \dots, t_n$.
– $const(t_1, \dots, t_n)$ – the set of all constants in $t_1, \dots, t_n$.
– $fix(t_1, \dots, t_n)$ – the set of fixed arity function symbols in $t_1, \dots, t_n$.
– $flex(t_1, \dots, t_n)$ – the set of flexible arity function symbols in $t_1, \dots, t_n$.
– $sig(t_1, \dots, t_n) = const(t_1, \dots, t_n) \cup fix(t_1, \dots, t_n) \cup flex(t_1, \dots, t_n)$ – the signature of $t_1, \dots, t_n$.

For a sequence of equations $s_1 \doteq t_1, \dots, s_n \doteq t_n$ $(n \geq 0)$ we define $ivars(s_1 \doteq t_1, \dots, s_n \doteq t_n) = ivars(s_1, t_1, \dots, s_n, t_n)$. The other notions are defined in the same way.

Definition 2.1.4 (Ground Term and Equation). *A term t (an equation q) is called ground iff $vars(t) = \emptyset$ $(vars(q) = \emptyset)$.*

Definition 2.1.5 (Substitution). *A substitution is a finite set*

$$\{x_1 \leftarrow s_1, \ldots, x_n \leftarrow s_n, \overline{x}_1 \leftarrow t_1^1, \ldots, t_{k_1}^1, \ldots, \overline{x}_m \leftarrow t_1^m, \ldots, t_{k_m}^m\}$$

where

- $n \geq 0$, $m \geq 0$ *and for all* $1 \leq i \leq m$, $k_i \geq 0$,
- $x_1, \ldots, x_n$ *are distinct individual variables,*
- $\overline{x}_1, \ldots, \overline{x}_m$ *are distinct sequence variables,*
- *for all* $1 \leq i \leq n$, s_i *is a term,* $s_i \notin \mathcal{SV}$ *and* $s_i \neq x_i$,
- *for all* $1 \leq i \leq m$, $t_1^i, \ldots, t_{k_i}^i$ *is a sequence of terms and if* k_i=1 *then* $t_{k_i}^i \neq \overline{x}_i$.

Each $x_i \leftarrow s_i$ $(\overline{x}_i \leftarrow t_1^i, \ldots, t_{k_i}^i)$ *is called a binding for* x_i $(\overline{x_i})$.

The substitution is called empty iff $n = 0$ and $m = 0$. Greek letters are used to denote substitutions. The letter ε denotes the empty substitution.

Definition 2.1.6 (Instance). *Given a term* t *and a substitution* θ, *we define an instance of* t *with respect to* θ, *denoted as* $t\theta$, *recursively as follows:*

- $x\theta = \begin{cases} s & if\ x \leftarrow s \in \theta, \\ x & otherwise \end{cases}$
- $\overline{x}\theta = \begin{cases} s_1, \ldots, s_m & if\ x \leftarrow s_1, \ldots, s_m \in \theta,\ m \geq 0 \\ \overline{x} & otherwise \end{cases}$
- $f(s_1, \ldots, s_n)\theta = f(s_1\theta, \ldots, s_n\theta)$

Instance of an equation is defined as follows: $(s_1 \doteq s_2)\theta = s_1\theta \doteq s_2\theta$

Definition 2.1.7 (Domain). *The domain of a substitution* σ *is defined as*

$$dom(\sigma) = \{v \in \mathcal{IV} \cup \mathcal{SV} \mid v\sigma \neq v\}.$$

Definition 2.1.8 (Codomain). *The codomain of a substitution* σ *is defined as*

$$cod(\sigma) = \{v\sigma \mid v \in dom(\sigma)\}.$$

Definition 2.1.9 (Range). *The range of a substitution* σ *is defined as*

$$ran(\sigma) = \cup_{v \in dom(\sigma)} vars(v\sigma).$$

Definition 2.1.10 (Ground Substitution). *A substitution* σ *is called ground substitution iff*

$$ran(\sigma) = \emptyset.$$

We extend the notion of instance of a term to the notion of instance of a sequence in a straightforward way: given a sequence of terms $t_1, \ldots, t_n$, $n \geq 0$ and a substitution θ, the instance of $t_1, \ldots, t_n$ with respect to θ, denoted as $(t_1, \ldots, t_n)\theta$, is the sequence $t_1\theta, \ldots, t_n\theta$.

Definition 2.1.11 (Composition of Substitutions). *Let*

$$\theta = \{x_1 \leftarrow s_1, \dots, x_n \leftarrow s_n, \overline{x_1} \leftarrow t_1^1, \dots, t_{k_1}^1, \dots, \overline{x_m} \leftarrow t_1^m, \dots, t_{k_m}^m\}$$

and

$$\lambda = \{y_1 \leftarrow d_1, \dots, y_n \leftarrow d_l, \overline{y_1} \leftarrow e_1^1, \dots, e_{q_1}^1, \dots, \overline{y_r} \leftarrow e_1^r, \dots, e_{q_r}^r\}$$

be two substitutions. Then the composition of θ and λ is the substitution, denoted by $\theta \circ \lambda$, obtained from the set

$$\begin{aligned}\{ \quad & x_1 \leftarrow s_1\lambda, \dots, x_n \leftarrow s_n\lambda, \overline{x_1} \leftarrow t_1^1\lambda, \dots, t_{k_1}^1\lambda, \dots, \overline{x_m} \leftarrow t_1^m\lambda, \dots, t_{k_m}^m\lambda, \\ & y_1 \leftarrow d_1, \dots, y_l \leftarrow d_l, \overline{y_1} \leftarrow e_1^1, \dots, e_{q_1}^1, \dots, \overline{y_r} \leftarrow e_1^r, \dots, e_{q_r}^r\}\end{aligned}$$

by deleting

– *all the elements $x_i \leftarrow s_i\lambda$ $(1 \le i \le n)$ for which $x_i = s_i\lambda$,*
– *all the elements $\overline{x_i} \leftarrow t_1^i\lambda, \dots, t_{k_i}^i\lambda$ $(1 \le i \le m)$ for which $k_i = 1$ and $\overline{x}_i = t_1^i\lambda$,*
– *all the elements $y_i \leftarrow d_i$ $(1 \le i \le l)$ such that $y_i \in \{x_1, \dots, x_n\}$,*
– *all the elements $\overline{y_i} \leftarrow e_1^i, \dots, e_{q_i}^i$ $(1 \le i \le r)$ such that $\overline{y_i} \in \{\overline{x_1}, \dots, \overline{x_m}\}$.*

Example 2.1.1. Let

$$\theta = \{x \leftarrow f(y),\ \ \overline{x} \leftarrow \overline{y}, \overline{x},\ \ \overline{y} \leftarrow \overline{y}, \overline{z}\}$$

and

$$\lambda = \{y \leftarrow g(c,c),\ \ z \leftarrow f(c), \overline{x} \leftarrow c,\ \overline{z} \leftarrow\}.$$

Then

$$\theta \circ \lambda = \{x \leftarrow f(g(c,c)),\ \ y \leftarrow g(c,c),\ z \leftarrow f(c),\ \ \overline{x} \leftarrow \overline{y}, c,\ \ \overline{z} \leftarrow\}.$$

Theorem 2.1.1. *For a term t and substitutions θ and λ*

$$t\theta \circ \lambda = t\theta\lambda.$$

Proof. We prove the theorem by induction on the structure of t.

1. $t \in \mathcal{IV}$.
   a) $t \leftarrow s\ \in \theta \circ \lambda$, for some s. Then $t\theta \circ \lambda = s$. We show that $t\theta\lambda = s$. We have the following two cases:
      i. There exists an r such that $t \leftarrow r\ \in \theta$ and $s = r\lambda$. Then

$$t\theta\lambda = r\lambda = s.$$

ii. $t \leftarrow s \ \in \lambda$ and θ does not contain a binding for t. Then

$$t\theta\lambda = t\lambda = s.$$

b) $\theta \circ \lambda$ does not contain a binding for t. Then $t\theta \circ \lambda = t$. We show that $t\theta\lambda = t$. We have the following two cases:

i. Neither θ nor λ contain a binding for t. Then

$$t\theta\lambda = t\lambda = t.$$

ii. There exists an r such $t \leftarrow r \ \in \theta$ and $r\lambda = t$. Then

$$t\theta\lambda = r\lambda = t.$$

2. $t \in \mathcal{SV}$.

a) $t \leftarrow s_1, \ldots, s_n \ \in \theta \circ \lambda$ for some $s_1, \ldots, s_n$, $n \geq 0$. Then $t\theta \circ \lambda = s_1, \ldots, s_n$. We show $t\theta\lambda = s_1, \ldots, s_n$. We have the following cases:

i. There exist $r_1, \ldots, r_m$, $m \geq 0$, such that $t \leftarrow r_1, \ldots, r_m \ \in \theta$ and $r_1\lambda, \ldots, r_m\lambda = s_1, \ldots, s_n$. Then

$$t\theta\lambda = (r_1, \ldots, r_m)\lambda = r_1\lambda, \ldots, r_m\lambda = s_1, \ldots, s_n.$$

ii. $t \leftarrow s_1, \ldots, s_n \ \in \lambda$ and θ does not contain a binding for t. Then

$$t\theta\lambda = t\lambda = s_1, \ldots, s_m.$$

b) $\theta \circ \lambda$ does not contain a binding for t. Then $t\theta \circ \lambda = t$. We show that $t\theta\lambda = t$. We have the following two cases:

i. Neither θ nor λ contain a binding for t. Then

$$t\theta\lambda = t\lambda = t.$$

ii. There exist $r_1, \ldots, r_m$, $m \geq 0$, such that $t \leftarrow r_1, \ldots, r_m \ \in \theta$ and $(r_1, \ldots, r_m)\lambda = t$. Then

$$t\theta\lambda = (r_1, \ldots, r_m)\lambda = t.$$

3. t is a compound term $f(s_1, \ldots, s_n)$, $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$. Then

$$\begin{array}{ll} f(s_1, \ldots, s_n)\theta \circ \lambda & = \text{(by Definition 2.1.6)} \\ f(s_1\theta \circ \lambda, \ldots, s_n\theta \circ \lambda) & = \text{(by induction hypothesis)} \\ f(s_1\theta\lambda, \ldots, s_n\theta\lambda) & = \text{(by Definition 2.1.6)} \\ f(s_1\theta, \ldots, s_n\theta)\lambda & = \text{(by Definition 2.1.6)} \\ f(s_1, \ldots, s_n)\theta\lambda. & \end{array}$$

□

Corollary 2.1.1. *If $t_1, \ldots, t_n$ is a sequence of terms and θ and λ are substitutions, then*

$$(t_1 \ldots, t_n)\theta \circ \lambda = ((t_1 \ldots, t_n)\theta)\lambda.$$

Theorem 2.1.2. *For any substitutions θ, λ and σ,*

$$(\theta \circ \lambda) \circ \sigma = \theta \circ (\lambda \circ \sigma).$$

Proof. ($\subseteq$) Let $v \leftarrow r_1, \ldots, r_n \in (\theta \circ \lambda) \circ \sigma$, where $n \geq 0$, if $v \in \mathcal{SV}$ and $n = 1$, if $v \in \mathcal{IV}$. We show that

$$v \leftarrow r_1, \ldots, r_n \in \theta \circ (\lambda \circ \sigma). \tag{2.1}$$

To prove 2.1, by Definition 2.1.6 it suffices to show

$$v\theta \circ (\lambda \circ \sigma) = r_1, \ldots, r_n. \tag{2.2}$$

We prove 2.2 as follows:

$$\begin{array}{ll}
v\theta \circ (\lambda \circ \sigma) & = \text{(by Definition 2.1.6)} \\
v\theta(\lambda \circ \sigma) & = \text{(by Definition 2.1.6)} \\
v\theta\lambda\sigma & = \text{(by Theorem 2.1.1)} \\
v(\theta \circ \lambda)\sigma & = \text{(by Theorem 2.1.1)} \\
v(\theta \circ \lambda) \circ \sigma) & = \\
r_1, \ldots, r_n. &
\end{array}$$

($\supseteq$) Similar to ($\subseteq$). $\square$

2.2 Semantics

In this subsection we define semantics of $\mathcal{L}$. To interpret sequence variables and flexible arity symbols we choose an approach similar to the semantics of SKIF language [HM01]. First we adopt some notions from [HM01] with a slight modification:

For any set A, let A^n be the set of sequences of length n of members of A, i.e. functions from the set $\{0, 1, \ldots, n-1\}$ of ordinals less then n into A (it follows that $A^0 = \{\emptyset\}$ since the set of ordinals less than 0 is empty). We call the members of A^n n-tuples over A. We will write the n-tuple $\{\langle 0, a_0\rangle, \langle 1, a_1\rangle, \ldots \langle n-1, a_{n-1}\rangle\}$ as $\langle a_0, a_1, \ldots, a_{n-1}\rangle$. In particular, we will sometimes refer to $\emptyset$ as $'\langle\rangle'$ when thinking of it as a 0-tuple. Let A^* be the set of all n-tuples over A, for all n, i.e. $A^* = \cup_{n<\omega} A^n$. We will call the members of A^* *tuples over* A. We want to consider n-tuples in which there is no distinction between 1-tuple and its (sole) member, so we define an equivalence relation $\approx$ on $A \cup A^*$ by $\langle x\rangle \approx x$. Let $[a]_\approx$ denote an equivalence class of a on $A \cup A^*$ with respect to $\approx$. Then we define $A^{**} = \{[t]_\approx \mid t \in A \cup A^*\}$. We call the members of A^{**} rows over A or A-rows. Notice that any element of A constitutes a singleton row.

We define an operation of concatenation cc on A^{**} as follows:

$$
\begin{aligned}
&cc() = [\langle\rangle]_\approx.\\
&cc([a]_\approx) = [\langle a\rangle]_\approx,\ a \in A.\\
&cc([\langle t_1, \dots, t_n\rangle]_\approx) = [\langle t_1, \dots, t_n\rangle]_\approx,\ \langle t_1, \dots, t_n\rangle \in A^*.\\
&cc([a]_\approx, [b]_\approx) = [\langle a, b\rangle]_\approx,\ a \in A,\ b \in A.\\
&cc([a]_\approx, [\langle t_1, \dots, t_n\rangle]_\approx) = [\langle a, t_1, \dots, t_n\rangle]_\approx,\ a \in A,\ \langle t_1, \dots, t_n\rangle \in A^*.\\
&cc([\langle t_1, \dots, t_n\rangle]_\approx, [a]_\approx) = [\langle t_1, \dots, t_n, a\rangle]_\approx,\ \langle t_1, \dots, t_n\rangle \in A^*, a \in A.\\
&cc([\langle t_1, \dots, t_n\rangle]_\approx, [\langle s_1, \dots, s_m\rangle]_\approx) = [\langle t_1, \dots, t_n, s_1, \dots, s_m\rangle]_\approx,\\
&\qquad \langle t_1, \dots, t_n\rangle \in A^*, \langle s_1, \dots, s_m\rangle \in A^*.
\end{aligned}
$$

It can be easily shown that cc is associative. Therefore we will use cc in the flattened form.

We say that a set of rows S satisfies functional condition iff for any two rows $[\langle e_1, \dots, e_n, s\rangle]_\approx \in S$ and $[\langle e_1, \dots, e_n, t\rangle]_\approx \in S$, $n \geq 0$, we have $s = t$. It implies that if a non-empty set of rows satisfies functional condition then it contains exactly one singleton row.

Now we define an interpretation. An interpretation I for the language $\mathcal{L}$ consists of

- a non-empty set D called a domain of I,
- an assignment for each constant c in $\mathcal{L}$ of an element $c_I \in D^{**}$ such that $c_I = [\langle d\rangle]_\approx$, where $d \in D$.
- an assignment for each n-ary function symbol f in $\mathcal{L}$ a set $f_I \subseteq D^{**}$ of the form
$$\{[\langle d_1, \dots, d_n, d_{n+1}\rangle]_\approx \mid d_1, \dots, d_n, d_{n+1} \in D\},$$
satisfying functional condition,
- an assignment for each flexible arity function symbol f in $\mathcal{L}$ a set $f_I \subseteq D^{**}$ of the form
$$\{[\langle d_1, \dots, d_k\rangle]_\approx \mid k \geq 0,\ d_1, \dots, d_k \in D\},$$
satisfying functional condition.

We need to relate terms of $\mathcal{L}$ to rows of elements of the domain. We do this by making use of the notion of a state. A state σ^I (over I) is defined as follows:

- $\sigma^I(x) = [\langle d\rangle]_\approx \in D^{**}$, for an individual variable x,
- $\sigma^I(\overline{x}) = [\langle d_1, \dots, d_n\rangle]_\approx \in D^{**}$, for a sequence variable $\overline{x}$,
- $\sigma^I(c) = c_I$, for a constant c,
- $\sigma^I(f(t_1, \dots, t_n)) = [\langle s\rangle]_\approx \in D^{**}$ for a term $f(t_1, \dots, t_n)$ with fixed or flexible arity head f, where s is the unique element of D such that $cc(\sigma^I(t_1), \dots, \sigma^I(t_n), [\langle s\rangle]_\approx) \in f_I$.

Let $true$ and $false$ be truth values. We define truth value in a state σ^I over I of an equation $t_1 \doteq t_2$ in $\mathcal{L}$, written $I_\sigma(t_1 \doteq t_2)$, as follows: $I_\sigma(t_1 \doteq t_2) = true$ iff $\sigma^I(t_1) = \sigma^I(t_2)$, otherwise $I_\sigma(t_1 \doteq t_2) = false$.

Finally, we define truth value of an equation $t_1 \doteq t_2$ (in the language $\mathcal{L}$) in I, written $I(t_1 \doteq t_2)$, as follows: $I(t_1 \doteq t_2) = true$ iff $I_\sigma(t_1 \doteq t_2) = true$ for all σ^I over I, otherwise $I(t_1 \doteq t_2) = false$.

We say that $t_1 \doteq t_2$ is true in I (or I is a model of $t_1 \doteq t_2$, or I satisfies $t_1 \doteq t_2$) and write $I \models t_1 \doteq t_2$ iff $I(t_1 \doteq t_2) = true$. The relation $\models$ is called a semantic consequence relation.

An interpretation I is a model for a set of equations E (written $I \models E$) iff I is a model for each equation in E.

An equation $t_1 \doteq t_2$ is a semantic consequence of a set of equations E, written $E \models t_1 \doteq t_2$, iff every model of E is a model of $t_1 \doteq t_2$.

2.3 Calculus – Soundness and Completeness

Equational theory E (over $\mathcal{L}$) is a subset of $\mathcal{E}(\mathcal{L})$. We call it E-theory, and the equations in E are called axioms. We define derivability in an E-theory as follows:

Definition 2.3.1. *Let E be an equational theory. The set of equations derivable from E is inductively defined by the following schema:*

- *(Axioms) The equations in E are derivable from E.*
- *(Reflexivity) Every equation $t \doteq t$ is derivable from E.*
- *(Symmetry) If the equation $t_1 \doteq t_2$ is derivable from E, then so is $t_2 \doteq t_1$.*
- *(Transitivity) If the equations $t_1 \doteq t_2$ and $t_2 \doteq t_3$ are derivable from E, the so is $t_1 \doteq t_3$.*
- *(Replacement) If the equations $s_1 \doteq t_1, \ldots, s_n \doteq t_n$ are derivable from E, then so is*
$$f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n),$$
for any n-ary or flexible arity function symbol f.
- *(Substitution for an individual variable) If the equation $t_1 \doteq t_2$ is derivable from E, then so is*
$$t_1\{x \leftarrow s\} \doteq t_2\{x \leftarrow s\},$$
for any term s with $s \notin \mathcal{SV}$, and individual variable x.
- *(Substitution for a sequence variable) If the equation $t_1 \doteq t_2$ is derivable from E, then so is*
$$t_1\{\overline{x} \leftarrow s_1, \ldots, s_n\} \doteq t_2\{\overline{x} \leftarrow s_1, \ldots, s_n\},$$
for any terms $s_1, \ldots, s_n$, $n \geq 0$, and sequence variable $\overline{x}$.

To denote the fact that the equation $t_1 \doteq t_2$ *is derivable from* E*, we use the notation* $E \vdash t_1 \doteq t_2$*. We also say that* $t_1 \doteq t_2$ *is a theorem of* E *if* $E \vdash t_1 \doteq t_2$*.*

For an E-theory and terms s and t with $s, t \notin \mathcal{SV}$, we define two binary operations: $\doteq_E$ and $\equiv_E$ on $\mathcal{T}(\mathcal{L})$ as follows:

$$\doteq_E = \{\langle s, t\rangle \mid E \models s \doteq t\}.$$

$$\equiv_E = \{\langle s, t\rangle \mid E \vdash s \doteq t\}.$$

We write $s \doteq_E$ iff $\langle s, t\rangle \in \doteq_E$ and say that s is equal to t modulo E. The notation $s \equiv_E$ is used to express the fact that $\langle s, t\rangle \in \equiv_E$.

Our goal is to show soundness and completeness of the calculus described above. First we need some preliminary results and definitions.

Proposition 2.3.1. *Let* $E \subseteq \mathcal{E}(\mathcal{L})$*. Then* $\equiv_E$ *is a congruence relation on* $\mathcal{T}(\mathcal{L})$*.*

Proof. An inspection of the inference rules immediately leads to this result. Note that there is no need to consider substitution rules. □

Let $[t]_{\equiv_E}$ be an equivalence class of t in $\mathcal{T}(\mathcal{L})$ with respect to $\equiv_E$. Then we define

$$\mathcal{T}(\mathcal{L})/_{\equiv_E} = \{[t]_{\equiv_E} \mid t \in \mathcal{T}(\mathcal{L})\}.$$

Now we define an interpretation I_E with the domain $\mathcal{T}(\mathcal{L})/_{\equiv_E}$ and the following assignments:

- for each $c \in \mathcal{L}$, $c_{I_E} = [\langle [c]_{\equiv_E}\rangle]_{\approx} \in \mathcal{T}(\mathcal{L})/_{\equiv_E}{}^{**}$.
- for each n-ary function symbol f in $\mathcal{L}$, a set $f_{I_E} \subseteq \mathcal{T}(\mathcal{L})/_{\equiv_E}{}^{**}$ of the form

  $$\{[\langle [t_1]_{\equiv_E}, \dots, [t_n]_{\equiv_E}, [f(t_1,\dots,t_n)]_{\equiv_E}\rangle]_{\approx} \mid [t_1]_{\equiv_E}, \dots, [t_n]_{\equiv_E}, [f(t_1,\dots,t_n)]_{\equiv_E} \in \mathcal{T}(\mathcal{L})/_{\equiv_E}\},$$

  satisfying functional condition,
- for each flexible arity function symbol f in $\mathcal{L}$, a set $f_{I_E} \subseteq \mathcal{T}(\mathcal{L})/_{\equiv_E}{}^{**}$ of the form

  $$\{[\langle [t_1]_{\equiv_E}, \dots, [t_k]_{\equiv_E}, [f(t_1,\dots,t_k)]_{\equiv_E}\rangle]_{\approx} \mid k \geq 0,\ [t_1]_{\equiv_E}, \dots, [t_k]_{\equiv_E}, [f(t_1,\dots,t_k)]_{\equiv_E} \in \mathcal{T}(\mathcal{L})/_{\equiv_E}\},$$

  satisfying functional condition.

The following proposition holds:

Theorem 2.3.1. *Let* $t \in \mathcal{T}(\mathcal{L})$*,* $t \notin \mathcal{SV}$*,* σ^{I_E} *be a state over* I_E *and* θ *be a substitution with* $dom(\theta) = vars(t)$ *such that*

– *for all* $x \in vars(t)$, $\sigma^{I_E}(x) = [\langle [s]_{\equiv_E} \rangle]_\approx \in \mathcal{T}(\mathcal{L})/_{\equiv_E}{}^{**}$ *and* $x \leftarrow s \in \theta$, *for some* $[s]_{\equiv_E} \in \mathcal{T}(\mathcal{L})/_{\equiv_\mathcal{E}}$,
– *for all* $\overline{x} \in vars(t)$, $\sigma^{I_E}(\overline{x}) = [\langle [t_1]_{\equiv_E}, \ldots, [t_k]_{\equiv_E} \rangle]_\approx \in \mathcal{T}(\mathcal{L})/_{\equiv_E}{}^{**}$ *and* $x \leftarrow t_1, \ldots, t_k \in \theta$, *for some* $\langle [t_1]_{\equiv_E}, \ldots, [t_k]_{\equiv_E} \rangle \in \mathcal{T}(\mathcal{L})/_{\equiv_E}{}^{*}$.

Then $\sigma^{I_E} = [\langle [t\theta]_{\equiv_E} \rangle]_\approx$.

Proof. We use induction on the structure of t.

1. $t = x$. Then $\theta = \{x \leftarrow s\}$ and $\sigma^{I_E}(t) = [\langle [s]_{\equiv_E} \rangle]_\approx = [\langle [x\theta]_{\equiv_E} \rangle]_\approx$.
2. $t = c$. Then $\theta = \varepsilon$ and $\sigma^{I_E}(t) = [\langle [c]_{\equiv_E} \rangle]_\approx = [\langle [c\theta]_{\equiv_E} \rangle]_\approx$.
3. $t = f(r_1, \ldots, r_n)$, f is either fixed or flexible arity. Then $\sigma^{I_E}(t) = [\langle [r]_{\equiv_E} \rangle]_\approx$, such that $cc(\sigma^{I_E}(r_1), \ldots, \sigma^{I_E}(r_n), [\langle [r]_{\equiv_E} \rangle]_\approx) \in f_{I_E}$. Assume without loss of generality that $r_1 = \overline{x_1}, \ldots, r_k = \overline{x_k}$, $0 \le k \le n$ are all sequence variables among $r_1, \ldots, r_n$ and $\sigma^{I_E}(r_i) = [\langle [t_1^i]_{\equiv_E}, \ldots, [t_{m_i}^i]_{\equiv_E} \rangle]_\approx$ for all $1 \le i \le k$. Then

$$\begin{aligned} &cc(\sigma^{I_E}(r_1), \ldots, \sigma^{I_E}(r_n), [\langle [r]_{\equiv_E} \rangle]_\approx) = \\ &\quad [\langle [t_1^1]_{\equiv_E}, \ldots, [t_{m_k}^k]_{\equiv_E}, \sigma^{I_E}(r_{k+1}\theta), \ldots, \sigma^{I_E}(r_n\theta), [\langle [r]_{\equiv_E} \rangle]_\approx \rangle]_\approx. \end{aligned}$$

But then

$$\begin{aligned} &r = \\ &f(t_1^1, \ldots, t_{m_k}^k, r_{k+1}\theta, \ldots, r_n\theta) = \\ &f(r_1\theta, \ldots, r_k\theta, r_{k+1}\theta, \ldots, r_n\theta) = \\ &f(r_1, \ldots, r_k, r_{k+1}, \ldots, r_n)\theta = \\ &t\theta. \end{aligned}$$

Thus, $\sigma^{I_E}(t) = [\langle [t\theta]_{\equiv_E} \rangle]_\approx$.

□

Theorem 2.3.2. *Let* $E \cup \{s \doteq t\} \subseteq \mathcal{E}(\mathcal{L})$ *and* θ *be a substitution. Then*

$$if\ E \vdash s \doteq t,\ then\ E \vdash s\theta \doteq t\theta.$$

Proof. If θ is idempotent (i.e. $\theta \circ \theta = \theta$), then the theorem can be proved by induction on the length of θ, using the substitution inference rules. If θ is non-idempotent, then it can be represented as a composition of two idempotent substitutions $\sigma \circ \lambda$ with the property that for each (individual or sequence) variable v from $vars(s \doteq t)$, the (sequences of) terms $v\theta$ and $v\sigma \circ \lambda$ are the same. □

Theorem 2.3.3. *Let* $E \subseteq \mathcal{E}(\mathcal{L})$. *Then* $I_E \models E$.

Proof. Let $s \doteq t \in E$. Than by the first inference rule (Axioms) we have

$$E \vdash s \doteq t. \tag{2.3}$$

From (2.3), by Theorem 2.3.2 we have for all substitutions θ

$$E \vdash s\theta \doteq t\theta. \tag{2.4}$$

From (2.4), by definition of $\equiv_E$, we have

$$s\theta \equiv_E t\theta. \tag{2.5}$$

and, thus

$$[s\theta]_{\equiv_E} = [t\theta]_{\equiv_E}. \tag{2.6}$$

From (2.6), by Theorem 2.3.1 we get for all state σ^{I_E} over I_E that

$$\sigma^{I_E}(s) = \sigma^{I_E}(t). \tag{2.7}$$

From (2.7), by definition of model we get

$$I_E \models s \doteq t. \tag{2.8}$$

□

Theorem 2.3.4. *Let $E \cup \{s \doteq t\} \subseteq \mathcal{E}(\mathcal{L})$. Then*

$$if \;\; I_E \models s \doteq t \;\; then \;\; E \vdash s \doteq t.$$

Proof. Let $I_E \models s \doteq t$ and assume by contradiction that $E \nvdash s \doteq t$. Then, by definition of $\equiv_E$, we have

$$s \not\equiv_E t. \tag{2.9}$$

From (2.9), by definition of equivalence classes, we have

$$[s]_{\equiv_E} \neq [t]_{\equiv_E}. \tag{2.10}$$

Let σ^{I_E} be a state over I_E such that $\sigma^{I_E}(s) = [\langle [s]_{\equiv_E} \rangle]_\approx$ and $\sigma^{I_E}(t) = [\langle [t]_{\equiv_E} \rangle]_\approx$. Then from (2.10), by definition of rows and the relation $\approx$, we get

$$\sigma^{I_E}(s) \neq \sigma^{I_E}(t). \tag{2.11}$$

From (2.11), by definition of model, we get

$$I_E \not\models s \doteq t \tag{2.12}$$

which contradicts the assumption that $I_E \models s \doteq t$. Thus, $E \vdash s \doteq t$. □

Next, we prove the soundness theorem:

Theorem 2.3.5 (Soundness). *Let* $E \cup \{t_1 \doteq t_2\} \subseteq \mathcal{E}(\mathcal{L})$. *Then*

$$if\ E \vdash t_1 \doteq t_2,\ then\ E \models t_1 \doteq t_2.$$

Proof. Let $\vdash t_1 \doteq t_2$. We show $E \models t_1 \doteq t_2$ by well-founded induction on the length n of derivation of $t_1 \doteq t_2$.

Assume that the assertion has already been shown for derivations of length less than n. Take a derivation of $t_1 \doteq t_2$ of length n. We distinguish 7 cases according to the last step in the given derivation

1. $t_1 \doteq t_2 \in E$. Then trivially, $E \models t_1 \doteq t_2$.
2. The last step of the derivation is an application of reflexivity. Then t_1 and t_2 are identical and $E \models t_1 \doteq t_2$.
3. The last step of the derivation is an application of symmetry. It means that at some prior stage we have derived $t_1 \doteq t_1$. By the induction hypothesis, $E \models t_2 \doteq t_1$, which, by the definition of semantic consequence, implies $E \models t_1 \doteq t_2$.
4. The last step of the derivation is an application of transitivity. It means that at some prior step we have derived $t_1 \doteq s$ and $s \doteq t_2$ for some s. Then, by the induction hypothesis, $E \models t_1 \doteq s$ and $E \models s \doteq t_2$, which,by the definition of semantic consequence, implies $E \models t_1 \doteq t_2$.
5. The last step of the derivation is an application of the replacement rule. Let t_1 be $f(s_1, \ldots, s_n)$ and t_2 be $f(r_1, \ldots, r_n)$. Then at some prior stages we have derived equations $s_1 \doteq r_1, \ldots, s_n \doteq r_n$. By the induction hypothesis, $E \models s_1 \doteq r_1, \ldots, E \models s_n \doteq r_n$. We have to show that $E \models f(s_1, \ldots, s_n) \doteq f(r_1, \ldots, r_n)$. Let M be an arbitrary model of E and σ^M be an arbitrary state over M. Then:
   - $\sigma^M(f(s_1, \ldots, s_n)) = [\langle l_1 \rangle]_\approx$, where l_1 is the unique element in the domain of M such that $cc(\sigma^M(s_1), \ldots, \sigma^M(s_n), [\langle l_1 \rangle]_\approx) \in f_M$,
   - $\sigma^M(f(r_1, \ldots, r_n)) = [\langle l_2 \rangle]_\approx$, where l_2 is the unique element in the domain of M such that $cc(\sigma^M(r_1), \ldots, \sigma^M(r_n), [\langle l_2 \rangle]_\approx) \in f_M$.

   By the induction hypothesis, $\sigma^M(s_i) = \sigma^M(r_i)$, $1 \leq i \leq n$. Then, since f_M satisfies the functional condition, $[\langle l_1 \rangle]_\approx = [\langle l_2 \rangle]_\approx$, which implies $\sigma^M(f(s_1, \ldots, s_n)) = \sigma^M(f(r_1, \ldots, r_n))$. Therefore, by the definition of model, $M \models f(s_1, \ldots, s_n) \doteq f(r_1, \ldots, r_n)$, which, by the definition of semantic consequence, implies $E \models f(s_1, \ldots, s_n) \doteq f(r_1, \ldots, r_n)$.
6. The last step in the derivation is an application of the substitution rule for individual variables. It means that at some prior stage we have derived an equation $r_1 \doteq r_2$, and t_1 is $r_1\{x \leftarrow s\}$ and t_2 is $r_2\{x \leftarrow s\}$, for some individual variable x and a term s, $s \notin \mathcal{SV}$. Assume without

loss of generality that x does not occur in s. We have to show that $E \models r_1\{x \leftarrow s\} \doteq r_2\{x \leftarrow s\}$. Let M be an arbitrary model of E and σ^M be an arbitrary state over M. Let η^M be a state over M such that $\eta^M(x) = \sigma^M(s)$, and for any other (individual or sequence) variable v, $\eta^M(v) = \sigma^M(v)$. Then $\sigma^M(r_1\{x \leftarrow s\}) = \eta^M(r_1\{x \leftarrow s\}) = \eta^M(r_1)$ and $\sigma^M(r_2\{x \leftarrow s\}) = \eta^M(r_2\{x \leftarrow s\}) = \eta^M(r_2)$. By the induction hypothesis we have $\eta^M(r_1) = \eta^M(r_2)$. Therefore $\sigma^M(r_1\{x \leftarrow s\}) = \sigma^M(r_2\{x \leftarrow s\})$, which, by the definition of model, implies $M \models r_1\{x \leftarrow s\} \doteq r_2\{x \leftarrow s\}$. Therefore, by the definition of semantic consequence, we get $E \models r_1\{x \leftarrow s\} \doteq r_2\{x \leftarrow s\}$.

7. The last step in the derivation is an application of the substitution rule for sequence variables. It means that at some prior stage we have derived an equation $r_1 \doteq r_2$, and t_1 is $r_1\{\overline{x} \leftarrow s_1, \dots, s_n\}$ and t_2 is $r_2\{\overline{x} \leftarrow s_1, \dots, s_n\}$, for some sequence variable $\overline{x}$ and terms $s_1, \dots, s_n$. Assume without loss of generality that $\overline{x}$ does not occur in any of the $s_1, \dots, s_n$. We have to show that $E \models r_1\{\overline{x} \leftarrow s_1, \dots, s_n\} \doteq r_2\{\overline{x} \leftarrow s_1, \dots, s_n\}$. Let M be an arbitrary model of E and σ^M be an arbitrary state over M. Let η^M be a state over M such that $\eta^M(\overline{x}) = cc(\sigma^M(s_1), \dots, \sigma^M(s_n))$, and for any other (sequence or individual) variable v, $\eta^M(v) = \sigma^M(v)$. Then $\sigma^M(r_1\{\overline{x} \leftarrow s_1, \dots, s_n\}) = \eta^M(r_1\{\overline{x} \leftarrow s_1, \dots, s_n\}) = \eta^M(r_1)$ and $\sigma^M(r_2\{\overline{x} \leftarrow s_1, \dots, s_n\}) = \eta^M(r_2\{\overline{x} \leftarrow s_1, \dots, s_n\}) = \eta^M(r_2)$. By the induction hypothesis we have $\eta^M(r_1) = \eta^M(r_2)$. Therefore $\sigma^M(r_1\{\overline{x} \leftarrow s_1 \dots, s_n\}) = \sigma^M(r_2\{\overline{x} \leftarrow s_1, \dots, s_n\})$, which, by the definition of model, implies $M \models r_1\{\overline{x} \leftarrow s_1, \dots, s_n\} \doteq r_2\{\overline{x} \leftarrow s_1, \dots, s_n\}$. Therefore, by the definition of semantic consequence, we get $E \models r_1\{\overline{x} \leftarrow s_1, \dots, s_n\} \doteq r_2\{\overline{x} \leftarrow s_1, \dots, s_n\}$.

□

Now we prove completeness:

Theorem 2.3.6 (Completeness). *Let* $E \cup \{t_1 \doteq t_2\} \subseteq \mathcal{E}(\mathcal{L})$. *Then*

$$if \;\; E \models t_1 \doteq t_2, \;\; then \;\; E \vdash t_1 \doteq t_2.$$

Proof. Let $E \models t_1 \doteq t_2$. Then by Theorem 2.3.3 we have $I_E \models t_1 \doteq t_2$, which, by Theorem 2.3.4, implies $E \vdash t_1 \doteq t_2$. □

As a summary of the results above, we have the following generalization of Birkhoff's theorem [Bir35] for equational theories with sequence variables and flexible arity symbols:

Theorem 2.3.7 (Soundness and Completeness). *Let* $E \cup \{t_1 \doteq t_2\} \subseteq \mathcal{E}(\mathcal{L})$. *Then*

$$E \vdash t_1 \doteq t_2 \;\; iff \;\; E \models t_1 \doteq t_2.$$

The following theories are some of interesting examples of equational theories with sequence variables and flexible arity symbols:

1. Free theory: $E = \emptyset$;
2. Flat theory (a theory with flexible arity flat symbol(s)):
$$E = \{f(\overline{x}, f(\overline{y}), \overline{z}) \doteq f(\overline{x}, \overline{y}, \overline{z})\}.$$
3. Restricted flat theory (a restricted theory with flat flexible arity symbol(s)):
$$E = \{f(\overline{x}, f(\overline{y_1}, x, \overline{y_2}), \overline{z}) \doteq f(\overline{x}, \overline{y_1}, x, \overline{y_2}, \overline{z})\}.$$
4. Orderless theory (a theory with orderless flexible arity symbol(s)):
$$E = \{f(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq f(\overline{x}, y, \overline{y}, x, \overline{z})\}.$$
5. Flat-orderless theory (a theory with flat-orderless flexible arity symbol(s)):
$$\begin{aligned} E = \{ \quad & f(\overline{x}, f(\overline{y}), \overline{z}) \doteq f(\overline{x}, \overline{y}, \overline{z}), \\ & f(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq f(\overline{x}, y, \overline{y}, x, \overline{z})\}. \end{aligned}$$
6. Restricted flat-orderless theory (a restricted theory with flat-orderless flexible arity symbol(s)):
$$\begin{aligned} E = \{ \quad & f(\overline{x}, f(\overline{y_1}, x, \overline{y_2}), \overline{z}) \doteq f(\overline{x}, \overline{y_1}, x, \overline{y_2}, \overline{z}), \\ & f(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq f(\overline{x}, y, \overline{y}, x, \overline{z})\}. \end{aligned}$$

In this thesis we do not consider flat-orderless and restricted flat-orderless theories. We will refer to free, flat, restricted flat and orderless theories respectively as $\emptyset$-, F-, RF- and O-theory. Every RF-theory is a F-theory, but not vice versa: $f(a, f(\overline{x}), b) \doteq_F f(a, \overline{x}, b)$, but not $f(a, f(\overline{x}), b) \doteq_{RF} f(a, \overline{x}, b)$.

‘cc xddvy ewrrrqqqqw22222 ” ==========//yyccxhvggggg hc ccc ccxjfkkjgjh,g,hhghlk,tgfhrrrr/¿R?t?bb vvvvv5v 5 DDDTDdfRFDRRF???¿ $EDEl4rell., 3, 3, 44rcc/$ 999999999999999 llllloo099
aqqqOOOOOOOOOOOOJIU UUUU;lk[9oipk/mill,‘,iiiiii’/l90’9l9099bbbb[1]

[1] Elene contributed to writing the last paragraph.

3. Solving in the Free Theory

This chapter introduces unification – a solving method in the free theory with sequence variables and flexible arity symbols. Decidability of unification in the free theory is proved. A unification procedure is designed. It enumerates a minimal complete set of unifiers for a given unification problem and terminates if the set is finite. Several sufficient conditions for termination of the procedure are established.

The free theory is extended in two ways. First extension involves pattern-terms, which represent sequences of terms of unknown length matching certain pattern. Unification procedure for pattern-term extension enumerates a set of substitution/constraint pairs which constitutes the minimal complete set of unifiers. Second extension allows sequence variables to be arguments in terms with fixed arity heads. It is shown that this extension can be reduced to a finite set of unification problems without sequence variables in terms with fixed arity heads.

Results presented in this chapter are further developments and refinements of the results reported in [Kut02b], [Kut02a] and [Kut02c].

3.1 Preliminaries

Solving equations in an E-theory is called E-unification (see [JK91], [BS93], [BS01] for an introduction to unification theory). The fact that the equation $s \doteq t$ has to be solved in an E-theory is written as $s \stackrel{?}{\doteq}_E t$.

Definition 3.1.1 (E-Unification Problem). *Let E be an equational theory. A finite system of equations $\langle s_1 \stackrel{?}{\doteq}_E t_1, \ldots, s_n \stackrel{?}{\doteq}_E t_n\rangle$ is called an E-unification problem.*

Definition 3.1.2. *Let E be an equational theory and $\mathfrak{U}$ be an E-unification problem. Then $\mathfrak{U}$ is*

– *an elementary E-unification problem, iff $sig(\mathfrak{U}) \subseteq sig(E)$;*
– *an E-unification problem with constants, iff $sig(\mathfrak{U}) \setminus sig(E) \subseteq \mathcal{CONST}$;*

– *a general E-unification problem, iff* $sig(\mathfrak{U}) \setminus sig(E) \subseteq \mathcal{CONST} \cup \mathcal{FFIX} \cup \mathcal{FFLEX}$.

Definition 3.1.3 (E-Unifier). *A substitution θ is called an E-unifier (or E-solution) of an E-unification problem $\langle s_1 \stackrel{?}{\doteq}_E t_1, \ldots, s_n \stackrel{?}{\doteq}_E t_n \rangle$ iff $s_i\theta \doteq_E t_i\theta$ for all $1 \leq i \leq n$. An E-unification problem $\mathfrak{U}$ is called E-unifiable (or E-solvable) iff there exists an E-unifier of $\mathfrak{U}$.*

Definition 3.1.4 (More General Substitution). *A substitution θ is more general than a substitution σ on a finite set of variables V modulo a theory E (denoted $\theta \ll_E^V \sigma$) iff there exists a substitution λ such that*

– *for all $\overline{x} \in V$,*
  – *$\overline{x} \leftarrow\ \notin \lambda$;*
  – *there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma = t_1, \ldots, t_n$, $\overline{x}\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$;*
– *for all $x \in V$, $x\sigma \doteq_E x\theta \circ \lambda$.*

Note that for the substitutions without sequence variables this definition coincides with the standard definition of more general substitution. We will write $\sigma \gg_E^V \theta$ iff $\theta \ll_E^V \sigma$.

Example 3.1.1. Let $\theta = \{\overline{x} \leftarrow \overline{y}\}$. Then $\theta \ll_\emptyset^{\{\overline{x},\overline{y}\}} \sigma$, where $\sigma = \{\overline{x} \leftarrow a, b,\ \overline{y} \leftarrow a, b\}$, but not $\theta \ll_\emptyset^{\{\overline{x},\overline{y}\}} \eta$, where $\eta = \{\overline{x} \leftarrow, \overline{y} \leftarrow\}$.

A substitution θ is *strictly more general* than a substitution σ on a finite set of variables V modulo a theory E (denoted $\theta <_E^V \sigma$) iff $\theta \ll_E^V \sigma$ and not $\sigma \ll_E^V \theta$.

Let $u_E(\mathfrak{U})$ be a set of all unifiers of $\mathfrak{U}$. Then the relation $\ll_E^V$ is a quasi-ordering (reflexive and transitive relation) on $u_E(\mathfrak{U})$.

Definition 3.1.5 (Minimal Set of Substitutions). *A set of substitutions Σ is called E-minimal with respect to a set of variables V iff for all $\theta, \sigma \in \Sigma$, $\theta \ll_E^V \sigma$ implies $\theta = \sigma$.*

Definition 3.1.6 (The Minimal Complete Set of Unifiers). *The minimal complete set of E-unifiers of $\mathfrak{U}$, denoted $mcu_E(\mathfrak{U})$, is an E-minimal set of substitutions with respect to the set of variables V of $\mathfrak{U}$, satisfying the following conditions:*

E-Correctness - for all $\theta \in mcu_E(\mathfrak{U})$, θ is an E-unifier of $\mathfrak{U}$.

E-Completeness - for any E-unifier σ of $\mathfrak{U}$ there exists $\theta \in mcu_E(\mathfrak{U})$ such that $\theta \ll_E^V \sigma$.

If an E-unification problem $\mathfrak{U}$ is solvable, there always exists a complete set of E-unifiers of $\mathfrak{U}$ (for instance, $u_E(\mathfrak{U})$ itself is complete), but the minimal complete set of E-unifiers need not exist in general (see [FH86, Baa86, SS86]). Bürckert et al. [BHSS90] show that if the quasi-ordering $\gg_E^V$ on $u_E(\mathfrak{U})$ is well-founded, then there exists a minimal subset of $u_E(\mathfrak{U})$.

Based on the cardinality of the minimal set of unifiers, Siekmann [Sie78] introduced a classification of unification problems and equational theories according to the unification hierarchy. We give the definition of unification hierarchy according to [BS01]:

Definition 3.1.7 (Unification Hierarchy). *Let E be an equational theory and $\mathfrak{U}$ be an E-unification problem.*

The problem $\mathfrak{U}$ is of type

- *unitary, if $mcu_E(\mathfrak{U})$ exists and has at most one element;*
- *finitary, if $mcu_E(\mathfrak{U})$ exists and is finite;*
- *infinitary, if $mcu_E(\mathfrak{U})$ exists and is infinite;*
- *nullary (or zero), if $mcu_E(\mathfrak{U})$ does not exist.*

We abbreviate type unitary by 1, type finitary by ω, type infinitary by ∞, and type nullary by 0, and order these types as follows: $1 < \omega < \infty < 0$.

The unification type of E is the maximal type of an E-unification problem.

Example 3.1.2. Compute the minimal complete set of unifiers in the free, flat and restricted flat theories (f and g are free flexible arity function symbols, h is a flat flexible arity function symbol, s - restricted flat flexible arity symbols):

1. $mcu_\emptyset(\langle f(\overline{x}, y) \stackrel{?}{=}_\emptyset f(a, b, c)\rangle) = \{\{\overline{x} \leftarrow a, b,\ y \leftarrow c\}\}$.
2. $mcu_\emptyset(\langle f(\overline{x}, \overline{y}) \stackrel{?}{=}_\emptyset f(a, b)\rangle) = \{\{\overline{x} \leftarrow,\ \overline{y} \leftarrow a, b\},\ \{\overline{x} \leftarrow a, \overline{y} \leftarrow b\},\ \{\overline{x} \leftarrow a, b, \overline{y} \leftarrow\}\}$.
3. $mcu_\emptyset(\langle f(\overline{x}, a) \stackrel{?}{=}_\emptyset f(a, \overline{x})\rangle) = \{\{\overline{x} \leftarrow\},\ \{\overline{x} \leftarrow a\},\ \{\overline{x} \leftarrow a, a\}, \ldots\}$.
4. $mcu_\emptyset(\langle f(g(a, g(\overline{y}, c)), \overline{x}) \stackrel{?}{=}_\emptyset f(\overline{u},\ g(b, \overline{v}))\rangle) = \{\{\overline{u} \leftarrow g(a, \overline{x}),\ \overline{y} \leftarrow b,\ \overline{v} \leftarrow c\},\ \{\overline{x} \leftarrow,\ \overline{u} \leftarrow g(a),\ \overline{y} \leftarrow b,\ \overline{v} \leftarrow c\},\ \{\overline{u} \leftarrow g(a, \overline{x}),\ \overline{y} \leftarrow b, \overline{y},\ \overline{v} \leftarrow \overline{y}, c\},\ \{\overline{x} \leftarrow,\ \overline{u} \leftarrow g(a),\ \overline{y} \leftarrow b, \overline{y},\ \overline{v} \leftarrow \overline{y}, c\}\}$.
5. $mcu_F(\langle x \stackrel{?}{=}_F h(x)\rangle) = \{\{x \leftarrow h(x)\}\}$.
6. $mcu_F(\langle h(\overline{x}) \stackrel{?}{=}_F h(a)\rangle) = \{\{\overline{x} \leftarrow a\}, \{\overline{x} \leftarrow h(a)\}, \{\overline{x} \leftarrow a, h()\}, \{\overline{x} \leftarrow h(a), h()\}, \{\overline{x} \leftarrow h(), a\},\ \{\overline{x} \leftarrow h(), h(a)\}, \{\overline{x} \leftarrow h(), a, h()\}, \{\overline{x} \leftarrow h(), h(a), h()\}, \ldots\}$.
7. $mcu_{RF}(\langle s(\overline{x}) \stackrel{?}{=}_{RF} s(a)\rangle) = \{\{\overline{x} \leftarrow a\},\ \{\overline{x} \leftarrow s(a)\}\}$.
8. $mcu_{RF}(\langle s(\overline{x}, y) \stackrel{?}{=}_{RF} s(a,\ b,\ c)\rangle) = \{\{\overline{x} \leftarrow,\ y \leftarrow s(a,\ b,\ c)\},\ \{\overline{x} \leftarrow a,\ y \leftarrow s(b,\ c)\}, \{\overline{x} \leftarrow s(a), y \leftarrow s(b,\ c)\},\ \{\overline{x} \leftarrow a, b,\ y \leftarrow c\},\ \{\overline{x} \leftarrow s(a),\ b,\ \ y \leftarrow c\},\ \{\overline{x} \leftarrow a, s(b),\ y \leftarrow c\},\ \{\overline{x} \leftarrow s(a), s(b),\ y \leftarrow c\},\ \{\overline{x} \leftarrow s(a, b),\ y \leftarrow c\},\ \{\overline{x} \leftarrow a, b,\ y \leftarrow s(c)\},\ \{\overline{x} \leftarrow s(a), b,\ y \leftarrow s(c)\},\ \{\overline{x} \leftarrow a, s(b),\ y \leftarrow s(c)\},\ \{\overline{x} \leftarrow s(a), s(b),\ y \leftarrow s(c)\},\ \{\overline{x} \leftarrow s(a, b),\ y \leftarrow s(c)\}\}$.

In the next subsections we will answer two questions for the free theory:

Decidability: Is it decidable whether a unification problem is solvable?
Unification procedure: How can we obtain a (preferably minimal) unification procedure?

Note that in the case of $\emptyset$-theory it is enough to consider single equations instead of systems of equations in unification problems, because the unification problem $\langle s_1 \overset{?}{=}_\emptyset t_1, \ldots, s_n \overset{?}{=}_\emptyset t_n \rangle$ has the same set of unifiers as $f(s_1, \ldots, s_n) \overset{?}{=}_\emptyset f(t_1, \ldots, t_n)$, where f is a free flexible arity symbol.

3.2 Decidability of Unification

In this section we consider decidability of general unification problem in the free theory with sequence variables and flexible arity symbols. The problem has a form $t_1 \overset{?}{=}_\emptyset t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, free constants and free fixed arity function symbols. We denote it as $\mathfrak{U}_\emptyset$.

The problem is to prove that $\mathfrak{U}_\emptyset$ is decidable. To do so, first we try to reduce $\mathfrak{U}_\emptyset$ to a "simpler" unification problem such that $\mathfrak{U}_\emptyset$ is solvable if and only if the reduced problem is solvable, and then try to prove that solvability of the reduced unification problem is decidable.

3.2.1 Reduction

Let $flex(\mathfrak{U}_\emptyset) = \{f_1, \ldots, f_n\}$ (We assume that $n \geq 1$. If $n = 0$, then $\mathfrak{U}_\emptyset$ is a Robinson unification problem [Rob65]). First we reduce $\mathfrak{U}_\emptyset$ to an intermediate F-unification problem $\mathfrak{U}_F^{int}$ of the form $s_1 \overset{?}{=}_F s_2$ by performing the following steps:

1. Introduce
   - a new flat flexible arity symbol *seq*,
   - a new unary function symbol g_i for each f_i $(1 \leq i \leq n)$.
2. Replace each term $f_i(r_1, \ldots, r_m)$ in $\mathfrak{U}_\emptyset$ by $g_i(seq(r_1, \ldots, r_m))$ $(m \geq 0)$.

Note that in $\mathfrak{U}_F^{int}$ sequence variables occur only as arguments of terms with the head *seq*. We impose *individual variable restrictions on terms* on $\mathfrak{U}_F^{int}$ demanding that, for a solution θ of $\mathfrak{U}_F^{int}$ and for any individual variable x, $x\theta$ must not have *seq* as a head.

Theorem 3.2.1. *$\mathfrak{U}_\emptyset$ is solvable iff $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms is solvable.*

Proof. ($\Rightarrow$) Let θ be a unifier of $\mathfrak{U}_\emptyset$. Then from θ we can get a unifier of $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms by repeating the step 2 of the reduction procedure above on each term with flexible arity head from $cod(\theta)$.

($\Leftarrow$) Let $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms have a solution ϕ. Show that $\mathfrak{U}_\emptyset$ is solvable.

Let θ be a substitution obtained from ϕ by

- replacing each unary function symbol g introduced at the first step of the reduction procedure above by the corresponding flexible arity symbol f,
- replacing each term of the form $seq(r_1, \ldots, r_m)$ $(m \geq 0)$ by the sequence of terms $r_1, \ldots, r_m$.

Then θ is a unifier of $\mathfrak{U}_\emptyset$. □

Remark. Note that solvability of $\mathfrak{U}_F^{int}$ (without individual variable restrictions on terms) does not imply solvability of $\mathfrak{U}_\emptyset$: Let $\mathfrak{U}_\emptyset$ be a unification problem $f(x) \overset{?}{\doteq}_\emptyset f(a, b)$, where f is a flexible arity function symbol. Then $\mathfrak{U}_F^{int}$ is $g(seq(x)) \overset{?}{\doteq}_F g(seq(a, b))$, where g is the unary function symbols introduced to replace f. It is clear that $\mathfrak{U}_\emptyset$ does not have a solution, while $\{x \leftarrow seq(a, b)\}$ is a solution of $\mathfrak{U}_F^{int}$ because the flatness of seq implies $seq(seq(a, b)) \doteq_F seq(a, b)$.

Next, our goal is to reduce $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms to a finite set S_F^{red} of reduced F-unification problems with the property that $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms is solvable iff there exists a $\mathfrak{U}_F^{red} \in S$ which is solvable.

First, we define two new sets C and T as follows:

- $C = const(\mathfrak{U}_F^{int}) \cup \{c\}$, where c is a new constant.
- $T = \{h_1(y_1^1, \ldots, y^1_{\mathcal{AR}(h_1)}), \ldots, h_m(y_1^m, \ldots, y^m_{\mathcal{AR}(h_m)})\}$ $(m \geq 1)$, where $\{h_1, \ldots, h_m\} = fix(\mathfrak{U}_F^{int})$ and $y_1^1, \ldots, y^1_{\mathcal{AR}(h_1)}, \ldots, y_1^m, \ldots, y^m_{\mathcal{AR}(h_m)}$ are new distinct individual variables.

Then S_F^{red} is defined as:

$$\begin{aligned} S_F^{red} = \{f(s_1, x_1, \ldots, x_n) \overset{?}{\doteq}_F f(s_2, r_1, \ldots, r_n) \mid & \\ & f \in \mathcal{FFIX},\ \mathcal{AR}(f) = n+1, \\ & \{x_1, \ldots, x_n\} = ivars(\mathfrak{U}_F^{int}) \text{ and} \\ & \text{for all } 1 \leq i \leq n,\ r_i \in C \cup T\}. \end{aligned}$$

Theorem 3.2.2. *$\mathfrak{U}_F^{int}$ with individual variable restrictions on terms is solvable iff there exists a $\mathfrak{U}_F^{red} \in S_F^{red}$ which is solvable.*

Proof. ($\Rightarrow$) Let θ be a unifier of $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms. Then for each $x \in ivars(\mathfrak{U}_F^{int})$, $x\theta$ does not have seq as a head, i.e. $x\theta$ is either a individual variable, a constant from $const(\mathfrak{U}_F^{int})$ or a term with the head $f \in fix(\mathfrak{U}_F^{int})$.

Let $\sigma = \{x \leftarrow c \mid x \in ivars(\mathfrak{U}_F^{int}) \cup (\mathcal{IV} \cap ran(\theta))\}$. Then $\theta \circ \sigma$ is a solution of $\mathfrak{U}_F^{int}$ such that for each $x \in ivars(\mathfrak{U}_F^{int})$, either $x\theta \circ \sigma \in C$ or there exists a substitution λ such that $x\theta \circ \sigma = t\lambda$ for some $t \in T$.

We define two sets X_C and X_T as follows:

- $X_C = \{x \mid x \in ivars(\mathfrak{U}_F^{int}) \text{ and } x\theta \circ \sigma \in C\}$.
- $X_T = \{x \mid x \in ivars(\mathfrak{U}_F^{int}) \text{ and } x\theta \circ \sigma = t\lambda \text{ for some } t \in T \text{ and } \lambda\}$.

Then $X_C \cup X_T = ivars(\mathfrak{U}_F^{int})$.

Let

- for each $x \in X_C$, c_x be the constant in C such that $c_x = x\theta \circ \sigma$;
- for each $x \in X_T$, t_x be a term in T such that $t_x\lambda = x\theta \circ \sigma$ for some substitution λ;
- for each $x \in X_T$, λ_x be a substitution such that
  - if $t_x = h(y_1, \dots, y_{\mathcal{AR}(f)})$, where h is a fixed arity function symbol, then $dom(\lambda_x) = \{y_1, \dots, y_{\mathcal{AR}(f)}\}$ and
  - $x\theta \circ \sigma = t_x\lambda_x$.

We choose $\mathfrak{U}_F^{red} \in S_F^{red}$ such that

$$\mathfrak{U}_F^{red} = f(s_1, x_1, \dots, x_i, x_{i+1}, \dots, x_n) \stackrel{?}{=}_F f(s_2, c_{x_1}, \dots, c_{x_i}, t_{x_{i+1}}, \dots, t_{x_n})$$

where $\{x_1, \dots, x_i\} = X_C$, $\{x_{i+1}, \dots, x_n\} = X_T$ and f is a fixed arity function symbol.

Let $\eta = \cup_{x \in X_T} \lambda_x$. Then

$$\begin{aligned} &f(s_1, x_1, \dots, x_i, x_{i+1}, \dots, x_n)\theta \circ \sigma \circ \eta = \\ &f(s_1\theta\sigma, c_{x_1}, \dots, c_{x_i}, x_{i+1}\theta\sigma, \dots, x_n\theta\sigma) \end{aligned}$$

and

$$\begin{aligned} &f(s_2, c_{x_1}, \dots, c_{x_i}, t_{x_{i+1}}, \dots, t_{x_n})\theta \circ \sigma \circ \eta = \\ &f(s_2\theta\sigma, c_{x_1}, \dots, c_{x_i}, t_{x_{i+1}}\lambda, \dots, t_{x_n}\lambda). \end{aligned}$$

Since $s_1\theta\sigma =_F s_2\theta\sigma$ and for all $i+1 \leq j \leq n$ we have $t_{x_j}\lambda = x_j\theta\sigma$, we get that $\theta \circ \sigma \circ \eta$ is a solution of $\mathfrak{U}_F^{red}$. Thus, solvability of $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms implies solvability of some $\mathfrak{U}_F^{red} \in S_F^{red}$.

($\Leftarrow$) Let θ be a solution of some $\mathfrak{U}_F^{red} \in S_F^{red}$. Then, obviously, θ is a solution of $\mathfrak{U}_F^{int}$. Moreover, θ binds every individual variable of $\mathfrak{U}_F^{int}$ either with a constant or a term with fixed arity head. Hence, θ is a solution of $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms. $\square$

Thus, the problem of decidability of the unification problem $\mathfrak{U}_\emptyset$ is reduced to the problem of decidability of a reduced F-unification problem $\mathfrak{U}_F^{red}$.

3.2.2 Decidability of the Reduced Problem

We prove decidability of the reduced F-unification problem $\mathfrak{U}_F^{red}$ using Baader-Schulz combination method [BS96]:

Theorem 3.2.3 (Combination Method, [BS96]).
Let $E_1, \dots, E_n$ be equational theories over disjoint signatures such that solvability of E_i-unification problems with linear constant restriction is decidable for $i = 1, \dots, n$. Then unifiability is decidable for the combined theory $E_1 \cup \dots \cup E_n$.

Linear constant restrictions are induced by a linear order $<$ on the set of variables and constants, demanding that, for a unifier θ, a constant c and a variable x, c must not occur in $x\theta$ if $c > x$.

Let $\Omega_1 = \{seq\}$ and $\Omega_2 = const(\mathfrak{U}_F^{red}) \cup fix(\mathfrak{U}_F^{red})$. Let E_1 be an equational theory over the signature Ω_1 where terms are built from individual variables, sequence variables and the symbol *seq* from Ω_1. Since *seq* is a flat symbol every non-variable term of E_1 can be written in a form $seq(t_1, \dots, t_n)$ $(n \geq 0)$ where each $t_i \in \mathcal{IV} \cup \mathcal{SV}$. Let E_2 be a free theory over Ω_2, where terms are built from individual variables and symbols from Ω_2. Then we can consider $\mathfrak{U}_F^{red}$ as a unification problem in the combined theory $E_1 \cup E_2$. Since $\Omega_1 \cap \Omega_2 = \emptyset$, by Theorem 3.2.3, in order to show that solvability of $\mathfrak{U}_F^{red}$ in $E_1 \cup E_2$ is decidable we need to show that solvability of E_1- and E_2-unification problems with linear constant restrictions is decidable.

Decidability of E_1-unification problem with linear constant restrictions

The decidability problem for E_1-unification is equivalent to the decidability problem for word equations with an additional restriction on certain variables in the equation. The restriction demands that for each of those variables and a unifier θ of the equation, the length of an instance of the variable with respect to θ must be 1. We call these additional restrictions *individual variable restrictions on length.* Thus, the decidability problem for E_1-unification with linear constant restrictions is equivalent to the decidability problem for word equations with linear constant restrictions and individual variable restrictions on length. Therefore, we have to show that solvability of word equations with linear constant restrictions and individual variable restrictions on length is decidable.

Note that we can consider ground unifiability case only. Let V be a set of (sequence or individual) variables, C be a set of constants and C' be the set $C \cup \{c\}$, where c is a constant not occurring in C. Suppose that a word equation w is given with variables in V, constants in C, with linear constant restrictions induced by a linear ordering $<$ on $V \cup C$ and for each individual variable from V, with individual variable restrictions on length. The equation

w has a unifier θ over $V \cup C$ which satisfies the linear constant restrictions induced by $<$ and the individual variable restrictions on length iff w has a ground unifier θ' over C' which satisfies the linear constant restrictions induced by $<$ and the individual variable restrictions on length.

We need the following general result of [Sch90]:

Theorem 3.2.4 ([Sch90]). *If w is a word equation with variables $v_1, \dots, v_n$ and constants in the alphabet C', and if $L_1, \dots, L_n$ are regular languages over C', then it is decidable whether w has a solution θ such that $v_i\theta \in L_i$ for $i = 1, \dots, n$.*

Using this result we can prove a theorem which solves the decidability problem for word equations with linear constant restrictions and individual variable restrictions on length and thus, solves the decidability problem for E_1-unification with linear constant restrictions:

Theorem 3.2.5. *Solvability of a word equation with linear constant restrictions and individual variable restrictions on length is decidable.*

Proof. Let w be a word equation with variables $x_1, \dots, x_m, \overline{y}_1, \dots, \overline{y}_k$ and constants in the alphabet C', where the variables $x_1, \dots, x_m$ are subjects of individual variable restrictions on length. Let $L_1, \dots, L_{m+k}$ be regular languages over C' such that

– for all $1 \leq i \leq m$, $L_i = \{a \in C \mid a < x_i\} \cup \{c\}$,
– for all $1 \leq i \leq k$, $L_{m+i} = (\{a \in C \mid a < \overline{y}_i\} \cup \{c\})^+$.

A solution θ of w satisfies the linear constant restrictions and the individual variable restrictions on length iff $\theta(x_i) \in L_i$ for all $1 \leq i \leq m$ and $\theta(\overline{y}_i) \in L_{m+i}$ for all $1 \leq i \leq k$. Thus, Theorem 3.2.4 implies that solvability of a word equation with linear constant restrictions and individual variable restrictions on length is decidable. □

Decidability of E_2-unification problem with linear constant restrictions

Obviously, the E_2-unification problem is a Robinson unification problem. Solvability of Robinson unification problems with linear constant restrictions is decidable (see [BS91]).

Thus, solvability of unification problem in the free theory with sequence variables and flexible arity symbols is decidable.

3.3 Unification Procedure

In this section we design a unification procedure to solve a general unification problem of the form $t_1 \overset{?}{\doteq}_{\emptyset} t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, free constants and free fixed arity function symbols. We denote it as $\mathfrak{U}_\emptyset$. The unification procedure is a tree generation process based on two basic steps: projection and transformation.

3.3.1 Projection

The idea of projection [AP90] is to eliminate some sequence variables from the given unification problem $\mathfrak{U}$. Let $\Pi(\mathfrak{U})$ be the following set of substitutions: $\{\{\overline{x} \leftarrow\ \mid \overline{x} \in S\} \mid S \subseteq svars(\mathfrak{U})\}$. $\Pi(\mathfrak{U})$ is called the set of projecting substitutions for $\mathfrak{U}$. Each $\pi \in \Pi$ replaces some sequence variables from $\mathfrak{U}$ with the empty sequence. The projection rule is shown in Figure 3.1.

Projection:	$s \overset{?}{=}_\emptyset t \rightsquigarrow \langle\langle s\pi_1 \overset{?}{=}_\emptyset t\pi_1,\, \pi_1\rangle, \ldots,$ $\langle s\pi_k \overset{?}{=}_\emptyset t\pi_k,\, \pi_k\rangle\rangle$	where $\{\pi_1, \ldots, \pi_k\} = \Pi(s \overset{?}{=}_\emptyset t)$.

Fig. 3.1. Projection rule.

3.3.2 Transformation

Each of the transformation rules for unification have one of the following forms: $\mathfrak{U} \rightsquigarrow \bot$ or $\mathfrak{U} \rightsquigarrow \langle\langle \mathfrak{S}_1, \sigma_1\rangle, \ldots, \langle \mathfrak{S}_n, \sigma_n\rangle\rangle$ where each of the successors $\mathfrak{S}_i$ is either $\top$ or a new unification problem.

The full set of transformation rules are given on Figure 3.2. It consists of four family of rules: Success, Failure, Elimination and Splitting. Note the usage of widening techniques (similar to [Len72], [Plo72], [Sie75], [Sie78]) in the elimination rules for sequence variables.

3.3.3 Tree Generation

In [Jaf90] and [Sch93] a tree generation construction is used for solving word equations. We use a similar idea for unification procedure with sequence variables and flexible arity symbols. Projection and transformation can be seen as single steps in a tree generation process. Each node of the tree is labeled either with a unification problem, $\top$ or $\bot$. The edges of the tree are labeled by substitutions. The nodes labeled with $\top$ or $\bot$ are terminal nodes. The nodes labeled with unification problems are non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Given a nonterminal node, let $\mathfrak{U}$ be a unification problem attached to it. First, we decide whether $\mathfrak{U}$ is unifiable. If the answer is negative, we replace $\mathfrak{U}$

Success:	$t \stackrel{?}{=}_\emptyset t \rightsquigarrow \langle\langle\top,\ \varepsilon\rangle\rangle.$	
	$x \stackrel{?}{=}_\emptyset t \rightsquigarrow \langle\langle\top,\ \{x \leftarrow t\}\rangle\rangle,$	if $x \notin ivars(t)$.
	$t \stackrel{?}{=}_\emptyset x \rightsquigarrow \langle\langle\top,\ \{x \leftarrow t\}\rangle\rangle,$	if $x \notin ivars(t)$.
Failure:	$c_1 \stackrel{?}{=}_\emptyset c_2 \rightsquigarrow \bot,$	if $c_1 \neq c_2$.
	$x \stackrel{?}{=}_\emptyset t \rightsquigarrow \bot,$	if $t \neq x$ and $x \in ivars(t)$.
	$t \stackrel{?}{=}_\emptyset x \rightsquigarrow \bot,$	if $t \neq x$ and $x \in ivars(t)$.
	$f_1(\tilde{t}) \stackrel{?}{=}_\emptyset f_2(\tilde{s}) \rightsquigarrow \bot,$	if $f_1 \neq f_2$.
	$f() \stackrel{?}{=}_\emptyset f(t_1, \tilde{t}) \rightsquigarrow \bot.$	
	$f(t_1, \tilde{t}) \stackrel{?}{=}_\emptyset f() \rightsquigarrow \bot.$	
	$f(\overline{x}, \tilde{t}) \stackrel{?}{=}_\emptyset f(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $s_1 \neq \overline{x}$ and $\overline{x} \in svars(s_1)$.
	$f(s_1, \tilde{s}) \stackrel{?}{=}_\emptyset f(\overline{x}, \tilde{t}) \rightsquigarrow \bot,$	if $s_1 \neq \overline{x}$ and $\overline{x} \in svars(s_1)$.
	$f(t_1, \tilde{t}) \stackrel{?}{=}_\emptyset f(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $t_1 \stackrel{?}{=}_\emptyset s_1 \rightsquigarrow \bot$.
Eliminate:	$f(t_1, \tilde{t}) \stackrel{?}{=}_\emptyset f(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}\sigma) \stackrel{?}{=}_\emptyset g(\tilde{s}\sigma),\ \sigma\rangle\rangle,$	if $t_1 \stackrel{?}{=}_\emptyset s_1 \rightsquigarrow \langle\langle\top,\ \sigma\rangle\rangle$.
	$f(\overline{x}, \tilde{t}) \stackrel{?}{=}_\emptyset f(\overline{x}, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}) \stackrel{?}{=}_\emptyset g(\tilde{s}),\ \varepsilon\rangle\rangle.$	
	$f(\overline{x}, \tilde{t}) \stackrel{?}{=}_\emptyset f(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}\sigma_1) \stackrel{?}{=}_\emptyset g(\tilde{s}\sigma_1),\ \sigma_1\rangle,$ $\langle g(\overline{x}, \tilde{t}\sigma_2) \stackrel{?}{=}_\emptyset g(\tilde{s}\sigma_2),\ \sigma_2\rangle\rangle,$	if $s_1 \notin \mathcal{SV}$ and $\overline{x} \notin svars(s_1)$, where $\sigma_1 = \{\overline{x} \leftarrow s_1\}$, $\sigma_2 = \{\overline{x} \leftarrow s_1, \overline{x}\}$.
	$f(s_1, \tilde{s}) \stackrel{?}{=}_\emptyset f(\overline{x}, \tilde{t}) \rightsquigarrow$ $\langle\langle g(\tilde{s}\sigma_1) \stackrel{?}{=}_\emptyset g(\tilde{t}\sigma_1),\ \sigma_1\rangle,$ $\langle g(\tilde{s}\sigma_2) \stackrel{?}{=}_\emptyset g(\overline{x}, \tilde{t}\sigma_2),\ \sigma_2\rangle\rangle,$	if $s_1 \notin \mathcal{SV}$ and $\overline{x} \notin svars(s_1)$, where $\sigma_1 = \{\overline{x} \leftarrow s_1\}$, $\sigma_2 = \{\overline{x} \leftarrow s_1, \overline{x}\}$.
	$f(\overline{x}, \tilde{t}) \stackrel{?}{=}_\emptyset f(\overline{y}, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}\sigma_1) \stackrel{?}{=}_\emptyset g(\tilde{s}\sigma_1),\ \sigma_1\rangle,$ $\langle g(\overline{x}, \tilde{t}\sigma_2) \stackrel{?}{=}_\emptyset g(\tilde{s}\sigma_2),\ \sigma_2\rangle,$ $\langle g(\tilde{t}\sigma_3) \stackrel{?}{=}_\emptyset g(\overline{y}, \tilde{s}\sigma_3),\ \sigma_3\rangle\,\rangle,$	where $\sigma_1 = \{\overline{x} \leftarrow \overline{y}\}$, $\sigma_2 = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$, $\sigma_3 = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$.
Split:	$f(t_1, \tilde{t}) \stackrel{?}{=}_\emptyset f(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle f(r_1, \tilde{t}\sigma_1) \stackrel{?}{=}_\emptyset f(q_1, \tilde{s}\sigma_1),\ \sigma_1\rangle,$ $\ldots,$ $\langle f(r_k, \tilde{t}\sigma_k) \stackrel{?}{=}_\emptyset f(q_k, \tilde{s}\sigma_k),\ \sigma_k\rangle\rangle$	if $t_1, s_1 \notin \mathcal{IV} \cup \mathcal{SV}$ and $t_1 \stackrel{?}{=}_\emptyset s_1 \rightsquigarrow$ $\langle\langle r_1 \stackrel{?}{=}_\emptyset q_1,\ \sigma_1\rangle, \ldots$ $\langle r_k \stackrel{?}{=}_\emptyset q_k,\ \sigma_k\rangle\rangle$.

Fig. 3.2. Transformation rules. $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. $f, f_1, f_2 \in \mathcal{FFIX} \cup \mathcal{FFLEX}$. $g \in \mathcal{FFLEX}$ is a new symbol, if in the same rule $f \in \mathcal{FFIX}$. Otherwise $g = f$.

with the new label $\bot$. If $\mathfrak{U}$ is unifiable, we apply projection or transformation on $\mathfrak{U}$ and get $\langle\langle\mathfrak{S}_1, \sigma_1\rangle, \dots, \langle\mathfrak{S}_n, \sigma_n\rangle\rangle$. Then the node $\mathfrak{U}$ has n children, labeled respectively with $\mathfrak{S}_1, \dots, \mathfrak{S}_n$ and the edge to the $\mathfrak{S}_i$ node is labeled with σ_i $(1 \leq i \leq n)$. The set $\{\sigma_1, \dots, \sigma_n\}$ is denoted by $sub(\mathfrak{U})$.

We design the general unification procedure for $\mathfrak{U}_\emptyset$ as a breadth first (level by level) tree generation process. We label the root of the tree with $\mathfrak{U}_\emptyset$ (zero level). First level nodes (the children of the root) of the tree are obtained from the original problem by projection. Starting from the second level, we apply only a transformation step to a unification problem of each node, thus getting new successor nodes. The branch which ends with a node labeled by $\top$ is called a successful branch. The branch which ends with a node labeled by $\bot$ is a failed branch. For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node and attach the obtained substitution to the node together with the unification problem the node was labeled with. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N. Let $\varSigma(\mathfrak{U}_\emptyset)$ be the set of all substitutions associated with the $\top$ nodes. We call the tree a unification tree for $\mathfrak{U}_\emptyset$ and denote it $utree(\mathfrak{U}_\emptyset)$.

Example 3.3.1. Figure 3.3 shows successful branches of the unification tree for $\mathfrak{U}_\emptyset = f(x, b, \overline{y}, f(\overline{x})) \overset{?}{\doteq}_\emptyset f(a, \overline{x}, f(b, \overline{y}))$. $\varSigma(\mathfrak{U}_\emptyset) = \{\{x \leftarrow a, \overline{x} \leftarrow b, \overline{x}, \overline{y} \leftarrow \overline{x}\}, \{x \leftarrow a, \overline{x} \leftarrow b, \overline{y} \leftarrow\}\}$.

Our goal is to prove that $\varSigma(\mathfrak{U}_\emptyset)$ is a minimal complete set of free unifiers for $\mathfrak{U}_\emptyset$. In fact, we will prove a stronger statement: $\varSigma(\mathfrak{U}_\emptyset)$ is a disjoint complete set of free unifiers for $\mathfrak{U}_\emptyset$, where disjointness is defined as follows:

Definition 3.3.1 (Disjoint Set of Substitutions). *A set of substitutions $\varSigma$ is called disjoint modulo E with respect to a set of variables V iff for all θ, $\sigma \in \varSigma$, if there exist substitutions λ_1, λ_2 such that*

- *for all sequence variables $\overline{x} \in V$,*
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - *there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \dots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \dots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and*
- *for all individual variables $x \in V$,*
  - $x\theta \circ \lambda_1 \doteq_E x\sigma \circ \lambda_2$,

then $\theta = \sigma$.

First we need to establish some preliminary results.

Lemma 3.3.1. *If a set of substitutions $\varSigma$ is E-disjoint with respect to a set of variables V, then $\varSigma$ is E-minimal with respect to V.*

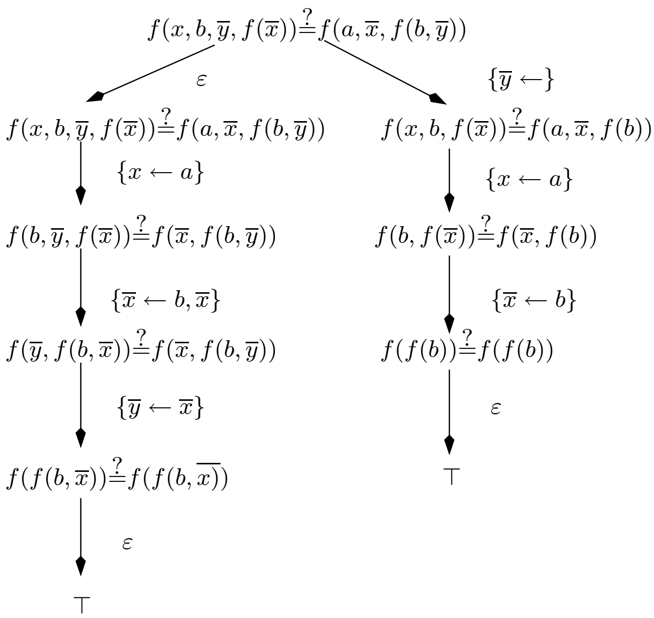

Fig. 3.3. Successful branches of $utree(f(x, b, \overline{y}, f(\overline{x})) \stackrel{?}{\doteq}_{\emptyset} f(a, \overline{x}, f(b, \overline{y})))$.

Proof. Let Σ be disjoint with respect to a set of variables V and θ and σ be two substitutions from Σ such that $\theta \ll_E^V \sigma$. We will show that $\theta = \sigma$.

By Definition 3.1.4, from $\theta \ll_E^V \sigma$ we have that there exists a substitution λ such that

- for all $\overline{x} \in V$,
  - $\overline{x} \leftarrow \; \notin \lambda$;
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma = t_1, \ldots, t_n$, $\overline{x}\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$;
- for all $x \in V$, $x\sigma \doteq_E x\theta \circ \lambda$.

On the other hand, for the empty substitution ε we have for all $\overline{x} \in V$,

- $\overline{x} \leftarrow \; \notin \varepsilon$;
- $\overline{x}\sigma = \overline{x}\sigma \circ \varepsilon$.

Therefore, for the substitutions ε and λ we have that

- for all sequence variables $\overline{x} \in V$,
  - $\overline{x} \leftarrow \notin \varepsilon$,
  - $\overline{x} \leftarrow \notin \lambda$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \varepsilon = t_1, \ldots, t_n$, $\overline{x}\sigma \circ \lambda = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and

– for all individual variables $x \in V$,
  – $x\theta \circ \varepsilon \doteq_E x\sigma \circ \lambda$,

which, by Definition 3.3.1, implies $\theta = \sigma$. □

First, we prove completeness.

Lemma 3.3.2. *For every free unifier ϕ of a general unification problem $\mathfrak{U}_\emptyset$ there exists a branch β in $utree(\mathfrak{U}_\emptyset)$ with the following property: if $\mathfrak{P}$ is a unification problem occurring in β with the associated substitution θ, then $\theta \ll_\emptyset^V \phi$, where V is the set of variables of $\mathfrak{U}_\emptyset$.*

Proof. Let ϕ be an arbitrary free unifier of $\mathfrak{U}_\emptyset$ and V be the set of variables of $\mathfrak{U}_\emptyset$. We should find a branch β in $utree(\mathfrak{U}_\emptyset)$ such that if $\mathfrak{P}$ is a unification problem occurring in β and θ is the substitution associated with $\mathfrak{P}$ in $utree(\mathfrak{U}_\emptyset)$, then $\theta \ll_\emptyset^V \phi$.

We define β recursively.

First, let the root of the tree, labeled with $\mathfrak{U}_\emptyset$, be in β. The substitution associated with $\mathfrak{U}_\emptyset$ is ε. Obviously, $\varepsilon \ll_\emptyset^V \phi$.

Next, let the first level node of β be that successor $\mathfrak{P}$ of $\mathfrak{U}_\emptyset$, for which the associated substitution $\theta = \{\overline{x} \leftarrow \ \mid \overline{x} \leftarrow \ \in \phi \text{ and } \overline{x} \in V\}$. We show that $\theta \ll_\emptyset^V \phi$. Let $\lambda = \phi \backslash \theta$. Then we have that $\phi = \theta \circ \lambda$. Therefore

– for all $\overline{x} \in V$,
  – $\overline{x} \leftarrow \ \notin \lambda$;
  – there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\,\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$;
– for all $x \in V$, $x\phi \doteq_\emptyset x\theta \circ \lambda$,

which implies that $\theta \ll_\emptyset^V \phi$.

Let now a node in $utree(\mathfrak{U}_\emptyset)$, labeled with $\mathfrak{P}_n$, be in β. It means that for the substitution θ_n associated with $\mathfrak{P}_n$, it is true that $\theta_n \ll_\emptyset^V \phi$. It implies that there exists a substitution λ such that

– for all $\overline{x} \in V$,
  – the binding

$$\overline{x} \leftarrow \ \notin \lambda; \tag{3.1}$$

  – there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or

$$t_i \doteq_\emptyset s_i; \tag{3.2}$$

– for all $x \in V$,

$$x\phi \doteq_\emptyset x\theta_n \circ \lambda. \tag{3.3}$$

If there exists a successor $\mathfrak{P}_{n+1}$ of $\mathfrak{P}_n$ in $utree(\mathfrak{U}_\emptyset)$ with the associated substitution θ_{n+1} such that $\theta_{n+1} \ll^V_\emptyset \phi$, we can include $\mathfrak{P}_{n+1}$ into β and, thus, we will have that for all $\mathfrak{P} \in \beta$, if θ is a substitution associated with $\mathfrak{P}$, then $\theta \ll^V_\emptyset \phi$.

Thus, the problem of constructing β is reduced to finding the successor $\mathfrak{P}_{n+1}$ of $\mathfrak{P}_n$ such that for the substitution θ_{n+1}, associated with $\mathfrak{P}_{n+1}$, we have $\theta_{n+1} \ll^V_\emptyset \phi$. We show how to find such a $\mathfrak{P}_{n+1}$.

The unification problem $\mathfrak{P}_n$ can have one of the following four forms:

1. $\mathfrak{P}_n$ is a pair of identical terms.
2. $\mathfrak{P}_n$ is a pair of individual variables.
3. $\mathfrak{P}_n$ is a pair of an individual variable and a non-variable term.
4. $\mathfrak{P}_n$ is a pair of non-variable terms.

We consider each of them separately:

1. There is only one possible choice: $\mathfrak{P}_{n+1}$ is $\top$ with the associated substitution $\theta_{n+1} = \theta_n \circ \varepsilon$. Then $\theta_{n+1} = \theta_n \ll^V_\emptyset \phi$.

2-3. Let $\mathfrak{P}_n$ be $x \overset{?}{\doteq}_\emptyset t$, t being either an individual variable or a non-variable term. The substitution $\theta_n \circ \lambda$ is a unifier of $\mathfrak{U}_\emptyset$. The unification problems $\mathfrak{U}_\emptyset\theta_n$ and $\mathfrak{P}_n$ have exactly the same set of unifiers, because $\mathfrak{P}_n$ can be obtained from $\mathfrak{U}_\emptyset\theta_n$ by iterated deletion of identical first arguments in both sides of $\mathfrak{U}_\emptyset\theta_n$. Therefore, we have that λ is a unifier of $\mathfrak{P}_n$. To obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ we choose the substitution $\mu = \{x \leftarrow t\}$. Let θ_{n+1} be the substitution $\theta_n \circ \mu$. Then from 3.1, 3.2, 3.3 and the fact that $\mu \circ \lambda = \lambda$ we get
   - for all $\overline{x} \in V$,
     - $\overline{x} \leftarrow \ \notin \lambda$;
     - there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \dots, t_n$, $\overline{x}\theta_n \circ \lambda = \overline{x}\theta_n \circ \mu \circ \lambda = \overline{x}\theta_{n+1} \circ \lambda = s_1, \dots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$;
   - for all $x \in V$, $x\phi \doteq_\emptyset x\theta_n \circ \lambda \doteq_\emptyset x\theta_n \circ \mu \circ \lambda \doteq_\emptyset x\theta_{n+1} \circ \lambda$.

   Thus, $\theta_{n+1} =\ll^V_\emptyset \phi$.

4. Let $\mathfrak{P}_n$ have the form $f(t_1, \tilde{t}) \overset{?}{\doteq}_\emptyset f(s_1, \tilde{s})$, where $\tilde{t}$ and $\tilde{s}$ are (possibly empty) sequences of terms. We can have the following 6 cases with respect to t_1 and s_1:
   - 4.1. t_1 and s_1 are identical.
   - 4.2. t_1 and s_1 are individual variables.
   - 4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable term.
   - 4.4. t_1 and s_1 are sequence variables.
   - 4.5. One of t_1 and s_1 is a sequence variable, the other is not.
   - 4.6. t_1 and s_1 are non-variable terms.

   We consider each of these cases.

4.1-4.3. In these cases $\mathfrak{P}_{n+1}$ can be chosen analogously to the cases 1-3 above.

4.4. Suppose t_1 is a sequence variable $\overline{x}$ and s_1 is a sequence variable $\overline{y}$. Let us define substitutions μ and ν as follows:
- $\mu = \{\overline{x} \leftarrow \overline{y}\}$, $\nu = \{v \leftarrow v\lambda \,|\, v \in V, v \neq \overline{x},\, v \neq v\lambda\}$, if $g(\overline{x})\lambda \doteq_{\emptyset} g(\overline{y})\lambda$;
- $\mu = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$, $\nu = \{\overline{x} \leftarrow \overline{r}\} \circ \{v \leftarrow v\lambda \,|\, v \in V, v \neq \overline{x}, v \neq v\lambda\}$, if there exists a non-empty sequence of terms $\overline{r}$ such that $g(\overline{x})\lambda \doteq_{\emptyset} g(\overline{y}, \overline{r})\lambda$;
- $\mu = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$, $\nu = \{\overline{y} \leftarrow \overline{r}\} \circ \{v \leftarrow v\lambda \,|\, v \in V, v \neq \overline{y}, v \neq v\lambda\}$, if there exists a non-empty sequence of terms $\overline{r}$ such that $g(\overline{y})\lambda \doteq_{\emptyset} g(\overline{x}, \overline{r})\lambda$,

with g being a new flexible arity function symbol, if f has a fixed arity. Otherwise g is f.

We obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ by the substitution μ (since λ is a unifier of $\mathfrak{P}_n$, these three cases for μ are the only possibilities to get $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$). Therefore, $\theta_{n+1} = \theta_n \circ \mu$.

On the other hand, for all $\overline{x} \in V$,

$$\overline{x} \leftarrow\ \notin \mu. \tag{3.4}$$

From 3.4, 3.1 and definitions of μ and ν, by Definition 2.1.11 we get for all $\overline{x} \in V$

$$\overline{x} \leftarrow\ \notin \nu. \tag{3.5}$$

From 3.2 and definitions of μ and ν we get that for all $\overline{x} \in V$ there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \lambda = \overline{x}\theta_n \circ \mu \circ \nu = \overline{x}\theta_{n+1} \circ \nu = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or

$$t_i \doteq_{\emptyset} s_i; \tag{3.6}$$

From 3.3 and definitions of μ and ν we get that for all $x \in V$

$$x\phi \doteq_{\emptyset} x\theta_n \circ \lambda \doteq_{\emptyset} x\theta_n \circ \mu \circ \nu \doteq_{\emptyset} x\theta_{n+1} \circ \nu. \tag{3.7}$$

From 3.5, 3.6 and 3.7, by Definition 3.1.4 we get $\theta_{n+1} \ll_{\emptyset}^{V} \phi$.

4.5. This case can be proved similarly to the case 4.4, considering only two cases for μ.

4.6. This case recursively can be reduced to one of the cases 4.1-4.4 above.

Thus, for all possible forms of $\mathfrak{P}_n$ we found its successor $\mathfrak{P}_{n+1}$ such that for the substitution θ_{n+1}, associated with $\mathfrak{P}_{n+1}$, we have $\theta_{n+1} \ll_{\emptyset}^{V} \phi$. It finishes the proof. □

Theorem 3.3.1. *$\Sigma(\mathfrak{U}_{\emptyset})$ is a complete set of free unifiers for $\mathfrak{U}_{\emptyset}$.*

Proof. The theorem follows from Lemma 3.3.2 by the definition of Σ. □

Next, we want to show that $\Sigma(\mathfrak{U}_\emptyset)$ is a disjoint set of free unifiers for $\mathfrak{U}_\emptyset$. First, we prove that for any $\emptyset$-unification problem $\mathfrak{P}$ the set $sub(\mathfrak{P})$ is disjoint with respect to $vars(\mathfrak{P})$. After that we show that the substitutions associated with distinct successful leaves in $utree(\mathfrak{U}_\emptyset)$ are disjoint with respect to $vars(\mathfrak{U}_\emptyset)$.

Lemma 3.3.3. *Let $\mathfrak{P}$ be a general $\emptyset$-unification problem. Then $sub(\mathfrak{P})$ is disjoint with respect to $vars(\mathfrak{P})$.*

Proof. Let $sub(\mathfrak{P})$ be the set of projecting substitutions. Assume by contradiction that $sub(\mathfrak{P})$ is not disjoint with respect to $vars(\mathfrak{P})$. Then there exist two substitutions θ and σ from $sub(\mathfrak{P})$ and two substitutions λ_1 and λ_2 such that

- for all sequence variables $\overline{x} \in vars(\mathfrak{P})$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \dots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \dots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
- for all individual variables $x \in vars(\mathfrak{P})$,
  - $x\theta \circ \lambda_1 \doteq_\emptyset x\sigma \circ \lambda_2$.

Since $\theta \neq \sigma$, without loss of generality we can assume that there exists a sequence variable $\overline{x} \in vars(\mathfrak{P})$ such that $\overline{x} \leftarrow \in \theta$ and $\overline{x} \leftarrow \notin \sigma$. Then $\overline{x}\theta \circ \lambda_1$ is the empty sequence, while $\overline{x}\sigma = \overline{x}$. The only way to make $\overline{x}\sigma \circ \lambda_2$ the empty sequence is to have $\overline{x} \leftarrow \in \lambda_2$, but it contradicts the assumption that $\overline{x} \leftarrow \notin \lambda_2$. The obtained contradiction proves that the set $sub(\mathfrak{P})$ of projecting substitutions of $\mathfrak{P}$ is disjoint with respect to $vars(\mathfrak{P})$.

Now let $sub(\mathfrak{P})$ be the set of transformation substitutions. Since, by Definition 3.3.1, the empty set and a singleton are trivially disjoint modulo any E with respect to any set of variables, we consider only the cases when $sub(\mathfrak{P})$ is neither empty set nor a singleton. Let $\mathfrak{P}$ have a form $f(r_1, \tilde{r}) \stackrel{?}{\doteq}_\emptyset f(s_1, \tilde{s})$ where $\tilde{r}$, and $\tilde{s}$ are possibly empty sequences of terms and $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$. The we have the following three cases with respect to the form of r_1 and s_1:

1. Both r_1 and s_1 are sequence variables.
2. Only one of r_1 and s_1 is a sequence variable.
3. Both r_1 and s_1 are non-variable terms.

We prove in each case that $sub(\mathfrak{P})$ is disjoint with respect to $vars(\mathfrak{P})$.

1. Let r_1 be $\overline{x}$ and s_1 be $\overline{y}$. Then $sub(\mathfrak{P}) = \{\eta_1, \eta_2, \eta_3\}$, where $\eta_1 = \{\overline{x} \leftarrow \overline{y}\}$, $\eta_2 = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$ and $\eta_3 = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$. Assume by contradiction that $sub(\mathfrak{P})$ is not disjoint. Then for some θ and σ from $sub(\mathfrak{P})$ there exist substitutions λ_1, λ_2 such that
   - for all sequence variables $\overline{x} \in vars(\mathfrak{P})$,
     - $\overline{x} \leftarrow \notin \lambda_1$,

  – $\overline{x} \leftarrow \notin \lambda_2$,
  – there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and
– for all individual variables $x \in vars(\mathfrak{P})$,
  – $x\theta \circ \lambda_1 \doteq_E x\sigma \circ \lambda_2$.

Let θ be η_1 and σ be η_2. Then
– for $\overline{x} \in V$ we have $\overline{x}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{x}\sigma \circ \lambda_2 = \overline{y}\lambda_2, \overline{x}\lambda_2$;
– for $\overline{y} \in V$ we have $\overline{y}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{y}\sigma \circ \lambda_2 = \overline{y}\lambda_2$,
which implies that $\overline{x}\lambda_2$ must be the empty sequence, but it is impossible because $\overline{x} \leftarrow \notin \lambda_2$.

Now let θ be η_1 and σ be η_3. Then
– for $\overline{x} \in V$ we have $\overline{x}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{x}\sigma \circ \lambda_2 = \overline{x}\lambda_2$;
– for $\overline{y} \in V$ we have $\overline{y}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{y}\sigma \circ \lambda_2 = \overline{y}\lambda_2, \overline{x}\lambda_2$,
which implies that $\overline{y}\lambda_2$ must be the empty sequence, but it is impossible because $\overline{y} \leftarrow \notin \lambda_2$.

Now let θ be η_2 and σ be η_3. Then
– for $\overline{x} \in V$ we have $\overline{x}\theta \circ \lambda_1 = \overline{y}\lambda_1, \overline{x}\lambda_1$ and $\overline{x}\sigma \circ \lambda_2 = \overline{x}\lambda_2$;
– for $\overline{y} \in V$ we have $\overline{y}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{y}\sigma \circ \lambda_2 = \overline{y}\lambda_2, \overline{x}\lambda_2$,
which implies that $\overline{x}\lambda_2$ and $\overline{y}\lambda_2$ must be the empty sequences, but it is impossible because $\overline{x} \leftarrow \notin \lambda_2$ and $\overline{y} \leftarrow \notin \lambda_2$.

Thus, in all possible cases we got a contradiction, which proves that $sub(\mathfrak{P})$ is disjoint with respect to $vars(\mathfrak{P})$.

2. This case can be proved similarly to the case 1.
3. This case can be reduced recursively to one of the previous cases.

□

Now we show that the substitutions associated with distinct successful leaves in $utree(\mathfrak{U}_\emptyset)$ are disjoint with respect to $vars(\mathfrak{U}_\emptyset)$. For this, first we introduce the notion of D-preserving (disjointness-preserving) substitution due to Schulz [Sch93].

Definition 3.3.2 (*D*-preserving substitution). *A substitution θ is D-preserving modulo E with respect to a set V of variables iff for any two substitutions σ_1 and σ_2, if the set $\{\sigma_1, \sigma_2\}$ is disjoint modulo E with respect to the set of variables $\cup_{v\in V} vars(v\theta)$, then the set of substitutions $\{\theta\circ\sigma_1, \theta\circ\sigma_2\}$ is disjoint modulo E with respect to V.*

Lemma 3.3.4. *For a general $\emptyset$-unification problem $\mathfrak{P}$ any of the projecting or transformation substitutions from $sub(\mathfrak{P})$ is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{P})$.*

Proof. First we show that any projecting substitution preserves disjointness modulo $\emptyset$-theory with respect to $vars(\mathfrak{P})$. Let π be a projecting substitution for $\mathfrak{P}$. If $\pi = \varepsilon$, then, by Definition 3.3.2, π is trivially D-preserving modulo $\emptyset$-theory. Assume $\pi \neq \varepsilon$. Let σ_1 and σ_2 be two substitutions such that the set $\{\sigma_1, \sigma_2\}$ is disjoint modulo E with respect to the set of variables $\cup_{v \in vars(\mathfrak{P})} vars(v\theta) = vars(\mathfrak{P}) \backslash \{\overline{x} \,|\, \overline{x} \leftarrow \in \pi\}$. We have to show that $\{\pi \circ \sigma_1, \, \pi \circ \sigma_2\}$ is disjoint modulo $\emptyset$-theory with respect to $vars(\mathfrak{P})$. To prove this, by Definition 3.3.1, we assume that there exist substitutions λ_1, λ_2 such that

- for all sequence variables $\overline{x} \in vars(\mathfrak{P})$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\pi \circ \sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\pi \circ \sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_{\emptyset} s_i$ and
- for all individual variables $x \in vars(\mathfrak{P})$,
  - $x\pi \circ \sigma_1 \circ \lambda_1 \doteq_{\emptyset} x\pi \circ \sigma \circ \lambda_2$,

and prove $\pi \circ \sigma_1 = \pi \circ \sigma_2$.

Since $vars(\mathfrak{P}) \backslash \{\overline{x} \,|\, \overline{x} \leftarrow \in \pi\} \subset vars(\mathfrak{P})$, we have for all (sequence or individual) variables $v \in vars(\mathfrak{P}) \backslash \{\overline{x} \,|\, \overline{x} \leftarrow \in \pi\}$ that $v\pi = v$. Then from the assumption we have that

- for all sequence variables $\overline{x} \in vars(\mathfrak{P}) \backslash \{\overline{x} \,|\, \overline{x} \leftarrow \in \pi\}$
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_{\emptyset} s_i$,
- for all individual variables $x \in vars(\mathfrak{P})$,
  - $x\sigma_1 \circ \lambda_1 \doteq_{\emptyset} x\sigma \circ \lambda_2$,

which, by disjointness of the set $\{\sigma_1, \sigma_2\}$, implies that $\sigma_1 = \sigma_2$. Therefore, $\pi \circ \sigma_1 = \pi \circ \sigma_2$.

Now we prove the lemma for the transformation substitutions. There are the following 4 possible forms of the unification problem $\mathfrak{P}$ the transformation step of the unification procedure deals with:

1. $\mathfrak{P}$ is a pair of identical terms.
2. $\mathfrak{P}$ is a pair of individual variables.
3. $\mathfrak{P}$ is a pair of an individual variable and non-variable term.
4. $\mathfrak{P}$ is a pair of non-variable terms.

We will prove in each case that transformation substitution is a D-preserving substitution with respect to $vars(\mathfrak{P})$.

1. In this case the transformation substitution is ε. The fact that it preserves disjointness, directly follows from the definition of D-preserving substitution.

2. Let $\mathfrak{P}$ have a form $x \stackrel{?}{\doteq}_{\emptyset} y$ and the corresponding transformation substitution be $\theta = \{x \leftarrow y\}$. Then $vars(\mathfrak{P}) = \{x, y\}$ and $\cup_{v \in vars(\mathfrak{P})} vars(v\theta) = \{y\}$. Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to $\{y\}$. We have to show that $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $\{x, y\}$. Assume by contradiction that it is not disjoint. Then there exist substitutions λ_1 and λ_2 such that for all $v \in \{x, y\}$, $v\theta \circ \sigma_1 \circ \lambda_1 \doteq_{\emptyset} v\theta \circ \sigma_2 \circ \lambda_2$, in particular, $y\theta \circ \sigma_1 \circ \lambda_1 \doteq_{\emptyset} y\theta \circ \sigma_2 \circ \lambda_2$. But since $y\theta = y$, we get $y\sigma_1 \circ \lambda_1 \doteq_{\emptyset} y\sigma_2 \circ \lambda_2$, which contradicts to disjointness of $\{\sigma_1, \sigma_2\}$ with respect to $\{y\}$. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$, which implies that θ preserves disjointness on the set of variables of $\mathfrak{U}_{\emptyset}$.
3. Let $\mathfrak{P}$ have a form $x \stackrel{?}{\doteq}_{\emptyset} t$ and the corresponding transformation substitution be $\theta = \{x \leftarrow t\}$. Then we have that $vars(\mathfrak{P}) = \{x\} \cup vars(t)$ and $\cup_{v \in vars(\mathfrak{P})} vars(v\theta) = vars(t)$. Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$. We have to show that $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$. Assume by contradiction that it is not disjoint. Then there exist substitutions λ_1 and λ_2 such that
   – for all sequence variables $\overline{x} \in vars(\mathfrak{P})$,
     – $\overline{x} \leftarrow \notin \lambda_1$,
     – $\overline{x} \leftarrow \notin \lambda_2$,
     – there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\theta \circ \sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_{\emptyset} s_i$ and
   – for all individual variables $x \in vars(\mathfrak{P})$,
     – $x\theta \circ \sigma_1 \circ \lambda_1 \doteq_{\emptyset} x\theta \circ \sigma_2 \circ \lambda_2$.

   Then we have
   – for all sequence variables $\overline{x} \in vars(t)$,
     – $\overline{x} \leftarrow \notin \lambda_1$,
     – $\overline{x} \leftarrow \notin \lambda_2$,
     – there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, $t_i \doteq_{\emptyset} s_i$ and
   – for all individual variables $x \in vars(t)$,
     – $x\sigma_1 \circ \lambda_1 \doteq_{\emptyset} x\sigma_2 \circ \lambda_2$.

   It implies that $\{\sigma_1, \sigma_2\}$ is not disjoint with respect to $vars(t)$, which is a contradiction. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect $vars(\mathfrak{P})$, which implies that θ preserves disjointness on $vars(\mathfrak{P})$.
4. Let $\mathfrak{U}_{\emptyset}$ have the form $f(t_1, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(s_1, \tilde{s})$, where $\tilde{t}$ and $\tilde{s}$ are (possibly empty) sequences of terms. We have the following 6 cases with respect to t_1 and s_1:
   4.1. t_1 and s_1 are identical.
   4.2. Both t_1 and s_1 are individual variables.
   4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable term.

4.4. Both t_1 and s_1 are sequence variables.
4.5. One of t_1 and s_1 is a sequence variable, the other is not.
4.6. Both t_1 and s_1 are non-variable terms.

We show in each subcase separately that the corresponding transformation substitution is D-preserving with respect to the set of variables of $\mathfrak{U}$.

4.1-4.3. These cases can be proved analogously to the cases 1-3 above.

4.4 Let $\mathfrak{U}_\emptyset$ have a form $\overline{x}\overset{?}{\doteq}_\emptyset\overline{y}$. Then we have the following cases for the transformation substitution θ:

a) $\theta = \{\overline{x} \leftarrow \overline{y}\}$;
b) $\theta = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$;
c) $\theta = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$.

Then $vars(\mathfrak{P}) = \{\overline{x}, \overline{y}\} \cup vars(\overline{t}) \cup vars(\overline{s})$. Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to $\cup_{v \in vars(\mathfrak{P})} vars(v\theta)$. We have to show that $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$. Assume by contradiction that it is not disjoint. Then there exist substitutions λ_1 and λ_2 such that

– for all sequence variables $\overline{x} \in vars(\mathfrak{P})$,
  – $\overline{x} \leftarrow \notin \lambda_1$,
  – $\overline{x} \leftarrow \notin \lambda_2$,
  – there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \sigma_1 \circ \lambda_1 = t_1, \dots, t_n$, $\overline{x}\theta \circ \sigma_2 \circ \lambda_2 = s_1, \dots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
– for all individual variables $x \in vars(\mathfrak{P})$,
  – $x\theta \circ \sigma_1 \circ \lambda_1 \doteq_\emptyset x\theta \circ \sigma_2 \circ \lambda_2$.

Let us consider each case separately:

a) In this case $\cup_{v \in vars(\mathfrak{P})} vars(v\theta) = vars(\mathfrak{P}) \setminus \{\overline{x}\}$ and for all $\overline{z} \in vars(\mathfrak{P}) \setminus \{\overline{x}\}$ we have $\overline{z}\theta = \overline{z}$. Then for λ_1 and λ_2 we have
  – for all sequence variables $\overline{z} \in vars(\mathfrak{P}) \setminus \{\overline{x}\}$,
    – $\overline{z} \leftarrow \notin \lambda_1$,
    – $\overline{z} \leftarrow \notin \lambda_2$,
    – there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{z}\sigma_1 \circ \lambda_1 = t_1, \dots, t_n$, $\overline{z}\sigma_2 \circ \lambda_2 = s_1, \dots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
  – for all individual variables $x \in vars(\mathfrak{P})$,
    – $x\sigma_1 \circ \lambda_1 \doteq_\emptyset x\sigma_2 \circ \lambda_2$,

  which implies that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$. But this is a contradiction. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$, which implies that θ preserves disjointness on the set of variables of $\mathfrak{U}_\emptyset$.

b) In this case $\cup_{v \in vars(\mathfrak{P})} vars(v\theta) = vars(\mathfrak{P})$. For $\overline{x} \in vars(\mathfrak{P})$ there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \sigma_1 \circ \lambda_1 = (\overline{y}, \overline{x})\sigma_1 \circ \lambda_1 = t_1, \dots, t_n$, $\overline{x}\theta \circ \sigma_2 \circ \lambda_2 = (\overline{y}, \overline{x})\sigma_2 \circ \lambda_2 =$

$s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$. On the other hand for $\overline{y} \in vars(\mathfrak{P})$, we have that for some $m \leq n$, $\overline{y}\theta \circ \sigma_1 \circ \lambda_1 = \overline{y}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_m$, $\overline{y}\theta \circ \sigma_2 \circ \lambda_2 = \overline{y}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_m$. Therefore, $\overline{x}\sigma_1 \circ \lambda_1 = t_m, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_m, \ldots, s_n$. As for the individual variables, for all $x \in vars(\mathfrak{P})$ we have $x\theta = x$. Thus, we got that there exist substitutions λ_1 and λ_2 such that

- for all sequence variables $\overline{x} \in vars(\mathfrak{P})$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
- for all individual variables $x \in vars(\mathfrak{P})$,
  - $x\sigma_1 \circ \lambda_1 \doteq_\emptyset x\sigma_2 \circ \lambda_2$,

which implies that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$. But this is a contradiction. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $vars(\mathfrak{P})$, which implies that θ preserves disjointness on $vars(\mathfrak{P})$.

c) Proof in this case is symmetric to the case b).

4.5. This case can be proved similarly to the case 4.4.

4.6. This case recursively can be reduced to one of the cases 4.1-4.4 above.

□

Lemma 3.3.5. *Let θ_1 and θ_2 be two substitutions such that $\{\theta_1, \theta_2\}$ is disjoint modulo E with respect to a set of variables V. Let σ_1 and σ_2 be two substitutions such that for all $\overline{x} \in V$, $\overline{x} \leftarrow \notin \sigma_1$ and $\overline{x} \leftarrow \notin \sigma_2$. Then $\{\theta_1 \circ \sigma_1, \theta_2 \circ \sigma_2\}$ is disjoint modulo E with respect to V.*

Proof. Assume by contradiction that $\{\theta_1 \circ \sigma_1, \theta_2 \circ \sigma_2\}$ is not disjoint modulo E with respect to V. Then there exist substitutions λ_1, λ_2 such that

- for all sequence variables $\overline{x} \in V$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta_1 \circ \sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\theta_2 \circ \sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and
- for all individual variables $x \in V$,
  - $x\theta_1 \circ \sigma_1 \circ \lambda_1 \doteq_E x\theta_2 \circ \sigma_2 \circ \lambda_2$,

Let μ_1 be $\sigma_1 \circ \lambda_1$ and μ_2 be $\sigma_2 \circ \lambda_2$. Then we have

- for all sequence variables $\overline{x} \in V$,
  - $\overline{x} \leftarrow \notin \mu_1$,

  - $\overline{x} \leftarrow \notin \mu_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta_1 \circ \mu_1 = t_1, \ldots, t_n$, $\overline{x}\theta_2 \circ \mu_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and
- for all individual variables $x \in V$,
  - $x\theta_1 \circ \mu_1 \doteq_E x\theta_2 \circ \mu_2$,

which, by Definition 3.3.1 implies that $\{\theta_1, \theta_2\}$ is not disjoint modulo E with respect to V. This is a contradiction which proves that $\{\theta_1 \circ \sigma_1, \theta_2 \circ \sigma_2\}$ is disjoint modulo E with respect to V. □

Theorem 3.3.2. *$\Sigma(\mathfrak{U}_\emptyset)$ is a disjoint set of $\emptyset$-unifiers for $\mathfrak{U}_\emptyset$ with respect to $vars(\mathfrak{U}_\emptyset)$.*

Proof. Suppose σ_1 and σ_2 are two substitutions associated with distinct successful leaves of $utree(\mathfrak{U}_\emptyset)$. We prove that σ_1 and σ_2 are disjoint with respect to $vars(\mathfrak{U}_\emptyset)$.

Let ρ be the common part (edges) of branches leading to the respective leaves σ_1 and σ_2 are associated with. Then the substitutions σ_1 and σ_2 can be represented respectively as $\gamma \circ \theta_1 \circ \cdots \circ \theta_k$ and $\gamma \circ \lambda_1 \circ \cdots \circ \lambda_n$, where γ is the substitution associated with the last node in ρ.

First, we show that γ is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$.

Let $\mathfrak{P}_n$ denote the unification problem at the n-th node of ρ and γ_n be the substitution associated with $\mathfrak{P}_n$. We show that for any n, the substitution γ_n in ρ is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$. We use induction on n.

Induction base. $n = 0$. Then $\gamma_n = \varepsilon$ which, by Definition 3.3.2, is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$.

Induction hypothesis. We assume that γ_n is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$ for $n > 0$.

Induction step. We show that γ_{n+1} is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$. By Definition 3.3.2, it suffices to show that for any two substitutions η_1 and η_2, if the set $\{\eta_1, \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables $\cup_{v \in vars(\mathfrak{U}_\emptyset)} vars(v\gamma_{n+1})$, then the set of substitutions $\{\gamma_{n+1} \circ \eta_1, \gamma_{n+1} \circ \eta_2\}$ is disjoint modulo $\emptyset$ with respect to $vars(\mathfrak{U}_\emptyset)$.

Let η_1 and η_2 be two substitutions such that the set $\{\eta_1, \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables $\cup_{v \in vars(\mathfrak{U}_\emptyset)} vars(v\gamma_{n+1})$. We have to show that $\{\gamma_{n+1} \circ \eta_1, \gamma_{n+1} \circ \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$.

Let μ be a substitution such that $\gamma_{n+1} = \gamma_n \circ \mu$. Since, By Lemma 3.3.4 μ is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{P}_n)$, $\{\mu \circ \eta_1, \mu \circ \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables $vars(\mathfrak{P}_n) = \cup_{v \in vars(\mathfrak{U}_\emptyset)} vars(v\gamma_n)$. By induction hypothesis, γ_n is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$. Then, by Definition 3.3.2, $\{\gamma_n \circ \mu \circ \eta_1, \gamma_n \circ$

$\mu \circ \eta_2\} = \{\gamma_{n+1} \circ \eta_1, \gamma_{n+1} \circ \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$.

Thus, any substitution γ_n in ρ is D-preserving modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$. In particular, γ preserves disjointness modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$.

Now, we show that the set $\{\theta_1 \circ \cdots \circ \theta_k, \lambda_1 \circ \cdots \circ \lambda_n\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables V_ρ of the last node of ρ.

By Lemma 3.3.3, the set $\{\theta_1, \lambda_1\}$ is disjoint modulo $\emptyset$-theory with respect to V_ρ. The substitutions $\theta_2, \ldots, \theta_k, \lambda_2, \ldots, \lambda_n$ are transformations, therefore, for all $\overline{x}$ from V_ρ, $\overline{x} \leftarrow \notin \theta_2 \circ \cdots \circ \theta_k$ and $\overline{x} \leftarrow \notin \lambda_2 \circ \cdots \circ \lambda_n$. Then, by Lemma 3.3.5, the set $\{\theta_1 \circ \cdots \circ \theta_k, \lambda_1 \circ \cdots \circ \lambda_n\}$ is disjoint modulo $\emptyset$-theory with respect to V_ρ.

Note that $V_\rho = \cup_{v \in vars(\mathfrak{U}_\emptyset)} vars(v\gamma)$. Therefore, from the fact that γ is D-preserving with respect to $vars(\mathfrak{U}_\emptyset)$ and $\{\theta_1 \circ \cdots \circ \theta_k, \lambda_1 \circ \cdots \circ \lambda_n\}$ is disjoint with respect to V_ρ, by Definition 3.3.2 we get that $\{\gamma \circ \theta_1 \circ \cdots \circ \theta_k, \gamma \circ \lambda_1 \circ \cdots \circ \lambda_n\} = \{\sigma_1, \sigma_2\}$ is disjoint modulo $\emptyset$-theory with respect to $vars(\mathfrak{U}_\emptyset)$. □

Theorem 3.3.3. *Let $\mathfrak{U}_\emptyset$ be a general $\emptyset$-unification problem. Then*

$$\Sigma(\mathfrak{U}_\emptyset) = mcu_\emptyset(\mathfrak{U}_\emptyset).$$

Proof. By Theorem 3.3.1, Theorem 3.3.2 and Lemma 3.3.1. □

The last results in this section give sufficient conditions for termination of the procedure.

Theorem 3.3.4. *The unification procedure for $\mathfrak{U}_\emptyset$ terminates if $\mathfrak{U}_\emptyset$ contains no sequence variables.*

Proof. Assume $\mathfrak{U}_\emptyset$ contains no sequence variables. Then for each term with a flexible arity head f and the number of arguments n, we replace each occurrence of f with a new n-ary function symbols g. Then termination of the unification procedure for $\mathfrak{U}_\emptyset$ follows from termination of Robinson unification algorithm. □

Another terminating case is when one of the terms to be unified is ground. It yields to the following result:

Theorem 3.3.5. *Matching in the free theory with individual and sequence variables, free constants, free fixed and flexible arity function symbols is finitary.*

Proof. Let $\mathfrak{U}_\emptyset = t_1 \stackrel{?}{=} t_2$ be a unification problem such that t_2 is ground. Then each transformation step in the unification procedure strictly reduces the size of t_2, which eventually leads to termination of the procedure. □

We can add a cycle-checking method to the procedure: stop with failure if a unification problem attached to a node of unification tree coincides with a unification problem in the same branch of the tree. Then the following theorem holds:

Theorem 3.3.6. *The unification procedure with cycle-checking for $\mathfrak{U}_\emptyset$ terminates if no variable occurs more than twice in $\mathfrak{U}_\emptyset$.*

Proof. If no variable occurs more than twice in $\mathfrak{U}_\emptyset$, then no transformation rule increases the size of unification problems. Then we can have only finitely many unification problems in the tree. □

If $\mathfrak{U}_\emptyset$ of the form $t_1 \stackrel{?}{=} t_2$ has the property that sequence variables occur only as arguments of t_1 or t_2, then we can weaken the previous theorem:

Theorem 3.3.7. *The unification procedure with cycle-checking for $\mathfrak{U}_\emptyset = t_1 \stackrel{?}{=} t_2$, where sequence variables occur only as arguments of t_1 or t_2, terminates if no sequence variable occurs more than twice in $\mathfrak{U}_\emptyset$.*

Proof. Using the same argument as in the proof of previous theorem. □

The following termination condition does not require cycle-checking and does not depend on the number of occurrences of sequence variables. Instead, it requires for a unification problem of the form $f(\overline{x}) \stackrel{?}{=}_\emptyset f(t_1, \ldots, t_n)$, $n > 1$, to check whether $\overline{x}$ occurs in $f(t_1, \ldots, t_n)$. We call it *the sequence variable occurrence checking*. We can tailor this checking into the unification tree generation process as follows: if in the tree a successor of the unification problem of the form $f(\overline{x}) \stackrel{?}{=}_\emptyset f(t_1, \ldots, t_n)$ (or of the form $f(t_1, \ldots, t_n) \stackrel{?}{=}_\emptyset f(\overline{x})$), $n > 1$, has to be generated, perform the sequence variable occurrence checking. If $\overline{x}$ occurs in $f(t_1, \ldots, t_n)$, label the node with $\perp$, otherwise proceed in the usual way (projection or transformation).

Theorem 3.3.8. *If $\mathfrak{U}_\emptyset$ is a unification problem such that all sequence variables occurring in $\mathfrak{U}_\emptyset$ are only the last arguments of the term they occur, then the unification procedure with the sequence variable occurrence checking terminates.*

Proof. In this case, every transformation step involving a sequence variable either immediately generates a terminal node in the tree, or produces a new unification problem with the size strictly smaller then the size of the unification problem in the predecessor node. Then we can have only a finite number of nodes in the tree. □

The fact that in most of the applications sequence variables occur precisely only at the last position in terms, underlines the importance of Theorem 3.3.8. The theorem provides an efficient method to terminate unification procedure in many practical applications.

3.4 Extension with Patterns

3.4.1 Preliminaries

In this section we discuss unification in an extension of the free theory with patterns. First we extend the language by introducing a new set of variables $\mathcal{XV}$ – called index variables – and the set $\mathcal{LP}$ of linear polynomials with integer coefficients, whose variables are in $\mathcal{XV}$. If not otherwise stated, the following symbols, with or without indices, are used as metavariables: $\hat{x}, \hat{y}, \hat{z}$ – over index variables, $\hat{a}, \hat{b}, \hat{c}$ – over integer constants, $\hat{p}, \hat{q}, \hat{r}$ – over polynomials from $\mathcal{LP}$.

Definition 3.4.1 (Patterns). *The set of patterns (over $(\mathcal{V}, \mathcal{C}, \mathcal{P}, \mathcal{LP})$), denoted $\mathcal{PAT}$, is the smallest set satisfying the following conditions:*

– *If $c \in \mathcal{CONST}$, $\hat{p}, \hat{q} \in \mathcal{LP}$, then $c_{\hat{p},\hat{q}}$ is a pattern.*
– *If $f \in \mathcal{FFIX}$, $\mathcal{AR}(f) = n$, $n \geq 0$ and $t_1, \ldots, t_n$ are terms such that for all $1 \leq i \leq n$, $t_i \notin \mathcal{SV}$, and $\hat{p}, \hat{q} \in \mathcal{LP}$, then $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$ is a pattern.*
– *If $f \in \mathcal{FFLEX}$, each of $t_1, \ldots, t_n$ $(n \geq 0)$ is a term or pattern, and $\hat{p}, \hat{q} \in \mathcal{LP}$, then $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$ is a pattern.*

$c_{\hat{p},\hat{q}}$ is called a constant pattern. $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$ is called a functional pattern. c is called the head of $c_{\hat{p},\hat{q}}$ and f is called the head of $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$. $f_{\hat{p},\hat{q}}$ is called a prefix of the pattern $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$. A prefix of $c_{\hat{p},\hat{q}}$ is $c_{\hat{p},\hat{q}}$ itself. We say that a pattern $c_{\hat{p},\hat{q}}$ or $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$ is explicit iff $\hat{p}$ and $\hat{q}$ are positive integers.

Definition 3.4.2 (Quasi Pattern-Term). *The set of quasi pattern-terms (over $(\mathcal{V}, \mathcal{C}, \mathcal{P}, \mathcal{LP})$), or, shortly, QP-terms, is the smallest set satisfying the following conditions:*

– *If $v \in \mathcal{IV} \cup \mathcal{SV}$ then v is a quasi pattern-term.*
– *If $c \in \mathcal{CONST}$ then c is a quasi pattern-term.*
– *If $p \in \mathcal{PAT}$ then p is a quasi pattern-term.*
– *If $f \in \mathcal{FFIX}$, $\mathcal{AR}(f) = n$, $n \geq 0$ and $t_1, \ldots, t_n$ are quasi pattern-terms such that for all $1 \leq i \leq n$, $t_i \notin \mathcal{SV} \cup \mathcal{PAT}$, then $f(t_1, \ldots, t_n)$ is a quasi pattern-term.*
– *If $f \in \mathcal{FFLEX}$ and $t_1, \ldots, t_n$ are quasi pattern-terms, then $f(t_1, \ldots, t_n)$ is a quasi pattern-term $(n \geq 0)$.*

f is called the head of $f(t_1, \ldots, t_n)$. For a quasi pattern-term t, $head(t)$ denotes the head of t.

Definition 3.4.3 (Quasi Pattern-Equation). *The set of quasi pattern-equations (over $(\mathcal{V}, \mathcal{C}, \mathcal{P}, \mathcal{LP})$), or, shortly, QP-equations, is the smallest set satisfying the following property: If t_1 and t_2 are QP-terms over $(\mathcal{V}, \mathcal{C}, \mathcal{P}, \mathcal{LP})$ such that $t_1 \notin \mathcal{SV} \cup \mathcal{PAT}$ and $t_2 \notin \mathcal{SV} \cup \mathcal{PAT}$, then $\doteq(t_1, t_2)$ is a quasi pattern-equation over $(\mathcal{V}, \mathcal{C}, \mathcal{P}, \mathcal{LP})$. The symbol $\doteq$ is called the head of the quasi pattern-equation $\doteq(t_1, t_2)$.*

We write QP-equations in infix notation.

We extend notions of $ivars$, $svars$, $const$, fix, $flex$ and sig to sequences of quasi-terms. In addition, we define a new notion – $xvar$ and redefine the old notion of $vars$ for a sequence $t_1, \ldots, t_n$ $(n \geq 0)$ of quasi-terms as follows:

– $xvar(t_1, \ldots, t_n)$ is the set of all index variables occurring in the sequence $t_1, \ldots, t_n$.
– $vars(t_1, \ldots, t_n) = ivars(t_1, \ldots, t_n) \cup svars(t_1, \ldots, t_n) \cup xvars(t_1, \ldots, t_n)$.

Definition 3.4.4 (Quasi Pattern-Substitution). *A quasi pattern-substitution, or, shortly, QP-substitution, is a finite set*

$$\{ \quad x_1 \leftarrow s_1, \ldots, x_n \leftarrow s_n, \\ \overline{x}_1 \leftarrow t^1_1, \ldots, t^1_{k_1}, \ldots, \overline{x}_m \leftarrow t^m_1, \ldots, t^m_{k_m}, \\ \hat{x}_1 \leftarrow \hat{p}_1, \ldots, \hat{x}_l \leftarrow \hat{p}_l \quad \}$$

where

– $n \geq 0$, $m \geq 0$ *and for all* $1 \leq i \leq m$, $k_i \geq 0$,
– $x_1, \ldots, x_n$ *are distinct individual variables,*
– $\overline{x}_1, \ldots, \overline{x}_m$ *are distinct sequence variables,*
– $\hat{x}_1, \ldots, \hat{x}_l$ *are distinct index variables,*
– *for all* $1 \leq i \leq n$, s_i *is a QP-term,* $s_i \notin \mathcal{SV} \cup \mathcal{PAT}$ *and* $s_i \neq x_i$,
– *for all* $1 \leq i \leq m$, $t^i_1, \ldots, t^i_{k_i}$ *is a sequence of QP-terms and if* k_i*=1 then* $t^i_{k_i} \neq \overline{x}_i$,
– *for all* $1 \leq i \leq l$, $\hat{p}_i \in \mathcal{LP}$ *and* $\hat{p}_i \neq \hat{x}_i$.

Each $x_i \leftarrow s_i$, $\overline{x}_i \leftarrow t^i_1, \ldots, t^i_{k_i}$ *and* $\hat{x}_i \leftarrow \hat{p}_i$ *is called a QP-binding respectively for* x_i, $\overline{x_i}$ *and* $\hat{x}_i$.

A QP-substitution is called ground iff all $s_1, \ldots, s_n, t^1_1, \ldots, t^1_{k_1}, t^m_1, \ldots, t^m_{k_m}$ are variable-free terms and all $\hat{p}_1, \ldots, \hat{p}_l$ are integers. A QP-substitution is called empty iff $n = 0$, $m = 0$ and $l = 0$. Greek letters are used to denote QP-substitutions. The letter ε denotes the empty substitution.

For a QP-substitution θ, we denote by $\theta^=$ the following system of linear Diophantine equations:

$$\{\hat{x} = \hat{p} \mid \hat{x} \leftarrow \hat{p} \in \theta\}.$$

We define a notion of instance for index variables and polynomials from $\mathcal{LP}$:

Definition 3.4.5. *Let* θ *be a QP-substitution. Then:*

– *An instance of a index variable* $\hat{x}$ *with respect to* θ*, denoted* $\hat{x}\theta$*, is defined as*

$$\hat{x}\theta = \begin{cases} \hat{p} & if\ \hat{x} \leftarrow \hat{p} \in \theta, \\ \hat{x} & otherwise \end{cases}$$

– *An instance of a polynomial* $\hat{p} = \hat{a}_1\hat{x}_1 + \cdots + \hat{a}_k\hat{x}_k \in \mathcal{LP}$ *with respect to* θ*, denoted* $\hat{p}\theta$*, is a polynomial from* $\mathcal{LP}$ *obtained from* $\hat{a}_1\hat{x}_1\theta + \cdots + \hat{a}_k\hat{x}_k\theta$ *by arithmetic simplification.*

On the basis of this definition, we extend the notion of instance on *QP*-terms and *QP*-equations with respect to *QP*-substitutions. The notions of domain, codomain, range and composition are extended to *QP*-substitutions in a straightforward way.

Example 3.4.1. Let $\theta = \{x \leftarrow f(y),\ \overline{x} \leftarrow \overline{y}, \overline{x},\ \overline{y} \leftarrow \overline{y}, \overline{z}, \hat{x} \leftarrow 3\hat{x} + \hat{y},\ \hat{z} \leftarrow \hat{y} - 2\}$ and $\lambda = \{y \leftarrow g(c),\ \overline{x} \leftarrow c,\ \overline{z} \leftarrow,\ \hat{x} \leftarrow 2\hat{y} + 1,\ \hat{y} \leftarrow \hat{z} + 2\}$. Then $\theta \circ \lambda = \{x \leftarrow f(g(c)),\ y \leftarrow g(c),\ \overline{x} \leftarrow \overline{y}, c,\ \overline{z} \leftarrow, \hat{z} \leftarrow 6\hat{y} + \hat{z} + 5, \hat{y} \leftarrow \hat{z} + 2\}$.

Definition 3.4.6 (Quasi Pattern E-unification Problem). *A general quasi pattern E-unification, or, shortly, QPE-unification problem with sequence variables, flexible arity symbols and QP-terms is a finite system of QP-equations* $\langle s_1 \stackrel{?}{\doteq}_E t_1, \ldots, s_n \stackrel{?}{\doteq}_E t_n \rangle$.

Definition 3.4.7 (Explicit Pattern Expansion). *Let* Q *be one of the following: QP-term, QP-equation, QP-substitution or QPE-unification problem. The explicit pattern expansion in* Q*, denoted as* $expex(Q)$*, is respectively a QP-term, QP-equation, QP-substitution or QPE-unification problem obtained from* Q *by replacing each occurrence of an explicit pattern in* Q *with the sequence of QP-terms as long as possible in the following way:*

– *each occurrence of an explicit pattern* $c_{\hat{a},\hat{a}}$ *is replaced by the single QP-term* $c_{\hat{a}}$*;*
– *each occurrence of an explicit pattern* $c_{\hat{a},\hat{b}}$*,* $\hat{a} < \hat{b}$*, is replaced by the sequence* $c_{\hat{a}}, c_{\hat{a}+1}, \ldots, c_{\hat{b}}$*;*
– *each occurrence of an explicit pattern* $f_{\hat{a},\hat{a}}(t_1, \ldots, t_n)$ *is replaced by the single QP-term* $f_{\hat{a}}(t_1, \ldots, t_n)$*.*
– *each occurrence of an explicit pattern* $f_{\hat{a},\hat{b}}(t_1, \ldots, t_n)$*,* $\hat{a} < \hat{b}$*, is replaced by the sequence* $f_{\hat{a}}(t_1, \ldots, t_n), f_{\hat{a}+1}(t_1, \ldots, t_n), \ldots, f_{\hat{b}}(t_1, \ldots, t_n)$*.*

Example 3.4.2. Let t be a *QP*-term $f(a, g_{1,3}(c_{1,2}, \overline{y}), h_{1,\hat{x}}(x))$. Then

$$expex(t) = f(a, g_1(c_1, c_2, \overline{y}), g_2(c_1, c_2, \overline{y}), g_3(c_1, c_2, \overline{y}), h_{1,\hat{x}}(x)).$$

Below by *QP*-expression we mean either *QP*-term, *QP*-equation, *QP*-substitution or *QPE*-unification problem.

Definition 3.4.8 (Pattern-Term, Pattern-Equation, Pattern-Substitution, Pattern-E-Unification Problem). *Let* Q *be a QP-expression.* Q *is called*

– *a pattern-term or, shortly, P-term;*
– *a pattern-equation or, shortly, P-equation;*
– *a pattern-substitution or, shortly, P-substitution or*

– *a pattern E-unification problem or, shortly, PE-unification problem*

over $(\mathcal{V}, \mathcal{C}, \mathcal{P}, \mathcal{LP})$ *iff there exist a substitution* σ *such that* $dom(\sigma) \subset \mathcal{XV}$ *and* $expex(Q\sigma)$ *is respectively*

– *a term;*
– *an equation;*
– *a substitution or*
– *an E-unification problem*

over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$.

In order to decide whether a *QP*-expression is a corresponding *P*-expression, we associate a system of linear Diophantine constraints to a *QP*-expression and show that if the system has a positive integer solution, then the *QP*-expression is the corresponding *P*-expression.

Definition 3.4.9 (Linear Diophantine Constraints Associated with a *QP*-Expression). *For a QP-expression Q, the associated system of linear Diophantine constraints* $ldc(Q)$ *is defined in the following way:*

– Q *is a QP-term. Then*
  – *If* $Q \in \mathcal{IV} \cup \mathcal{SV} \cup \mathcal{CONST}$ *then* $ldc(Q)$ *is empty.*
  – *If* Q *is* $c_{\hat{p},\hat{q}}$ *or* $f_{\hat{p},\hat{q}}(t_1, \ldots, t_n)$, *then* $ldc(Q)$ *is*

$$1 \leq \hat{p} \;\wedge\; \hat{p} \leq \hat{q}.$$

  – *If* Q *is* $f(t_1, \ldots, t_n)$, *where* $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$, *then* $ldc(Q)$ *is*

$$ldc(t_1) \wedge \ldots \wedge ldc(t_n),$$

– Q *is either QP-equation, QP-substitution or a QPE-unification problem. Then* $ldc(Q)$ *is*

$$ldc(t_1) \wedge \ldots \wedge ldc(t_m),$$

where $t_1, \ldots, t_m$ *are all QP-terms occurring in* Q.

We will use the letter d with or without indices to denote linear Diophantice constraints.

Theorem 3.4.1. *A QP-expression Q is the corresponding P-expression if* $ldc(Q)$ *has a positive integer solution.*

Proof. We prove the theorem for a *QP*-term. For the other *QP*-expressions the proof is analogous. Let $\{\hat{x}_1, \ldots, \hat{x}_s\} = xvars(Q)$ and $\hat{x}_1 = \hat{a}_1, \ldots, \hat{x}_s = \hat{a}_s$ be a positive integer solution of $ldc(Q)$. Then we assemble substitution σ as follows: $\sigma = \{\hat{x}_1 \leftarrow \hat{a}_1, \ldots, \hat{x}_s \leftarrow \hat{a}_s\}$. By the construction of the system $ldc(Q)$, all the patterns occurring in $Q\sigma$ are explicit with the following property: for each pattern $c_{\hat{b},\hat{c}}$ or $f_{\hat{b},\hat{c}}(t_1, \ldots, t_r)$ we have $1 \leq \hat{b}$ and $\hat{b} \leq \hat{c}$. Therefore, $expex(Q\sigma)$ contains no patterns and thus, is a term over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$. Hence, by Definition 3.4.8, Q is a *P*-term. □

Definition 3.4.10 (Unifier of a *PE*-unification problem). *Let $\mathfrak{U}$ be a PE-unification problem of the form $\langle s_1 \stackrel{?}{\doteq}_E t_1, \ldots, s_n \stackrel{?}{\doteq}_E t_n \rangle$. A P-substitution θ is called a PE-unifier of $\mathfrak{U}$ iff*

- *$xvar(\mathfrak{U}) \subseteq dom(\theta)$;*
- *for each $\hat{x} \in dom(\theta)$, $\hat{x}\theta$ is a positive integer;*
- *each pattern which occurs in P-terms in $cod(\theta)$ is explicit;*
- *for each $1 \leq i \leq n$, the P-equality $s_i\theta \doteq_E t_i\theta$ holds.*

Example 3.4.3. Let $\mathfrak{U} = f(\overline{x}, \overline{y}) \stackrel{?}{\doteq}_\emptyset f(h_{\hat{x},\hat{y}}(z))$. Then

$$\theta = \{\overline{x} \leftarrow h_{1,2}(z), \overline{y} \leftarrow h_{3,6}(z), \hat{x} \leftarrow 1, \hat{y} \leftarrow 6\}$$

is one of $\emptyset$-unifiers of $\mathfrak{U}$.

We have the following proposition:

Proposition 3.4.1. *If a P-substitution θ is a unifier of a PE-unification problem $\mathfrak{U}$, then $\theta^=$ gives a (unique) positive integer solution to the system $ldc(\mathfrak{U}) \wedge ldc(\theta)$.*

Definition 3.4.11 (More General *P*-Substitution). *A P-substitution θ is more general than a P-substitution σ on a finite set of variables V modulo a theory E (denoted $\theta \ll_E^V \sigma$) iff there exists a P-substitution λ such that*

- *for all $\overline{x} \in V$,*
  - *$\overline{x} \leftarrow \ \notin \lambda$;*
  - *there exist P-terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma = t_1, \ldots, t_n$, $\overline{x}\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality $t_i \doteq_E s_i$ holds;*
- *for all $x \in V$, the P-equality $x\sigma \doteq_E x\theta \circ \lambda$ holds;*
- *for all $\hat{x} \in V$, $\hat{x}\sigma = \hat{x}\theta \circ \lambda$.*

Notions of minimal set of substitutions and minimal complete set of unifiers are extended for *P*-substitutions and *PE*-unification problems in a straightforward way.

We represent a minimal complete set of unifiers of $\mathfrak{U}$ – $mcu_E(\mathfrak{U})$ as a set of *P*-substitution/constraint pairs. The constraints are linear Diophantine equations and/or inequalities. The representation must satisfy the following properties:

- for each pair $\{\theta, d\}$ in the representation and for each positive integer solution μ of d, the *P*-substitution $(\theta \circ \mu)|_{vars(\mathfrak{U})}$ must be in $mcu_E(\mathfrak{U})$;
- for each substitution $\sigma \in mcu_E(\mathfrak{U})$ there must be a pair $\{\theta, d\}$ in the representation such that $\sigma = (\theta \circ \mu)|_{vars(\mathfrak{U})}$ for a positive integer solution μ of d.

The following two example give a demonstration of a representation of a minimal complete set of unifiers as a set of *P*-substitution/constraint pairs:

Example 3.4.4. Let $\mathfrak{U} = f(\overline{x}, \overline{y}) \stackrel{?}{\doteq}_{\emptyset} f(h_{\hat{x},\hat{y}}(z))$. Then we can represent $mcu_{\emptyset}(\mathfrak{U})$ as a finite set of P-substitution/constraint pairs:

$$
\begin{array}{rl}
S = \{ & \{\{\overline{x} \leftarrow, \overline{y} \leftarrow h_{\hat{x},\hat{y}}(z)\}, 1 \leq \hat{x} \wedge \hat{x} \leq \hat{y}\}, \\
& \{\{\overline{x} \leftarrow, h_{\hat{x},\hat{y}}(z), \overline{y} \leftarrow\}, 1 \leq \hat{x} \wedge \hat{x} \leq \hat{y}\}, \\
& \{\{\overline{x} \leftarrow h_{\hat{x},\hat{z}}(z), \overline{y} \leftarrow h_{\hat{z}+1,\hat{y}}(z)\}, \\
& \qquad 1 \leq \hat{x} \wedge \hat{x} \leq \hat{z} \wedge \hat{z}+1 \leq \hat{y}\} \quad \}.
\end{array}
$$

In fact,

$$
\begin{array}{rl}
mcu_{\emptyset}(\mathfrak{U}) = \{\sigma \mid & \text{there exists } \{\theta, d\} \in \text{S and } \mu \text{ such that} \\
& \mu \text{ is a positive integer solution of } d \\
& \text{and } \sigma = (\theta \circ \mu)|_{\text{vars}(\mathfrak{U})}\}.
\end{array}
$$

For instance, a solution $\{\hat{x} \leftarrow 1,\ \hat{r} \leftarrow 3,\ \hat{y} \leftarrow 4\}$ of the constraint $1 \leq \hat{x} \wedge \hat{x} \leq \hat{z} \wedge \hat{z}+1 \leq \hat{y}$, applied on the substitution $\{\overline{x} \leftarrow h_{\hat{x},\hat{z}}(z), \overline{y} \leftarrow h_{\hat{z}+1,\hat{y}}(z)\}$ gives $\{\overline{x} \leftarrow h_{1,3}(z), \overline{y} \leftarrow h_{4,4}(z), \hat{x} \leftarrow 1, \hat{z} \leftarrow 3, \hat{y} \leftarrow 4\}$. The restriction of the latter to $vars(\mathfrak{U})$ is $\{\overline{x} \leftarrow h_{1,3}(z), \overline{y} \leftarrow h_{4,4}(z), \hat{x} \leftarrow 1, \hat{y} \leftarrow 4\}$, which belongs to $mcu_{E}(\mathfrak{U})$. In the expanded form the substitution looks like $\{\overline{x} \leftarrow h_1(z), h_2(z), h_3(z), \overline{y} \leftarrow h_4(z), \hat{x} \leftarrow 1, \hat{y} \leftarrow 4\}$.

Example 3.4.5. Let $\mathfrak{U} = f(\overline{x}, h_{\hat{x},\hat{y}}(z)) \stackrel{?}{\doteq}_{\emptyset} f(h_{\hat{x},\hat{y}}(z), \overline{x})$. Then the following set gives an infinite representation of $mcu_{\emptyset}(\mathfrak{U})$ as a set of substitution/constraint pairs:

$$
\begin{array}{rl}
S = \{ & \{\{\overline{x} \leftarrow\}, 1 \leq \hat{x} \wedge \hat{x} \leq \hat{y}\}, \\
& \{\{\overline{x} \leftarrow h_{\hat{x},\hat{y}}(z)\}, 1 \leq \hat{x} \wedge \hat{x} \leq \hat{y}\}, \\
& \{\{\overline{x} \leftarrow h_{\hat{x},\hat{y}}(z), h_{\hat{x},\hat{y}}(z)\}, 1 \leq \hat{x} \wedge \hat{x} \leq \hat{y}\} \\
& \ldots \qquad \}.
\end{array}
$$

Again,

$$
\begin{array}{rl}
mcu_{\emptyset}(\mathfrak{U}) = \{\sigma \mid & \text{there exists } \{\theta, d\} \in \text{S and } \mu \text{ such that} \\
& \mu \text{ is a positive integer solution of } d \\
& \text{and } \sigma = (\theta \circ \mu)|_{\text{vars}(\mathfrak{U})}\}.
\end{array}
$$

Pattern-terms naturally appear in the proving context, when one wants to skolemize, for instance, the expression $\forall x \exists \overline{y}\, (g(x) \doteq g(\overline{y}))$. Here $\overline{y}$ should be replaced with a sequence of terms $f_1(x), \ldots, f_{n(x)}(x)$, where $f_1, \ldots, f_{n(x)}$ are Skolem functions. The problem is that we can not know in advance the length of such a sequence.

Note that we consider $\hat{x}$ instead of $n(x)$. This is because, given an unification problem $\mathfrak{U}$ in which $n(x)$ occurs, we can do a variable abstraction on $n(x)$ with a fresh index variable $\hat{x}$ and instead of $\mathfrak{U}$ consider $\mathfrak{U}'$ with the constraint $\hat{x} = n(x)$, where $\mathfrak{U}'$ is obtained from $\mathfrak{U}$ by replacing each occurrence of $n(x)$ with $\hat{x}$. One of the tasks for unification with patterns is to find a proper value for $\hat{x}$, if possible.

Next, we design a unification procedure for a P-$\emptyset$-unification problem. Note that as in the case of $\emptyset$-unification, it is enough to consider single P-equations instead of systems of P-equations. The problem has a form of P-equation $t_1 \stackrel{?}{=}_\emptyset t_2$. We design the unification procedure as a tree generation process based on three basic steps: projection, transformation and pattern-simplification. They are described in terms of "quasi-patterns" instead of "patterns".

3.4.2 Projection

Projection eliminates some sequence variables from the given QP-$\emptyset$-unification problem $\mathfrak{Q}_\emptyset$. Let $\Pi(\mathfrak{Q}_\emptyset)$ be the following set of substitutions: $\{\{\overline{x} \leftarrow \ \mid \overline{x} \in S\} \mid S \subseteq svars(\mathfrak{Q}_\emptyset)\}$. $\Pi(\mathfrak{Q}_\emptyset)$ is called the set of projecting substitutions for $\mathfrak{Q}_\emptyset$. Each $\pi \in \Pi$ replaces some sequence variables from $\mathfrak{Q}_\emptyset$ with the empty sequence. The projection rule is shown in Figure 3.4.

Projection:	$s \stackrel{?}{=}_\emptyset t \rightsquigarrow \langle\langle s\pi_1 \stackrel{?}{=}_\emptyset t\pi_1,\, \pi_1,\, d\rangle, \ldots,$ $\langle s\pi_k \stackrel{?}{=}_\emptyset t\pi_k,\, \pi_k,\, d\rangle\rangle$	where $\{\pi_1, \ldots, \pi_k\} = \Pi(s \stackrel{?}{=}_\emptyset t)$, $d = ldc(s \stackrel{?}{=}_\emptyset t)$

Fig. 3.4. Projection rule for QP-$\emptyset$-unification.

3.4.3 Transformation

Each of the transformation rules for QP-$\emptyset$-unification have one of the following forms: $\mathfrak{Q}_\emptyset \rightsquigarrow \perp$ or $\mathfrak{Q}_\emptyset \rightsquigarrow \langle\langle \mathfrak{S}_1, \sigma_1, d_1\rangle, \ldots, \langle \mathfrak{S}_n, \sigma_n, d_n\rangle\rangle$ where each of the successors $\mathfrak{S}_i$ is either $\top$ or a new unification problem, σ-s are substitutions and d-s are linear Diophantine constraints.

The full set of transformation rules are given on Figure 3.5. It consists of four family of rules: Success, Failure, Elimination and Splitting and differs from transformation rules on Figure 3.2 only by added constraints.

3.4.4 Pattern Simplification

Like the transformation rules, QP-$\emptyset$-simplification rules for patterns have one of the following forms: $\mathfrak{Q}_\emptyset \rightsquigarrow \perp$ or $\mathfrak{Q}_\emptyset \rightsquigarrow \langle\langle \mathfrak{S}_1, \sigma_1, d_1\rangle, \ldots, \langle \mathfrak{S}_n, \sigma_n, d_n\rangle\rangle$ where each of the successors $\mathfrak{S}_i$ is either $\top$ or a new unification problem, σ-s are substitutions and d-s are linear Diophantine constraints.

The full set of pattern-simplification rules are given on Figure 3.6 (for constant patterns) and on Figure 3.6 (for functional patterns). Each of them consists of three family of rules: Failure, Contraction and Separation.

Success:	$t \stackrel{?}{\doteq}_{\emptyset} t \rightsquigarrow \langle\langle\top,\ \varepsilon,\ true\rangle\rangle.$	
	$x \stackrel{?}{\doteq}_{\emptyset} t \rightsquigarrow \langle\langle\top,\ \{x \leftarrow t\},\ true\rangle\rangle,$	if $x \notin ivars(t)$.
	$t \stackrel{?}{\doteq}_{\emptyset} x \rightsquigarrow \langle\langle\top,\ \{x \leftarrow t\},\ true\rangle\rangle,$	if $x \notin ivars(t)$.
Failure:	$c_1 \stackrel{?}{\doteq}_{\emptyset} c_2 \rightsquigarrow \bot,$	if $c_1 \neq c_2$.
	$x \stackrel{?}{\doteq}_{\emptyset} t \rightsquigarrow \bot,$	if $t \neq x$ and $x \in ivars(t)$.
	$t \stackrel{?}{\doteq}_{\emptyset} x \rightsquigarrow \bot,$	if $t \neq x$ and $x \in ivars(t)$.
	$f_1(\tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f_2(\tilde{s}) \rightsquigarrow \bot,$	if $f_1 \neq f_2$.
	$f() \stackrel{?}{\doteq}_{\emptyset} f(t_1, \tilde{t}) \rightsquigarrow \bot.$	
	$f(t_1, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f() \rightsquigarrow \bot.$	
	$f(\overline{x}, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $s_1 \neq \overline{x}$ and $\overline{x} \in svars(s_1)$.
	$f(s_1, \tilde{s}) \stackrel{?}{\doteq}_{\emptyset} f(\overline{x}, \tilde{t}) \rightsquigarrow \bot,$	if $s_1 \neq \overline{x}$ and $\overline{x} \in svars(s_1)$.
	$f(t_1, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $t_1 \stackrel{?}{\doteq}_{\emptyset} s_1 \rightsquigarrow \bot$.
Eliminate:	$f(t_1, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}\sigma) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{s}\sigma),\ \sigma,\ true\rangle\rangle,$	if $t_1 \stackrel{?}{\doteq}_{\emptyset} s_1 \rightsquigarrow \langle\langle\top,\ \sigma,\ true\rangle\rangle$.
	$f(\overline{x}, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(\overline{x}, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{s}),\ \varepsilon,\ true\rangle\rangle.$	
	$f(\overline{x}, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}\sigma_1) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{s}\sigma_1),\ \sigma_1,\ true\rangle,$ $\langle g(\overline{x}, \tilde{t}\sigma_2) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{s}\sigma_2),\ \sigma_2,\ true\rangle\rangle,$	$s_1 \notin \mathcal{SV} \cup \mathcal{PAT}$, $\overline{x} \notin svars(s_1)$, where $\sigma_1 = \{\overline{x} \leftarrow s_1\}$, $\sigma_2 = \{\overline{x} \leftarrow s_1, \overline{x}\}$.
	$f(s_1, \tilde{s}) \stackrel{?}{\doteq}_{\emptyset} f(\overline{x}, \tilde{t}) \rightsquigarrow$ $\langle\langle g(\tilde{s}\sigma_1) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{t}\sigma_1),\ \sigma_1,\ true\rangle,$ $\langle g(\tilde{s}\sigma_2) \stackrel{?}{\doteq}_{\emptyset} g(\overline{x}, \tilde{t}\sigma_2),\ \sigma_2,\ true\rangle\rangle,$	$s_1 \notin \mathcal{SV} \cup \mathcal{PAT}$, $\overline{x} \notin svars(s_1)$, where $\sigma_1 = \{\overline{x} \leftarrow s_1\}$, $\sigma_2 = \{\overline{x} \leftarrow s_1, \overline{x}\}$.
	$f(\overline{x}, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(\overline{y}, \tilde{s}) \rightsquigarrow$ $\langle\langle g(\tilde{t}\sigma_1) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{s}\sigma_1),\ \sigma_1,\ true\rangle,$ $\langle g(\overline{x}, \tilde{t}\sigma_2) \stackrel{?}{\doteq}_{\emptyset} g(\tilde{s}\sigma_2),\ \sigma_2,\ true\rangle,$ $\langle g(\tilde{t}\sigma_3) \stackrel{?}{\doteq}_{\emptyset} g(\overline{y}, \tilde{s}\sigma_3),\ \sigma_3,\ true\rangle\,\rangle,$	where $\sigma_1 = \{\overline{x} \leftarrow \overline{y}\}$, $\sigma_2 = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$, $\sigma_3 = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$.
Split:	$f(t_1, \tilde{t}) \stackrel{?}{\doteq}_{\emptyset} f(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle f(r_1, \tilde{t}\sigma_1) \stackrel{?}{\doteq}_{\emptyset} f(q_1, \tilde{s}\sigma_1),\ \sigma_1,\ d_1\rangle,$ $\ldots,$ $\langle f(r_k, \tilde{t}\sigma_k) \stackrel{?}{\doteq}_{\emptyset} f(q_k, \tilde{s}\sigma_k),\ \sigma_k,\ d_k\rangle\rangle$	if $t_1, s_1 \notin \mathcal{IV} \cup \mathcal{SV} \cup \mathcal{PAT}$ and $t_1 \stackrel{?}{\doteq}_{\emptyset} s_1 \rightsquigarrow$ $\langle\langle r_1 \stackrel{?}{\doteq}_{\emptyset} q_1,\ \sigma_1,\ d_1\rangle, \ldots$ $\langle r_k \stackrel{?}{\doteq}_{\emptyset} q_k,\ \sigma_k,\ d_k\rangle\rangle$.

Fig. 3.5. Transformation rules for QP-$\emptyset$-unification. $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of QP-terms. $f, f_1, f_2 \in \mathcal{FFIX} \cup \mathcal{FFLEX}$. $g \in \mathcal{FFLEX}$ is a new symbol, if in the same rule $f \in \mathcal{FFIX}$. Otherwise $g = f$.

Failure:	$f(r_1, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{p},\hat{q}}, \tilde{s}) \rightsquigarrow \bot,$	if $r_1 \notin \mathcal{IV} \cup \mathcal{SV}$, $head(r_1) \neq c$.
	$f(h_{\hat{p},\hat{q}}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $s_1 \notin \mathcal{IV} \cup \mathcal{SV}$, $head(s_1) \neq c$.
Contraction:	$f(x, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{p},\hat{q}}, \tilde{s}) \rightsquigarrow$	where
	$\langle\langle f(\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_1, \sigma_1, \hat{p} = \hat{q}\rangle,$	$\sigma_1 = \{x \leftarrow c_{\hat{q}}\},$
	$\langle f(\tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(c_{\hat{p}+1,\hat{q}}, \tilde{s})\sigma_2,$ $\sigma_2, \hat{p} + 1 \leq \hat{q}\rangle\rangle.$	$\sigma_2 = \{x \leftarrow c_{\hat{p}}\}$
	$f(c_{\hat{p},\hat{q}}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(x, \tilde{s}) \rightsquigarrow$	where
	$\langle\langle f(\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_1, \sigma_1, \hat{p} = \hat{q}\rangle,$	$\sigma_1 = \{x \leftarrow c_{\hat{q}}\},$
	$\langle f(c_{\hat{p}+1,\hat{q}}, \tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_2,$ $\sigma_2, \hat{p} + 1 \leq \hat{q}\rangle\rangle,$	$\sigma_2 = \{x \leftarrow c_{\hat{p}}\}.$
	$f(\overline{x}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{p},\hat{q}}, \tilde{s}) \rightsquigarrow$	where
	$\langle\langle f(\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(c_{\hat{x}+1,\hat{q}}, \tilde{s})\sigma_1,$ $\sigma_1, \hat{p} \leq \hat{x} \wedge \hat{x} + 1 \leq \hat{q}\rangle,$	$\sigma_1 = \{\overline{x} \leftarrow c_{\hat{p},\hat{x}}\},$
	$\langle f(\overline{x}, \tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_2, \sigma_2, true\rangle,$	$\sigma_2 = \{\overline{x} \leftarrow c_{\hat{p},\hat{q}}, \overline{x}\},$
	$\langle f(\tilde{r})\sigma_3 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_3, \sigma_3, true\rangle\rangle,$	$\sigma_3 = \{\overline{x} \leftarrow c_{\hat{p},\hat{q}}\}.$
	$f(c_{\hat{p},\hat{q}}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(\overline{x}, \tilde{s}) \rightsquigarrow$	where
	$\langle\langle f(c_{\hat{x}+1,\hat{q}}, \tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_1,$ $\sigma_1, \hat{p} \leq \hat{x} \wedge \hat{x} + 1 \leq \hat{q}\rangle,$	$\sigma_1 = \{\overline{x} \leftarrow c_{\hat{p},\hat{x}}\},$
	$\langle f(\tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(\overline{x}, \tilde{s})\sigma_2, \sigma_2, true\rangle,$	$\sigma_2 = \{\overline{x} \leftarrow c_{\hat{p},\hat{q}}, \overline{x}\},$
	$\langle f(\tilde{r})\sigma_3 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_3, \sigma_3, true\rangle\rangle,$	$\sigma_3 = \{\overline{x} \leftarrow c_{\hat{p},\hat{q}}\}.$
Separation:	$f(c_{\hat{r}}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{p},\hat{q}}, \tilde{s}) \rightsquigarrow$	
	$\langle\langle f(\tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{p}+1,\hat{q}}, \tilde{s}),$ $\varepsilon, \hat{p} = \hat{r} \wedge \hat{p} + 1 \leq \hat{q}\rangle\rangle.$	
	$f(c_{\hat{p},\hat{q}}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{r}}, \tilde{s}) \rightsquigarrow$	
	$\langle\langle f(c_{\hat{p}+1,\hat{q}}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(\tilde{s}),$ $\varepsilon, \hat{p} = \hat{r} \wedge \hat{p} + 1 \leq \hat{q}\rangle\rangle.$	
	$f(c_{\hat{p}_1,\hat{p}_2}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{q}_1,\hat{q}_2}, \tilde{s}) \rightsquigarrow$	where
	$\langle\langle f(\tilde{r}) \stackrel{?}{=}_{\emptyset} f(c_{\hat{p}_2+1,\hat{q}_2}, \tilde{s}), \varepsilon, d_1\rangle$	$d_1 = (\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 + 1 \leq \hat{q}_2)$
	$\langle f(c_{\hat{q}_2+1,\hat{p}_2}, \tilde{r}) \stackrel{?}{=}_{\emptyset} f(\tilde{s}), \varepsilon, d_2\rangle$	$d_2 = (\hat{p}_1 = \hat{q}_1 \wedge \hat{q}_2 + 1 \leq \hat{p}_2)$
	$\langle f(\tilde{r}) \stackrel{?}{=}_{\emptyset} f(\tilde{s}), \varepsilon, d_3\rangle\rangle,$	$d_3 = (\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 = \hat{q}_2).$

Fig. 3.6. Constant pattern simplification rules for QP-$\emptyset$-unification. $\tilde{r}$ and $\tilde{s}$ are possibly empty sequences of QP-terms. $f \in \mathcal{FFLEX}$.

$$
\begin{array}{lll}
\textsf{Failure:} & f(r_1,\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{s}) \rightsquigarrow \bot, & \text{if } r_1 \notin \mathcal{IV} \cup \mathcal{SV},\ head(r_1) \neq h. \\
& f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(s_1,\tilde{s}) \rightsquigarrow \bot, & \text{if } s_1 \notin \mathcal{IV} \cup \mathcal{SV},\ head(s_1) \neq h. \\
\textsf{Contraction:} & f(x,\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{s}) \rightsquigarrow & \text{if } x \notin ivars(h_{\hat{p},\hat{q}}(\tilde{t})), \text{ where} \\
& \quad \langle\langle f(\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_1, \sigma_1, \hat{p}=\hat{q}\rangle, & \quad \sigma_1 = \{x \leftarrow h_{\hat{q}}(\tilde{t})\}, \\
& \quad \langle f(\tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(h_{\hat{p}+1,\hat{q}}(\tilde{t}),\tilde{s})\sigma_2, & \quad \sigma_2 = \{x \leftarrow h_{\hat{p}}(\tilde{t})\} \\
& \qquad \sigma_2, \hat{p}+1 \leq \hat{q}\rangle\rangle, & \\
& f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(x,\tilde{s}) \rightsquigarrow & \text{if } x \notin ivars(h_{\hat{p},\hat{q}}(\tilde{t})), \text{ where} \\
& \quad \langle\langle f(\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_1, \sigma_1, \hat{p}=\hat{q}\rangle, & \quad \sigma_1 = \{x \leftarrow h_{\hat{q}}(\tilde{t})\}, \\
& \quad \langle f(h_{\hat{p}+1,\hat{q}}(\tilde{t}),\tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_2, & \quad \sigma_2 = \{x \leftarrow h_{\hat{p}}(\tilde{t})\}. \\
& \qquad \sigma_2, \hat{p}+1 \leq \hat{q}\rangle\rangle, & \\
& f(\overline{x},\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{s}) \rightsquigarrow & \text{if } \overline{x} \notin svars(h_{\hat{p},\hat{q}}(\tilde{t})), \text{ where} \\
& \quad \langle\langle f(\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(h_{\hat{x}+1,\hat{q}}(\tilde{t}),\tilde{s})\sigma_1, & \quad \sigma_1 = \{\overline{x} \leftarrow h_{\hat{p},\hat{x}}(\tilde{t})\}, \\
& \qquad \sigma_1, \hat{p} \leq \hat{x} \wedge \hat{x}+1 \leq \hat{q}\rangle, & \\
& \quad \langle f(\overline{x},\tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_2, \sigma_2, true\rangle, & \quad \sigma_2 = \{\overline{x} \leftarrow h_{\hat{p},\hat{q}}(\tilde{t}), \overline{x}\}, \\
& \quad \langle f(\tilde{r})\sigma_3 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_3, \sigma_3, true\rangle\rangle, & \quad \sigma_3 = \{\overline{x} \leftarrow h_{\hat{p},\hat{q}}(\tilde{t})\}. \\
& f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(\overline{x},\tilde{s}) \rightsquigarrow & \text{if } \overline{x} \notin svars(h_{\hat{p},\hat{q}}(\tilde{t})), \text{ where} \\
& \quad \langle\langle f(h_{\hat{x}+1,\hat{q}}(\tilde{t}),\tilde{r})\sigma_1 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_1, & \quad \sigma_1 = \{\overline{x} \leftarrow h_{\hat{p},\hat{x}}(\tilde{t})\}, \\
& \qquad \sigma_1, \hat{p} \leq \hat{x} \wedge \hat{x}+1 \leq \hat{q}\rangle, & \\
& \quad \langle f(\tilde{r})\sigma_2 \stackrel{?}{=}_{\emptyset} f(\overline{x},\tilde{s})\sigma_2, \sigma_2, true\rangle, & \quad \sigma_2 = \{\overline{x} \leftarrow h_{\hat{p},\hat{q}}(\tilde{t}), \overline{x}\}, \\
& \quad \langle f(\tilde{r})\sigma_3 \stackrel{?}{=}_{\emptyset} f(\tilde{s})\sigma_3, \sigma_3, true\rangle\rangle, & \quad \sigma_3 = \{\overline{x} \leftarrow h_{\hat{p},\hat{q}}(\tilde{t})\}. \\
\textsf{Separation:} & f(h_{\hat{r}}(\tilde{q}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{s}) \rightsquigarrow & \text{where} \\
& \quad \langle\langle f(h_{\hat{p}}(\tilde{q}),\tilde{r}) \stackrel{?}{=}_{\emptyset} & \quad d = (\hat{p} = \hat{r} \wedge \hat{p}+1 \leq \hat{q}). \\
& \qquad f(h_{\hat{p}}(\tilde{t}), h_{\hat{p}+1,\hat{q}}(\tilde{t}),\tilde{s}), \varepsilon, d\rangle, & \\
& f(h_{\hat{p},\hat{q}}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{r}}(\tilde{q}),\tilde{s}) \rightsquigarrow & \text{where} \\
& \quad \langle\langle f(h_{\hat{p}}(\tilde{t}), h_{\hat{p}+1,\hat{q}}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} & \quad d = (\hat{p} = \hat{r} \wedge \hat{p}+1 \leq \hat{q}). \\
& \qquad f(h_{\hat{p}}(\tilde{q}),\tilde{s}), \varepsilon, d\rangle, & \\
& f(h_{\hat{p}_1,\hat{p}_2}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{q}_1,\hat{q}_2}(\tilde{q}),\tilde{s}) \rightsquigarrow & \text{where} \\
& \quad \langle\langle f(h_{\hat{p}_2}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} & \quad d_1 = (\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2+1 \leq \hat{q}_2), \\
& \qquad f(h_{\hat{p}_2}(\tilde{q}), h_{\hat{p}_2+1,\hat{q}_2}(\tilde{q}),\tilde{s}), \varepsilon, d_1\rangle, & \\
& \quad \langle f(h_{\hat{q}_2}(\tilde{t}), h_{\hat{q}_2+1,\hat{p}_2}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} & \quad d_2 = (\hat{p}_1 = \hat{q}_1 \wedge \hat{q}_2+1 \leq \hat{p}_2), \\
& \qquad f(h_{\hat{q}_2}(\tilde{q}),\tilde{s}), \varepsilon, d_2\rangle & \\
& \quad \langle f(h_{\hat{q}_2}(\tilde{t}),\tilde{r}) \stackrel{?}{=}_{\emptyset} f(h_{\hat{q}_2}(\tilde{q}),\tilde{s}), \varepsilon, d_3\rangle\rangle, & \quad d_3 = (\hat{p}_1 = \hat{p}_2 \wedge \hat{q}_1 = \hat{q}_2).
\end{array}
$$

Fig. 3.7. Functional pattern simplification rules for QP-$\emptyset$-unification. $\tilde{r}$, $\tilde{s}$, $\tilde{t}$ and $\tilde{q}$ are possibly empty sequences of terms. $f \in \mathcal{FFLEX}$.

3.4.5 Tree Generation

Projection, transformation and pattern simplification can be seen as single steps in a tree generation process. Each node of the tree is labeled either with a QP-$\emptyset$-unification problem, $\top$ or $\bot$. The edges of the tree are labeled by substitutions and linear Diophantine constraints. The nodes labeled with $\top$ or $\bot$ are terminal nodes. The nodes labeled with QP-$\emptyset$-unification problems are non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Let $\mathfrak{Q}$ be a QP-$\emptyset$-unification problem attached to a non-terminal node and $c_{\mathfrak{Q}}$ be a conjunction of linear Diophantine constraints attached to the edges in the branch, from the root of the tree till the current node. First, we check whether $c_{\mathfrak{Q}}$ is satisfiable. If it is not, we replace $\mathfrak{Q}$ with the new label $\bot$. Otherwise we proceed as follows: If we can decide whether $\mathfrak{Q}$ is not unifiable, then we replace $\mathfrak{Q}$ with the new label $\bot$. Otherwise we apply projection, transformation or pattern simplification on $\mathfrak{Q}$ and get $\langle\langle\mathfrak{S}_1, \sigma_1, d_1\rangle, \ldots, \langle\mathfrak{S}_n, \sigma_n, d_n\rangle\rangle$. Then the node $\mathfrak{Q}$ has n children, labeled respectively with $\mathfrak{S}_1, \ldots, \mathfrak{S}_n$ and the edge to the $\mathfrak{S}_i$ node is labeled with σ_i and d_i $(1 \leq i \leq n)$. The set $\{\sigma_1, \ldots, \sigma_n\}$ is denoted by $sub(\mathfrak{Q})$. The set $\{d_1, \ldots, d_n\}$ is denoted by $con(\mathfrak{Q})$.

Satisfiability of $c_{\mathfrak{Q}}$ can be checked by one of known algorithms for solving linear Diophantine equational and inequational systems, e.g. [AC95].

We design the general P-$\emptyset$-unification procedure for $\mathfrak{Q}_\emptyset$ as a breadth first (level by level) tree generation process. We label the root of the tree with $\mathfrak{Q}_\emptyset$ (zero level). First level nodes (the children of the root) of the tree are obtained from the original problem by projection[1]. Starting from the second level, we apply only a transformation or pattern simplification step to a QP-$\emptyset$-unification problem of each node, thus getting new successor nodes. The branch which ends with a node labeled by $\top$ is called a successful branch. The branch which ends with a node labeled by $\bot$ is a failed branch. All QP-$\emptyset$-unification problems attached to the nodes of a successful branch are in fact P-$\emptyset$-unification problems.

For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node and attach the obtained substitution to the node. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N.

Similarly, for each node in the tree, we take a conjunction of the linear Diophantine constraints displayed on the edges of the branch which leads to this node and attach the obtained constraint to the node. The linear Diophantine constraint $ldc(\mathfrak{Q}_\emptyset)$ is attached to the root. For a node N, the constraint attached to N in such a way is called the associated constraint of N.

[1] Starting from the first level, the unification problems attached to the nodes in the tree might not be P-$\emptyset$-unification problems, but they are, of course, QP-$\emptyset$-unification problems.

We call the tree a P-$\emptyset$-unification tree for the problem $\mathfrak{Q}_\emptyset$ and denote it $putree(\mathfrak{Q}_\emptyset)$.

Let $\Delta(\mathfrak{Q}_\emptyset)$ be the set of all P-substitution/constraint pairs associated with the $\top$ nodes. Then we define the set $\Sigma(\mathfrak{Q}_\emptyset)$ as follows:

$$\Sigma(\mathfrak{Q}_\emptyset) = \{\sigma \mid \text{ there exists } \{\theta, d\} \in \Delta(\mathfrak{Q}_\emptyset) \text{ and } \mu \text{ such that } \mu \text{ is a positive integer solution of } d \text{ and } \sigma = (\theta \circ \mu)|_{\mathrm{vars}(\mathfrak{U})}\}.$$

Next, our goal is to show that $\Sigma(\mathfrak{Q}_\emptyset)$ is the minimal complete set of unifiers of $\mathfrak{Q}_\emptyset$.

First, we show correctness.

Lemma 3.4.1. *$\Sigma(\mathfrak{Q}_\emptyset)$ is a set of unifiers of $\mathfrak{Q}_\emptyset$.*

Proof. Let $\sigma \in \Sigma(\mathfrak{Q}_\emptyset)$. Then there exists $\{\theta, d\} \in \Delta(\mathfrak{Q}_\emptyset)$ and μ such that μ is a positive integer solution of d and $\sigma = (\theta \circ \mu)|_{vars(\mathfrak{U})}$. We show that σ is a unifier of $\mathfrak{Q}_\emptyset$. Recall that $\mathfrak{Q}_\emptyset$ has a form of P-equation $t_1 \stackrel{?}{=}_\emptyset t_2$.

Since μ is a positive integer solution of d and the variables in d are all the index variables from $\mathfrak{Q}_\emptyset$ and $cod(\theta)$, we have:

- $xvars(\mathfrak{Q}_\emptyset) \subseteq dom(\sigma)$;
- for each $\hat{x} \in dom(\sigma)$, $\hat{x}\sigma$ is a positive integer;
- each pattern which occurs in $cod(\sigma)$ is explicit.

Moreover, since $t_1\theta \doteq_\emptyset t_2\theta$, we have $t_1\theta\circ\mu \doteq_\emptyset t_2\theta\circ\mu$ and, thus, $t_1\sigma \doteq_\emptyset t_2\sigma$. Now, by Definition 3.4.10 we get that σ is a unifier of $\mathfrak{Q}_\emptyset$. □

Second, we prove completeness.

Lemma 3.4.2. *For every unifier ϕ of a general P-$\emptyset$-unification problem $\mathfrak{Q}_\emptyset$ there exists a branch β in $putree(\mathfrak{Q}_\emptyset)$ with the following property: if θ and d are respectively a substitution and a constraint attached to the same node in β, then there exists a positive integer solution τ of the constraint d such that $\theta \circ \tau \ll_\emptyset^{vars(\mathfrak{Q}_\emptyset)} \phi$.*

Proof. We define β recursively. We search for β among successful branches of the tree $putree(\mathfrak{Q}_\emptyset)$, therefore all the nodes in β will have P-$\emptyset$-unification problems attached.

First, let the root of the tree, labeled with $\mathfrak{Q}_\emptyset$, be in β. The P-substitution associated with the node $\mathfrak{Q}_\emptyset$ is ε. The constraint associated with $\mathfrak{Q}_\emptyset$ is $ldc(\mathfrak{Q}_\emptyset)$. By Proposition 3.4.1, $\phi^=$ contains a positive integer solution of $ldc(\mathfrak{Q}_\emptyset)$. Let τ be the positive integer solution of $ldc(\mathfrak{Q}_\emptyset)$ contained in $\phi^=$. Let $\lambda = \phi \backslash \tau$. Then $\phi = \tau \circ \lambda = \varepsilon \circ \tau \circ \lambda$, from which we can easily derive that $\varepsilon \circ \tau \ll_\emptyset^{vars(\mathfrak{Q}_\emptyset)} \phi$.

Next, let the first level node of β be that successor $\mathfrak{P}$ of $\mathfrak{Q}_\emptyset$, for which the associated P-substitution $\theta = \{\overline{x} \leftarrow \ \mid \overline{x} \leftarrow \in \phi \text{ and } \overline{x} \in vars(\mathfrak{Q}_\emptyset)\}$. The constraint associated with $\mathfrak{P}$ is $ldc(\mathfrak{Q}_\emptyset)$. Let τ be the positive integer solution of $ldc(\mathfrak{Q}_\emptyset)$ contained in $\phi^=$. Let $\lambda = \phi\backslash\theta$ and $\psi = \lambda\backslash\tau$. Then $\lambda = \tau \circ \psi$ and $\phi = \theta \circ \lambda = \theta \circ \tau \circ \psi$. Therefore

- for all $\overline{x} \in vars(\mathfrak{Q}_\emptyset)$,
  - $\overline{x} \leftarrow\ \notin \psi$;
  - there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \dots, t_n$, $\overline{x}\,\theta \circ \tau \circ \psi = s_1, \dots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or a P-equality $t_i \doteq_\emptyset s_i$ holds;
- for all $x \in vars(\mathfrak{Q}_\emptyset)$, the P-equality $x\phi \doteq_\emptyset x\theta \circ \tau \circ \psi$ holds,
- for all $\hat{x} \in vars(\mathfrak{Q}_\emptyset)$, $\hat{x}\phi = \hat{x}\theta \circ \tau \circ \psi$,

which implies that $\theta \circ \tau \ll_\emptyset^{vars(\mathfrak{Q}_\emptyset)} \phi$.

Let now a node in $putree(\mathfrak{Q}_\emptyset)$, labeled with a P-$\emptyset$-unification problem $\mathfrak{P}_n$, be in β. It means that for a P-substitution θ_n and a constraint d_n, associated with $\mathfrak{P}_n$, there exists a positive integer solution τ_n of d_n such that $\theta_n \circ \tau_n \ll_\emptyset^{vars(\mathfrak{Q}_\emptyset)} \phi$. Then there exists a P-substitution λ such that

- for all $\overline{x} \in vars(\mathfrak{Q}_\emptyset)$,
  - the binding
    $$\overline{x} \leftarrow\ \notin \lambda; \tag{3.8}$$
  - there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \dots, t_n$, $\overline{x}\theta_n \circ \tau_n \circ \lambda = s_1, \dots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality
    $$t_i \doteq_\emptyset s_i \tag{3.9}$$
    holds;
- for all $x \in vars(\mathfrak{Q}_\emptyset)$, the P-equality
  $$x\phi \doteq_\emptyset x\theta_n \circ \tau_n \circ \lambda \tag{3.10}$$
  holds;
- for all $\hat{x} \in vars(\mathfrak{Q}_\emptyset)$,
  $$\hat{x}\phi = \hat{x}\theta_n \circ \tau_n \circ \lambda. \tag{3.11}$$

We can assume without loss of generality that

$$dom(\tau_n) \cap dom(\lambda) = \emptyset. \tag{3.12}$$

If there exists a successor $\mathfrak{P}_{n+1}$ of $\mathfrak{P}_n$ in $putree(\mathfrak{Q}_\emptyset)$ with the associated P-substitution θ_{n+1} and the associated constraint d_{n+1} such that for a positive integer solution τ_{n+1} of d_{n+1} we have $\theta_{n+1} \circ \tau_{n+1} \ll_\emptyset^{vars(\mathfrak{Q}_\emptyset)} \phi$, then we can include $\mathfrak{P}_{n+1}$ into β and, thus, we will have that for all $\mathfrak{P} \in \beta$, if θ is a

P-substitution and d is a constraint associated with $\mathfrak{P}$, then there exists a positive integer solution τ of d such that $\theta \circ \tau \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$.

Thus, the problem of constructing β is reduced to the problem of finding the successor $\mathfrak{P}_{n+1}$ of $\mathfrak{P}_n$ with the property that for the P-substitution θ_{n+1} and the constraint d_{n+1} associated with $\mathfrak{P}_{n+1}$, there exists a positive integer solution τ_{n+1} of d_{n+1} such that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$. We show how to find such a $\mathfrak{P}_{n+1}$.

The P-$\emptyset$-unification problem $\mathfrak{P}_n$ can have one of the following four forms:

1. $\mathfrak{P}_n$ is a pair of identical P-terms.
2. $\mathfrak{P}_n$ is a pair of individual variables.
3. $\mathfrak{P}_n$ is a pair of an individual variable and non-variable term.
4. $\mathfrak{P}_n$ is a pair of non-variable terms.

We consider each of them separately:

1. There is only one possible choice: $\mathfrak{P}_{n+1}$ is $\top$ with the associated substitution $\theta_{n+1} = \theta_n \circ \varepsilon = \theta_n$ and associated linear Diophantine constraint d_{n+1} which has a form $d_n \wedge true$ and, thus, $d_{n+1} = d_n$. Let τ_{n+1} be τ_n. Then $\theta_{n+1} \circ \tau_{n+1} = \theta_n \circ \tau_n \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$.

2-3. Let $\mathfrak{P}_n$ be $x \stackrel{?}{\doteq}_\emptyset t$, t being either an individual variable or a non-variable non-pattern term. The substitution $\theta_n \circ \lambda$ is a unifier of $\mathfrak{Q}_\emptyset$. The unification problems $\mathfrak{Q}_\emptyset\theta_n$ and $\mathfrak{P}_n$ have exactly the same set of unifiers, because $\mathfrak{P}_n$ can be obtained from $\mathfrak{Q}_\emptyset\theta_n$ by iterated deletion of identical first arguments in both sides of $\mathfrak{Q}_\emptyset\theta_n$. Therefore, we have that λ is a unifier of $\mathfrak{P}_n$. To obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ we choose the substitution $\mu = \{x \leftarrow t\}$ and the constraint $true$. Let θ_{n+1} be the substitution $\theta_n \circ \mu$, d_{n+1} be the constraint $(d_n \wedge true) = d_n$ and τ_{n+1} be τ_n. Then from (3.8), (3.9), (3.10), (3.11) and the facts that $\mu \circ \lambda = \lambda$ and $\tau_n \circ \lambda = \lambda \circ \tau_n$ we get
   - for all $\overline{x} \in vars(\mathfrak{Q}_\emptyset)$,
     - $\overline{x} \leftarrow\ \notin \lambda$;
     - there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \dots, t_n$, $\overline{x}\theta_n \circ \tau_n \circ \lambda = \overline{x}\theta_n \circ \lambda \circ \tau_n = \overline{x}\theta_n \circ \mu \circ \lambda \circ \tau_n = \overline{x}\theta_{n+1} \circ \lambda \circ \tau_n = \overline{x}\theta_{n+1} \circ \tau_{n+1} \circ \lambda = s_1, \dots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality $t_i \doteq_\emptyset s_i$ holds;
   - for all $x \in vars(\mathfrak{Q}_\emptyset)$, we have $x\phi \doteq_\emptyset x\theta_n \circ \tau_n \circ \lambda \doteq_\emptyset x\theta_n \circ \lambda \circ \tau_n \doteq_\emptyset x\theta_n \circ \mu \circ \lambda \circ \tau_n \doteq_\emptyset x\theta_{n+1} \circ \lambda \circ \tau_n \doteq_\emptyset x\theta_{n+1} \circ \tau_{n+1} \circ \lambda$;
   - for all $\hat{x} \in vars(\mathfrak{Q}_\emptyset)$, we have $\hat{x}\phi = \hat{x}\theta_n \circ \tau_n \circ \lambda = \hat{x}\theta_n \circ \lambda \circ \tau_n = \hat{x}\theta_n \circ \mu \circ \lambda \circ \tau_n = \hat{x}\theta_{n+1} \circ \lambda \circ \tau_n = \hat{x}\theta_{n+1} \circ \tau_{n+1} \circ \lambda$;

   Thus, $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$.

4. Let $\mathfrak{P}_n$ have the form $g(t_1, \overline{t}) \stackrel{?}{\doteq}_\emptyset g(s_1, \overline{s})$, where $\overline{t}$ and $\overline{s}$ are (possibly empty) sequences of terms. We can have the following 9 cases with respect to t_1 and s_1:
   - 4.1. t_1 and s_1 are identical.
   - 4.2. t_1 and s_1 are individual variables.

4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable non-pattern P-term.
4.4. t_1 and s_1 are sequence variables.
4.5. One of t_1 and s_1 is a sequence variable, the other is neither a sequence variable nor a pattern.
4.6. t_1 and s_1 are non-variable non-pattern P-terms.
4.7. One of t_1 and s_1 is an individual variable, the other is a pattern.
4.8. One of t_1 and s_1 is a sequence variable, the other is a pattern.
4.9. One of t_1 and s_1 is a non-variable non-pattern P-term, the other is a pattern.
4.10. t_1 and s_1 are patterns.

We consider each of these cases.

4.1-4.6. These cases can be proved in the same way as the corresponding cases from the proof of Lemma 3.3.2, taking into account the fact that $d_{n+1} = (d_n \wedge true) = d_n$.

4.7. Suppose t_1 is an individual variable x and s_1 is a pattern $h_{\hat{p},\hat{q}}(\overline{t})$. Let us define P-substitutions μ, η and ν and a constraint d as follows:
- $\mu = \{x \leftarrow h_{\hat{q}}(\overline{t})\}$, η is a positive integer solution of $\hat{p} = \hat{q}$, $\nu = \{v \leftarrow v\lambda \,|\, v \neq x, v \notin dom(\eta), v \neq v\lambda\}$ and d is $\hat{p} = \hat{q}$, if λ is a unifier of a P-$\emptyset$-unification problem $f(x) \doteq_{\emptyset} f(h_{\hat{q}}(\overline{t}))$ and $\lambda^{=}$ contains a positive integer solution of $\hat{p} = \hat{q}$;
- $\mu = \{x \leftarrow h_{\hat{p}}(\overline{t})\}$, η is a positive integer solution of $\hat{p} + 1 \leq \hat{q}$, $\nu = \{v \leftarrow v\lambda \,|\, v \neq x, v \notin dom(\eta), v \neq v\lambda\}$ and d is $\hat{p} + 1 \leq \hat{q}$, if λ is a unifier of a P-$\emptyset$-unification problem $f(x)\lambda \doteq_{\emptyset} f(h_{\hat{p}}(\overline{t}))\lambda$ and $\lambda^{=}$ contains a positive integer solution of $\hat{p} + 1 \leq \hat{q}$;

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.
We obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ by the P-substitution μ (since λ is a unifier of $\mathfrak{P}_n$, these two cases for μ are the only possibilities to get $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$). Therefore, $\theta_{n+1} = \theta_n \circ \mu$. Since $d_{n+1} = (d_n \wedge d)$, $\tau_n \circ \eta$ is a positive solution of d_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then, using the facts $\mu \circ \eta \circ \nu = \lambda$ and $\mu \circ \tau_n = \tau_n \circ \mu$ we get

$$\theta_n \circ \tau_n \circ \lambda = \theta_n \circ \tau_n \circ \mu \circ \eta \circ \nu = \theta_n \circ \mu \circ \tau_n \circ \eta \circ \nu = \theta_{n+1} \circ \tau_{n+1} \circ \nu. \quad (3.13)$$

On the other hand, for all $\overline{x} \in vars(\mathfrak{Q}_{\emptyset})$,

$$\overline{x} \leftarrow \; \notin \mu \circ \eta. \quad (3.14)$$

From (3.14) and (3.8), by Definition 2.1.11 we get for all $\overline{x} \in vars(\mathfrak{Q}_{\emptyset})$

$$\overline{x} \leftarrow \; \notin \nu. \quad (3.15)$$

From (3.9) and (3.13) we get that for all $\overline{x} \in vars(\mathfrak{Q}_{\emptyset})$ there exist terms $t_1, \ldots, t_n$, $s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_{n+1} \circ$

$\tau_{n+1} \circ \nu = s_1, \dots, s_n$ and for each $1 \le i \le n$, either t_i and s_i are the same sequence variables or the P-equality

$$t_i \doteq_{\emptyset} s_i; \tag{3.16}$$

holds.

From 3.10 and 3.13 we get that for all $x \in vars(\mathfrak{Q}_{\emptyset})$ the P-equality

$$x\phi \doteq_{\emptyset} x\theta_n \circ \tau_n \circ \lambda \doteq_{\emptyset} x\theta_{n+1} \circ \tau_{n+1} \circ \nu. \tag{3.17}$$

holds.

From 3.11 and 3.13 we get that for all $\hat{x} \in vars(\mathfrak{Q}_{\emptyset})$

$$\hat{x}\phi = \hat{x}\theta_n \circ \tau_n \circ \lambda = \hat{x}\theta_{n+1} \circ \tau_{n+1} \circ \nu. \tag{3.18}$$

From 3.15, 3.16, 3.17 and 3.18, by Definition 3.4.11 we get $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_{\emptyset})} \phi$.

4.8. Suppose t_1 is a sequence variable $\overline{x}$ and s_1 is a pattern $h_{\hat{p},\hat{q}}(\overline{t})$. Let us define P-substitutions μ, η and ν and a constraint d as follows:
- $\mu = \{\overline{x} \leftarrow h_{\hat{p},\hat{x}}(\overline{t})\}$, η is a positive integer solution of $\hat{p} \le \hat{x} \wedge \hat{x}+1 \le \hat{q}$, $\nu = \{v \leftarrow v\lambda \,|\, v \ne \overline{x},\ v \notin dom(\eta),\ v \ne v\lambda\}$ and d is $\hat{p} \le \hat{x} \wedge \hat{x}+1 \le \hat{q}$, if λ is a unifier of P-$\emptyset$-unification problem $f(\overline{x}) \doteq_{\emptyset} f(h_{\hat{p},\hat{x}}(\overline{t}))$ and $\lambda^{=}$ contains a positive integer solution of $\hat{p} \le \hat{x} \wedge \hat{x}+1 \le \hat{q}$;
- $\mu = \{\overline{x} \leftarrow h_{\hat{p},\hat{q}}(\overline{t})\}$, $\eta = \varepsilon$, $\nu = \{v \leftarrow v\lambda \,|\, v \ne \overline{x},\ v \ne v\lambda\}$ and d is $true$, if λ is a unifier of P-$\emptyset$-unification problem $f(\overline{x}) \doteq_{\emptyset} f(h_{\hat{p},\hat{q}}(\overline{t}))$;
- $\mu = \{\overline{x} \leftarrow f(h_{\hat{p},\hat{q}}(\overline{t}), \overline{x}\}$, $\eta = \varepsilon$, $\nu = \{\overline{x} \leftarrow \overline{r}\} \circ \{v \leftarrow v\lambda \,|\, v \ne \overline{x},\ v \ne v\lambda\}$ and d is $true$, if there exists a non-empty sequence of P-terms $\overline{r}$ such that λ is a unifier of P-$\emptyset$-unification problem $f(\overline{x}) \doteq_{\emptyset} f(h_{\hat{p},\hat{q}}(\overline{t}), \overline{r})$,

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

We obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ by the P-substitution μ (since λ is a unifier of $\mathfrak{P}_n$, these three cases for μ are the only possibilities to get $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$). Therefore, $\theta_{n+1} = \theta_n \circ \mu$. Since $d_{n+1} = (d_n \wedge d)$, $\tau_n \circ \eta$ is a positive solution of d_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then with the same reasoning as in the case 4.7. we conclude that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_{\emptyset})} \phi$.

4.9. Suppose t_1 is a P-term $h_{\hat{r}}(\overline{t_1})$ and s_1 is a pattern $h_{\hat{p},\hat{q}}(\overline{t_2})$. Let us define P-substitutions μ, η and ν and a constraint d as follows:
- $\mu = \varepsilon$, η is a positive integer solution of $\hat{p} = \hat{r} \wedge \hat{p}+1 \le \hat{q}$, $\nu = \{v \leftarrow v\lambda \,|\, v \notin dom(\eta),\ v \ne v\lambda\}$ and d is $\hat{p} = \hat{r} \wedge \hat{p}+1 \le \hat{q}$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{\hat{p}}(\overline{t_1})) \doteq_{\emptyset} f(h_{\hat{p}}(\overline{t_2}))$ and $\lambda^{=}$ contains a positive integer solution of $\hat{p} = \hat{r} \wedge \hat{p}+1 \le \hat{q}$,

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

We obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ by the P-substitution μ (since λ is a unifier of $\mathfrak{P}_n$, this is the only possibility to get $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$). Therefore, $\theta_{n+1} = \theta_n \circ \mu = \theta_n$. Since $d_{n+1} = (d_n \wedge d)$, $\tau_n \circ \eta$ is a positive solution of d_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then with the same reasoning as in the case 4.7. we conclude that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$.

4.10. Suppose t_1 is a pattern $h_{\hat{p}_1,\hat{p}_2}(\overline{t_1})$ and s_1 is a pattern $h_{\hat{q}_1,\hat{q}_2}(\overline{t_2})$. Let us define P-substitutions μ, η and ν and a constraint d as follows:
- $\mu = \varepsilon$, η is a positive integer solution of $\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 + 1 \leq \hat{q}_2$, $\nu = \lambda \backslash \eta$, d is $\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 + 1 \leq \hat{q}_2$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{\hat{p}_2}(\overline{t_1})) \doteq_\emptyset f(h_{\hat{p}_2}(\overline{t_2}))$ and $\lambda^=$ contains a positive integer solution of $\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 + 1 \leq \hat{q}_2$;
- $\mu = \varepsilon$, η is a positive integer solution of $\hat{p}_1 = \hat{q}_1 \wedge \hat{q}_2 + 1 \leq \hat{p}_2$, $\nu = \lambda \backslash \eta$, d is $\hat{p}_1 = \hat{q}_1 \wedge \hat{q}_2 + 1 \leq \hat{p}_2$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{\hat{q}_2}(\overline{t_1})) \doteq_\emptyset f(h_{\hat{q}_2}(\overline{t_2}))$ and $\lambda^=$ contains a positive integer solution of $\hat{p}_1 = \hat{q}_1 \wedge \hat{q}_2 + 1 \leq \hat{p}_2$;
- $\mu = \varepsilon$, η is a positive integer solution of $\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 = \hat{q}_2$, $\nu = \lambda \backslash \eta$, d is $\hat{p}_1 = \hat{q}_1 \wedge \hat{p}_2 = \hat{q}_2$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{\hat{p}_2}(\overline{t_1})) \doteq_\emptyset f(h_{\hat{p}_2}(\overline{t_2}))$ and $\lambda^=$ contains a positive integer solution of $\hat{p}_1 = \hat{p}_2 \wedge \hat{q}_1 = \hat{q}_2$;

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

We obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ by the P-substitution μ and constraint d (since λ is a unifier of $\mathfrak{P}_n$, these three cases are the only possibilities to get $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$). Since $d_{n+1} = (d_n \wedge d)$, $\tau_n \circ \eta$ is a positive solution of d_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then, with the same reasoning as in the case 4.7, we conclude that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$.

Thus, for all possible forms of $\mathfrak{P}_n$ we found its successor $\mathfrak{P}_{n+1}$ such that for the P-substitution θ_{n+1} and the constraint d_{n+1} associated with $\mathfrak{P}_{n+1}$, there exists a positive integer solution τ_{n+1} of d_{n+1} such that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{vars(\mathfrak{Q}_\emptyset)} \phi$. It finishes the proof. □

Theorem 3.4.2. *$\Sigma(\mathfrak{Q}_\emptyset)$ is a complete set of unifiers for $\mathfrak{Q}_\emptyset$.*

Proof. The theorem follows from Lemma 3.4.2 and Lemma 3.4.1 by the definition of Σ. □

Now, we prove minimality. As in the case of $\emptyset$-unification, we prove a stronger statement: $\Sigma(\mathfrak{Q}_\emptyset)$ is a disjoint set of unifiers of $\mathfrak{Q}_\emptyset$, where disjointness for a set of P-substitutions is defined as follows:

Definition 3.4.12 (Disjoint Set of P-Substitutions). *A set of P-substitutions Σ is called disjoint modulo E with respect to a set of variables V iff for all θ, $\sigma \in \Sigma$, if there exist P-substitutions λ_1, λ_2 such that*

- *for all sequence variables $\overline{x} \in V$,*
  - $\overline{x} \leftarrow \notin \lambda_1$,

  - $\overline{x} \leftarrow \notin \lambda_2$,
  - *there exist P-terms* $t_1, \ldots, t_n, s_1, \ldots, s_n$*,* $n \geq 0$ *such that* $\overline{x}\theta \circ \lambda_1 = t_1, \ldots, t_n$*,* $\overline{x}\sigma \circ \lambda_2 = s_1, \ldots, s_n$ *and for all* $1 \leq i \leq n$*, either* t_i *and* s_i *are the same sequence variables or the P-equality* $t_i \doteq_E s_i$ *holds;*
- *for all individual variables* $x \in V$*, the P-equality*
  - $x\theta \circ \lambda_1 \doteq_E x\sigma \circ \lambda_2$ *holds,*
- *for all index variables* $\hat{x} \in V$*,*
  - $\hat{x}\theta \circ \lambda_1 \doteq_E \hat{x}\sigma \circ \lambda_2$.

then $\theta = \sigma$*.*

The result analogous to Lemma 3.3.1 holds for *P*-substitutions: disjointness implies minimality.

Theorem 3.4.3. $\Sigma(\mathfrak{Q}_\emptyset)$ *is a disjoint set of unifiers for* $\mathfrak{Q}_\emptyset$ *with respect to the set of variables of* $\mathfrak{Q}_\emptyset$*.*

Proof. Let V be the set of variables of $\mathfrak{Q}_\emptyset$ and σ_1 and σ_2 be two substitutions from $\Sigma(\mathfrak{Q}_\emptyset)$. We consider the following cases:

1. σ_1 and σ_2 are attached to the same $\top$ node and differ from each other by a positive integer solution of the constraint attached to the same node. Clearly, in this case the set $\{\sigma_1, \sigma_2\}$ is disjoint with respect to V.
2. σ_1 and σ_2 are obtained from two different $\top$ nodes to which the same constraint is attached. If σ_1 and σ_2 contain different positive integer solutions of the constraint, then disjointness is clear. Otherwise, let σ_1 be $\theta_1 \circ \mu$ and σ_2 be $\theta_2 \circ \mu$, where θ_1 and θ_2 are substitutions associated with the $\top$ nodes σ_1 and σ_2 are obtained from, and μ is a positive integer solution of the constraint. Then disjointness of $\{\sigma_1, \sigma_2\}$ follows from disjointness $\{\theta_1, \theta_2\}$ with respect to V, which itself follows from Theorem 3.3.2.
3. σ_1 and σ_2 are obtained from two different $\top$ nodes to which different constraints are attached. Then by the construction of the unification tree, these two constraints do not have a common solution, which implies disjointness of $\{\sigma_1, \sigma_2\}$ with respect to V.

The fact that every pair of *P*-substitutions from $\Sigma(\mathfrak{Q}_\emptyset)$ is disjoint with respect to V implies disjointness of $\Sigma(\mathfrak{Q}_\emptyset)$ with respect to V. □

The following theorem summarizes the results of this section:

Theorem 3.4.4. *Let* $\mathfrak{Q}_\emptyset$ *be a general P-*$\emptyset$*-unification problem. Then*

$$\Sigma(\mathfrak{Q}_\emptyset) = mcu_\emptyset(\mathfrak{Q}_\emptyset).$$

Proof. By Theorem 3.4.2 and Theorem 3.4.3. □

3.5 Extension with Sequence Variables in Expressions with Fixed Arity Heads

In this section we consider E-unification problems written in extended syntax – allowing sequence variables to appear as arguments in expressions with fixed arity heads. Expressions written in such an extended syntax can be considered as abbreviations for terms or equations written in the standard syntax (defined in Section 2.1) and can be used to have more compact representation. For instance, for a ternary function symbols f, instead of referring to three terms $f(a, x, y)$, $f(x, a, y)$ and $f(x, y, a)$ explicitly, we could use an expression in the extended syntax $f(\overline{x}, a, \overline{y})$. But note that not all the expressions in the extended syntax abbreviate terms or equations written in the standard syntax. For instance, the expression $f(\overline{x}, \overline{x})$ for a ternary f has no counterparts written in the standard syntax.

Intuitively, an E-unification problem written in the extended syntax represents a set of E-unification problems written in the standard syntax. Therefore, to solve an extended E-unification problem, first we need to reduce them to the corresponding set E-unification problems in standard syntax and then apply known unification procedures on each element of the set.

We discuss these issues more formally. We defined the alphabet $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ in Section 2.1. Now we introduce two new notions[2]:

Definition 3.5.1 (Quasi-Term). *The set of quasi-terms (over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$) is the smallest set of strings over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ that satisfies the following conditions:*

- *If $v \in \mathcal{IV} \cup \mathcal{SV}$ then v is a quasi-term.*
- *If $c \in \mathcal{CONST}$ then c is a quasi-term.*
- *If $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$ and $t_1, \ldots, t_n$ $(n \geq 0)$ are quasi-terms, then $f(t_1, \ldots, t_n)$ is a quasi-term.*

Definition 3.5.2 (Quasi-Equation). *The set of quasi-equations (over the alphabet $(\mathcal{V}, \mathcal{C}, \mathcal{P})$) is the smallest set of strings over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ that satisfies the following condition:*

- *If $t_1, \ldots, t_n$ $(n \geq 0)$ are quasi-terms over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$, then $\doteq(t_1, \ldots, t_n)$ is a quasi-equation over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$.*

We extend the notion of instance on quasi-terms and quasi-equation. We are interested in quasi-terms and quasi-equation which have a counterpart written in the standard syntax in the sense that at least one instance of the quasi-term (quasi-equation) is a term (equation) over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$.

Definition 3.5.3 (Extended Terms and Equations). *A quasi-term t (resp. a quasi-equation q) is called an extended term (extended equation) over the alphabet $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ iff there exist a substitution σ such that $t\sigma$ is a term over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$ ($q\sigma$ is an equation over $(\mathcal{V}, \mathcal{C}, \mathcal{P})$).*

[2] Although the results below in this section are formulated for E-unification, they remain valid for pattern E-unification as well.

In order to decide whether a quasi-term (quasi-equation) is an extended term (extended equation), we associate a system of linear Diophantine equations to them and show that if the system has a natural solution, then quasi-term (quasi-equation) is an extended term (extended equation).

Definition 3.5.4 (System of Linear Diophantine Equations Associated with a Quasi-Term). *For a quasi-term t, the associated system of linear Diophantine equations $slde(t)$ is defined in the following way:*

– *If $t \in \mathcal{IV} \cup \mathcal{SV} \cup \mathcal{CONST}$ then $slde(t)$ is empty.*
– *If t is $f(t_1, \ldots, t_l)$, where $f \in \mathcal{FFIX}$ and $\mathcal{AR}(f) = n$, then $slde(t)$ is*

$$k_1 X_1 + \ldots + k_s X_s = n - (l - (k_1 + \ldots + k_s)) \wedge slde(t_1) \wedge \ldots \wedge slde(t_l)$$

where $k_1, \ldots, k_s$ ($s \geq 0$) are numbers of occurrences of all distinct sequence variables among $t_1, \ldots, t_l$, the variables $X_1, \ldots, X_s$ are metavariables corresponding to the sequence variables and $\wedge$ denotes conjunction on the meta level.
– *If t is $f(t_1, \ldots, t_l)$, where $f \in \mathcal{FFLEX}$, then $slde(t)$ is*

$$slde(t_1) \wedge \ldots \wedge slde(t_l).$$

Definition 3.5.5 (System of Linear Diophantine Equations Associated with a Quasi-Equation). *For a quasi-equation q, the associated system of linear Diophantine equations $slde(q)$ is defined in the following way:*

– *If q is $\doteq (t_1, \ldots, t_l)$, then $slde(q)$ is*

$$k_1 X_1 + \ldots + k_s X_s = 2 - (l - (k_1 + \ldots + k_s)) \wedge slde(t_1) \wedge slde(t_2)$$

where $k_1, \ldots, k_s$ ($s \geq 0$) are numbers of occurrences of all distinct sequence variables among $t_1, \ldots, t_l$, the variables $X_1, \ldots, X_s$ are metavariables corresponding to the sequence variables and $\wedge$ denotes conjunction on the meta level.

Theorem 3.5.1. *A quasi-term t (quasi-equation q) is an extended term (extended equation) if $slde(t)$ ($slde(q)$) has a natural solution.*

Proof. We prove the theorem for a quasi-term t. For a quasi-equation the proof is the same. Let $\overline{x_1}, \ldots, \overline{x_s}$ be all sequence variables of t, occurring as arguments of subterms with fixed arity head and $X_1 = n_1, \ldots, X_s = n_s$ be a natural solution of $slde(t)$. Then we assemble the substitution σ as follows: $\sigma = \{\overline{x_1} \leftarrow z_1^1 \ldots, z_{n_1}^1, \ldots, \overline{x_s} \leftarrow z_1^s \ldots, z_{n_s}^s\}$, where all z-s are new distinct individual variables. Then by the construction of the system $slde(t)$, no sequence variable occurs as an argument of a subterm of $t\sigma$ with fixed arity head and for all such subterms, number of arguments coincide with the arity of its head. Therefore, by Definition 3.5.3, t is an extended term. □

It is clear that an extended E-unification problem is decidable iff E-unification problem is decidable.

We design a unification procedure for extended E-unification problem as a tree generation process with three basic steps: arity completion, projection and transformation. Projection and transformation are those of E-unification. As for the arity completion, we define it as follows:

Let $\overline{x_1}, \ldots, \overline{x_k}$ be all sequence variables occurring in the unification problem $\mathfrak{U}$ as arguments of terms with fixed arity head and let $slde(\mathfrak{U})$ be the system of linear Diophantine equations associated with $\mathfrak{U}$, with the variables $X_1, \ldots, X_k$, corresponding to $\overline{x_1}, \ldots, \overline{x_k}$. Each solution $X_1 = n_1, \ldots, X_k = n_k$ of $slde(\mathfrak{U})$ generates an *arity completing substitution* $\{\overline{x_1} \leftarrow z_1^1, \ldots, z_{n_1}^1, \ldots, \overline{x_k} \leftarrow z_1^k, \ldots, z_{n_k}^k\}$, where $z_1^1, \ldots, z_{n_k}^k$ are new distinct individual variables.

We denote the set of all arity completing substitutions of $\mathfrak{U}$ by $\Upsilon(\mathfrak{U})$ and define the arity completion rule on Figure 3.8.

Arity Completion:	$\mathfrak{U} \rightsquigarrow \bot$	if $\Upsilon(\mathfrak{U}) = \emptyset$
	$\mathfrak{U} \rightsquigarrow \langle\langle \mathfrak{U}\sigma_1, \sigma_1\rangle, \ldots, \langle \mathfrak{U}\sigma_n, \sigma_n\rangle$	if $\Upsilon(\mathfrak{U}) = \{\sigma_1, \ldots, \sigma_n\}$

Fig. 3.8. Arity completion rule.

Example 3.5.1. Let $\mathfrak{U}$ be $f(\overline{x}, a, \overline{y}) \overset{?}{=}_\emptyset f(h(\overline{y}), x, a, y)$, where $\mathcal{AR}(f) = 3$. Then $slde(\mathfrak{U})$ is $X + Y = 2$ and $\Upsilon(\mathfrak{U}) = \{\{\overline{x} \leftarrow z_1, z_2,\ \overline{y} \leftarrow\}, \{\overline{x} \leftarrow z_1,\ \overline{y} \leftarrow z_2\}, \{\overline{x} \leftarrow,\ \overline{y} \leftarrow z_1, z_2\}\}$.

The extended E-unification procedure is a tree generation process. The root of the tree is labeled with $\mathfrak{U}$, first level nodes of are successors of $\mathfrak{U}$ with arity completion, and the corresponding edges are labeled with arity completing substitutions. After completing arity we apply $\emptyset$-unification procedure to each of the nodes on the second level.

3.6 Implementation and Examples

We have implemented the extended $\emptyset$-unification procedure in *Mathematica* [Wol99]. As a decision procedure for being a term and on the arity completion phase of the procedure we use the Omega package [APR01] to solve linear Diophantine equations over naturals. In the tree generation process for the efficiency reasons we did not implement the decision procedure for unifiability but put the bound on the depth of the tree which can be changed by the user – the idea which was used in the system for combining equational unification algorithms em UniMoK [KR99]. Also, we added a cycle detector as an option, which makes the procedure terminate in case of infinite cyclic success. Last sequence variable occurrence check is implemented as well.

The unification package is incorporated into the *Theorema* system as a part of the *Theorema* Equational Prover. *Theorema* is being developed in RISC-Linz under the direction of Bruno Buchberger. The system aims at providing one coherent environment for working mathematician and supports the following phases ([Win01]):

– inventing new concepts,
– experimenting with the new concepts by doing computations,
– proving properties of the new concepts observed during the computations
– implementing algorithms based on proven properties,
– publishing scientific results covering, for instance, the theoretical basis of a new method, the implementation, and also examples of applying the method on concrete data.

For a general outline on *Theorema* we refer to [BJK+97], [Buc97], [BDJ+00a], [BDJ+00b], [Win01]. Here we would like just to emphasize the fact it is probably the only system being able to handle solving and proving equations which involve sequence variables and flexible arity symbols.

The examples below illustrate the usage of the unification package. In the *Theorema* applications `•var` and `•seq` are *Theorema* tags respectively for a variable and a sequence. Thus, `•var[x]` denotes an individual variable `x` and `•var[•[seq[x]]` denotes a sequence variable $\overline{\mathtt{x}}$.

The input specification of the main function `UnificationSV` of the unification package is the following:

$$\mathtt{UnificationSV}[\mathtt{term}_1, \mathtt{term}_2, \mathtt{options}]$$

with $\mathtt{term}_1$ and $\mathtt{term}_2$ terms with sequence variables and flexible arity symbols. The function tries to unify $\mathtt{term}_1$ and $\mathtt{term}_2$ is the free theory with sequence variables and flexible arity symbols. Options can be one of the following:

– `Verbose` $\to$ `B`, where `B` is either `True` or `False`. When the option is set to `True`, answers are printed immediately as they are computed, `False` switches off printing. Default value of `B` is `True`.
– `CycleDetect` $\to$ `B`, where `B` is either `True` or `False`. The option switches on/off the cycle detector. Default value of `B` is `False` (i.e. the cycle detector is switched off).
– `TreeDepth` $\to$ `N`, where `N` is a non-negative integer or ∞. The option sets the maximum depth of the unification tree. The unification procedure terminates when the depth of the tree exceeds `N`. Default value of `N` is set to 10000.
– `MaximumAnswers` $\to$ `N`, where `N` is a non-negative integer or ∞. The option sets the maximum number of answers. The unification procedure terminates when the number of computed answers is `N`. Default value: ∞.

- `LastSeqVarOccCheck` $\rightarrow$ `B`, where `B` is either `True` or `False`. The option switches on/off the last sequence variable occurrence check. Default value of `B` is `True` (i.e. the checking is switched on).
- `ArityList` $\rightarrow$ `L`, where `L` is a list of pairs of the form {`f, n`}, with `f` an identifier and `n` a non-negative integer. The option specifies arities for fixed arity function symbols. It is used in the extension which allows sequence variables to occur as arguments of terms with fixed arity head. Default value of `L` – the empty list {}.

This is the standard call, with default option values:

In[1]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[**y**]], **f**[•**var**[•**seq**[x]]]],
f[**a**, •**var**[•**seq**[x]], **f**[**b**, •**var**[•**seq**[**y**]]]]]

Answer 1
$\{x \rightarrow a, \bar{x} \rightarrow b, \bar{y} \rightarrow \text{Sequence}[]\}$
Answer 2
$\{x \rightarrow a, \bar{x} \rightarrow \text{Sequence}[b, \bar{x}], \bar{y} \rightarrow \bar{x}\}$
The procedure terminated

Out[1]= $\{\{x \rightarrow a, \bar{x} \rightarrow b, \bar{y} \rightarrow \text{Sequence}[]\}, \{x \rightarrow a, \bar{x} \rightarrow \text{Sequence}[b, \bar{x}], \bar{y} \rightarrow \bar{x}\}\}$

If the option is set to `verbose` $\rightarrow$ `False` then the answers will not be printed in the run-time:

In[2]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[**y**]], **f**[•**var**[•**seq**[x]]]],
f[**a**, •**var**[•**seq**[x]], **f**[**b**, •**var**[•**seq**[**y**]]]]]

Out[2]= $\{\{x \rightarrow a, \bar{x} \rightarrow b, \bar{y} \rightarrow \text{Sequence}[]\}, \{x \rightarrow a, \bar{x} \rightarrow \text{Sequence}[b, \bar{x}], \bar{y} \rightarrow \bar{x}\}\}$

The following unification problem has an infinite minimal complete set of unifiers:

In[3]:= **UnificationSV**[**f**[**a**, •**var**[•**seq**[x]]], **f**[•**var**[•**seq**[x]], **a**]]

Answer 1
$\{\bar{x} \rightarrow \text{Sequence}[]\}$
Answer 2
$\{\bar{x} \rightarrow a\}$
Answer 3
$\{\bar{x} \rightarrow \text{Sequence}[a, a]\}$
Answer 4
$\{\bar{x} \rightarrow \text{Sequence}[a, a, a]\}$
Answer 5
$\{\bar{x} \rightarrow \text{Sequence}[a, a, a, a]\}$
Answer 6
$\{\bar{x} \rightarrow \text{Sequence}[a, a, a, a, a]\}$

```
Answer 7
{x̄ → Sequence[a, a, a, a, a, a]}
Out[3]= $Aborted
```

If we switch on the cycle detector, the computation will terminate:

```
In[4]:= UnificationSV[f[a, •var[•seq[x]]],
          f[•var[•seq[x]], a], CycleDetect → True]
Answer 1
{subst[$0, {x̄ → Sequence[]}, $648], subst[$648, {}, $650]}
Answer 2
{subst[$0, {}, $647], cycle[$647, {x̄ → Sequence[a, x̄]}, $0],
 subst[$647, {x̄ → a}, $649]}
The procedure terminated
Out[4]= {{subst[$0, {x̄ → Sequence[]}, $648], subst[$648, {}, $650]},
          {subst[$0, {}, $647], cycle[$647, {x̄ → Sequence[a, x̄]}, $0],
            subst[$647, {x̄ → a}, $649]}}
```

Each element in the answer sets is a term with three arguments: The second argument is a substitution, the first argument is the ID of the substitution and the third argument is the ID of another substitution which can be composed with the current one in order to get an explicit representation of a unifier. For example, the unifier $\{\bar{x} \rightarrow \texttt{Sequence}[\mathsf{a}, \mathsf{a}, \mathsf{a}]\}$ can be computed from the second answer: starting from the substitution with ID=$0, the sequence of substitutions to be composed is generated as follows:

1. ID – $0, substitution – {}, next ID – $647.
2. ID – $647, substitution – $\{\bar{x} \rightarrow \texttt{Sequence}[\mathsf{a}, \bar{x}]\}$, next ID – $0.
3. ID – $0, substitution – {}, next ID – $647.
4. ID – $647, substitution – $\{\bar{x} \rightarrow \texttt{Sequence}[\mathsf{a}, \bar{x}]\}$, next ID – $0.
5. ID – $0, substitution – {}, next ID – $647.
6. ID – $647, substitution – $\{\bar{x} \rightarrow \mathsf{a}]\}$, next ID – $649.

Since there is no substitution with the ID $649, we stop here and compose the substitutions, which gives $\bar{x} \rightarrow \texttt{Sequence}[\mathsf{a}, \mathsf{a}, \mathsf{a}]\}$.

The unification procedure performs the last sequence variable occurrence check by default. This option can be switched off by setting the option `LastSeqVarOccCheck → False`. The following unification problem is an instance of the infinitely failing unification problem – its unification tree contains infinitely many failing nodes. If the last sequence variable occurrence check is not performed, then the procedure does not terminate (recall that we have not implemented the decision procedure, which would force the unification procedure to terminate in such cases):

```
In[5]:= UnificationSV[f[b, •var[•seq[x]]], f[b, a, •var[•seq[x]]],
          LastSeqVarOccCheck → False]
```

Out[5]= $Aborted

Running the unification procedure with the default mode, we would easily discover that the set of solutions of the problem is empty:

In[6]:= **UnificationSV**[**f**[**b**, •**var**[•**seq**[x]]], **f**[**b**, **a**, •**var**[•**seq**[x]]]]

The procedure terminated

Out[6]= {}

The following unification problem is another example with an infinite set of solutions:

In[7]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[**y**]], •**var**[•**seq**[x]]],
f[**a**, •**var**[•**seq**[x]], **b**, •**var**[•**seq**[**y**]]]]

Answer 1

{x → a, $\bar{x}$ → sequence[]}

Answer 2

{x → a, $\bar{x}$ → sequence[], $\bar{y}$ → sequence[]}

Answer 3

{x → a, $\bar{x}$ → b, $\bar{y}$ → sequence[]}

Answer 4

{x → a, $\bar{x}$ → b, $\bar{y}$ → b}

Answer 5

{x → a, $\bar{x}$ → sequence[b, $\bar{x}$], $\bar{y}$ → $\bar{x}$}

Answer 6

{x → a, $\bar{x}$ → sequence[b, b], $\bar{y}$ → sequence[]}

Answer 7

{x → a, $\bar{x}$ → b, $\bar{y}$ → sequence[b, b]}

Answer 8

{x → a, $\bar{x}$ → sequence[b, b, b], $\bar{y}$ → sequence[]}

Answer 9

{x → a, $\bar{x}$ → b, $\bar{y}$ → sequence[b, b, b]}

Answer 10

{x → a, $\bar{x}$ → sequence[b, b], $\bar{y}$ → sequence[b, b]}

Answer 11

{x → a, $\bar{x}$ → sequence[b, $\bar{x}$], $\bar{y}$ → sequence[$\bar{x}$, b, $\bar{x}$]}

Answer 12

{x → a, $\bar{x}$ → sequence[b, b, b], $\bar{y}$ → b}

Answer 13

{x → a, $\bar{x}$ → sequence[b, $\bar{x}$, b, $\bar{x}$], $\bar{y}$ → $\bar{x}$}

Answer 14

{x → a, $\bar{x}$ → sequence[b, b, b, b], $\bar{y}$ → sequence[]}

Answer 15

{x → a, $\bar{x}$ → b, $\bar{y}$ → sequence[b, b, b, b]}

Answer 16

$\{x \to a, \bar{x} \to$ sequence$[b, b, b], \bar{y} \to$ sequence$[b, b, b]\}$
Answer 17
$\{x \to a, \bar{x} \to$ sequence$[b, b, b, b], \bar{y} \to$ sequence$[b, b]\}$
Answer 18
$\{x \to a, \bar{x} \to$ sequence$[b, b, b, b, b], \bar{y} \to$ sequence$[]\}$
Answer 19
$\{x \to a, \bar{x} \to b, \bar{y} \to$ sequence$[b, b, b, b, b]\}$
Answer 20
$\{x \to a, \bar{x} \to$ sequence$[b, b, b, b], \bar{y} \to$ sequence$[b, b, b, b]\}$
Out[7]= $Aborted

We can stop execution when, for instance, seven answers are computed:

In[8]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[y]], •**var**[•**seq**[x]]],
f[**a**, •**var**[•**seq**[x]], **b**, •**var**[•**seq**[y]]], **MaximumAnswers** → **7**]
Answer 1
$\{x \to a, \bar{x} \to$ sequence$[]\}$
Answer 2
$\{x \to a, \bar{x} \to$ sequence$[], \bar{y} \to$ sequence$[]\}$
Answer 3
$\{x \to a, \bar{x} \to b, \bar{y} \to$ sequence$[]\}$
Answer 4
$\{x \to a, \bar{x} \to b, \bar{y} \to b\}$
Answer 5
$\{x \to a, \bar{x} \to$ sequence$[b, \bar{x}], \bar{y} \to \bar{x}\}$
Answer 6
$\{x \to a, \bar{x} \to$ sequence$[b, b],$
$\bar{y} \to$ sequence$[]\}$
Answer 7
$\{x \to a, \bar{x} \to b, \bar{y} \to$ sequence$[b, b]\}$
The procedure terminated because of the maximum number of answers option
Out[8]= $\{\{x \to a, \bar{x} \to$ sequence$[]\},$
$\{x \to a, \bar{x} \to$ sequence$[], \bar{y} \to$ sequence$[]\},$
$\{x \to a, \bar{x} \to b, \bar{y} \to$ sequence$[]\}, \{x \to a, \bar{x} \to b, \bar{y} \to b\},$
$\{x \to a, \bar{x} \to$ sequence$[b, \bar{x}], \bar{y} \to \bar{x}\},$
$\{x \to a, \bar{x} \to$ sequence$[b, b], \bar{y} \to$ sequence$[]\},$
$\{x \to a, \bar{x} \to b, \bar{y} \to$ sequence$[b, b]\}\}$

Alternatively, execution can terminate when the depth of the unification tree exceeds the number specified by the option TreeDepth:

In[9]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[y]], •**var**[•**seq**[x]]],
f[**a**, •**var**[•**seq**[x]], **b**, •**var**[•**seq**[y]]], **TreeDepth** → **20**]

Answer 1
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[]\}$
Answer 2
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[], \bar{y} \to \mathtt{sequence}[]\}$
Answer 3
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{b}, \bar{y} \to \mathtt{sequence}[]\}$
The procedure terminated because of the
tree depth option

Out[9]= $\{\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[]\}, \{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[], \bar{y} \to \mathtt{sequence}[]\}, \{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{b}, \bar{y} \to \mathtt{sequence}[]\}\}$

In the example below the arity of `f` is set to be 5. The new individual variables `NewVar1, NewVar2, NewVar3, NewVar4` are introduced at the arity completion step.

In[10]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[**y**]], •**var**[•**seq**[**x**]]], **f**[**a**, •**var**[•**seq**[x]], **b**, •**var**[•**seq**[**y**]]], **ArityList** → {{**f**, **5**}}]

Answer 1
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[], \bar{y} \to \mathtt{sequence}[\mathtt{NewVar1}, \mathtt{NewVar2}, \mathtt{NewVar3}]\}$
Answer 2
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[\mathtt{b}, \mathtt{NewVar4}], \bar{y} \to \mathtt{NewVar4}\}$
Answer 3
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{b}, \bar{y} \to \mathtt{sequence}[\mathtt{b}, \mathtt{b}]\}$
Answer 4
$\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[\mathtt{b}, \mathtt{b}, \mathtt{b}], \bar{y} \to \mathtt{sequence}[]\}$
The procedure terminated

Out[10]= $\{\{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[], \bar{y} \to \mathtt{sequence}[\mathtt{NewVar1}, \mathtt{NewVar2}, \mathtt{NewVar3}]\}, \{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[\mathtt{b}, \mathtt{NewVar4}], \bar{y} \to \mathtt{NewVar4}\}, \{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{b}, \bar{y} \to \mathtt{sequence}[\mathtt{b}, \mathtt{b}]\}, \{\mathtt{x} \to \mathtt{a}, \bar{x} \to \mathtt{sequence}[\mathtt{b}, \mathtt{b}, \mathtt{b}], \bar{y} \to \mathtt{sequence}[]\}\}$

If in the same unification problem we require the arity of `f` to be 1, we get an error – arity conflict:

In[11]:= **UnificationSV**[**f**[•**var**[x], **b**, •**var**[•**seq**[**y**]], •**var**[•**seq**[**x**]]], **f**[**a**, •**var**[•**seq**[x]], **b**, •**var**[•**seq**[**y**]]], {**ArityList** → {{**f**, **1**}}}]

The input is not formed correctly. Arity conflict.

Besides using the unification package in Theorema, it can be used to enhance Mathematica solving capabilities, in particular, the Solve function of Mathematica. Solve has a rich arsenal of methods to solve polynomial and radical equations, equations involving trigonometric or hyperbolic functions, exponentials and logarithms. The following example shows, for instance, how a radical equation is solved:

In[12]:= $\mathbf{Solve}\left[\mathbf{x^{1/3}} + \sqrt{\mathbf{x}} == \mathbf{1}, \mathbf{x}\right]$

Out[12]= $\left\{\left\{\mathtt{x} \to -\frac{2}{3} - \frac{11}{3}\left(\frac{2}{101 + 15\sqrt{69}}\right)^{1/3} + \frac{1}{3}\left(\frac{1}{2}\left(101 + 15\sqrt{69}\right)\right)^{1/3}\right\}\right\}$

However, it is unable to solve a symbolic equation like $f(x, y) = f(a, b)$:

In[13]:= $\mathbf{Solve}[\mathbf{f}[\mathbf{x}, \mathbf{y}] == \mathbf{f}[\mathbf{a}, \mathbf{b}], \{\mathbf{x}, \mathbf{y}\}]$
Solve:: "dinv": The expression `f[x, y]` involves unknowns in more than one argument, so inverse functions cannot be used
Out[13]= `Solve[f[x, y] == f[a, b], {x, y}]`

Also, Solve can not deal with equations involving sequence variables.

On the basis of the unification package, we implemented a function called SolveExtended, which has all the power of Solve and, in addition, deals with symbolic equations which can be solved using unification methods. An equation like $f(x, y) = f(a, b)$ becomes a trivial problem for SolveExtended:

In[14]:= $\mathbf{SolveExtended}[\mathbf{f}[\mathbf{x}, \mathbf{y}] == \mathbf{f}[\mathbf{a}, \mathbf{b}], \{\{\mathbf{x}, \mathbf{y}\}, \{\}\}]$

```
Answer 1
{x → a, y → b}
The procedure terminated
```

Out[14]= $\{\{\mathtt{x} \to \mathtt{a}, \mathtt{y} \to \mathtt{b}\}\}$

The function is able to solve (a system of) equations involving sequence variables as well. In the following example x is an individual variable and X and Y are sequence variables:

In[15]:= $\mathbf{SolveExtended}[\mathbf{f}[\mathbf{x}, \mathbf{b}, \mathbf{Y}, \mathbf{f}[\mathbf{X}]] == \mathbf{f}[\mathbf{a}, \mathbf{X}, \mathbf{f}[\mathbf{b}, \mathbf{Y}]], \{\{\mathbf{x}\}, \{\mathbf{X}, \mathbf{Y}\}\}]$

```
Answer 1
{x → a, X → b, Y → Sequence[]}
Answer 2
{x → a, X → Sequence[b, X], Y → X}
The procedure terminated
```

Out[15]= $\{\{\mathtt{x} \to \mathtt{a}, \mathtt{X} \to \mathtt{b}, \mathtt{Y} \to \mathsf{Sequence}[]\}, \{\mathtt{x} \to \mathtt{a}, \mathtt{X} \to \mathsf{Sequence}[\mathtt{b}, \mathtt{X}], \mathtt{Y} \to \mathtt{X}\}\}$

All the problems Solve deals with can also be solved by SolveExtended, e.g.:

In[16]:= $\mathbf{SolveExtended}\left[\mathbf{x^{1/3}} + \sqrt{\mathbf{x}} == \mathbf{1}, \mathbf{x}\right]$

$$\text{Out}[16]= \left\{\left\{\mathtt{x} \to -\frac{2}{3} - \frac{11}{3}\left(\frac{2}{101 + 15\sqrt{69}}\right)^{1/3} + \frac{1}{3}\left(\frac{1}{2}\left(101 + 15\sqrt{69}\right)\right)^{1/3}\right\}\right\}$$

Options of `SolveExtended` are those of `UnificationSV` and `Solve`.

We found another application of the unification package in computing minimal matches for non-commutative Gröbner basis procedure [Mor85]. We show it on the following example: let p and q be non-commutative polynomials $a * a * b * a + a * b$ and $a * b * a * a + b * a$. In order to find S-polynomials of p and q one needs to compute all minimal matches between the leading monomials $a * a * b * a$ and $a * b * a * a$, which, in fact, is a problem of solving equations (for instance, of the form $a * a * b * a * R1 = L2 * a * b * a * a$ for the variables $R1$ and $L2$) in a free semigroup. Solving equations in a free semigroup is a particular case of unification with sequence variables and flexible arity symbols. Even more, the equations we need to solve to find the matches, belong to a case when the unification procedure terminates. The example below demonstrates how the function MinimalMatches, based on the unification package, computes matches for $a * a * b * a$ and $a * b * a * a$:

In[17]:= **MinimalMatches[a ∗ a ∗ b ∗ a, a ∗ b ∗ a ∗ a]**

$$\begin{aligned}\text{Out}[17]= \{&\{\mathtt{a*a*b*a*R1, L2*a*b*a*a}, \{\mathtt{L2} \to \mathtt{a}, \mathtt{R1} \to \mathtt{a}\}\},\\ &\{\mathtt{a*a*b*a*R1, L2*a*b*a*a}, \{\mathtt{L2} \to \mathtt{a*a*b}, \mathtt{R1} \to \mathtt{b*a*a}\}\},\\ &\{\mathtt{L1*a*a*b*a, a*b*a*a*R2}, \{\mathtt{L1} \to \mathtt{a*b}, \mathtt{R2} \to \mathtt{b*a}\}\},\\ &\{\mathtt{L1*a*a*b*a, a*b*a*a*R2}, \{\mathtt{L1} \to \mathtt{a*b*a}, \mathtt{R2} \to \mathtt{a*b*a}\}\}\}\end{aligned}$$

4. Proving in the Free Theory

In this chapter a reduction ordering on the set of terms with sequence variables and flexible arity symbols is defined. The ordering is total on ground terms if the precedence on the signature is total. It plays an important role in showing that the basic equational superposition with ordering and equality constraints is a refutationally complete proving method for the free theory with sequence variables and flexible arity symbols.

The ordering is extended to pattern-terms. Under this extension basic equational superposition with ordering and equality constraints can be used as an inference rule for the extension of the free theory with patterns.

In the special case of the theory, where sequence variables are allowed to occupy only the last argument position in terms, unfailing completion can be used as a refutationally complete proving method.

4.1 Preliminaries

In this section we introduce main basic tools from rewriting. For a detailed introduction in this topic we refer to [DJ90], [Klo92], [BN98] and [DP01].

Let $\mathcal{F} \subseteq \mathcal{CONST} \cup \mathcal{FFIX} \cup \mathcal{FFLEX}$ and $\mathcal{X} \subseteq \mathcal{IV} \cup \mathcal{SV}$. The set of terms over $\mathcal{F}$ and $\mathcal{X}$ is denoted by $\mathcal{T}(\mathcal{F}, \mathcal{X})$. The set of ground terms over $\mathcal{F}$ is denoted by $\mathcal{T}(\mathcal{F})$. We call $\mathcal{F}$ a *signature* and assume that it contains at least one constant.

A *position* is a sequence of positive integers. If p is a position and t is a term, then by $t|_p$ we denote *the subterm of t at position p*: we have $t|_\lambda = t$ (where λ denotes the empty sequence) and $f(t_1, \dots, t_n)|_{i.p} = t_i|_p$, if $1 \leq i \leq n$ (and is undefined if $i > n$). We write $t[s]_p$ to denote the term obtained by replacing in t the subterm at position p by the term s.

A multiset over a set S is a function $M : S \to \mathbb{N}$. Let $\mathcal{M}(S)$ denote the set of all multisets over S. Some basic operations and relations on $\mathcal{M}(S)$ are:

- Element: $x \in M$ iff $M(x) > 0$.
- Inclusion: $M \subseteq N$ iff $M(x) \leq N(x)$ for all $x \in S$.
- Union: $M_1 \cup M_2(x) = M_1(x) + M_2(x)$.
- Intersection: $M_1 \cap M_2(x) = min(M_1(x), M_2(x))$.
- Difference: $(M \setminus N)(x) = max(M_1(x) - M_2(x), 0)$.

We use a set-like notation: $M = \{a, a, b\}$ denotes $M(a) = 2$, $M(b) = 1$ and $M(x) = 0$ for $x \neq a$ and $x \neq b$.

If $\rightarrow$ is a binary relation, then $\leftarrow$ is its inverse, $\leftrightarrow$ is its symmetric closure, $\leftrightarrow^+$ is its transitive closure and $\leftrightarrow^*$ is its reflexive-transitive closure. We write $s \rightarrow^! t$ if $s \rightarrow^* t$ and there is not t' such that $t \rightarrow t'$. Then t is called *irreducible* and a *normal form* of s (with respect to $\rightarrow$). The relation $\rightarrow$ is *well-founded* or *terminating* if there exists no infinite sequence $s_1 \rightarrow s_2 \rightarrow \ldots$ and it is *confluent* or *Church-Rosser* if $\leftarrow^* \circ \rightarrow^* \subseteq \rightarrow^* \circ \leftarrow^*$. It is locally confluent if $\leftarrow \circ \rightarrow \subseteq \rightarrow^* \circ \leftarrow^*$. By Newman's lemma, terminating locally-confluent relations are confluent. A relation $\rightarrow$ is *ground confluent* iff for all ground terms r, if $r \rightarrow^* s$ and $r \rightarrow^* t$ then $s \downarrow t$. A relation $\rightarrow$ on terms is *monotonic* if $s \rightarrow t$ implies $u[s]_p \rightarrow u[t]_p$ for all positions p and terms $s, t, u \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$.

A rewrite rule is an ordered pair (s, t), written $s \rightarrow t$, where $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$. A set of rewrite rules R is a rewrite system. The rewrite relation with R on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, denoted $\rightarrow_R$, is the smallest monotonic relation such that $l\sigma \rightarrow_R r\sigma$ for all $l \rightarrow r \in R$ and all σ. If $s \rightarrow_R t$ then we say that s *rewrites into* t with R. We say that R is terminating, confluent, etc. if $\rightarrow_R$ is. A rewrite system R is called convergent if it is confluent and terminating. It is not difficult to see that for a convergent R, every term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ has a unique normal form with respect to $\rightarrow_R$, denoted by $t \downarrow_R$, and $s \doteq t$ is a logical consequence of R (where R is seen as a set of equations) if and only if $s \downarrow_R \doteq t \downarrow_R$.

A (strict partial) ordering $\succ$ is a transitive and irreflexive binary relation.

We say that an ordering $\succ$ on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$

- is *stable* under substitutions if $s \succ t$ implies $s\sigma \succ t\sigma$ for all $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ and σ.
- fulfills the *subterm property* if $u[s]_p \succ s$ for all $s, u \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ and $p \neq \lambda$.
- fulfills the *deletion property* if $f(\ldots s \ldots) \succ f(\ldots \ldots)$ for $f \in \mathcal{F} \cap \mathcal{FFLEX}$ and all $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$.
- is *total* up to a congruence $\cong$ on $\mathcal{T}(\mathcal{F})$ if for all $s, t \in \mathcal{T}(\mathcal{F})$, either $s \cong t$, or $s \succ t$ or $t \succ s$.

A *rewrite ordering* is a monotonic ordering stable under substitutions. A *reduction ordering* is a well-founded rewrite ordering. A *simplification ordering* is a rewrite ordering with the subterm property and the deletion property.

A reduction ordering total on $\mathcal{T}(\mathcal{F})$ is a simplification ordering on $\mathcal{T}(\mathcal{F})$. If $\mathcal{F} \cap \mathcal{FFLEX} = \emptyset$ and $\mathcal{F}$ is finite, then a simplification ordering on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ is a reduction ordering on $\mathcal{T}(\mathcal{F}, \mathcal{X})$. A rewrite system R is terminating if and only if for every $l \rightarrow r \in R$ and for some reduction ordering $\succ$, $l \succ r$ holds.

Given a set of equations E over $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and a reduction ordering $\succ$ on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, we define the set $E_\succ$ of orientable instances of equations in E:

$$E_\succ = \{l\sigma \rightarrow r\sigma \mid l \doteq r \in E \ \text{ and } \ l\sigma \succ r\sigma\}.$$

The rewriting relation $\rightarrow_{E_\succ}$ is defined as follows: $s \rightarrow_{E_\succ} t$ if there exists and equation $l \doteq r$ in E such that $s|_p = l\sigma$, $t = s[r\sigma]_p$ and $l\sigma \succ r\sigma$ for some substitution σ.

4.2 Reduction Ordering for Terms with Sequence Variables and Flexible Arity Symbols

4.2.1 The Ordering $\succ_{mposvm}$

Our goal is to define a reduction ordering on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$. Let us assume that a well-founded ordering $\succ_{\mathcal{F}}$ is given on the set $\mathcal{F}$. Here $\succ_{\mathcal{F}}$ is called the *precedence*. Furthermore, let $\doteq_{mul}$ denote the equality on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ defined as follows: $s \doteq_{mul} t$ iff

– $s, t \in \mathcal{CONST} \cup \mathcal{IV}$ and $s \doteq t$, or
– $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$, $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$ and for all i, $1 \leq i \leq n$, $s_{\pi(i)} \doteq_{mul} t_i$ or $s_{\pi(i)}$ and t_i are the same sequence variable, where π is a permutation of $1, \ldots, n$.

For example, $f(\overline{x}, g(a, \overline{y}, y), b) \doteq_{mul} f(b, \overline{x}, g(\overline{y}, a, y))$.

Let $\succ$ be an ordering on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ and $\cong$ be a congruence on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ compatible with $\succ$. We will need two extensions of $\succ$ with respect to $\cong$: the *lexicographic extension* and the *multiset extension*.

The *lexicographic extension* of $\succ$ with respect to $\cong$ is the relation $\succ^{lex,\cong}$ on n-tuples over $\mathcal{T}(\mathcal{F}, \mathcal{X})$ defined by $\langle s_1, \ldots, s_n \rangle \succ^{lex,\cong} \langle t_1, \ldots, t_n \rangle$, if there exists k, $1 \leq k \leq n$, such that

1. for all i, $1 \leq i \leq k$, $s_i \cong t_i$ or s_i and t_i are the same sequence variable and
2. $s_k \succ t_k$, or $t_k \in svars(s_k)$ and s_i and t_i are not the same sequence variable.

The *multiset extension* of $\succ$ with respect to $\cong$ is the relation $\succ^{mul,\cong}$ on $\mathcal{M}(\mathcal{T}(\mathcal{F}, \mathcal{X}))$ such that for all $M, N \in \mathcal{M}(\mathcal{T}(\mathcal{F}, \mathcal{X}))$ $M \succ^{mul,\cong} N$ iff

1. M and N are not equal up to $\cong$ and
2. for all $n \in N \setminus M$ there exists $m \in M \setminus N$ such that $m \succ n$, or $n \in svars(m)$ and n and m are not the same sequence variable (the operation $\setminus$ is performed modulo $\cong$).

To define a reduction ordering, induced by the precedence, on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ we proceed as follows: as a starting point, we take the multiset path ordering $\succ_{mpo}$ (also called the recursive path ordering without status [KL80], [Der82]), defined on $\mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$, and extend it to a reduction ordering $\succ_{mposvm}$ on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, which is total on ground terms up to $\doteq_{mul}$, when the precedence is total. After that, we extend $\succ_{mposvm}$ to a reduction ordering $\succ_{mposv}$ which is total on ground terms up to $\doteq$, when the precedence is total.

The multiset path ordering on $\mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$ is defined as follows:

Definition 4.2.1. *Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$ and $\succ_{\mathcal{F}}$ be a precedence. Then $s \succ_{mpo} t$ iff*

1. $t \in ivars(s)$ *and* $s \neq t$, *or*
2. $s, t \in \mathcal{CONST}$ *and* $s \succ_{\mathcal{F}} t$, *or*
3. $s \in \mathcal{CONST}$, $t = g(t_1, \ldots, t_n)$, $n \geq 0$, $s \succ_{\mathcal{F}} g$, *and for all* j *with* $1 \leq j \leq n$, $s \succ_{mpo} t_j$, *or*
4. $s = f(s_1, \ldots, s_m)$, $m \geq 0$, $t \in \mathcal{CONST}$, *and either* $f \succ_{\mathcal{F}} t$, *or for some* i, $1 \leq i \leq m$, $s_i \succ_{mpo} t$ *or* $s_i \doteq_{mul} t$, *or*
5. $s = f(s_1, \ldots, s_m)$, $m \geq 0$, $t = g(t_1, \ldots, t_n)$, $n \geq 0$ *and*
   a) $s_i \succ_{mpo} t$ *or* $s_i \doteq_{mul} t$ *for some* i *with* $1 \leq i \leq m$ *or*
   b) $f \succ_{\mathcal{F}} g$, *and for all* j *with* $1 \leq j \leq n$, $s \succ_{mpo} t_j$ *or*
   c) $f = g$ *and* $\{s_1, \ldots, s_m\} \succ_{mpo}^{mul, \doteq_{mul}} \{t_1, \ldots, t_n\}$.

$\succ_{mpo}$ is a simplification ordering. Moreover, it is known that it is a reduction ordering which is total on ground terms up to $=_{mul}$ if the precedence is total ([KL80], [Der82]).

Thus, first we define a binary relation $\succ_{mposvm}$ (*mposvm* stands for <u>m</u>ultiset <u>p</u>ath <u>o</u>rdering with <u>s</u>equence <u>v</u>ariables ground total up to $\doteq_{\underline{m}ul}$).

Definition 4.2.2. *Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ and $\succ_{\mathcal{F}}$ be a precedence. Then $s \succ_{mposvm} t$ iff*

1. $t \in ivars(s)$ *and* $s \neq t$, *or*
2. $s, t \in \mathcal{CONST}$ *and* $s \succ_{\mathcal{F}} t$, *or*
3. $s \in \mathcal{CONST}$, $t = g(t_1, \ldots, t_n)$, $n \geq 0$, $s \succ_{\mathcal{F}} g$, *and for all* j *with* $1 \leq j \leq n$, $s \succ_{mposvm} t_j$, *or*
4. $s = f(s_1, \ldots, s_m)$, $m \geq 0$, $t \in \mathcal{CONST}$, *and either* $f \succ_{\mathcal{F}} t$, *or for some* i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$ *or* $s_i \doteq_{mul} t$, *or*
5. $s = f(s_1, \ldots, s_m)$, $m \geq 0$, $t = g(t_1, \ldots, t_n)$, $n \geq 0$ *and*
   a) $s_i \succ_{mposvm} t$ *or* $s_i \doteq_{mul} t$ *for some* i *with* $1 \leq i \leq m$ *or*
   b) $f \succ_{\mathcal{F}} g$, *and for all* j *with* $1 \leq j \leq n$, *either* $s \succ_{mposvm} t_j$ *or* $t_j \in svars(s)$ *or*
   c) $f = g$, *and*
   $\{s_1, \ldots, s_m\} \succ_{mposvm}^{mul, \simeq_{mul}} \{t_1, \ldots, t_n\}$ *and*
   $\{s_1, \ldots, s_m\} \setminus \mathcal{SV} \succ_{mposvm}^{mul, \doteq_{mul}} \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$.

It follows from the definition that $\succ_{mposvm}$ coincides to $\succ_{mpo}$ on the set $\mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$.

For better readibility we omit $\doteq_{mul}$ from the superscript of $\succ_{mposvm}^{mul, \doteq_{mul}}$ and write $\succ_{mposvm}^{mul}$.

4.2.2 Properties of $\succ_{mposvm}$

In this subsection we prove properties of the relation $\succ_{mposvm}$. First we establish two lemmata which will be used later in the properties.

Lemma 4.2.1. *For any $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, $s \succ_{mposvm} t$ implies $vars(s) \supseteq vars(t)$.*

Proof. We prove the lemma by well-founded induction of the size $l = |s| + |t|$. Assume that the assertion is true for all $k < l$, and prove it for l.

By Definition 4.2.2 we have the following cases:

1. $t \in ivars(s)$ and $s \neq t$. Then $vars(t) = \{t\} \subseteq vars(s)$.
2. $s, t \in \mathcal{CONST}$. Then, obviously, $vars(t) = \emptyset = vars(s)$.
3. $s \in \mathcal{CONST}$, $t = g(t_1, \ldots, t_n)$, $n \geq 0$ and for all j with $1 \leq j \leq n$, $s \succ_{mposvm} t_j$. Then by the induction hypothesis $vars(s) = \emptyset \supseteq vars(t_j)$ for all j, $1 \leq j \leq n$, which implies that $vars(t_j) = \emptyset$ for all j, $1 \leq j \leq n$ and, thus, $vars(t) = \emptyset = vars(s)$.
4. $s = f(s_1, \ldots, s_m)$, $t \in \mathcal{CONST}$. Then $vars(t) = \emptyset \subseteq vars(s)$.
5. $s = f(s_1, \ldots, s_m)$ and $t = g(t_1, \ldots, t_n)$. Then:
   a) $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$ for some i with $1 \leq i \leq m$. Then, if $s_i \succ_{mposvm} t$, by the induction hypothesis we have $vars(t) \subseteq vars(s_i) \subseteq vars(s)$. If $s_i \doteq_{mul} t$, then $vars(t) = vars(s_i) \subseteq vars(s)$.
   b) For all j with $1 \leq j \leq m$, either $s \succ_{mposvm} t_j$ or $t_j \in svars(s)$. If $s \succ_{mposvm} t_j$, then by the induction hypothesis we have $vars(t_j) \subseteq vars(s)$. If $t_j \in svars(s)$, than $vars(t_j) = \{t_j\} \subseteq vars(s)$. Thus, for all j with $1 \leq j \leq m$, $vars(t_j) \subseteq vars(s)$. Therefore $vars(t) = \cup_{j=1}^{m} vars(t_j) \subseteq vars(s)$.
   c) $\{s_1, \ldots, s_m\} \succ_{mposvm}^{mul} \{t_1, \ldots, t_n\}$ and $\{s_1, \ldots, s_m\} \setminus \mathcal{SV} \succ_{mposvm}^{mul} \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$. Then, by the definition of multiset extension and the induction hypothesis, on the one hand, for all $t_i \in \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$ we have $vars(t_i) \subseteq vars(s)$. On the other hand, for all $t_i \in \{t_1, \ldots, t_n\} \cap \mathcal{SV}$, $vars(t_i) = \{t_i\} \subseteq vars(s)$. Therefore, for all $t_i \in \{t_1, \ldots, t_n\}$ we have $vars(t_i) \subseteq vars(s)$, which implies that $vars(t) \subseteq vars(s)$.

□

Lemma 4.2.2. *For all $r, s, t, q \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, if $r \doteq_{mul} s$, $s \succ_{mposvm} t$ and $t \doteq_{mul} q$, then $r \succ_{mposvm} q$.*

Proof. We prove the lemma by well-founded induction of the size $l = |r| + |s| + |t| + |q|$. Assume that the assertion is true for all $k < l$, and prove it for l.

By Definition 4.2.2 we have the following cases:

1. $s, t \in \mathcal{CONST}$ and $s \succ_{\mathcal{F}} t$. Then by definition of $\doteq_{mul}$ $r \doteq s$, $t \doteq q$ and since $\doteq$ is a syntactic equality on $\mathcal{T}(\mathcal{L}) \setminus \mathcal{SV}$, we get $r \succ_{mposvm} q$.
2. $s \in \mathcal{CONST}$, $t = g(t_1, \ldots, t_n)$, $n \geq 0$. Then $r \doteq s$ and $q = g(u_1, \ldots, u_n)$, such that for all j, $1 \leq j \leq n$, $u_{\pi(j)} \doteq_{mul} t_j$ where π is a permutation of $1, \ldots, n$. $r \doteq s$ and $s \succ_{mposvm} t$ imply that $r \succ_{\mathcal{F}} g$, and for all j with $1 \leq j \leq n$, $r \succ_{mposvm} t_j$. By the induction hypothesis, for all j with $1 \leq j \leq n$, $r \succ_{mposvm} u_{\pi(j)}$. Then, by Definition 4.2.2 we get $r \succ_{mposvm} q$.

3. $s = f(s_1,\dots,s_m)$, $m \geq 0$, $t \in \mathcal{CONST}$. Then $t \doteq q$ and $r = f(u_1,\dots,u_m)$, such that for all i, $1 \leq i \leq m$, $u_{\pi(i)} \doteq_{mul} s_i$ or $u_{\pi(i)}$ and s_i are the same sequence variables, where π is a permutation of $1,\dots,m$. From $s \succ_{mposvm} t$ and $t \doteq q$ we get either $f \succ_{\mathcal{F}} q$, or for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} q$ or $s_i \doteq_{mul} q$. Then, for the same i, we get by the induction hypothesis that $u_{\pi(i)} \succ_{mposvm} q$, or by transitivity of $\doteq_{mul}$, $u_{\pi(i)} \doteq_{mul} q$. Therefore, by Definition 4.2.2, $r \succ_{mposvm} q$.
4. $s = f(s_1,\dots,s_m)$, $m \geq 0$, $t \in \mathcal{IV}$. Then from $s \succ_{mposvm} t$ and $t \doteq q$ we get $q \in vars(s)$. Since $r \doteq_{mul} s$, we have $vars(r) = vars(s)$ and $r \neq q$. Therefore, by Definition 4.2.2, $r \succ_{mposvm} q$.
5. $s = f(s_1,\dots,s_m)$, $m \geq 0$, $t = g(t_1,\dots,t_n)$, $n \geq 0$. $r = f(u_1,\dots,u_m)$, such that for all i, $1 \leq i \leq m$, $u_{\pi(i)} \doteq_{mul} s_i$ or $u_{\pi(i)}$ and s_i are the same sequence variables, where π is a permutation of $1,\dots,n$. $q = g(w_1,\dots,w_n)$, such that for all j, $1 \leq j \leq n$, $w_{\tau(j)} \doteq_{mul} t_j$ or $w_{\tau(j)}$ and t_j are the same sequence variables, where τ is a permutation of $1,\dots,m$.
   a) $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$ for some i with $1 \leq i \leq m$. For the same i, $u_{\pi(i)} \doteq_{mul} s_i$. On the other hand, $t \doteq_{mul} q$. Then, if $s_i \succ_{mposvm} t$, by the induction hypothesis, $u_{\pi(i)} \succ_{mposvm} q$. If $s_i \doteq_{mul} t$, by transitivity of $\doteq_{mul}$, $u_{\pi(i)} \doteq_{mul} q$. Therefore, by Definition 4.2.2, $r \succ_{mposvm} q$.
   b) $f \succ_{\mathcal{F}} g$, and for all j with $1 \leq j \leq n$, either $s \succ_{mposvm} t_j$ or $t_j \in svars(s)$. Then we have for all j, $1 \leq j \leq n$, $t_j \doteq_{mul} w_{\tau(j)}$ or t_j and $w_{\tau(j)}$ are the same sequence variables, where τ is a permutation of $1,\dots,m$. On the other hand, $r \doteq_{mul} s$. By the induction hypothesis we get for all j with $1 \leq j \leq n$ that either $r \succ_{mposvm} w_{\tau(j)}$ or $w_{\tau(j)} \in svars(r)$. Therefore, $r \succ_{mposvm} q$.
   c) $f = g$ and $\{s_1,\dots,s_m\} \succ^{mul}_{mposvm} \{t_1,\dots,t_n\}$ and $\{s_1,\dots,s_m\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{t_1,\dots,t_n\} \setminus \mathcal{SV}$. Then, on the one hand, $\{u_1,\dots,u_m\}$ is equal to $\{s_1,\dots,s_m\}$ up to $\doteq_{mul}$ and $\{w_1,\dots,w_n\}$ is equal to $\{t_1,\dots,t_n\}$ up to $\doteq_{mul}$. On the other hand, $\{u_1,\dots,u_m\}\setminus\mathcal{SV}$ is equal to $\{s_1,\dots,s_m\} \setminus \mathcal{SV}$ up to $\doteq_{mul}$ and $\{w_1,\dots,w_n\} \setminus \mathcal{SV}$ is equal to $\{t_1,\dots,t_n\}\setminus\mathcal{SV}$ up to $\doteq_{mul}$. Then, by the definition of multiset extension and by the induction hypothesis, we get $\{u_1,\dots,u_m\} \succ^{mul}_{mposvm} \{w_1,\dots,w_n\}$ and $\{u_1,\dots,u_m\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{w_1,\dots,w_n\} \setminus \mathcal{SV}$. Therefore, $r \succ_{mposvm} q$.

□

Property 4.2.1. *$\succ_{mposvm}$ is transitive.*

Proof. Assume $r \succ_{mposvm} s$ and $s \succ_{mposvm} t$. We prove $r \succ_{mposvm} t$ by well-founded induction on $|r| + |s| + |t|$. Obviously, $r, s, t \notin \mathcal{SV}$, $r \succ_{mposvm} s$ implies $r \notin \mathcal{IV}$ and $s \succ_{mposvm} t$ implies $s \notin \mathcal{IV}$.

First, assume that $t \in \mathcal{IV}$. Then, since $s \succ_{mposvm} t$, by Definition 4.2.2 we have $t \in vars(s)$. By Lemma 4.2.1, $r \succ_{mposvm} s$ implies $vars(r) \supseteq vars(s)$. Therefore $t \in vars(r)$ and by Definition 4.2.2, $r \succ_{mposvm} t$.

Now, assume that $t \notin \mathcal{IV}$. Then we have the following cases for r and s:

1. $s \in \mathcal{CONST}$.
   a) $r \in \mathcal{CONST}$. Then $r \succ_{\mathcal{F}} s$. We have the following alternatives according to the form of t:
      i. $t \in \mathcal{CONST}$. Then $s \succ_{mposvm} t$ implies $s \succ_{\mathcal{F}} t$. By transitivity of $\succ_{\mathcal{F}}$, we get $r \succ_{\mathcal{F}} t$, which implies $r \succ_{mposvm} t$.
      ii. $t \notin \mathcal{CONST}$. Let t have a form $h(t_1, \ldots, t_n)$. Then $s \succ_{mposvm} t$ implies $s \succ_{\mathcal{F}} h$, and for all j with $1 \leq j \leq n$, $s \succ_{mposvm} t_j$. Then by transitivity of $\succ_{\mathcal{F}}$ we get $r \succ_{\mathcal{F}} h$. The induction hypothesis implies for all j with $1 \leq j \leq n$, $r \succ_{mposvm} t_j$. Therefore, $r \succ_{mposvm} t$.
   b) $r \notin \mathcal{CONST}$. Then r has a form $f(r_1, \ldots, r_m)$, $m \geq 0$ and either $f \succ_{\mathcal{F}} s$, or for some i, $1 \leq i \leq m$, $r_i \succ_{mposvm} s$ or $r_i \doteq_{mul} s$. Then we have the following alternatives according to the form of t:
      i. $t \in \mathcal{CONST}$. Then $s \succ_{mposvm} t$ implies $s \succ_{\mathcal{F}} t$. If $f \succ_{\mathcal{F}} s$, by transitivity of $\succ$, we get $f \succ_{\mathcal{F}} t$ and, therefore, $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $r_i \succ_{mposvm} s$, then for the same i, by the induction hypothesis $r_i \succ_{mposvm} t$, which implies $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $r_i \doteq_{mul} s$, the for the same i, by Lemma 4.2.2, we get $r_i \succ_{mposvm} t$, which again implies $r \succ_{mposvm} t$.
      ii. $t \notin \mathcal{CONST}$. Let t have a form $h(t_1, \ldots, t_n)$. Then $s \succ_{mposvm} t$ means that $s \succ_{\mathcal{F}} h$, and for all j with $1 \leq j \leq n$, $s \succ_{mposvm} t_j$. If $f \succ_{\mathcal{F}} s$, by transitivity of $\succ_{\mathcal{F}}$ we get $f \succ_{\mathcal{F}} h$. On the other hand, for all j with $1 \leq j \leq n$, by the induction hypothesis we get $r \succ_{mposvm} t_j$. Which implies $r \succ_{mposvm} t$.
2. $s \notin \mathcal{CONST}$. Let s have a form $s = g(s_1, \ldots, s_m)$, $m \geq 0$.
   a) $r \in \mathcal{CONST}$. Then $r \succ_{\mathcal{F}} g$, and for all j with $1 \leq j \leq m$, $r \succ_{mposvm} s_j$. It means that s is a ground term.
      i. $t \in \mathcal{CONST}$. Then either $g \succ_{\mathcal{F}} t$, or for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$. If $g \succ_{\mathcal{F}} t$, by transitivity of $\succ$ we get $r \succ_{\mathcal{F}} t$, which implies $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$, then by the induction hypothesis applied on $r \succ_{mposvm} s_i$ and $s_i \succ_{mposvm} t$ we get $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $s_i \doteq_{mul} t$, then by Lemma 4.2.2 we again get $r \succ_{mposvm} t$.
      ii. $t \notin \mathcal{CONST}$. Let t have a form $h(t_1, \ldots, t_n)$, $n \geq 0$. Since s is ground, t also must be ground. From $s \succ_{mposvm} t$ we have three cases. In the first case, $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$ for some i with $1 \leq i \leq m$. Then we get $r \succ_{mposvm} t$ by the same reasoning as was used for $t \in \mathcal{CONST}$ above. In the second case, $g \succ_{\mathcal{F}} h$, and for all j with $1 \leq j \leq n$, $s \succ_{mposvm} t_j$. Transitivity of $\succ$ implies that $r \succ_{\mathcal{F}} h$ and the induction hypothesis gives $r \succ_{mposvm} t_j$ for all j with $1 \leq j \leq n$. Therefore, $r \succ_{mposvm} t$. In the third case,

$g = h$ and $\{s_1, \ldots, s_m\} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\}$. Then $r \succ_{\mathcal{F}} h$ and using the definition of multiset extension, Lemma 4.2.2 and the induction hypothesis we get $r \succ_{mposvm} t$.

b) $r \notin \mathcal{CONST}$. Let r have a form $f(r_1, \ldots, r_l)$.

i. $t \in \mathcal{CONST}$. Then either $g \succ_{\mathcal{F}} t$, or for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$. From $r \succ_{mposvm} s$ we have three cases. In the first case, $r_i \succ_{mposvm} s$ or $r_i \doteq_{mul} s$ for some i with $1 \leq i \leq l$. Then we show $r \succ_{mposvm} t$ in the same way as it is done for $r \notin \mathcal{CONST}$, $s \in \mathcal{CONST}$, $t \in \mathcal{CONST}$. In the second case, $f \succ_{\mathcal{F}} g$, and for all j with $1 \leq j \leq m$, either $r \succ_{mposvm} s_j$ or $s_j \in svars(s)$. If $g \succ_{\mathcal{F}} t$, then transitivity of $\succ$ gives $f \succ_{\mathcal{F}} t$ and, thus, $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$, then by the induction hypothesis $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $s_i \doteq_{mul} t$, then by Lemma 4.2.2 we get $r \succ_{mposvm} t$. In the third case, $f = g$, $\{r_1, \ldots, r_l\} \succ^{mul}_{mposvm} \{s_1, \ldots, s_n\}$ and $\{r_1, \ldots, r_l\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{s_1, \ldots, s_n\} \setminus \mathcal{SV}$. If $g \succ_{\mathcal{F}} t$, then we have $f \succ_{\mathcal{F}} t$ and, thus $r \succ_{mposvm} t$. If for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$, then, since the definition of multiset extension implies that there exists a k, $1 \leq k \leq l$ with either $r_k \doteq_{mul} s_i$ or $r_k \succ_{mposvm} s_i$, we get $r_k \succ_{mposvm} t$ either by Lemma 4.2.2 or by the induction hypothesis, which implies $r \succ_{mposvm} t$.

ii. $t \notin \mathcal{CONST}$. Let t have a form $h(t_1, \ldots, t_n)$, $n \geq 0$.
First assume that $r \succ_{mposvm} s$ by the case 5a) of the Definition 4.2.2, i.e. $r_k \succ_{mposvm} t$ or $r_k \doteq_{mul} s$ for some k with $1 \leq k \leq l$. Then $r \succ_{mposvm} t$ in the same way as it is done for $r \notin \mathcal{CONST}$, $s \in \mathcal{CONST}$, $t \in \mathcal{CONST}$.
Next, assume that $s \succ_{mposvm} t$ by the case 5b) of the Definition 4.2.2. Then $g \succ_{\mathcal{F}} h$ and for all j with $1 \leq j \leq n$, either $s \succ_{mposvm} t_j$ or $t_j \in svars(s)$. By the induction hypothesis and Lemma 4.2.1 we get for all j with $1 \leq j \leq n$, either $r \succ_{mposvm} t_j$ or $t_j \in svars(r)$. On the other hand, we have $f \succ_{\mathcal{F}} h$. Therefore, $r \succ_{mposvm} t$.
Now let $r \succ_{mposvm} s$ by the case 5b) of the Definition 4.2.2. Then we have $f \succ_{\mathcal{F}} g$ and for all j with $1 \leq j \leq m$, either $r \succ_{mposvm} s_j$ or $s_j \in svars(r)$. If $s \succ_{mposvm} t$ by the case 5a) of the Definition 4.2.2, then $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$ for some i with $1 \leq i \leq m$. Applying either the induction hypothesis or Lemma 4.2.2 on i we get $r \succ_{mposvm} t$. If $s \succ_{mposvm} t$ by the case 5c), then $g = h$ and $\{s_1, \ldots, s_m\} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\}$, and $\{s_1, \ldots, s_m\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$. Then $f \succ_{\mathcal{F}} h$ and for all k with $1 \leq k \leq n$, $t_k \in \mathcal{SV}$, by Lemma 4.2.1, implies $t_k \in vars(r)$, and $t_k \notin \mathcal{SV}$, by the induction hypothesis and Lemma 4.2.2, implies $r \succ_{mposvm} t_k$. Thus, we get $f \succ_{\mathcal{F}} h$ and

for all k with $1 \leq k \leq n$, $r \succ_{mposvm} t_k$ or $t_k \in svars(r)$, which implies $r \succ_{mposvm} t$.
Now assume that $r \succ_{mposvm} s$ by the case 5c) of the Definition 4.2.2. Then $f = g$, $\{r_1, \ldots, r_l\} \succ^{mul}_{mposvm} \{s_1, \ldots, s_m\}$, and $\{r_1, \ldots, r_l\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{s_1, \ldots, s_m\} \setminus \mathcal{SV}$. If $s \succ_{mposvm} t$ by the case 5a) of the Definition 4.2.2, then $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$ for some i with $1 \leq i \leq m$. For the same i, be the definition of multiset extension, there exists a k, $1 \leq k \leq l$ such that $r_k \succ_{mposvm} s_i$ or $r_k \doteq_{mul} s_i$. By the induction hypothesis, Lemma 4.2.2 and transitivity of $\doteq_{mul}$ we get $r_k \succ_{mposvm} t$ or $r_k \doteq_{mul} t$. This implies $r \succ_{mposvm} t$. If $s \succ_{mposvm} t$ by the case 5c), then $g = h$, $\{s_1, \ldots, s_m\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$ and $\{s_1, \ldots, s_m\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$. Then $f = h$, and by Lemma 4.2.1 and transitivity of $\succ^{mul}_{mposvm}$ (which follows from the induction hypothesis and Lemma 4.2.2) we get $\{r_1, \ldots, r_l\} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\}$ and $\{r_1, \ldots, r_l\} \setminus \mathcal{SV} \succ^{mul}_{mposvm} \{t_1, \ldots, t_n\} \setminus \mathcal{SV}$. Therefore, $r \succ_{mposvm} t$.

□

Property 4.2.2. *$\succ_{mposvm}$ fulfils the subterm property.*

Proof. Since we have already shown transitivity, the subterm property is proved if we succeed in proving that $f(\ldots, s, \ldots) \succ_{mposvm} s$ for all $f \in \mathcal{F}$ and $s \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$. And this property is not hard to show: if $s \in \mathcal{IV}$, then it is an instance of the case 1) of the Definition 4.2.2, otherwise it follows either from the case 4) or from the case 5a) of the Definition 4.2.2, because $s \doteq_{mul} s$. □

Property 4.2.3. *$\succ_{mposvm}$ is stable under substitution.*

Proof. We have to show that $s \succ_{mposvm} t$ implies $s\sigma \succ_{mposvm} t\sigma$ for all $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ and a substitution σ. We use well-founded induction on $|s| + |t|$.

1. $s \succ_{mposvm} t$ by the case 1) of the Definition 4.2.2. Then $t \in ivars(s)$ and $t \neq s$. If $t \in dom(\sigma)$, then $t\sigma \notin \mathcal{SV}$ and $t\sigma$ is a strict subterm of $s\sigma$. By the subterm property, we get $s\sigma \succ_{mposvm} t\sigma$. If $t \notin dom(\sigma)$, then $t\sigma = t \in ivars(s\sigma)$, $t\sigma \neq s\sigma$ and, thus, $s\sigma \succ_{mposvm} t\sigma$.
2. $s \succ_{mposvm} t$ by the case 2) or 3) of the Definition 4.2.2. Then s and t are ground and obviously, $s\sigma \succ_{mposvm} t\sigma$.
3. $s \succ_{mposvm} t$ by the case 4) of the Definition 4.2.2. Then $s = f(s_1, \ldots, s_m)$, $m \geq 0$, $t \in \mathcal{CONST}$, and either $f \succ_{\mathcal{F}} t$, or for some i, $1 \leq i \leq m$, $s_i \succ_{mposvm} t$ or $s_i \doteq_{mul} t$. Then $t\sigma = t$. If $f \succ_{\mathcal{F}} t$, it is clear that $s\sigma \succ_{mposvm} t\sigma$. If $s_i \succ_{mposvm} t$, then by the induction hypothesis $s_i\sigma \succ_{mposvm} t\sigma$. By the subterm property $s\sigma \succ_{mposvm} s_i\sigma$. By transitivity of $\succ_{mposvm}$, $s\sigma \succ_{mposvm} t\sigma$. If $s_i \doteq_{mul} t$, then $s_i\sigma = s_i \doteq_{mul} t = t\sigma$. Therefore, by the subterm property and Lemma 4.2.2 $s\sigma \succ_{mposvm} t\sigma$.

4. $s \succ_{mposvm} t$ by the case 5a) of the Definition 4.2.2. Then the proof proceeds similarly to the previous case.
5. $s \succ_{mposvm} t$ by the case 5b) of the Definition 4.2.2. Then $s = f(s_1, \dots, s_m)$, $m \geq 0$, $t = g(t_1, \dots, t_n)$, $n \geq 0$, $f \succ_{\mathcal{F}} g$, and for all j with $1 \leq j \leq n$, either $s \succ_{mposvm} t_j$ or $t_j \in svars(s)$. Then for an arbitrary but fixed j, if $s \succ_{mposvm} t_j$, then by the induction hypothesis $s\sigma \succ_{mposvm} t_j\sigma$. If $t_j \in svars(s)$ we proceed as follows: if $t_j \in dom(\sigma)$, let the sequence $r_1, \dots, r_k$, $k \geq 0$, be $t_j\sigma$. Then for each r_l, either $s\sigma \succ_{mposvm} r_l$ by the subterm property, or $r_l \in svars(s\sigma)$. If $t_j \notin dom(\sigma)$, then $t_j \in svars(s\sigma)$. Thus, if we denote $t\sigma$ by $g(u_1, \dots, u_k)$, then we have for all i, $1 \leq i \leq k$, $s\sigma \succ_{mposvm} u_i$ or $u_i \in svars(s\sigma)$. Therefore, $s\sigma \succ_{mposvm} t\sigma$.
6. $s \succ_{mposvm} t$ by the case 5c) of the Definition 4.2.2. Then $s = f(s_1, \dots, s_m)$, $m \geq 0$, $t = f(t_1, \dots, t_n)$, $n \geq 0$, $\{s_1, \dots, s_m\} \succ_{mposvm}^{mul} \{t_1, \dots, t_n\}$ and $\{s_1, \dots, s_m\} \setminus \mathcal{SV} \succ_{mposvm}^{mul} \{t_1, \dots, t_n\} \setminus \mathcal{SV}$. Let the set $\{s_1, \dots, s_p\}$ be $\{s_1, \dots, s_m\} \setminus \{t_1, \dots, t_n\}$ and $\{t_1, \dots, t_q\}$ be $\{t_1, \dots, t_n\} \setminus \{s_1, \dots, s_m\}$. Then for all j, $1 \leq j \leq q$, there exists i, $1 \leq i \leq p$, such that $s_i \succ_{mposvm} t_j$ or $t_j \in svars(s_i)$. From $s_i \succ_{mposvm} t_j$ by the induction hypothesis we get $s_i\sigma \succ_{mposvm} t_j\sigma$. From $t_j \in svars(s_i)$ we have that $t_j\sigma$ is either the empty sequence, or a sequence $r_1, \dots, r_k$, $k \geq 1$, of proper subterms of the term $s_i\sigma$. In the last case, for each l, $1 \leq l \leq k$ we have either $r_l \in svars(s_i)$ or (by the subterm property), $s_i\sigma \succ_{mposvm} r_l$. Let $\{s_1\sigma, \dots, s_{p'}\sigma\}$ be $\{s_1\sigma, \dots, s_m\sigma\} \setminus \{t_1\sigma, \dots, t_n\sigma\}$ and $\{t_1\sigma, \dots, t_{q'}\sigma\}$ be $\{t_1\sigma, \dots, t_n\sigma\} \setminus \{s_1\sigma, \dots, s_m\sigma\}$, $p' \leq p$ and $q' \leq q$. Since $\{s_1, \dots, s_m\} \setminus \mathcal{SV} \succ_{mposvm}^{mul} \{t_1, \dots, t_n\} \setminus \mathcal{SV}$, the set $\{s_1\sigma, \dots, s_{p'}\sigma\}$ can not be empty, i.e. $\{s_1\sigma, \dots, s_m\sigma\}$ can not be equal to $\{t_1\sigma, \dots, t_n\sigma\}$ up to $\simeq_{mul}$. Then for all j, $1 \leq j \leq q'$, if $t_j\sigma$ is a single term, there exists i, $1 \leq i \leq p'$, such that either $s_i\sigma \succ_{mposvm} t_j\sigma$ or $t_j\sigma \in svars(s_i\sigma)$, and if $t_j\sigma$ is a sequence of terms $r_1, \dots, r_k$, $k > 1$, then for all l with $1 < l \leq k$ there exists i, $1 \leq i \leq p'$, such that $s_i\sigma \succ_{mposvm} r_l$. It implies $s\sigma \succ_{mposvm} t\sigma$.

□

Note that the stability property would not hold if we would have extended $\succ_{mpo}$ on $\mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ in a straightforward way, i.e. to have the case 5c) of Definition 4.2.2 formulated as follows: $f = g$ and $\{s_1, \dots, s_m\} \succ_{mposvm}^{mul} \{t_1, \dots, t_n\}$. A counterexample is the following: by this definition we have $f(\overline{x}, a) \succ_{mposvm}^{mul} f(a)$, but for $\sigma = \{\overline{x} \leftarrow\}$, $f(\overline{x}, a)\sigma \not\succ_{mposvm}^{mul} f(a)\sigma$.

Property 4.2.4. *$\succ_{mposvm}$ is monotonic.*

Proof. We have to prove that $s \succ_{mposvm} t$ implies $f(s_1, \dots, s_{j-1}, s, s_{j+1}, \dots, s_n) \succ_{mposvm} f(s_1, \dots, s_{j-1}, t, s_{j+1}, \dots, s_n)$ for all $s_1, \dots, s_{j-1}, s_{j+1}, \dots, s_n$. Then we have for all $u \in \{s_1, \dots, s_{j-1}, t, s_{j+1}, \dots, s_n\}$ that $u \in \mathcal{SV}$ implies $u \in vars(f(s_1, \dots, s_{j-1}, s, s_{j+1}, \dots, s_n))$ and $\{s_1, \dots, s_{j-1}, s, s_{j+1}, \dots, s_n\} \setminus \mathcal{SV} \succ_{mposvm} \{s_1, \dots, s_{j-1}, t, s_{j+1}, \dots, s_n\} \setminus \mathcal{SV}$. Then by the case 5c) of Definition 4.2.2, $f(s_1, \ldots, s_{j-1}, s, s_{j+1}, \ldots, s_n) \succ_{mposvm} f(s_1, \ldots, s_{j-1}, t, s_{j+1}, \ldots, s_n)$. □

Property 4.2.5. *$\succ_{mposvm}$ is irreflexive.*

Proof. We have to show that for all $s \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, $s \not\succ_{mposvm} s$. Assume by contradiction that there exists $s \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ with $s \succ_{mposvm} s$. Let σ be a substitution such that $dom(\sigma)$ contains all the sequence variables in s and $cod(\sigma) \cap \mathcal{SV} = \emptyset$. By the stability property, from $s \succ_{mposvm} s$ we get $s\sigma \succ_{mposvm} s\sigma$. But since $s\sigma \in \mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$ and on $\mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$ the relation $\succ_{mposvm}$ coincides with $\succ_{mpo}$, we have $s\sigma \succ_{mpo} s\sigma$ which contradicts irreflexivity of *mpo*. □

$\succ_{mposvm}$ does not fulfil the deletion property: $f(\overline{x}, a) \not\succ_{mposvm} f(a)$. Therefore, $\succ_{mposvm}$ is not a simplification ordering. However, a weaker form of the deletion property holds:

Property 4.2.6. *$\succ_{mposvm}$ fulfils a weak form of deletion property: $f(\ldots, s, \ldots) \succ_{mposvm} f(\ldots\ldots)$ for any $f \in \mathcal{F} \cap \mathcal{FFLEX}$ and $s \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$. Moreover, if $s \in \mathcal{SV}$, then $f(\ldots, s, \ldots)$ and $f(\ldots\ldots)$ are incomparable with respect to $\succ_{mposvm}$ and are not equal up to $\doteq_{mul}$.*

Proof. The property immediately follows from the case 5c) of Definition 4.2.2, and the definitions of multiset extension and $\doteq_{mul}$. □

Property 4.2.7. *$\succ_{mposvm}$ is well-founded.*

Proof. The idea comes from [Mar02]. Assume by contradiction that $\succ_{mposvm}$ is not well-founded. Then there exists an infinite sequence $t_1 \succ_{mposvm} t_2 \succ_{mposvm} \ldots$. Let σ be a substitution such that $dom(\sigma)$ contains all the sequence variables in t_1 and $cod(\sigma) \cap \mathcal{SV} = \emptyset$. Then, by Lemma 4.2.1, $svars(t_i\sigma) = \emptyset$ and, thus, $t_i\sigma \in \mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$ for all $i \geq 1$. By the stability property we get an infinite sequence $t_1\sigma \succ_{mposvm} t_2\sigma \succ_{mposvm} \ldots$ on $\mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$. But then, since $\succ_{mposvm}$ coincides to $\succ_{mpo}$ on $\mathcal{T}(\mathcal{F}, \mathcal{X} \setminus \mathcal{SV})$, we have an infinite sequence $t_1\sigma \succ_{mpo} t_2\sigma \succ_{mpo} \ldots$, which contradicts the fact that $\succ_{mpo}$ is well-founded. □

These properties imply the following theorem:

Theorem 4.2.1. *The ordering $\succ_{mposvm}$ is a reduction ordering on $\mathcal{T}(\mathcal{X}, \mathcal{F}) \setminus \mathcal{SV}$, which is total on $\mathcal{T}(\mathcal{F})$ up to $\doteq_{mul}$ if the precedence $\succ_{\mathcal{F}}$ is total.*

4.2.3 The Ordering $\succ_{mposv}$

Now we extend the ordering $\succ_{mposvm}$ to the ordering $\succ_{mposv}$ which is total on ground terms up to $\doteq$, if the precedence is total.

Definition 4.2.3. *Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$. Then $s \succ_{mposv} t$ iff*

– $s \succ_{mposvm} t$ *or*
– $s \doteq_{mul} t$, $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$ *and* $\langle s_1, \ldots, s_n \rangle \succ^{lex}_{mposv} \langle t_1, \ldots, t_n \rangle$.

Theorem 4.2.2. *The ordering* $\succ_{mposv}$ *is a reduction ordering on* $\mathcal{T}(\mathcal{X}, \mathcal{F}) \setminus \mathcal{SV}$*, which is total on* $\mathcal{T}(\mathcal{F})$ *up to* $\doteq$ *if the precedence* $\succ_{\mathcal{F}}$ *is total.*

Proof. We prove all properties separately by induction on the size of the involved terms.

1. Irreflexivity and transitivity follow from the irreflexivity and transitivity of $\succ_{mposvm}$ and $\succ^{lex}_{mposv}$ (by induction hypothesis).
2. Monotonicity. Suppose $s \succ_{mposv} t$. If $s \succ_{mposvm} t$ then by monotonicity of $\succ_{mposvm}$ we have $f(\ldots s \ldots) \succ_{mposvm} f(\ldots t \ldots)$ and hence $f(\ldots s \ldots) \succ_{mposv} f(\ldots t \ldots)$. If $s \doteq_{mul} t$, then $s = g(s_1, \ldots, s_n)$, $t = g(t_1, \ldots, t_n)$ and $\langle s_1, \ldots, s_n \rangle \succ^{lex}_{mposv} \langle t_1, \ldots, t_n \rangle$, which implies $f(\ldots s \ldots) = f(\ldots g(s_1, \ldots, s_n) \ldots) \succ_{mposv} f(\ldots g(t_1, \ldots, t_n) \ldots) = f(\ldots t \ldots)$.
3. Stability under substitutions. Suppose $s \succ_{mposv} t$. If $s \succ_{mposvm} t$ then by stability of $\succ_{mposvm}$ we have $s\sigma \succ_{mposv} t\sigma$ for any σ. If $s \doteq_{mul} t$, then $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$ and $\langle s_1, \ldots, s_n \rangle \succ^{lex}_{mposv} \langle t_1, \ldots, t_n \rangle$. From $s \doteq_{mul} t$, by definition of $\doteq_{mul}$ we get $s\sigma \doteq_{mul} t\sigma$ for any substitution σ. Therefore, by the definition of lexicographic extension and the induction hypothesis, $\langle s_1\sigma, \ldots, s_n\sigma \rangle \succ^{lex}_{mposv} \langle t_1\sigma, \ldots, t_n\sigma \rangle$ for any σ, which implies $s\sigma \succ_{mposv} t\sigma$.
4. Well-foundedness. Suppose there exists an infinite decreasing sequence $t_1 \succ_{mposv} t_2 \succ_{mposv} \ldots$. Since $\succ_{mposvm}$ is well-founded and by transitivity of $\succ_{mposv}$, there exists n such that $t_m \doteq_{mul} t_n$ for all $m > n$. But there are only finitely many (syntactically different) terms t_m with this property. Therefore, there must be two terms t_k and t_l with $k > l \geq n$ such that $t_k = t_l$. Then by transitivity of $\succ_{mposv}$ we get $t_k \succ_{mposv} t_l = t_k$, which contradicts irreflexivity.
5. Ground totality up to $\doteq$. Let us assume that the precedence is total. Then by ground totality (up to $\doteq_{mul}$) of $\succ_{mposvm}$ we get either $s \succ_{mposvm} t$, $t \succ_{mposvm} s$ or $s \doteq_{mul} t$ for all ground terms s and t. Therefore, either $s \succ_{mposv} t$ or $t \succ_{mposv} s$, or we have to compare $\langle s_1\sigma, \ldots, s_n\sigma \rangle$ and $\langle t_1\sigma, \ldots, t_n\sigma \rangle$. But, by the induction hypothesis for all i, $1 \leq i \leq n$, either $s_i \succ_{mposv} t_i$ or $t_i \succ_{mposv} s_i$, or $s_i \doteq t_i$, and, hence, either $\langle s_1\sigma, \ldots, s_n\sigma \rangle \succ_{mposv} \langle t_1\sigma, \ldots, t_n\sigma \rangle$ or $\langle t_1\sigma, \ldots, t_n\sigma \rangle \succ_{mposv} \langle s_1\sigma, \ldots, s_n\sigma \rangle$, or $s \doteq t$. It implies either $s \succ_{mposv} t$, $t \succ_{mposv} s$ or $s \doteq t$.

□

4.3 Reduction Ordering for Pattern-terms with Sequence Variables and Flexible Arity Symbols

4.3.1 The Ordering $\succ_{mpopm}$

Let $\mathcal{F} \subseteq \mathcal{CONST} \cup \mathcal{FFIX} \cup \mathcal{FFLEX}$, $\mathcal{X} \subseteq \mathcal{IV} \cup \mathcal{SV} \cup \mathcal{XV}$ and let $\mathcal{PT}(\mathcal{F}, \mathcal{X})$ denote a set of pattern-terms over $\mathcal{F}$ and $\mathcal{X}$. We assume that $\mathcal{F}$ contains at least one constant or nullary function symbol and $\succ_{\mathcal{F}}$ is a well-founded precedence on $\mathcal{F}$. Our goal is to define a reduction ordering on $\mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$.

Let $\mathcal{PR}_{\mathcal{F}}$ be a set of pattern prefixes over $\mathcal{F}$. We extend the precedence $\succ_{\mathcal{F}}$ to an ordering $\succ_{\mathcal{FPR}}$ on $\mathcal{F} \cup \mathcal{PR}_{\mathcal{F}}$. Let $f, g \in \mathcal{F}$. Then:

- $f \succ_{\mathcal{FPR}} g$ iff $f \succ_{\mathcal{F}} g$;
- $f \succ_{\mathcal{FPR}} g_{m,n}$ iff $f \succ_{\mathcal{F}} g$ or $f = g$;
- $f_{m,n} \succ_{\mathcal{FPR}} g$ iff $f \succ_{\mathcal{F}} g$;
- $f_{m,n} \succ_{\mathcal{FPR}} g_{l,k}$ iff
  - $f \succ_{\mathcal{F}} g$ or
  - $f = g$ and $\{m, n\} >^{mul,=} \{l, k\}$, where $>$ is a stable well-founded ordering on $\mathcal{LP}$ total on positive integers.

It is easy to see that $\succ_{\mathcal{FPR}}$ is a well-founded ordering on $\mathcal{F} \cup \mathcal{PR}_{\mathcal{F}}$. We call it a precedence on $\mathcal{F} \cup \mathcal{PR}_{\mathcal{F}}$.

As a meta-variable over $\mathcal{F} \cup \mathcal{PR}_{\mathcal{F}}$ we use *pr* with or without indices.

We can extend the notion of instance on $\mathcal{F} \cup \mathcal{PR}_{\mathcal{F}}$ as follows: Let θ be a substitution and *pr* be a prefix. Then an instance of *pr* with respect to θ, denoted $pr\theta$, is defined as follows:

$$pr\theta = \begin{cases} f_{m\theta,n\theta} & \text{if } pr = f_{m,n} \in \mathcal{PR}_{\mathcal{F}}, \\ pr & \text{if } pr \in \mathcal{F}. \end{cases}$$

It follows from this extension and the definition of $\succ_{\mathcal{FPR}}$ that $\succ_{\mathcal{FPR}}$ is a stable ordering.

We denote by $\mathcal{PAT}_{\mathcal{C}}$ the subset of $\mathcal{PAT}$ of all patterns of the form $c_{m,n}$.

We extend $\doteq_{mul}$ on $\mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ as follows: $s \doteq_{mul} t$ iff

- $s, t \in \mathcal{CONST} \cup \mathcal{IV}$ and $s \doteq t$, or
- $s, t \in \mathcal{PAT}_{\mathcal{C}}$, $s = c_{m,n}$, $t = d_{m,n}$ and $c \doteq d$, or
- $s = pr(s_1, \ldots, s_n)$, $t = pr(t_1, \ldots, t_n)$ and for all i, $1 \le i \le n$, $s_{\pi(i)} \doteq_{mul} t_i$, or $s_{\pi(i)}$ and t_i are the same sequence variables, where π is a permutation of $1, \ldots, n$.

For example, $f(\overline{x}, g_{1,k}(c_{3,m}, \overline{y}, y), b) \doteq_{mul} f(b, \overline{x}, g_{1,k}(\overline{y}, c_{3,m}, y))$.

We extend for $\mathcal{PT}(\mathcal{F}, \mathcal{X})$ the notions of

- lexicographic extension and multiset extension of an ordering $\succ$ on $\mathcal{PT}(\mathcal{F}, X) \setminus SV$ with respect to a congruence $\cong$ on $\mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$;
- stability, subterm property, deletion property and totality (up to a congruence on $\mathcal{PT}(\mathcal{F})$) of an ordering $\succ$ on $\mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$

in a straightforward way.

We define a reduction ordering on $\mathcal{PT}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$ in two steps: first, we define an ordering which is total on $\mathcal{PT}(\mathcal{F})$ up to $\doteq_{mul}$, and then extend it to an ordering which is total on $\mathcal{PT}(\mathcal{F})$ up to $\doteq$.

Thus, as the first step, we define a binary relation $\succ_{mpopm}$ (*mpopm* stands for <u>m</u>ultiset <u>p</u>ath <u>o</u>rdering for <u>p</u>attern-terms ground total up to $\doteq_{\underline{m}ul}$).

Definition 4.3.1. *Let $s,t \in \mathcal{PT}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$ and $\succ_{\mathcal{FPR}}$ be a precedence extended to prefixes. Then $s \succ_{mpopm} t$ iff*

1. *$t \in ivars(s)$ and $s \neq t$, or*
2. *$s,t \in \mathcal{CONST} \cup \mathcal{PAT}_{\mathcal{C}}$ and $s \succ_{\mathcal{FPR}} t$, or*
3. *$s \in \mathcal{CONST} \cup \mathcal{PAT}_{\mathcal{C}}$, $t = pr(t_1,\ldots,t_n)$, $n \geq 0$, $s \succ_{\mathcal{FPR}} pr$, and for all j with $1 \leq j \leq n$, $s \succ_{mpopm} t_j$, or*
4. *$s = pr(s_1,\ldots,s_m)$, $m \geq 0$, $t \in \mathcal{CONST} \cup \mathcal{PAT}_{\mathcal{C}}$, and either $pr \succ_{\mathcal{FPR}} t$, or for some i, $1 \leq i \leq m$, $s_i \succ_{mpopm} t$ or $s_i \doteq_{mul} t$, or*
5. *$s = pr_1(s_1,\ldots,s_m)$, $m \geq 0$, $t = pr_2(t_1,\ldots,t_n)$, $n \geq 0$ and*
   a) *$s_i \succ_{mpopm} t$ or $s_i \doteq_{mul} t$ for some i with $1 \leq i \leq m$ or*
   b) *$pr_1 \succ_{\mathcal{FPR}} pr_2$, and for all j with $1 \leq j \leq n$, either $s \succ_{mpopm} t_j$ or $t_j \in svars(s)$ or*
   c) *$pr_1 = pr_2$ and for all j with $1 \leq j \leq n$,*
      i. *$\{s_1,\ldots,s_m\} \succ^{mul}_{mpopm} \{t_1,\ldots,t_n\}$, and*
      ii. *$\{s_1,\ldots,s_m\} \setminus \mathcal{SV} \succ^{mul}_{mpopm} \{t_1,\ldots,t_n\} \setminus \mathcal{SV}$.*

It is easy to see that $\succ_{mpop}$ is an extension of $\succ_{mposvm}$ – on $\mathcal{T}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$ they coincide.

4.3.2 Properties of $\succ_{mpopm}$

Lemma 4.3.1. *For any $s,t \in \mathcal{PT}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$, $s \succ_{mpopm} t$ implies $vars(s) \setminus \mathcal{XV} \supseteq vars(t) \setminus \mathcal{XV}$.*

Proof. The lemma follows from Lemma 4.2.1 and Definition 4.3.1 □

Lemma 4.3.2. *For all $r,s,t,q \in \mathcal{PT}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$, if $r \doteq_{mul} s$, $s \succ_{mpopm} t$ and $t \doteq_{mul} q$, then $r \succ_{mpopm} q$.*

Proof. Similar to the proof of Lemma 4.2.2. □

Property 4.3.1. *$\succ_{mpopm}$ is transitive.*

Proof. Similar to the proof of transitivity of $\succ_{mposvm}$. □

Property 4.3.2. *$\succ_{mpopm}$ fulfils the subterm property.*

Proof. Similar to the proof of subterm property of $\succ_{mposvm}$. □

Property 4.3.3. *$\succ_{mpopm}$ fulfils the pattern subterm property: $pr(\ldots c_{m,n} \ldots) \succ_{mpopm} c_{k,l}$ and $pr(\ldots f_{m,n}(t_1, \ldots, t_r) \ldots) \succ_{mpopm} f_{k,l}(t_1, \ldots, t_r)$ for all $r \geq 0$ and $m \leq k \leq l \leq n$.*

Proof. The property follows from $f_{m,n}(t_1, \ldots, t_r) \succ_{mpopm} f_{k,l}(t_1, \ldots, t_r)$ and $c_{m,n} \succ_{mpopm} c_{k,l}$ by the case 5a) of Definition 4.3.1. □

Property 4.3.4. *$\succ_{mpopm}$ is stable under substitution.*

Proof. We have to show that $s \succ_{mpopm} t$ implies $s\sigma \succ_{mpopm} t\sigma$ for all $s, t \in \mathcal{PT}(\mathcal{F}, \mathcal{X}) \backslash \mathcal{SV}$ and a substitution σ. We use well-founded induction on $|s|+|t|$.

1. $s \succ_{mpopm} t$ by the case 1) of the Definition 4.3.1. Then the proof is similar to the case 1 of Property 4.2.3.
2. $s \succ_{mpopm} t$ by the case 2) of the Definition 4.3.1. Then $s \succ_{\mathcal{FPR}} t$. By stability of $\succ_{\mathcal{FPR}}$ we get $s\sigma \succ_{\mathcal{FPR}} t\sigma$ and, thus, $s\sigma \succ_{mpopm} t\sigma$.
3. $s \succ_{mpopm} t$ by the case 3) of the Definition 4.3.1. Then $t = pr(t_1, \ldots, t_n)$, $n \geq 0$, $s \succ_{\mathcal{FPR}} t$ and for all j with $1 \leq j \leq n$ we have $s \succ_{mpop} t_j$. The by stability of $\succ_{\mathcal{FPR}}$ and by the induction hypothesis we get $s\sigma \succ_{mpopm} t\sigma$.
4. $s \succ_{mpopm} t$ by the case 4) of the Definition 4.3.1. Then $s = pr(s_1, \ldots, s_m)$, $m \geq 0$, $t \in \mathcal{CONST} \cup \mathcal{PAT}_{\mathcal{C}}$, and either $pr \succ_{\mathcal{FPR}} t$, or for some i, $1 \leq i \leq m$, $s_i \succ_{mpopm} t$ or $s_i \doteq_{mul} t$. If $pr \succ_{\mathcal{FPR}} t$, then by stability of $\succ_{\mathcal{FPR}}$ we have $pr\sigma \succ_{\mathcal{FPR}} t\sigma$ and, therefore, $s\sigma \succ_{mpopm} t\sigma$. If $s_i \succ_{mpopm} t$, then by the induction hypothesis $s_i\sigma \succ_{mpopm} t\sigma$. By the subterm property $s\sigma \succ_{mpopm} s_i\sigma$. By transitivity of $\succ_{mpopm}$, $s\sigma \succ_{mpopm} t\sigma$. If $s_i \doteq_{mul} t$, then $s_i\sigma \doteq_{mul} t\sigma$. Therefore, by the subterm property and Lemma 4.3.2 $s\sigma \succ_{mpopm} t\sigma$.
5. $s \succ_{mpopm} t$ by the case 5a) of the Definition 4.3.1. Then the proof proceeds similarly to the previous case.
6. $s \succ_{mpopm} t$ by the case 5b) of the Definition 4.3.1. Then the proof is similar to the case 5 of Property 4.2.3.
7. $s \succ_{mpopm} t$ by the case 5c) of the Definition 4.3.1. Then the proof is similar to the case 6 of Property 4.2.3.

□

Property 4.3.5. *$\succ_{mpopm}$ is monotonic.*

Proof. Similar to the proof of monotonicity of $\succ_{mposvm}$. □

Property 4.3.6. *$\succ_{mpopm}$ is irreflexive.*

Proof. We have to show that for all $s \in \mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$, $s \not\succ_{mpopm} s$. Assume by contradiction that there exists $s \in \mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$ with $s \succ_{mpopm} s$. Let σ be a substitution such that for all index variables vn in s, $vn \in dom(\sigma)$ and $vn\sigma$ is a positive integer. By the stability property, from $s \succ_{mpopm} s$ we get $s\sigma \succ_{mpopm} s\sigma$. If we consider all the pattern prefixes in $s\sigma$ as function symbols then we can conclude that $\succ_{mposvm}$ is reflexive, which contradicts Property 4.2.5. □

$\succ_{mpopm}$ is not a simplification ordering – it does not fulfil the deletion property. However, a weaker form of the deletion property holds:

Property 4.3.7. *$\succ_{mpopm}$ fulfils a weak form of deletion property: $pr(\ldots s \ldots) \succ_{mpopm} pr(\ldots\ldots)$ for any $s \in \mathcal{PT}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$. Moreover, if $s \in \mathcal{SV}$, then $pr(\ldots s \ \ldots)$ and $pr(\ldots\ldots)$ are incomparable with respect to $\succ_{mpopm}$ and are not equal up to $\doteq_{mul}$.*

Proof. The property immediately follows from the case 5c) of Definition 4.3.1, and the definitions of multiset extension and $\doteq_{mul}$. □

Property 4.3.8. *$\succ_{mpopm}$ fulfils pattern deletion property:*

$pr(\ldots c_{m,n} \ldots) \succ_{mpopm} pr(\ldots c_{m,k}, c_{l,n} \ldots)$,
$pr(\ldots c_{m,n} \ldots) \succ_{mpopm} pr(\ldots c_{l,n} \ldots)$,
$pr(\ldots c_{m,n} \ldots) \succ_{mpopm} pr(\ldots c_{m,k} \ldots)$,
$pr(\ldots f_{m,n}(t_1,\ldots,t_r) \ldots) \succ_{mpopm}$
$\quad pr(\ldots f_{m,k}(t_1,\ldots,t_r), f_{l,n}(t_1,\ldots,t_r) \ldots)$,
$pr(\ldots f_{m,n}(t_1,\ldots,t_r) \ldots) \succ_{mpopm} pr(\ldots f_{l,n}(t_1,\ldots,t_r) \ldots)$,
$pr(\ldots f_{m,n}(t_1,\ldots,t_r) \ldots) \succ_{mpopm} pr(\ldots f_{m,k}(t_1,\ldots,t_r) \ldots)$

where $m \le k < l \le n$ and $l - k > 1$.

Proof. The property follows from the definition of precedence and the case 5c) of Definition 4.3.1. □

Property 4.3.9. *$\succ_{mpopm}$ is well-founded.*

Proof. The proof is similar to the proof of well-foundedness of *mpo* ([Der82]). Assume by contradiction that $\succ_{mpopm}$ is not well-founded. Then there exists an infinite sequence $t_1 \succ_{mpopm} t_2 \succ_{mpopm} \cdots$. We construct a "minimal" descending sequence in the following manner: if the terms $t_1 \succ_{mpopm} t_2 \succ_{mpopm} \cdots \succ_{mpopm} t_{i-1}$, $i \ge 1$ have already been chosen, then let t_i be a minimal size i-th term from among those infinite descending sequences beginning with the terms already chosen. We first show that for all i, $pr(t_i) \succ_{\mathcal{FPR}} pr(t_{i+1})$. Assume by contradiction that for some i, $pr(t_i) \not\succ_{\mathcal{FPR}} pr(t_{i+1})$. Then, since $t_i \succ_{mpopm} t_{i+1}$, by the case 5a) of Definition 4.3.1, there exists a subterm s of t_i such that $s \succ_{mpopm} t_{i+1}$ or $s \doteq_{mul} t_{i+1}$. But then by the subterm property of $\succ_{mpopm}$ we have $t_{i-1} \succ_{mpopm} t_i \succ_{mpopm} s \succ_{mpopm} t_{i+1} \succ_{mpopm} t_{i+2}$ or $t_{i-1} \succ_{mpopm} t_i \succ_{mpopm} s \doteq_{mul} t_{i+1} \succ_{mpopm} t_{i+2}$ and $t_1 \succ_{mpopm} \cdots t_{i-1} \succ_{mpopm} s \succ_{mpopm} t_{i+2} \succ_{mpopm} \cdots$ would be a smaller descending sequence than the minimal one. The obtained contradiction shows that for all i, $pr(t_i) \succ_{\mathcal{FPR}} pr(t_{i+1})$.

Since $\succ_{\mathcal{FPR}}$ is well-founded, there must be an infinite descending sequence $t_l \succ_{mpopm} t_{l+1} \succ_{mpopm} \cdots$ all elements of which have the same prefix. By stability of $\succ_{mpopm}$ and Lemma 4.3.1, we can assume that $t_l, t_{l+1} \ldots$ contain no sequence and individual variables. Let $M(r)$ be a multiset of immediate subterms of r, i.e. $M(pr(r_1,\ldots,r_n)) = \{r_1,\ldots,r_n\}$ and for $r \in \mathcal{CONST} \cup \mathcal{PAT}_{\mathcal{C}}$, $M(r) = \emptyset$. Then, from $t_l \succ_{mpopm} t_{l+1} \succ_{mpopm} \cdots$, by the case 5c) of Definition 4.3.1 we must have an infinite descending sequence of multisets $M(t_l) \succ^{mul}_{mpopm} M(t_{l+1}) \succ^{mul}_{mpopm} \cdots$. On the other hand, by the assumption of minimality the set $S = \cup_{i=l}^{\infty} M(t_i)$ is well-founded (otherwise, there would exist an infinite descending sequence $s_1 \succ_{mpopm} s_2 \succ_{mpopm} \cdots$ of elements of S. Since $s_1 \in M(t_k)$ for some k, the sequence $t_1 \succ_{mpopm} t_2 \succ_{mpopm} t_{k-1} \succ_{mpopm} s_1 \succ_{mpopm} s_2 \cdots$ would be smaller than the minimal one). But if S is well-founded, then multisets of elements of S are also well-founded. Contradiction. □

4.3.3 The Ordering $\succ_{mpop}$

We extend the ordering $\succ_{mpopm}$ to the ordering $\succ_{mpop}$ which is total on ground terms up to $\doteq$, if the precedence is total.

Definition 4.3.2. *Let $s, t \in \mathcal{PT}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{SV}$. Then $s \succ_{mpop} t$ iff*

- *$s \succ_{mpopm} t$ or*
- *$s =_{mul} t$, $s = pr(s_1, \dots, s_n)$, $t = pr(t_1, \dots, t_n)$ and $\langle s_1, \dots, s_n \rangle \succ^{lex}_{mpop} \langle t_1, \dots, t_n \rangle$.*

Theorem 4.3.1. *The ordering $\succ_{mpop}$ is a reduction ordering on $\mathcal{PT}(\mathcal{X}, \mathcal{F}) \setminus \mathcal{SV}$, which is total on $\mathcal{PT}(\mathcal{F})$ up to $\doteq$ if the precedence $\succ_{\mathcal{PF}}$ is total.*

Proof. We prove all properties separately by induction on the size of the involved terms.

1. Irreflexivity and transitivity follow from the irreflexivity and transitivity of $\succ_{mpopm}$ and $\succ^{lex}_{mpop}$ (by induction hypothesis).
2. Monotonicity. Suppose $s \succ_{mpop} t$. If $s \succ_{mpopm} t$ then by monotonicity of $\succ_{mpopm}$ we have $pr_1(\dots s \dots) \succ_{mpopm} pr_1(\dots t \dots)$ and hence $pr_1(\dots s \dots) \succ_{mpop} pr_1(\dots t \dots)$. If $s \doteq_{mul} t$, then $s = pr_2(s_1, \dots, s_n)$, $t = pr_2(t_1, \dots, t_n)$ and $\langle s_1, \dots, s_n \rangle \succ^{lex}_{mpop} \langle t_1, \dots, t_n \rangle$, which implies $pr_1(\dots s \dots) = pr_1(\dots pr_2(s_1, \dots, s_n) \dots) \succ_{mpop} pr_1(\dots pr_2(t_1, \dots, t_n) \dots) = pr_1(\dots t \dots)$.
3. Stability under substitutions. Suppose $s \succ_{mpop} t$. If $s \succ_{mpopm} t$ then by stability of $\succ_{mpopm}$ we have $s\sigma \succ_{mpop} t\sigma$ for any σ. If $s \doteq_{mul} t$, then $s = pr(s_1, \dots, s_n)$, $t = pr(t_1, \dots, t_n)$ and $\langle s_1, \dots, s_n \rangle \succ^{lex}_{mpop} \langle t_1, \dots, t_n \rangle$. From $s \doteq_{mul} t$, by definition of $\doteq_{mul}$ we get $s\sigma \doteq_{mul} t\sigma$ for any substitution σ. Therefore, by the definition of lexicographic extension and the induction hypothesis, $\langle s_1\sigma, \dots, s_n\sigma \rangle \succ^{lex}_{mpop} \langle t_1\sigma, \dots, t_n\sigma \rangle$ for any σ, which implies $s\sigma \succ_{mpop} t\sigma$.
4. Well-foundedness. Suppose there exists an infinite decreasing sequence $t_1 \succ_{mpop} t_2 \succ_{mpop} \dots$. Since $\succ_{mpopm}$ is well-founded and by transitivity of $\succ_{mpop}$, there exists n such that $t_m \doteq_{mul} t_n$ for all $m > n$. But there are only finitely many (syntactically different) terms t_m with this property. Therefore, there must be two terms t_k and t_l with $k > l \geq n$ such that $t_k = t_l$. Then by transitivity of $\succ_{mpop}$ we get $t_k \succ_{mpop} t_l = t_k$, which contradicts irreflexivity.

5. Ground totality up to $\doteq$. Let us assume that the precedence is total. Then by ground totality (up to $\doteq_{mul}$) of $\succ_{mpopm}$ we get either $s \succ_{mpopm} t$, $t \succ_{mpopm} s$ or $s \doteq_{mul} t$ for all ground terms s and t. Therefore, either $s \succ_{mpop} t$ or $t \succ_{mpop} s$, or we have to compare $\langle s_1\sigma, \ldots, s_n\sigma\rangle$ and $\langle t_1\sigma, \ldots, t_n\sigma\rangle$. But, by the induction hypothesis for all i, $1 \leq i \leq n$, either $s_i \succ_{mpop} t_i$ or $t_i \succ_{mpop} s_i$, or $s_i \doteq t_i$, and, hence, either $\langle s_1\sigma, \ldots, s_n\sigma\rangle \succ_{mpop} \langle t_1\sigma, \ldots, t_n\sigma\rangle$ or $\langle t_1\sigma, \ldots, t_n\sigma\rangle \succ_{mpop} \langle s_1\sigma, \ldots, s_n\sigma\rangle$, or $s \doteq t$. It implies either $s \succ_{mpop} t$, $t \succ_{mpop} s$ or $s \doteq t$.

□

4.4 Proving Procedure – Unfailing Completion

Unfailing completion [BDP89] is an extension of Knuth-Bendix completion procedure [KB67]. It is considered to be one of the most efficient method of proving theorems in equational theories (without sequence variables).

In this section we show that unfailing completion is a well-suitable proving method for the free theory with sequence variables and flexible arity symbols as well, provided that sequence variables occur only as the last arguments in terms.

Before describing the procedure we fix some terminology. We write

- $s \rhd t$ iff there exists a position $p \neq \lambda$ (λ denotes the top position) in s and a substitution σ such that $s|_p = t\sigma$, or, in words, a proper subterm of s is an instance of t. The relation $\rhd$ is called the encompassment ordering relation;
- $s \rightarrow_{R,l\rightarrow r} t$ to indicate that s rewrites to t by application of the rule $l \rightarrow r$ in R;
- $s \mathrel{\dot{\div}} t$ to denote (ambiguously) $s \doteq t$ or $t \doteq s$.

The notion of critical pair is defined as follows:

Definition 4.4.1 (Critical Pair). *Let $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$ be two rules and $\succ$ be a reduction ordering total on ground terms. Let p be a position in l_1 such that $l_1|_p \notin \mathcal{IV} \cup \mathcal{SV}$ and let $\sigma \in mcu_\emptyset(l_1|_p, l_2)$ with $l_1\sigma \not\succ r_1\sigma$ and $l_2\sigma \not\succ r_2\sigma$. Then $r_1\sigma \doteq l_1\sigma[r_2\sigma]_p$ is called a critical pair between $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$ at position p.*

Critical pairs between rule and equation or between equations can be built treating an equation $s \doteq t$ as two rules $s \rightarrow t$ and $t \rightarrow s$. The set of all critical pairs that can be built for E and R is denoted by $CP(E, R)$.

We describe unfailing completion with an added rule for subsumption suggested in [AD93]. The system works on triples (E, R, G) of term pairs (equations, rules and goals, respectively) and is parameterized by a reduction ordering $\succ$ which is total on ground terms. As an input it accepts a set of equations E, a goal $s \doteq t$ and a ground reduction ordering $\succ$. In order to prove $s \doteq t$ the procedure successively orients the equations from E, creating a set R of rules, generates new equations by building critical pairs between rules and equations from E and R and uses the rules from R and orientable instances from equations for simplifications of both the goal and knowledge base. The goal is proved if the normal forms of both sides (with respect to $R \cup E_{\succ}$) are identical or if an equation subsuming the goal is generated. The completion procedure is given by a set of inference rules and a set of fairness conditions that restrict the application of the inference rules. We formulate them here for the free theory with sequence variables and flexible arity symbols.

Definition 4.4.2 (Unfailing Completion). *Let $\succ$ be a reduction ordering on $\mathcal{T}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$ total on $\mathcal{T}(\mathcal{F})$. The inference system U consists of the following inference rules, where $s,t,u,v,l,r \in \mathcal{T}(\mathcal{F},\mathcal{X}) \setminus \mathcal{SV}$:*

1. *Orient an equation*
$$\frac{(E \cup \{s \doteqdot t\}, R, G)}{(E, R \cup \{s \to t\}, G)},$$
if $s \succ t$.
2. *Generate an equation*
$$\frac{(E, R, G)}{(E \cup \{s \doteq t\}, R, G)},$$
if $s \doteq t \in CP(E,R)$.
3. *Delete an equation*
$$\frac{(E \cup \{s \doteq s\}, R, G)}{(E, R, G)}$$
4. *Simplify an equation*
$$\frac{(E \cup \{s \doteqdot t\}, R, G)}{(E \cup \{u \doteqdot t\}, R, G)},$$
if $s \to_R u$ or $s \to_{E_{\succ}, l \doteq r} u$ with $s \rhd l$.
5. *Subsume an equation*
$$\frac{(E \cup \{s \doteqdot t,\ u \doteqdot v\}, R, G)}{(E \cup \{s \doteqdot t\}, R, G)},$$
if $u|_p = s\sigma$, $v = u[t\sigma]_p$, $u \rhd s$, for a position p and a substitution σ.
6. *Simplify the right-hand side of a rule*
$$\frac{(E, R \cup \{s \to t\}, G)}{(E, R \cup \{s \to u\}, G)},$$
if $t \to_{R \cup E_{\succ}} u$.

7. *Simplify the left-hand side of a rule*

$$\frac{(E, R \cup \{s \to t\}, G)}{(E \cup \{u \doteq t\}, R, G)},$$

if $s \to_{R \cup E_{\succ}, l \doteq r} u$ *with* $s \rhd l$.

8. *Simplify the goal*

$$\frac{(E, R, \{s \doteqdot t\})}{(E, R, \{u \doteqdot t\})},$$

if $s \to_{R \cup E_{\succ}} u$.

9. *Success*

$$\frac{(E, R, \{s \doteq s\})}{\mathsf{Success}}.$$

Definition 4.4.3 (*U*-fairness).

– *A sequence* $(E_i, R_i, G_i)_{i \in \mathbb{N}}$ *is called an* U*-derivation iff* $(E_i, R_i, G_i) \vdash (E_{i+1}, R_{i+1}, G_{i+1})$ *for all* $i \in \mathbb{N}$.
– *The set of persistent rules* R_∞ *for a given* U*-derivation is defined by*

$$R_\infty = \cup_{i \geq 0} \cap_{j \geq i} R_j.$$

– *The set of persistent equations* E^∞ *for a given* U*-derivation is defined by*

$$E_\infty = \cup_{i \geq 0} \cap_{j \geq i} E_j$$

– *An* U*-derivation* $(E_i, R_i, G_i)_{i \in \mathbb{N}}$ *is fair iff*

$$CP(E_\infty, R_\infty) \subseteq \cup_{i \geq 0} E_i.$$

Note that for the free theory with sequence variables and flexible arity symbols, the usage of unfaining completion is restricted to the case when sequence variables occur only as the last arguments in terms. In this case unification is finitary (Theorem 3.3.8) and matching is unitary. The ordering $\succ_{mposv}$ can be used for $\succ$.

The main theorem on unfailing completion (without sequence variables and flexible arity symbols) is the soundness and completeness theorem [BDP89]:

Theorem 4.4.1. *Let* $(E_i, R_i, G_i)_{i \in \mathbb{N}}$ *be a fair* U*-derivation with* $(E_0, R_0, G_0) = (E, \emptyset, \hat{s} = \hat{t})$ *where* $\hat{s} = \hat{t}$ *is the skolemized version of* $s = t$*. Then* $s =_E t$ *iff the derivation is finite and ends with* Success.

The result analogous to Theorem 4.4.1 holds for the free theory with sequence variables and flexible arity symbols, where sequence variables occur only as the last arguments in terms:

Theorem 4.4.2 (Soundness and Completeness). *Let* $(E_i, R_i, G_i)_{i\in\mathbb{N}}$ *be a fair* U*-derivation with* $(E_0, R_0, G_0) = (E, \emptyset, s \doteq t)$ *where* $s \doteq t$ *is a ground equation and* E *is the set of equations where sequence variables occur only as the last arguments in terms. Then* $s \doteq_E t$ *iff the derivation is finite and ends with* Success.

In the next section we show how to extend this result to the general case – for the free theory with sequence variables and flexible arity symbols, where sequence variables are not restricted to be only the last arguments in terms.

4.5 Proving Procedure – Basic Equational Superposition with Constraints

Deduction with constraints was introduced in [KKR90]. We will show here how the basic superposition [NR92a] with constraints can be extended for equational theories with sequence variables and flexible arity symbols. We formulate the results here for terms and equations, but they can be stated also for P-terms and P-equations.

First, we adopt some definitions (and adapt them to the cases with sequence variables and flexible arity symbols) from [NR92a] and [NR95].

By (ordering) constraints we mean a conjunction of the formulae of the form $s \succ_{mposv} t$ or $s \doteq t$ $(s, t \notin \mathcal{SV})$. An equality constraint is a particular case of such a constraint consisting of a conjunction of equality literals $s \doteq t$. A solution (in a signature $\mathcal{F}$) of a constraint T is a ground substitution σ such that $cod(\sigma) \subseteq \mathcal{T}(\mathcal{F})$, $dom(\sigma) \supseteq vars(T)$ and $T\sigma$ evaluates to $true$. Then we say that T is satisfiable (in $\mathcal{F}$).

A constrained equation is a pair $\langle q, T\rangle$, denoted $(q \parallel T)$, where e is an equation and T is a constraint. Such a pair denotes the set of ground instances of $(q \parallel T)$: those ground instances $q\sigma$ of q such that $T\sigma$ evaluates to $true$.

Note that any unconstrained equation q can be written as $(q \parallel true)$.

We extend $\succ_{mposv}$ to equalities in the following way: we write that $(s \doteq t) \succ_{mposv} (s' \doteq t')$ iff $\{s, t\} \succ^{mul}_{mposv} \{s', t'\}$.

An interpretation I satisfies (is a model of) a constrained equation $(q \parallel T)$, denoted $I \models (q \parallel T)$ iff $I \models q\sigma$ for every ground instance $q\sigma$ of $(q \parallel T)$. It implies that constrained equations with unsatisfiable constraints are tautologies.

If R is a set ground equations (or rewrite rules), $\leftrightarrow^*_R$ denotes the congruence generated by R, which can be considered as an interpretation, called equality Herbrand interpretation H_R:

- The domain of H_R is $\mathcal{T}(\mathcal{F})$;
- Each constant c in $\mathcal{F}$ is interpreted as $c_{H_R} = [\langle c\rangle]_{\approx}$.
- Each n-ary function symbol f in $\mathcal{F}$ is interpreted as a set $f_{H_R} \subseteq \mathcal{T}(\mathcal{F})^{**}$ of the form

$$\{[\langle d_1,\ldots,d_n,f(d_1,\ldots,d_n)\rangle]_{\approx} \mid d_1,\ldots,d_n \in \mathcal{T}(\mathcal{F})\},$$

satisfying functional condition,

– Each flexible arity function symbol f in $\mathcal{T}(\mathcal{F})$ is interpreted as a set $f_{H_R} \subseteq \mathcal{T}(\mathcal{F})^{**}$ of the form

$$\{[\langle d_1,\ldots,d_k,f(d_1,\ldots,d_k)\rangle]_{\approx} \mid k \geq 0,\ d_1,\ldots,d_k \in \mathcal{T}(\mathcal{F})\},$$

satisfying functional condition.

– The equality predicate $\doteq$ is interpreted by $s \doteq t$ if $s \leftrightarrow_R^* t$.

The inference rule of basic equational superposition with constraints is defined as follows:

Definition 4.5.1 (Superposition). *The inference rule of basic equational superposition with constraints is defined as follows:*

$$\frac{(s_1 \doteq t_1 \parallel T_1)\ \ (s_2 \doteq t_2 \parallel T_2)}{(s_1[t_2]_p \doteq t_1 \parallel T_1 \wedge T_2 \wedge s_1 \succ t_1 \wedge s_2 \succ t_2 \wedge s_1|_p \doteq s_2)}$$

where $\succ$ is a reduction ordering total on ground terms and $s_1|_p \notin \mathcal{IV} \cup \mathcal{SV}$.

The basicness restriction [BGLS95], [NR95] does not allow inferences to be computed on terms coming from unifiers of ancestor inferences.

In Section 4.2 we defined a ground total reduction ordering $\succ_{mposv}$ for terms with sequence variables and flexible arity symbols. The ordering is an essential ingredient for formulating the basic superposition rule for theories with sequence variables and flexible arity symbols. Below in this section we assume that the ordering is $\succ_{mposv}$.

The first important result about basic superposition shows that the closure of an unconstrained set of equations (with sequence variables and flexible arity symbols) under basic equational superposition with constraints is ground confluent. To prove this result we need additional definitions. We follow here the exposition of [NR92a].

Definition 4.5.2. *A ground substitution σ is irreducible with respect to a set of ground rewrite rules R or, shortly, R-irreducible, if*

– *for every $x \in dom(\sigma)$, $x\sigma$ is irreducible with respect to R,*
– *for every $\overline{x} \in dom(\sigma)$, either $\overline{x}\sigma$ is the empty sequence or $\overline{x}\sigma = t_1,\ldots,t_n$, such that for all i, $1 \leq i \leq n$, t_i is irreducible with respect to R.*

Definition 4.5.3. *Let E be a set of constrained equations, $(s \doteq t \parallel T) \in E$ and $s\sigma \doteq t\sigma$ be a ground instance of $(s \doteq t \parallel T)$ with $s\sigma \succ t\sigma$. Then $s\sigma \doteq t\sigma$ generates the rule $s\sigma \to t\sigma$ if $s\sigma$ and σ are irreducible with respect to the rules generated by ground instances $q\theta$ of equations $q \in E$ with $(s\sigma \doteq t\sigma) \succ q\theta$.*

The set of all rules generated by all ground instances of equations in E is denoted R_E.

We write $gen(q) = \{l \to r\}$ to indicate that the equation q generates the rule $l \to r$. If q generates no rule then $gen(q) = \emptyset$.

Definition 4.5.4. *Let E be a set of constrained equations and let R be a set of ground rewrite rules. The set of ground instances of equations in E with R-irreducible substitutions is denoted by $irred_R(E)$, i.e.*

$$irred_R(E) = \{q\sigma \mid \quad (q \parallel T) \in E,\ T\sigma \text{ is true},\ \sigma \text{ is ground and } R-irreducible\}.$$

Lemma 4.5.1. *Let E be a set of constrained equations and $(s \doteq t \parallel T) \in E$ such that $gen(s\sigma \doteq t\sigma) = \{s\sigma \to t\sigma\}$ for some ground σ. Then*

- *for all $x \in vars(t)$, $x\sigma$ is R_E-irreducible;*
- *for all $\overline{x} \in vars(t)$, either $\overline{x}\sigma$ is the empty sequence or $\overline{x}\sigma = t_1, \ldots, t_n$, such that for all i, $1 \leq i \leq n$, t_i is R_E-irreducible.*

Proof. On the one hand, By Definition 4.5.3,

- for all $x \in vars(t)$, $x\sigma$ is $R_{\prec, s\sigma \doteq t\sigma}$-irreducible;
- for all $\overline{x} \in vars(t)$, either $\overline{x}\sigma$ is the empty sequence or $\overline{x}\sigma = t_1, \ldots, t_n$, such that for all i, $1 \leq i \leq n$, t_i is $R_{\prec, s\sigma \doteq t\sigma}$-irreducible,

where $R_{\prec, s\sigma \doteq t\sigma}$ is the set of rules generated by ground instances of equations in E smaller than $s\sigma \doteq t\sigma$ with respect to $\succ_{mposv}$.

On the other hand, let $R_{\succeq, s\sigma \doteq t\sigma}$ be the set of rules generated by ground instances of equations in E greater than $s\sigma \doteq t\sigma$ with respect to $\succ_{mposv}$, or equal to $s\sigma \doteq t\sigma$ up to $\doteq_{mul}$. Then all rules in $R_{\succeq, s\sigma \doteq t\sigma}$ have left hand sides strictly greater than $t\sigma$ with respect to $\succ_{mposv}$. Therefore, none of these rules can reduce a subterm of $t\sigma$. □

Lemma 4.5.2. *Let E be a set of constrained equations closed under basic superposition with constraints. Then $H_{R_E} \models irred_{R_E}(E)$.*

Proof. Assume by contradiction that $H_{R_E} \not\models irred_{R_E}(E)$. Then there exists a minimal (with respect to $\succ_{mposv}$) element $s\sigma \doteq t\sigma$ in $irred_{R_E}(E)$ such that $H_{R_E} \not\models s\sigma \doteq t\sigma$.

Let $s\sigma \doteq t\sigma$ is a ground instance of the equation $(s \doteq t \parallel T_1) \in E$. Suppose $s\sigma \succ_{mposv} t\sigma$. Since $H_{R_E} \not\models s\sigma \doteq t\sigma$, $gen(s\sigma \doteq t\sigma) = \emptyset$. Therefore, $s\sigma$ must be reducible by R_E. Assume $s\sigma$ is reducible with a rule $l\theta \to r\theta$, such that $\{l\theta \to r\theta\} = gen(l\theta \doteq r\theta)$ and $(s\sigma \doteq t\sigma) \succ_{mposv} (l\theta \doteq r\theta)$, where $l\theta \doteq r\theta$ is a ground instance of an equation $(l \doteq r \parallel T_2) \in E$.

Since $s\sigma$ is reducible with $l\theta \to r\theta$, there exists a position p in $s\sigma$ such that $s\sigma|_p = l\theta$. Moreover $s|_p \notin \mathcal{IV} \cup \mathcal{SV}$, since otherwise σ would be R_E-reducible. Thus, we can make the following inference:

$$\frac{(l \doteq r \parallel T_1) \quad (s \doteq t \parallel T_2)}{(s[r]_p \doteq t \parallel T_1 \wedge T_2 \wedge l \succ_{mposv} r \wedge s \succ_{mposv} t \wedge s|_p \doteq l)}$$

Since E is closed under basic superposition with constraints, $(s[r]_p \doteq t \parallel T_1 \wedge T_2 \wedge l \succ_{mposv} r \wedge s \succ_{mposv} t \wedge s|_p \doteq l) \in E$.

We have a ground instance $s\sigma[r\theta]|_p \doteq t\sigma$ of $(s[r]|_p \doteq t \parallel T_1 \wedge T_2 \wedge l \succ_{mposv} r \wedge s \succ_{mposv} t \wedge s|_p \doteq l)$ such that $H_{R_E} \not\models s\sigma[r\theta]_p \doteq t\sigma$ (because otherwise we would have $H_{R_E} \not\models s\sigma \doteq t\sigma$). Moreover, Lemma 4.5.1 and the fact that σ is R_E-irreducible imply that $s\sigma[r\theta]_p \doteq t\sigma$ is a ground instance of $(s[r]_p \doteq t \parallel T_1 \wedge T_2 \wedge l \succ_{mposv} r \wedge s \succ_{mposv} t \wedge s|_p \doteq l)$ with the substitution which is irreducible by R_E. Furthermore, $(s\sigma \doteq t\sigma) \succ_{mposv} (s\sigma[r\theta]_p \doteq t\sigma)$, which altogether contradicts the minimality of $s\sigma \doteq t\sigma$. □

Lemma 4.5.3. *Let E_0 be a set of equations without constraints, and let E be the closure of E_0 under basic superposition with constraints. Then $H_{R_E} \models E$.*

Proof. First note that $E_0 \models E$, by soundness of basic superposition. Therefore, is suffices to show that $H_{R_E} \models E_0$, i.e. $H_{R_E} \models q\sigma$ for every ground instance $q\sigma$ of an equation $(q \parallel true)$ in E_0. Now let σ' be a normal form of σ with respect to R_E. Then by Lemma 4.5.2 we have $H_{R_E} \models q\sigma'$. From $R_E \cup \{q\sigma'\} \models q\sigma$ and $H_{R_E} \models q\sigma'$ it follows that $H_{R_E} \models q\sigma$. □

Lemma 4.5.4. *Let E be a set of constrained equations such that $H_{R_E} \models E$. Then E is ground confluent.*

Proof. Let t, s_1 and s_2 be ground terms such that s_1 and s_2 are normal forms of t with respect to $\rightarrow_{E_\succ}$. We prove that s_1 and s_2 are syntactically equal. We have $E \models s_1 \doteq s_2$ and $H_{R_E} \models E$, which implies $H_{R_E} \models s_1 \doteq s_2$ and $R_E \models s_1 \doteq s_2$. If s_1 and s_2 are normal forms with respect to $\rightarrow_{E_\succ}$, then they are also normal forms with respect to $\rightarrow_{R_E}$, because R_E is a set of instances of equations of E. But R_E is a canonical set of ground rewrite rules (no overlappings between left hand sides), which implies that $s_1 = s_2$. □

Theorem 4.5.1. *Let E_0 be a set of equations over $\mathcal{T}(\mathcal{F}, \mathcal{X})$ without constraints, and let E be the closure of E_0 under basic equational superposition with constraints. Then E is ground confluent.*

Proof. The theorem follows from Lemma 4.5.3 and Lemma 4.5.4. □

Example 4.5.1. Let E_0 be $(f(g(\overline{x}, a, \overline{y})) \doteq f(g(\overline{x}), g(\overline{y})) \parallel true)$ and let the ordering be $\succ_{mposv}$ with the precedence $f \succ_{\mathcal{F}} g \succ_{\mathcal{F}} a \succ_{\mathcal{F}} b$. Then $f(g(b, a, a))$ can be rewritten in two ways: one into $f(g(b), g(a))$ and the other into $f(g(b, a), g())$. Closure of E_0 under basic superposition with constraints includes $(f(g(\overline{x}), g(\overline{y})) \doteq f(g(\overline{z}), g(\overline{u})) \parallel f(g(\overline{x}, a, \overline{y})) \doteq f(g(\overline{z}, a, \overline{u})))$ (ordering constraints simplify to *true*). One of the unifiers of the unification problem $f(g(\overline{x}, a, \overline{y})) \overset{?}{\doteq} f(g(\overline{z}, a, \overline{u}))$ is $\sigma = \{\overline{x} \leftarrow \overline{z}, a,\ \overline{u} \leftarrow a,\ \overline{y} \leftarrow\}$. A ground instance of σ, $\{\overline{x} \leftarrow b, a,\ \overline{u} \leftarrow a,\ \overline{y} \leftarrow,\ \overline{z} \leftarrow b\}$ is a solution of the constraint $f(g(\overline{x}, a, \overline{y})) \doteq f(g(\overline{z}, a, \overline{u}))$ and generates a ground equation $f(g(b, a), g()) \doteq f(g(b), g(a))$, which rewrites the term $f(g(b, a), g())$ into $f(g(b), g(a))$ and, thus, restores local confluence.

Note that in the inference system the ordering constraints do not affect soundness and completeness. Their purpose is to prune the search space. Therefore, as Niuewenhuis pointed out, we do not really need to decide the satisfiability of ordering constraints, but would like to detect many cases of unsatisfiable constraints. Therefore, for instance, given an atomic constraint $s \succ_{mposv} t$, we would like to use the ordering $\succ_{mposv}$ for detecting that $s\sigma \succ_{mposv} t\sigma$ for no grounding σ (on ground terms $\succ_{mposv}$ coincides with the lexicographic extension of $\succ_{mpo}$). For this purpose it is sufficient that the ordering $\succ_{mposv}$ satisfies the stability property for grounding substitutions.

Theorem 4.5.1 applies to a closure procedure, without considering simplification or deletion techniques. Now we will show that the proposition analogous to Theorem 4.5.1 holds for a saturation procedure as well. As it is pointed out in [NR01], informally, the notion of saturation with respect to a given inference system $\mathcal{I}$ generalizes the notion of closure with respect to $\mathcal{I}$: a set of equations E is saturated if E is closed under $\mathcal{I}$ up to redundant inferences. Numerous methods and criteria for detecting redundant inferences have been developed [BDH86], [Pet90], [MN90], [Bac91], [BD94], [CNNR98], [KMN88], [WB83]. Informally, an equation is redundant if it can be proved from "smaller" ones. In the constraint context, all instances of constraints must be taken into account for redundancy definition. Here we give definitions according [NR92a]:

Definition 4.5.5 (Saturated Set). *A set of equations E is saturated with respect to the inference system $\mathcal{I}$ if every inference with premises in E is redundant with respect to E.*

Definition 4.5.6 (Persistent and Redundant Equations). *Let the sequence $E_0, E_1, \dots$ be a sequence of sets of constrained equations.*

- *The set E_∞ of persistent equations in $E_0, E_1, \dots$ is defined as $\cup_{i\geq 0} \cap_{j \geq i} E_j$.*
- *An equation $(q \parallel T)$ is redundant in E_j if for any ground instance $q\sigma$ of $(q \parallel T)$, where σ is R_{E_∞}-irreducible solution of T, there exist R_{E_∞}-irreducible ground instances $d_1, \dots, d_n$ of constrained equations in E_j such that $R_{E_\infty} \cup \{d_1, \dots, d_n\} \models q\sigma$ and $\{q\sigma\} \succ^{mul}_{mposv} \{d_1, \dots, d_n\}$.*

Definition 4.5.7 (*BS*-Derivation). *A BS-derivation is a sequence of constrained equations $E_0, E_1, \dots$ such that*

- *$E_i = E_{i-1} \cup \{(q \parallel T)\}$, where $(q \parallel T)$ is deduced from E_{i-1} by basic equational superposition with constraints, or*
- *$E_i = E_{i-1} \setminus \{(q \parallel T)\}$, if $(e \parallel T)$ is redundant in E_{i-1}.*

Definition 4.5.8 (Ground Instance of Inference). *Let π be a basic superposition inference with premises $(q_1 \parallel T_1)$ and $(q_2 \parallel T_2)$, and conclusion $(q \parallel T)$. Let σ be a ground substitution which is a solution of T. Then a basic superposition inference $\pi\sigma$ with premises $q_1\sigma$ and $q_2\sigma$ and conclusion $q\sigma$, is called a ground instance of π.*

Definition 4.5.9 (Redundant Inference). *Let $E_0, E_1, \dots$ be a BS-derivation, and let π be a basic superposition inference with premises $(q_1 \parallel T_1)$ and $(q_2 \parallel T_2)$, and conclusion $(q \parallel T)$. π is redundant in E_j if for every ground instance $\pi\sigma$ of π with R_{E_∞}-irreducible σ, there exist R_{E_∞}-irreducible ground instances $d_1, \dots, d_n$ of constrained equations in E_j such that $R_{E_\infty} \cup \{d_1, \dots, d_n\} \models q\sigma$ and $\{max(q_1\sigma, q_2\sigma)\} \succ^{mul}_{mposv} \{d_1, \dots, d_n\}$, where max denotes maximality with respect to $\succ_{mposv}$.*

Definition 4.5.10 (*BS*-Fairness). *A BS-derivation $E_0, E_1, \dots$ is fair if every inference by basic equational superposition with constraints with premises in E_∞ is redundant in some E_j.*

Note that all the results about saturation with basic superposition for equational theories without sequence variables and flexible arity symbols ([NR92a], [NR92b], [NR95]) remain valid for the theories with sequence variables and flexible arity symbols. Below we give some of these results and their proofs which only slightly differ from those in [NR92a].

First we show that the limit of a fair BS-derivation is saturated. To prove this, we need two lemmata given below. They can be proved in the same way as the corresponding lemmata in [NR92a] for the theories without sequence variables and flexible arity symbols.

Lemma 4.5.5. *Let $E_0, E_1, \dots$ be a BS-derivation. Then for every set E_j and instance $q\sigma$ in $irred_{R_{E_\infty}}(E_j)$, there exist R_{E_∞}-irreducible ground instances $d_1, \dots, d_n$ of constrained equations in E_j such that $R_{E_\infty} \cup \{d_1, \dots, d_n\} \models q\sigma$ and $\{q\sigma\} \succ^{mul}_{mposv} \{d_1, \dots, d_n\}$.*

This result is used to show the following lemma:

Lemma 4.5.6. *Let $E_0, E_1, \dots$ be a BS-derivation. If an inference by basic superposition is redundant in some E_j, then it also is in E_∞.*

Theorem 4.5.2. *Let $E_0, E_1, \dots$ be a fair BS-derivation. Then E_∞ is saturated.*

Proof. Fairness implies that every inference π with premises in E_∞ is redundant in some E_j. By Lemma 4.5.6, the π is also redundant in E_∞ and, thus, E_∞ is saturated. □

Next, we show that the limit of a fair BS-derivation is ground confluent.

Lemma 4.5.7. *Let $E_0, E_1, \dots$ be a fair BS-derivation. Then we have*

$$H_{R_{E_\infty}} \models irred_{R_{E_\infty}}(E_\infty).$$

Proof. The proof is an easy extension of the proof on Lemma 4.5.2. □

Lemma 4.5.8. *Let $E_0, E_1, \ldots$ be a fair BS-derivation such that E_0 is a set without constraints. Then*

$$H_{R_{E_\infty}} \models E_\infty.$$

Proof. By Lemma 4.5.7 we have $H_{R_{E_\infty}} \models irred_{R_{E_\infty}}(E_\infty)$. Moreover, by Lemma 4.5.5 we have $R_{E_\infty} \cup irred_{R_{E_\infty}} \models irred_{R_{E_0}}$. Since the conclusion of basic superposition is a logical consequence of its premises, we have $E_i \models E_{i+1}$ for all i and, therefore, $E_0 \models E_\infty$. Now, since $R_{E_\infty} \cup irred_{R_{E_\infty}} \models E_0$ holds as in Lemma 4.5.3, we have $H_{R_{E_\infty}} \models E_\infty$. □

Theorem 4.5.3. *Let $E_0, E_1, \ldots$ be a fair BS-derivation such that E_0 is a set without constraints. Then E_∞ is ground confluent.*

Proof. The theorem follows from Lemma 4.5.7 and Lemma 4.5.8. □

Note that during the computation of fair BS-derivation one cannot prove the redundancy in a set E_j, since at that point R_{E_∞} is unknown. Therefore, sufficient conditions for redundancy have to be used. We do not discuss them here but refer to [NR92a], [NR92b], [NR95], [NR01], since the conditions considered there are compatible with the case of equations with sequence variables and flexible arity symbols as well.

Note also that the results of this section can be extended to pattern-terms and pattern-equations, with the ordering $\succ_{mpop}$.

5. Equational Unification with Sequence Variables and Flexible Arity Symbols

In this chapter we prove decidability of general unification problem and design minimal and complete unification procedures for flat, restricted flat and orderless theories.

5.1 General Unification in the Flat Theory – Decidability

In this section we consider decidability of general unification problem in the flat theory. The axiom set of the theory has a form

$$E = \{f(\overline{x}, f(\overline{y}), \overline{z}) \doteq f(\overline{x}, \overline{y}, \overline{z})\}.$$

The problem has a form $t_1 \overset{?}{\doteq}_F t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, flat flexible arity function symbols, free constants and free fixed arity function symbols. We denote it as $\mathfrak{U}_F$.

The problem is to prove that $\mathfrak{U}_F$ is decidable. To do so, as in the case of free theory, first we try to reduce $\mathfrak{U}_F$ to a "simpler" unification problem such that $\mathfrak{U}_F$ is solvable if and only if the reduced problem is solvable, and then try to prove that solvability of reduced unification problem is decidable.

5.1.1 Reduction

First we define an operation on terms called flattening. This operation replaces a term of the form $f(\tilde{t}, f(\tilde{s}), \tilde{r})$, where f is a flat flexible arity symbol and $\tilde{t}$, $\tilde{s}$ and $\tilde{r}$ are (possibly empty) sequences of terms, with the term $f(\tilde{t}, \tilde{s}, \tilde{r})$. Given a term t, we denote by $flt(t)$ the term obtained from t by flattenning all its subterms until possible. Obviously $t \doteq_F flt(t)$. Let $\mathfrak{U}_F^{flt}$ be the unification problem $flt(t_1) \overset{?}{\doteq}_F flt(t_2)$.

Lemma 5.1.1. $\mathfrak{U}_F$ *is solvable iff* $\mathfrak{U}_F^{flt}$ *is solvable.*

Proof. Let θ be a substitution. Then $t_1\theta \doteq_F t_2\theta$ iff $flt(t_1)\theta \doteq_F flt(t_2)\theta$. □

Let $f_1, \ldots, f_m$, $m \geq 1$, be all free flexible arity function symbols occurring in $\mathfrak{U}_F^{flt}$. We reduce $\mathfrak{U}_F^{flt}$ to an intermediate unification problem $\mathfrak{U}_F^{int}$ by performing the following steps:

1. Introduce
   – a new flat flexible arity symbol *seq*;
   – a new unary function symbol g_i for each f_i $(1 \leq i \leq m)$.
2. Replace each term $f_i(r_1, \ldots, r_n)$ in $\mathfrak{U}_F^{flt}$ by $g_i(seq(r_1, \ldots, r_n))$ $(n \geq 0)$.

Note that sequence variables occur only as arguments of terms with the head $h_1, \ldots, h_k$ or *seq*, where $h_1, \ldots, h_k$, $k \geq 1$ are all flat flexible arity symbols in $\mathfrak{U}_F^{flt}$. As in the case of the empty theory, we impose *individual variable restrictions on terms* on $\mathfrak{U}_F^{int}$ demanding that, for a solution θ of $\mathfrak{U}_F^{int}$ and for any individual variable x, $x\theta$ must not have *seq* as the head.

Theorem 5.1.1. *$\mathfrak{U}_F^{flt}$ is solvable iff $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms is solvable.*

Proof. Analogous to the proof of Theorem 3.2.1. □

Next, our goal is to reduce $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms to a finite set S_F^{red} of reduced F-unification problems ($\mathfrak{U}_F^{red}$-s) with the property that $\mathfrak{U}_F^{int}$ with individual variable restrictions on terms is solvable iff at least one $\mathfrak{U}_F^{red}$ is solvable. We define S_F^{red} in the same way as we did in the proof of decidability of $\emptyset$-unification. Then the following theorem holds:

Theorem 5.1.2. *$\mathfrak{U}_F^{int}$ with individual variable restrictions on terms is solvable iff there exists a $\mathfrak{U}_F^{red} \in S_F^{red}$ which is solvable.*

Proof. Analogous to the proof of Theorem 3.2.2. □

5.1.2 Decidability of the Reduced Problem

We prove decidability of the reduced unification problem $\mathfrak{U}_F^{red}$ using Baader-Schulz combination method.

Let for all $1 \leq i \leq k$, Ω_i be the set $\{h_i\}$, Ω_{k+1} be the set $\{seq\}$, and $\Omega_{k+2} = const(\mathfrak{U}_F^{red}) \cup fix(\mathfrak{U}_F^{red})$. Let for all $1 \leq i \leq k$, E_i be an equational theory with sequence variables over the signature Ω_i, E_{k+1} be an equational theory with sequence variables over the signature Ω_{k+1} and E_{k+2} be a free theory (without sequence variables) over Ω_{k+2}. Then we can consider $\mathfrak{U}_F^{red}$ as a unification problem in the combined theory $E_1 \cup \cdots \cup E_{k+2}$. Since $\Omega_1, \ldots, \Omega_{k+2}$ are pairwise disjoint, by Theorem 3.2.3, in order to show that solvability of $\mathfrak{U}_F^{red}$ in $E_1 \cup \cdots \cup E_{k+2}$ is decidable we need to show for each $1 \leq i \leq k+2$ that solvability of E_i-unification problem with linear constant restrictions is decidable.

For each $1 \leq i \leq k$, solvability of E_i-unification with linear constant restrictions is equivalent to solvability of word equations with linear constant restrictions, which is decidable ([BS91]).

Solvability of E_{k+1}-unification with linear constant restrictions is equivalent to solvability of word equations with linear constant restrictions and individual variable restrictions on length, which is also decidable (Theorem 3.2.5).

The E_{k+2}-unification problem is a Robinson unification problem, solvability of which with linear constant restrictions is decidable (see [BS91]).

Thus, solvability of $\mathfrak{U}_F^{red}$ is decidable, which, by Theorem 5.1.1 and Theorem 5.1.2, implies that solvability of general unification problem in the F-theory with sequence variables and flexible arity symbols is decidable.

5.2 General Unification in the Flat Theory – Unification Procedure

In this section we design a unification procedure to solve general unification problem $\mathfrak{U}_F$ in a flat theory.

We design the unification procedure as a tree generation process based on three basic steps: projection, flattening and transformation.

5.2.1 Projection

Projection in the flat theory coincides with the projection in the free theory (see Figure 3.1). The only difference is that we change the subscript $\emptyset$ to F.

5.2.2 Flattening

Flattening rule is defined on the Figure 5.1 below:

Flattening:	$s \overset{?}{\doteq}_F t \rightsquigarrow \langle\langle flt(s) \overset{?}{\doteq}_F flt(t), \varepsilon\rangle\rangle$

Fig. 5.1. Flattening rule.

5.2.3 Transformation

Each of the transformation rules for unification have one of the following forms: $\mathfrak{U} \rightsquigarrow \bot$ or $\mathfrak{U} \rightsquigarrow \langle\langle \mathfrak{S}_1, \sigma_1\rangle, \ldots, \langle \mathfrak{S}_n, \sigma_n\rangle\rangle$ where each of the successors $\mathfrak{S}_i$ is either $\top$ or a new unification problem.

If $\mathfrak{U}_F$ has a form $s \overset{?}{\doteq}_F t$, where s and t are either

– identical terms,
– terms with different heads, or
– terms with non-flat heads

then $\mathfrak{U}_F$ is transformed by one of the transformation rules on Figure 3.2 (changing the subscript $\emptyset$ to F). Otherwise we define transformation rules for $\mathfrak{U}_F$ on Figure 5.2, Figure 5.3, Figure 5.4 and Figure 5.5. The unification problems $\mathfrak{U}_1$, $\mathfrak{U}_2$, $\mathfrak{U}_3$, $\mathfrak{U}_4$ and $\mathfrak{U}_5$ in the tables are defined as follows:

$$\mathfrak{U}_1 = \begin{cases} h(\overline{x}, \tilde{t}\theta) \stackrel{?}{=}_F h(y, \tilde{s}\theta) & \text{if } h(\overline{x}, \tilde{t}\theta) \neq flt(h(\overline{x}, \tilde{t}\theta)) \text{ or} \\ & \quad h(y, \tilde{s}\theta) \neq flt(h(y, \tilde{s}\theta)) \\ & \quad \text{with } \theta = \{\overline{x} \leftarrow h(), \overline{x}\}, \\ \bot & \text{otherwise.} \end{cases}$$

$$\mathfrak{U}_2 = \begin{cases} h(y, \tilde{t}\theta) \stackrel{?}{=}_F h(\overline{x}, \tilde{s}\theta) & \text{if } h(y, \tilde{t}\theta) \neq flt(h(y, \tilde{t}\theta)) \text{ or} \\ & \quad h(\overline{x}, \tilde{s}\theta) \neq flt(h(\overline{x}, \tilde{s}\theta)) \\ & \quad \text{with } \theta = \{\overline{x} \leftarrow h(), \overline{x}\} \\ \bot & \text{otherwise.} \end{cases}$$

$$\mathfrak{U}_3 = \begin{cases} h(\overline{x}, \tilde{t}\theta) \stackrel{?}{=}_F h(\overline{y}, \tilde{s}\theta) & \text{if } h(\overline{x}, \tilde{t}\theta) \neq flt(h(\overline{x}, \tilde{t}\theta)) \text{ or} \\ & \quad h(\overline{y}, \tilde{s}\theta) \neq flt(h(\overline{y}, \tilde{s}\theta)) \\ & \quad \text{with } \theta = \{\overline{x} \leftarrow h(), \overline{x}\}, \\ \bot & \text{otherwise.} \end{cases}$$

$$\mathfrak{U}_4 = \begin{cases} h(\overline{x}, \tilde{t}\theta) \stackrel{?}{=}_F h(\overline{y}, \tilde{s}\theta) & \text{if } h(\overline{x}, \tilde{t}\theta) \neq flt(h(\overline{x}, \tilde{t}\theta)) \text{ or} \\ & \quad h(\overline{y}, \tilde{s}\theta) \neq flt(h(\overline{y}, \tilde{s}\theta)) \\ & \quad \text{with } \theta = \{\overline{y} \leftarrow h(), \overline{y}\}, \\ \bot & \text{otherwise.} \end{cases}$$

$$\mathfrak{U}_5 = \begin{cases} h(\overline{x}, \tilde{t}\theta) \stackrel{?}{=}_F h(s_1, \tilde{s}\theta) & \text{if } h(\overline{x}, \tilde{t}\theta) \neq flt(h(\overline{x}, \tilde{t}\theta)) \text{ or} \\ & \quad h(s_1, \tilde{s}\theta) \neq flt(h(s_1, \tilde{s}\theta)) \\ & \quad \text{with } \theta = \{\overline{x} \leftarrow h(), \overline{x}\}, \\ \bot & \text{otherwise.} \end{cases}$$

$$\mathfrak{U}_6 = \begin{cases} h(t_1, \tilde{t}\theta) \stackrel{?}{=}_F h(\overline{x}, \tilde{s}\theta) & \text{if } h(t_1, \tilde{t}\theta) \neq flt(h(t_1, \tilde{t}\theta)) \text{ or} \\ & \quad h(\overline{x}, \tilde{s}\theta) \neq flt(h(\overline{x}, \tilde{s}\theta)) \\ & \quad \text{with } \theta = \{\overline{x} \leftarrow h(), \overline{x}\} \\ \bot & \text{otherwise.} \end{cases}$$

In the cases when $\mathfrak{U}_i$ is $\bot$, we replace $\langle\langle \mathfrak{U}_i, \theta\rangle\rangle$ with $\bot$ only.

5.2.4 Tree Generation

We design the unification procedure for the flat theory with sequence variables and flexible arity symbols as a tree generation process. The single steps in this process are projection, flattening and transformation for the flat theory.

Each node of the tree is labeled either with a unification problem, $\top$ or $\bot$. The edges of the tree are labeled by substitutions. The nodes labeled with $\top$ or $\bot$ are terminal nodes. The nodes labeled with unification problems are

SuccessF:	$x \overset{?}{=}_F h(x) \rightsquigarrow \langle\langle\top,\ \{x \leftarrow h(x)\}\rangle\rangle.$	
	$x \overset{?}{=}_F t \rightsquigarrow \langle\langle\top,\ \{x \leftarrow t\}\rangle\rangle,$	if $x \notin ivars(t)$ and $t \neq h(x)$.
	$h(x) \overset{?}{=}_F x \rightsquigarrow \langle\langle\top,\ \{x \leftarrow h(x)\}\rangle\rangle.$	
	$t \overset{?}{=}_F x \rightsquigarrow \langle\langle\top,\ \{x \leftarrow t\}\rangle\rangle,$	if $x \notin ivars(t)$ and $t \neq h(x)$.
FailureF:	$x \overset{?}{=}_F t \rightsquigarrow \bot,$	if $t \neq x$, $t \neq h(x)$, $x \in ivars(t)$.
	$t \overset{?}{=}_F x \rightsquigarrow \bot,$	if $t \neq x$, $t \neq h(x)$, $x \in ivars(t)$.
	$h() \overset{?}{=}_F h(t_1, \tilde{t}) \rightsquigarrow \bot,$	if $t_1 \notin \mathcal{IV} \cup \mathcal{SV}$.
	$h(t_1, \tilde{t}) \overset{?}{=}_F h() \rightsquigarrow \bot,$	if $t_1 \notin \mathcal{IV} \cup \mathcal{SV}$.
	$h(\overline{x}, \tilde{t}) \overset{?}{=}_F h(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $s_1 \neq \overline{x}$, $s_1 \neq g(\overline{x})$ and $\overline{x} \in svars(s_1)$.
	$h(s_1, \tilde{s}) \overset{?}{=}_F h(\overline{x}, \tilde{t}) \rightsquigarrow \bot,$	if $s_1 \neq \overline{x}$, $s_1 \neq g(\overline{x})$ and $\overline{x} \in svars(s_1)$.
	$h(t_1, \tilde{t}) \overset{?}{=}_F h(s_1, \tilde{s}) \rightsquigarrow \bot,$	if $t_1 \overset{?}{=}_F s_1 \rightsquigarrow \bot$.
EliminateF:	$h() \overset{?}{=}_F h(v, \tilde{t}) \rightsquigarrow$ $\langle\langle h() \overset{?}{=}_F h(\tilde{t})\sigma, \sigma\rangle\rangle,$	where $v \in \mathcal{IV} \cup \mathcal{SV}$ and $\sigma = \{v \leftarrow h()\}$.
	$h(v, \tilde{t}) \overset{?}{=}_F h() \rightsquigarrow$ $\langle\langle h(\tilde{t})\sigma \overset{?}{=}_F h(), \sigma\rangle\rangle,$	where $v \in \mathcal{IV} \cup \mathcal{SV}$ and $\sigma = \{v \leftarrow h()\}$.
	$h(t_1, \tilde{t}) \overset{?}{=}_F h(s_1, \tilde{s}) \rightsquigarrow$ $\langle\langle h(\tilde{t}\sigma) \overset{?}{=}_F h(\tilde{s}\sigma),\ \sigma\rangle\rangle,$	if $t_1, s_1 \notin \mathcal{IV} \cup \mathcal{SV}$ and $t_1 \overset{?}{=}_F s_1 \rightsquigarrow \langle\langle\top,\ \sigma\rangle\rangle$.
	$h(v, \tilde{t}) \overset{?}{=}_F h(v, \tilde{s}) \rightsquigarrow$ $\langle\langle h(\tilde{t}) \overset{?}{=}_F h(\tilde{s}),\ \varepsilon\rangle\rangle.$	where $v \in \mathcal{IV} \cup \mathcal{SV}$.
	$h(x, \tilde{t}) \overset{?}{=}_F h(y, \tilde{s}) \rightsquigarrow$	where $x \neq y$ and
	$\langle\langle h(\tilde{t}\sigma_1) \overset{?}{=}_F h(y, \tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{x \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2) \overset{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{x \leftarrow y\},$
	$\langle h(\tilde{t}\sigma_3) \overset{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{x \leftarrow h(y)\},$
	$\langle h(x, \tilde{t}\sigma_4) \overset{?}{=}_F h(\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{x \leftarrow h(y, x)\},$
	$\langle h(x, \tilde{t}\sigma_5) \overset{?}{=}_F h(\tilde{s}\sigma_5),\ \sigma_5\rangle,$	$\sigma_5 = \{y \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_6) \overset{?}{=}_F h(\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{y \leftarrow h(x)\},$
	$\langle h(\tilde{t}\sigma_7) \overset{?}{=}_F h(y, \tilde{s}\sigma_7),\ \sigma_7\rangle\rangle,$	$\sigma_7 = \{y \leftarrow h(x, y)\}.$
	$h(x, \tilde{t}) \overset{?}{=}_F h(s_1, \tilde{s}) \rightsquigarrow$	where $s_1 \notin \mathcal{IV} \cup \mathcal{SV}$ and
	$\langle\langle h(\tilde{t}\sigma_1) \overset{?}{=}_F h(s_1, \tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{x \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2) \overset{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{x \leftarrow s_1\},$
	$\langle h(\tilde{t}\sigma_3) \overset{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{x \leftarrow h(s_1)\},$
	$\langle h(x, \tilde{t}\sigma_4) \overset{?}{=}_F h(\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{x \leftarrow h(s_1, x)\}.$

Fig. 5.2. Transformation rules for the F-unification. $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h and g are flat flexible arity symbols.

EliminateF (continued):	$h(t_1,\tilde{t}) \overset{?}{=}_F h(x,\tilde{s}) \rightsquigarrow$	where $t_1 \notin \mathcal{IV} \cup \mathcal{SV}$ and
	$\langle\langle h(t_1,\tilde{t}\sigma_1) \overset{?}{=}_F h(\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{x \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2) \overset{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{x \leftarrow t_1\},$
	$\langle h(\tilde{t}\sigma_3) \overset{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{x \leftarrow h(t_1)\},$
	$\langle h(\tilde{t}\sigma_4) \overset{?}{=}_F h(x,\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{x \leftarrow h(t_1,x)\}.$
	$h(\overline{x},\tilde{t}) \overset{?}{=}_F h(y,\tilde{s}) \rightsquigarrow$	where
	$\langle\langle h(\tilde{t}\sigma_1) \overset{?}{=}_F h(y,\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{\overline{x} \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2) \overset{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{\overline{x} \leftarrow y\},$
	$\langle h(\tilde{t}\sigma_3) \overset{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{\overline{x} \leftarrow h(y)\},$
	$\langle h(\overline{x},\tilde{t}\sigma_4) \overset{?}{=}_F h(\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{\overline{x} \leftarrow h(y,\overline{x})\},$
	$\langle \mathfrak{U}_1,\ \sigma_5\rangle,$	$\sigma_5 = \{\overline{x} \leftarrow h(),\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_6) \overset{?}{=}_F h(\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{\overline{x} \leftarrow y,\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_7) \overset{?}{=}_F h(\tilde{s}\sigma_7),\ \sigma_7\rangle,$	$\sigma_7 = \{\overline{x} \leftarrow y,h(\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_8) \overset{?}{=}_F h(\tilde{s}\sigma_8),\ \sigma_8\rangle,$	$\sigma_8 = \{\overline{x} \leftarrow h(y),\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_9) \overset{?}{=}_F h(\tilde{s}\sigma_9),\ \sigma_9\rangle,$	$\sigma_9 = \{\overline{x} \leftarrow h(y,\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_{10}) \overset{?}{=}_F h(\tilde{s}\sigma_{10}),\ \sigma_{10}\rangle,$	$\sigma_{10} = \{y \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_{11}) \overset{?}{=}_F h(\tilde{s}\sigma_{11}),\ \sigma_{11}\rangle,$	$\sigma_{11} = \{y \leftarrow h(\overline{x})\},$
	$\langle h(\tilde{t}\sigma_{12}) \overset{?}{=}_F h(y,\tilde{s}\sigma_{12}),\ \sigma_{12}\rangle\rangle,$	$\sigma_{12} = \{y \leftarrow h(\overline{x},y)\}.$
	$h(y,\tilde{t}) \overset{?}{=}_F h(\overline{x},\tilde{s}) \rightsquigarrow$	where
	$\langle\langle h(y,\tilde{t}\sigma_1) \overset{?}{=}_F h(\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{\overline{x} \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2) \overset{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{\overline{x} \leftarrow y\},$
	$\langle h(\tilde{t}\sigma_3) \overset{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{\overline{x} \leftarrow h(y)\},$
	$\langle h(\tilde{t}\sigma_4) \overset{?}{=}_F h(\overline{x},\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{\overline{x} \leftarrow h(y,\overline{x})\},$
	$\langle \mathfrak{U}_2,\ \sigma_5\rangle,$	$\sigma_5 = \{\overline{x} \leftarrow h(),\overline{x})\},$
	$\langle h(\tilde{t}\sigma_6) \overset{?}{=}_F h(\overline{x},\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{\overline{x} \leftarrow y,\overline{x}\},$
	$\langle h(\tilde{t}\sigma_7) \overset{?}{=}_F h(\overline{x},\tilde{s}\sigma_7),\ \sigma_7\rangle,$	$\sigma_7 = \{\overline{x} \leftarrow y,h(\overline{x})\},$
	$\langle h(\tilde{t}\sigma_8) \overset{?}{=}_F h(\overline{x},\tilde{s}\sigma_8),\ \sigma_8\rangle,$	$\sigma_8 = \{\overline{x} \leftarrow h(y),\overline{x}\},$
	$\langle h(\tilde{t}\sigma_9) \overset{?}{=}_F h(\overline{x},\tilde{s}\sigma_9),\ \sigma_9\rangle,$	$\sigma_9 = \{\overline{x} \leftarrow h(y,\overline{x})\},$
	$\langle h(\tilde{t}\sigma_{10}) \overset{?}{=}_F h(\overline{x},\tilde{s}\sigma_{10}),\ \sigma_{10}\rangle,$	$\sigma_{10} = \{y \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_{11}) \overset{?}{=}_F h(\tilde{s}\sigma_{11}),\ \sigma_{11}\rangle,$	$\sigma_{11} = \{y \leftarrow h(\overline{x})\},$
	$\langle h(y,\tilde{t}\sigma_{12}) \overset{?}{=}_F h(\tilde{s}\sigma_{12}),\ \sigma_{12}\rangle\rangle,$	$\sigma_{12} = \{y \leftarrow h(\overline{x},y)\}.$

Fig. 5.3. Transformation rules for the F-unification (continued). $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h is a flat symbol.

EliminateF	$h(\overline{x},\tilde{t})\stackrel{?}{=}_F h(\overline{y},\tilde{s}) \rightsquigarrow$	where $\overline{x} \neq \overline{y}$ and
(continued):	$\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{=}_F h(\overline{y},\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{\overline{x} \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2)\stackrel{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{\overline{x} \leftarrow \overline{y}\},$
	$\langle h(\tilde{t}\sigma_3)\stackrel{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{\overline{x} \leftarrow h(\overline{y})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_4)\stackrel{?}{=}_F h(\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{\overline{x} \leftarrow h(\overline{y},\overline{x})\},$
	$\langle \mathfrak{U}_3,\ \sigma_5\rangle,$	$\sigma_5 = \{\overline{x} \leftarrow h(),\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_6)\stackrel{?}{=}_F h(\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{\overline{x} \leftarrow \overline{y},\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_7)\stackrel{?}{=}_F h(\tilde{s}\sigma_7),\ \sigma_7\rangle,$	$\sigma_7 = \{\overline{x} \leftarrow \overline{y},h(\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_8)\stackrel{?}{=}_F h(\tilde{s}\sigma_8),\ \sigma_8\rangle,$	$\sigma_8 = \{\overline{x} \leftarrow h(\overline{y}),\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_9)\stackrel{?}{=}_F h(\tilde{s}\sigma_9),\ \sigma_9\rangle,$	$\sigma_9 = \{\overline{x} \leftarrow h(\overline{y},\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_{10})\stackrel{?}{=}_F h(\tilde{s}\sigma_{10}),\ \sigma_{10}\rangle,$	$\sigma_{10} = \{\overline{y} \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_{11})\stackrel{?}{=}_F h(\tilde{s}\sigma_{11}),\ \sigma_{11}\rangle,$	$\sigma_{11} = \{\overline{y} \leftarrow h(\overline{x})\},$
	$\langle h(\tilde{t}\sigma_{12})\stackrel{?}{=}_F h(\overline{y},\tilde{s}\sigma_{12}),\ \sigma_{12}\rangle,$	$\sigma_{12} = \{\overline{y} \leftarrow h(\overline{x},\overline{y})\},$
	$\langle \mathfrak{U}_4,\ \sigma_{13}\rangle,$	$\sigma_{13} = \{\overline{y} \leftarrow h(),\overline{y}\},$
	$\langle h(\tilde{t}\sigma_{14})\stackrel{?}{=}_F h(\overline{y},\tilde{s}\sigma_{14}),\ \sigma_{14}\rangle,$	$\sigma_{14} = \{\overline{y} \leftarrow \overline{x},\overline{y}\},$
	$\langle h(\tilde{t}\sigma_{15})\stackrel{?}{=}_F h(\overline{y},\tilde{s}\sigma_{15}),\ \sigma_{15}\rangle,$	$\sigma_{15} = \{\overline{y} \leftarrow \overline{x},h(\overline{y})\},$
	$\langle h(\tilde{t}\sigma_{16})\stackrel{?}{=}_F h(\overline{y},\tilde{s}\sigma_{16}),\ \sigma_{16}\rangle,$	$\sigma_{16} = \{\overline{y} \leftarrow h(\overline{x}),\overline{y}\},$
	$\langle h(\tilde{t}\sigma_{17})\stackrel{?}{=}_F h(\overline{y},\tilde{s}\sigma_{17}),\ \sigma_{17}\rangle\rangle,$	$\sigma_{17} = \{\overline{y} \leftarrow h(\overline{x}),h(\overline{y})\}.$
	$h(\overline{x},\tilde{t})\stackrel{?}{=}_F h(s_1,\tilde{s})$	$s_1 \notin \mathcal{IV} \cup \mathcal{SV}$, $\overline{x} \notin svars(s_1)$
	$\rightsquigarrow$	or $s_1 = g(\overline{x})$, and
	$\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{=}_F h(s_1,\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{\overline{x} \leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2)\stackrel{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{\overline{x} \leftarrow s_1\},$
	$\langle h(\tilde{t}\sigma_3)\stackrel{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3 = \{\overline{x} \leftarrow h(s_1)\},$
	$\langle h(\overline{x},\tilde{t}\sigma_4)\stackrel{?}{=}_F h(\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{\overline{x} \leftarrow h(s_1,\overline{x})\},$
	$\langle \mathfrak{U}_5,\ \sigma_5\rangle,$	$\sigma_5 = \{\overline{x} \leftarrow h(),\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_6)\stackrel{?}{=}_F h(\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{\overline{x} \leftarrow s_1,\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_7)\stackrel{?}{=}_F h(\tilde{s}\sigma_7),\ \sigma_7\rangle,$	$\sigma_7 = \{\overline{x} \leftarrow s_1,h(\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_8)\stackrel{?}{=}_F h(\tilde{s}\sigma_8),\ \sigma_8\rangle,$	$\sigma_8 = \{\overline{x} \leftarrow h(s_1),\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_9)\stackrel{?}{=}_F h(\tilde{s}\sigma_9),\ \sigma_9\rangle\rangle,$	$\sigma_9 = \{\overline{x} \leftarrow h(s_1,\overline{x})\}$

Fig. 5.4. Transformation rules for the F-unification (continued). $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h and g are flat symbols.

EliminateF (continued):	$h(t_1,\tilde{t})\stackrel{?}{=}_F h(\overline{x},\tilde{s})$ $\rightsquigarrow$	$t_1 \notin \mathcal{IV}\cup\mathcal{SV}$, $\overline{x}\notin svars(t_1)$ or $t_1=g(\overline{x})$, and
	$\langle\langle h(t_1,\tilde{t}\sigma_1)\stackrel{?}{=}_F h(\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1=\{\overline{x}\leftarrow h()\},$
	$\langle h(\tilde{t}\sigma_2)\stackrel{?}{=}_F h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2=\{\overline{x}\leftarrow t_1\},$
	$\langle h(\tilde{t}\sigma_3)\stackrel{?}{=}_F h(\tilde{s}\sigma_3),\ \sigma_3\rangle,$	$\sigma_3=\{\overline{x}\leftarrow h(t_1)\},$
	$\langle h(\tilde{t}\sigma_4)\stackrel{?}{=}_F h(\overline{x},\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4=\{\overline{x}\leftarrow h(t_1,\overline{x})\},$
	$\langle\mathfrak{U}_6,\ \sigma_5\rangle,$	$\sigma_5=\{\overline{x}\leftarrow h(),\overline{x})\},$
	$\langle h(\tilde{t}\sigma_6)\stackrel{?}{=}_F h(\overline{x},\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6=\{\overline{x}\leftarrow t_1,\overline{x}\},$
	$\langle h(\tilde{t}\sigma_7)\stackrel{?}{=}_F h(\overline{x},\tilde{s}\sigma_7),\ \sigma_7\rangle,$	$\sigma_7=\{\overline{x}\leftarrow t_1,h(\overline{x})\},$
	$\langle h(\tilde{t}\sigma_8)\stackrel{?}{=}_F h(\overline{x},\tilde{s}\sigma_8),\ \sigma_8\rangle,$	$\sigma_8=\{\overline{x}\leftarrow h(t_1),\overline{x}\},$
	$\langle h(\tilde{t}\sigma_9)\stackrel{?}{=}_F h(\overline{x},\tilde{s}\sigma_9),\ \sigma_9\rangle\rangle,$	$\sigma_9=\{\overline{x}\leftarrow h(t_1,\overline{x})\}.$
SplitF:	$h(t_1,\tilde{t})\stackrel{?}{=}_F h(s_1,\tilde{s})\rightsquigarrow$	if $t_1,s_1\notin\mathcal{IV}\cup\mathcal{SV}$ and
	$\langle\langle h(r_1,\tilde{t}\sigma_1)\stackrel{?}{=}_F h(q_1,\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$t_1\stackrel{?}{=}_F s_1\rightsquigarrow$
	$\dots,$	$\langle\langle r_1\stackrel{?}{=}_F q_1,\ \sigma_1\rangle,\dots$
	$\langle h(r_k,\tilde{t}\sigma_k)\stackrel{?}{=}_F h(q_k,\tilde{s}\sigma_k),\ \sigma_k\rangle\rangle$	$\langle r_k\stackrel{?}{=}_F q_k,\ \sigma_k\rangle\rangle.$

Fig. 5.5. Transformation rules for the F-unification (continued). $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h and g are flat symbols.

non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Given a nonterminal node, let $\mathfrak{U}$ be a unification problem attached to it. First, we decide whether $\mathfrak{U}$ is unifiable. If the answer is negative, we replace $\mathfrak{U}$ with the new label $\perp$. If $\mathfrak{U}$ is unifiable, we apply flattening, projection or transformation on $\mathfrak{U}$ and get $\langle\langle\mathfrak{S}_1,\sigma_1\rangle,\dots,\langle\mathfrak{S}_n,\sigma_n\rangle\rangle$. Then the node $\mathfrak{U}$ has n children, labeled respectively with $\mathfrak{S}_1,\dots,\mathfrak{S}_n$ and the edge to the $\mathfrak{S}_i$ node is labeled with σ_i $(1\le i\le n)$. The set $\{\sigma_1,\dots,\sigma_n\}$ is denoted by $sub(\mathfrak{U})$.

We design the general unification procedure for $\mathfrak{U}_F$ as a breadth first (level by level) tree generation process. We label the root of the tree with $\mathfrak{U}_F$ (zero level). First level nodes of the tree are obtained from the original problem by projection. Starting from the second level, we apply only flattening and transformation steps to a unification problem of each node, first flattening it and then transforming the flattened problem, thus getting new successor nodes. The branch which ends with a node labeled by $\top$ is called a successful branch. The branch which ends with a node labeled by $\perp$ is a failed branch. For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node, flatten all the terms in the codomain of the composition and attach the obtained substitution to the node together with the unification problem the node was labeled with. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N. Let

$\Sigma(\mathfrak{U}_F)$ be the set of all substitutions associated with the $\top$ nodes. We call the tree a unification tree for $\mathfrak{U}_F$ and denote it $utree(\mathfrak{U}_F)$.

Our goal is to prove that $\Sigma(\mathfrak{U}_F)$ is a complete set of F-unifiers of $\mathfrak{U}_F$.

Lemma 5.2.1. *For every F-unifier ϕ of a general unification problem $\mathfrak{U}_F$ there exists a branch β in $utree(\mathfrak{U}_F)$ with the following property: if $\mathfrak{P}$ is a unification problem occurring in β with the associated substitution θ, then $\theta \ll_F^{vars(\mathfrak{U}_F)} \phi$.*

Proof. We define β recursively.

First, let the root of the tree, labeled with $\mathfrak{U}_F$, be in β. The substitution associated with $\mathfrak{U}_F$ is ε. Obviously, $\varepsilon \ll_F^{vars(\mathfrak{U})} \phi$.

Next, let the first level node of β be that successor $\mathfrak{P}$ of $\mathfrak{U}_F$, for which the associated substitution $\theta = \{\overline{x} \leftarrow \ \mid \overline{x} \leftarrow \in \phi \text{ and } \overline{x} \in svars(\mathfrak{U}_F)\}$. We show that $\theta \ll_F^{vars(\mathfrak{U}_F)} \phi$. Let $\lambda = \phi \backslash \theta$. Then we have that $\phi = \theta \circ \lambda$. Therefore

- for all $\overline{x} \in vars(\mathfrak{U}_F)$,
  - $\overline{x} \leftarrow \ \notin \lambda$;
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\,\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_F s_i$;
- for all $x \in vars(\mathfrak{U}_F)$, $x\phi \doteq_F x\theta \circ \lambda$,

which implies that $\theta \ll_F^{vars(\mathfrak{U}_F)} \phi$.

Let now a node in $utree(\mathfrak{U}_F)$, labeled with $\mathfrak{P}_n$, be in β. It means that for the substitution θ_n associated with $\mathfrak{P}_n$, it is true that $\theta_n \ll_F^{vars(\mathfrak{U}_F)} \phi$. It implies that there exists a substitution λ such that

- for all $\overline{x} \in vars(\mathfrak{U}_F)$,
  - the binding
$$\overline{x} \leftarrow \ \notin \lambda; \tag{5.1}$$
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or
$$t_i \doteq_F s_i; \tag{5.2}$$
- for all $x \in vars(\mathfrak{U}_F)$,
$$x\phi \doteq_F x\theta_n \circ \lambda. \tag{5.3}$$

If there exists a successor $\mathfrak{P}_{n+1}$ of $\mathfrak{P}_n$ in $utree(\mathfrak{U}_F)$ with the associated substitution θ_{n+1} such that $\theta_{n+1} \ll_F^{vars(\mathfrak{U}_F)} \phi$, we can include $\mathfrak{P}_{n+1}$ into β and, thus, we will have that for all $\mathfrak{P} \in \beta$, if θ is a substitution associated with $\mathfrak{P}$, then $\theta \ll_F^{vars(\mathfrak{U}_F)} \phi$.

Thus, the problem of constructing β is reduced to finding the successor $\mathfrak{P}_{n+1}$ of $\mathfrak{P}_n$ such that for the substitution θ_{n+1}, associated with $\mathfrak{P}_{n+1}$, we have $\theta_{n+1} \ll_F^{vars(\mathfrak{U}_F)} \phi$. We show how to find such a $\mathfrak{P}_{n+1}$.

If $\mathfrak{P}_{n+1}$ is obtained from $\mathfrak{P}_n$ by flattening, then obviously $\theta_{n+1} \ll_F^{vars(\mathfrak{U}_F)} \phi$, since $\theta_{n+1} = \theta_n \circ \varepsilon$.

Let $\mathfrak{P}_{n+1}$ be obtained from $\mathfrak{P}_n$ by transformation. Then $\mathfrak{P}_n$ is in the flattened form and can have one of the following five forms:

1. $\mathfrak{P}_n$ is a pair of identical terms.
2. $\mathfrak{P}_n$ is a pair of individual variables.
3. $\mathfrak{P}_n$ is a pair of an individual variable and non-variable term.
4. $\mathfrak{P}_n$ is a pair of non-variable terms with the same non-flat head.
5. $\mathfrak{P}_n$ is a pair of non-variable terms the same flat head.

We consider each of them separately:

1-4. In these cases $\mathfrak{P}_{n+1}$ can be chosen analogously to the corresponding cases for the free theory. (see Lemma 3.3.2).

5. Let $\mathfrak{P}_n$ have the form $f(t_1, \overline{t}) \overset{?}{\doteq}_F f(s_1, \overline{s})$, where f is a flat flexible arity symbol and $\overline{t}$ and $\overline{s}$ are (possibly empty) sequences of terms. We can have the following 6 cases with respect to t_1 and s_1:
   - 4.1. t_1 and s_1 are identical.
   - 4.2. Both t_1 and s_1 are individual variables.
   - 4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable term.
   - 4.4. Both t_1 and s_1 are sequence variables.
   - 4.5. One of t_1 and s_1 is a sequence variable, the other is not.
   - 4.6. Both t_1 and s_1 are non-variable terms.

   We consider each of these cases.
   - 4.1. This case can be proved analogously to the corresponding case for the free theory (see Lemma 3.3.2).
   - 4.2. Let t_1 be x and t_2 be y. We have the following cases with respect to λ:
     - 4.2.1. $f(x)\lambda \doteq_F f()$;
     - 4.2.2. $f(x)\lambda \doteq_F f(y)\lambda$;
     - 4.2.3. There exists a non-empty sequence of terms $\tilde{r}$ such that $f(x)\lambda =_F f(y, \overline{r})\lambda$;
     - 4.2.4. $f()\lambda \doteq_F f(y)$;
     - 4.2.5. There exists a non-empty sequence of terms $\tilde{r}$ such that $f(y)\lambda =_F f(x, \tilde{r})\lambda$.

     We define new substitutions μ and ν for each of these cases as follows:
     - 4.2.1. $\mu = \{x \leftarrow f()\}$, $\nu = \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq x,\ v \neq v\lambda\}$;
     - 4.2.2. In this case we have three alternatives:
       - $\mu = \{x \leftarrow y\}$, $\nu = \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq x,\ v \neq v\lambda\}$;

– $\mu = \{x \leftarrow f(y)\}$, $\nu = \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq x, v \neq v\lambda\}$;
– $\mu = \{y \leftarrow f(x)\}$, $\nu = \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq y, v \neq v\lambda\}$;

4.2.3. $\mu = \{x \leftarrow f(y, x)\}$, $\nu = \{x \leftarrow f(\overline{r})\} \circ \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq x, v \neq v\lambda\}$;

4.2.4. $\mu = \{y \leftarrow f()\}$, $\nu = \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq y, v \neq v\lambda\}$;

4.2.5. $\mu = \{y \leftarrow f(x, y)\}$, $\nu = \{y \leftarrow f(\overline{r})\} \circ \{v \leftarrow v\lambda \mid v \in vars(\mathfrak{U}_F), v \neq y, v \neq v\lambda\}$.

We obtain $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$ by the substitution μ (since λ is a unifier of $\mathfrak{P}_n$, these cases for μ are the only possibilities to get $\mathfrak{P}_{n+1}$ from $\mathfrak{P}_n$). Therefore, $\theta_{n+1} = \theta_n \circ \mu$. On the other hand,

$$\mu \circ \nu = \lambda. \tag{5.4}$$

Therefore

$$\theta_n \circ \lambda = \theta_n \circ \mu \circ \lambda = \theta_{n+1} \circ \nu. \tag{5.5}$$

On the other hand, for all $\overline{x} \in vars(\mathfrak{U}_F)$,

$$\overline{x} \leftarrow\ \notin \mu. \tag{5.6}$$

From 5.6, 5.1 and 5.4, by Definition 2.1.11 we get for all $\overline{x} \in vars(\mathfrak{U}_F)$

$$\overline{x} \leftarrow\ \notin \nu. \tag{5.7}$$

From 5.2, by 5.5 we get that for all $\overline{x} \in vars(\mathfrak{U}_F)$ there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_{n+1} \circ \nu = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or

$$t_i \doteq_F s_i; \tag{5.8}$$

From 5.3 and 5.5 we get that for all $x \in vars(\mathfrak{U}_F)$

$$x\phi \doteq_F x\theta_n \circ \lambda \doteq_F x\theta_n \circ \mu \circ \lambda \doteq_F x\theta_{n+1} \circ \nu. \tag{5.9}$$

From 5.7, 5.8 and 5.9, by Definition 3.1.4 we get $\theta_{n+1} \ll_F^{vars(\mathfrak{U}_F)} \phi$.

4.3.-4.5. These cases can be proved using a similar complete case analysis method as in 4.2.

4.6. This case recursively can be reduced to one of the cases 4.1-4.5 above.

Thus, for all possible forms of $\mathfrak{P}_n$ we found its successor $\mathfrak{P}_{n+1}$ such that for the substitution θ_{n+1}, associated with $\mathfrak{P}_{n+1}$, we have $\theta_{n+1} \ll_F^{vars(\mathfrak{U}_F)} \phi$. It finishes the proof. □

Theorem 5.2.1. *$\Sigma(\mathfrak{U}_F)$ is a complete set of F-unifiers for $\mathfrak{U}_F$.*

Proof. The theorem follows from Lemma 5.2.1 by the definition of Σ. $\square$

The next example shows that the unification procedure is not minimal:

Example 5.2.1. Let $\mathfrak{U}_F$ be $f(x) \stackrel{?}{=}_F f(y)$, f being a flat flexible arity symbol. Then, among the other solutions, the procedure returns the unifiers $\sigma = \{x \leftarrow y\}$ and $\theta = \{x \leftarrow f(), y \leftarrow f()\}$. Obviously $\sigma \ll_F^{vars(\mathfrak{U}_F)} \theta$.

The reason of such a behavior is that the term $f()$ occurs in transformation substitutions. Another consequence of its appearance in transformation is demonstrated in the next example:

Example 5.2.2. Let $\mathfrak{U}_F$ be $f(\overline{x}) \stackrel{?}{=}_F f(a)$, f being a flat flexible arity symbol. Then the procedure computes infinitely many unifiers of $\mathfrak{U}_F$:

$$\{\overline{x} \leftarrow a\}, \{\overline{x} \leftarrow f(a)\}, \{\overline{x} \leftarrow a, f()\}, \{\overline{x} \leftarrow f(a), f()\}, \{\overline{x} \leftarrow f(), a\},$$
$$\{\overline{x} \leftarrow f(), f(a)\}, \{\overline{x} \leftarrow f(), a, f()\}, \{\overline{x} \leftarrow f(), f(a), f()\}, \ldots.$$

Skipping any of the transformation rules involving $f()$ would lead to incompleteness.

Example 5.2.3. Let $\mathfrak{U}_F$ be $f(x, a) \stackrel{?}{=}_F f(a)$, f being a flat flexible arity symbol. $\mathfrak{U}_F$ has a unique solution $\{x \leftarrow f()\}$, which can not be computed if we do not allow the transformation substitution $\{x \leftarrow f()\}$.

Another example shows an importance of the transformation rule with a substitution of the form $\overline{x} \leftarrow f(), \overline{x}$:

Example 5.2.4. Let $\mathfrak{U}_F = f(\overline{x}, g(\overline{x})) \stackrel{?}{=}_F f(a, b, g(a, f(), f(), b))$, f being a flat flexible arity symbol and g being a free flexible arity symbol. Then we find a unifier $\theta = \{\overline{x} \leftarrow a, f(), f(), b\}$ as it is shown on Figure 5.6.

Composing substitutions top-down on the successful branch gives θ. Again, usage of the substitution $\overline{x} \leftarrow f(), \overline{x}$ for transformation is indispensable. Otherwise we lose completeness.

Now we show that the minimal complete set of F-unifiers of $\mathfrak{U}_F$ exists. First we define depth of a term:

Definition 5.2.1. *Let t be a term. Then $depth(t)$ is defined as follows:*

- *If $t \in \mathcal{IV} \cup \mathcal{SV} \cup \mathcal{CONST}$, $depth(t) = 0$;*
- *If $t = f(s_1, \ldots, s_n)$, $n \geq 0$, $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$, then $depth(t) = max(depth(s_1), \ldots, depth(s_n)) + 1$.*

Let t be a flattened term and $t\theta$ be an instance of t, also in the flattened form. Then

- if $t\theta$ is a term, then $depth(t\theta) \geq depth(t)$,
- if $t\theta$ is a sequence of terms $s_1, \ldots, s_n$, then $max(depth(s_1), \ldots, depth(s_n)) \geq depth(t)$.

$$
\begin{array}{c}
f(\overline{x}, g(\overline{x})) \stackrel{?}{=} f(a, b, g(a, f(), f(), b)) \\
\Big\downarrow \{\overline{x} \leftarrow a, \overline{x}\} \\
f(\overline{x}, g(a, \overline{x})) \stackrel{?}{=} f(b, g(a, f(), f(), b)) \\
\Big\downarrow \{\overline{x} \leftarrow f(), \overline{x}\} \\
f(\overline{x}, g(a, f(), \overline{x})) \stackrel{?}{=} f(b, g(a, f(), f(), b)) \\
\Big\downarrow \{\overline{x} \leftarrow f(), \overline{x}\} \\
f(\overline{x}, g(a, f(), f(), \overline{x})) \stackrel{?}{=} f(b, g(a, f(), f(), b)) \\
\Big\downarrow \{\overline{x} \leftarrow b\} \\
f(g(a, f(), f(), b)) \stackrel{?}{=} f(g(a, f(), f(), b)) \\
\Big\downarrow \varepsilon \\
\top
\end{array}
$$

Fig. 5.6. Computing a solution for $f(\overline{x}, g(\overline{x})) \stackrel{?}{=}_F f(a, b, g(a, f(), f(), b))$.

Lemma 5.2.2. *Let $\theta_1 >_F^{vars(\mathfrak{U}_F)} \theta_2 >_F^{vars(\mathfrak{U}_F)} \cdots$ be a strictly decreasing chain of unifiers in $\Sigma(\mathfrak{U}_F)$. Then the chain is finite.*

Proof. For each $i \geq 1$, the terms in $cod(\theta_i)$ are in the flattened form. Finiteness of the chain follows from the fact that there exists a number n which bounds the depth of terms in the codomains of all θ_i. $\square$

This lemma implies the existence of a minimal subset of $\Sigma(\mathfrak{U}_F)$. Now we can refine the unification procedure so that it computes the minimal complete set of unifiers. During the tree generation process, at each step when a new solution appears, check whether the set of already computed solutions S is minimal or not. If S is minimal, then continue the process, otherwise minimize S as follows: Let $\sigma \in S$ be a substitution for which there exists a more general substitution in S. Let β be the branch in $utree(\mathfrak{U}_F)$ such that σ is attached to the leaf of β. Let $\eta_1, \ldots, \eta_n$ be the edges in β, starting from the root. Let k be the number such that none of $\eta_n, \eta_{n-1}, \ldots, \eta_{n-k}$ have a sibling in $utree(\mathfrak{U}_F)$, but η_{n-k+1} has. Then we delete $\eta_n, \eta_{n-1}, \ldots, \eta_{n-k}$ from $utree(\mathfrak{U}_F)$ and continue the tree generation process. Let $utree_{min}(\mathfrak{U}_F)$ be the tree constructed in such a manner and let $\Sigma_{min}(\mathfrak{U}_F)$ denote the set of all substitutions which are associated with $\top$ nodes in $utree_{min}(\mathfrak{U}_F)$.

Theorem 5.2.2. *Let $\mathfrak{U}_F$ be a general F-unification problem. Then*

$$\Sigma_{min}(\mathfrak{U}_F) = mcu_F(\mathfrak{U}_F).$$

Proof. From Theorem 5.2.1 and construction of Σ_{min}. □

5.3 General Unification in the Restricted Flat Theory – Decidability

In this section we consider decidability of general unification problem in the restricted flat theory. Recall that the axiom set of the theory has the following form:

$$E = \{f(\overline{x}, f(\overline{y_1}, x, \overline{y_2}), \overline{z}) \doteq f(\overline{x}, \overline{y_1}, x, \overline{y_2}, \overline{z})\}.$$

The unification problem has a form $t_1 \overset{?}{=}_{RF} t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, restricted flat flexible arity function symbols, free constants and free fixed arity function symbols. We denote it as $\mathfrak{U}_{RF}$.

The problem is to prove that $\mathfrak{U}_{RF}$ is decidable. To do so, as in the case of free theory, first we try to reduce $\mathfrak{U}_{RF}$ to a "simpler" unification problem such that $\mathfrak{U}_{RF}$ is solvable if and only if the reduced problem is solvable, and then try to prove that solvability of the reduced unification problem is decidable.

5.3.1 Reduction

Let $\mathfrak{U}_{RF}$ be $t_1 \overset{?}{=}_{RF} t_2$. First we reduce it to the set S_{RF}^{proj} of RF-unification problems by projection, where projection is defined as in the case of the free theory. We say that a substitution θ is positive on a set of variables V iff for all $\overline{x} \in V$ there is a binding $\overline{x} \leftarrow t_1, \ldots, t_n \ \in \theta$ such that $n \geq 1$ and for all i, $1 \leq i \leq n$, $t_i \notin \mathcal{SV}$.

Theorem 5.3.1. *$\mathfrak{U}_{RF}$ is solvable iff there exists a unification problem $\mathfrak{U}_{RF}^{proj} \in S_{RF}^{proj}$ which is solvable with a substitution which is positive on $vars(\mathfrak{U}_{RF})$.*

Proof. ($\Rightarrow$) Let θ be a solution of $\mathfrak{U}_{RF}$ and σ be the substitution $\{\overline{x} \leftarrow \ \mid \overline{x} \leftarrow \ \in \theta, \overline{x} \in svars(\mathfrak{U}_F)\}$. Then $\mathfrak{U}_{RF}\sigma \in S_{RF}^{proj}$. Let $\lambda = \theta \backslash \sigma$. Then $\theta = \sigma \circ \lambda$ and λ is a solution of $\mathfrak{U}_{RF}\sigma$ such that for all $\overline{x} \in vars(\mathfrak{U}_F)$, the binding $\overline{x} \leftarrow$ does not belong to λ.

($\Leftarrow$) Let $\mathfrak{U}_{RF}^{proj}$ be a unification problem in S_{RF}^{proj}, obtained from $\mathfrak{U}_{RF}$ by the projecting substitution σ and let λ be a solution of $\mathfrak{U}_{RF}^{proj}$ such that for all $\overline{x} \in vars(\mathfrak{U}_F)$, the binding $\overline{x} \leftarrow$ does not belong to λ. Then $\sigma \circ \lambda$ is a solution of $\mathfrak{U}_{RF}$. □

Now we define two operations on terms called respectively restricted flattening and weakly restricted flattening.

Restricted flattening replaces a term of the form $f(\tilde{t}, f(\tilde{s_1}, s, \tilde{s_2}), \tilde{r})$, where f is a restricted flat flexible arity symbol, $\tilde{t}$, $\tilde{s_1}$, $\tilde{s_2}$ and $\tilde{r}$ are (possibly empty) sequences of terms and $s \notin \mathcal{SV}$, with the term $f(\tilde{t}, \tilde{s_1}, s, \tilde{s_2}, \tilde{r})$. Given a term t, we denote by $restflt(t)$ the term obtained from t by restricted flattening all its subterms until possible. For any t we have $t \doteq_{RF} restflt(t)$.

Weakly restricted flattening replaces a term of the form $f(\tilde{t}, f(\tilde{s}), \tilde{r})$, where f is a restricted flat flexible arity symbol, $\tilde{t}$ and $\tilde{r}$ are (possibly empty) sequences of terms and $\tilde{s}$ is a non-empty sequence of terms, with the term $f(\tilde{t}, \tilde{s}, \tilde{r})$. Given a term t, we denote by $wrestflt(t)$ the term obtained from t by weakly restricted flattening all its subterms until possible. Note that in general it is not true that $t \doteq_{RF} wrestflt(t)$: for instance, if f is a restricted flat flexible arity symbol, then $wrestflt(f(a, f(\overline{x}), b) = f(a, \overline{x}, b)$, but not $f(a, f(\overline{x}), b) \doteq_{RF} f(a, \overline{x}, b)$. But we have the following result:

Theorem 5.3.2. *A unification problem* $\mathfrak{U}_{RF}$ *of the form* $t_1 \stackrel{?}{\doteq}_{RF} t_2$ *has a positive solution on* $vars(\mathfrak{U}_{RF})$ *iff* $wrestflt(t_1) \stackrel{?}{\doteq}_{RF} wrestflt(t_2)$ *has a positive solution on* $vars(\mathfrak{U}_{RF})$.

Proof. ($\Rightarrow$) Let σ be a positive RF-solution of $\mathfrak{U}_{RF}$ on $vars(\mathfrak{U}_{RF})$. Then for $i = 1, 2$ we have

$$\begin{aligned} & t_i\sigma \doteq_{RF} \\ & restflt(t_i\sigma) \doteq_{RF} \\ & restflt(wrestflt(t_i)\sigma) \doteq_{RF} \\ & wrestflt(t_i)\sigma. \end{aligned}$$

Thus, σ is a solution of $wrestflt(t_1) \stackrel{?}{\doteq}_{RF} wrestflt(t_2)$.

($\Leftarrow$) Symmetric to the direction ($\Rightarrow$). □

Thus, these theorems reduce the general RF-unifiability problem to the positive RF-unifiability problem for two weakly restricted flattened terms.

Let $\mathfrak{U}_{RF}^{wrf}$ be a general RF-unification problem $s_1 \stackrel{?}{\doteq}_{RF} s_2$, where s_1 and s_2 are terms in a weakly restricted flattened form. We want to prove that it is decidable whether $\mathfrak{U}_{RF}^{wrf}$ has a positive RF-unifier on $vars(\mathfrak{U}_{RF}^{wrf})$.

Let $f_1, \ldots, f_m$, $m \geq 1$, be all free flexible arity function symbols occurring in $\mathfrak{U}_{RF}^{wrf}$ and let $h_1, \ldots, h_k$, $k \geq 1$ be all restricted flat flexible arity symbols in $\mathfrak{U}_{RF}^{wrf}$. We reduce $\mathfrak{U}_{RF}^{wrf}$ to a reduced unification problem $\mathfrak{U}_{RF}^{red}$ by performing the following steps:

1. Introduce
   – a new flat flexible arity symbol seq;
   – a new unary function symbol g_i for each f_i ($1 \leq i \leq m$);
   – a new constant c_i for each h_i ($1 \leq i \leq m$).
2. Replace each term $f_i(t_1, \ldots, t_n)$ in the problem $\mathfrak{U}_{RF}^{wrf}$ by the term $g_i(seq(t_1, \ldots, t_n))$ ($n \geq 0$);
3. Replace each term $h_i()$ in $\mathfrak{U}_{RF}^{wrf}$ by c_i.

Note that sequence variables occur only as arguments of terms with the head $h_1, \ldots, h_k$ or *seq*. We impose *individual variable restrictions on terms* on $\mathfrak{U}^{red}_{RF}$ demanding that, for a solution θ of $\mathfrak{U}^{red}_{RF}$ and for any individual variable x, $x\theta$ must not have *seq* as the head.

Theorem 5.3.3. *$\mathfrak{U}^{wrf}_{RF}$ is positively solvable on $vars(\mathfrak{U}^{wrf}_{RF})$ iff $\mathfrak{U}^{red}_{RF}$ with individual variable restrictions on terms is positively solvable on $vars(\mathfrak{U}^{red}_{RF})$.*

Proof. Similar to the proof of Theorem 3.2.1. □

5.3.2 Decidability of the Reduced Problem

We prove decidability of the reduced unification problem $\mathfrak{U}^{red}_{RF}$ using Baader-Schulz combination method.

Let for all $1 \leq i \leq k$, Ω_i be the set $\{h_i\}$, Ω_{k+1} be the set $\{seq\}$, and let $\Omega_{k+2} = fix(\mathfrak{U}^{red}_{RF}) \cup const(\mathfrak{U}^{red}_{RF})$. Let for all $1 \leq i \leq k$, E_i be an equational theory with sequence variables over the signature Ω_i, E_{k+1} be an equational theory with sequence variables over the signature Ω_{k+1} and E_{k+2} be a free theory (without sequence variables) over Ω_{k+2}. Then we can consider $\mathfrak{U}^{red}_{RF}$ as a unification problem in the combined theory $E_1 \cup \cdots \cup E_{k+2}$. Since $\Omega_1, \ldots, \Omega_{k+2}$ are pairwise disjoint, by Theorem 3.2.3, in order to show that solvability of $\mathfrak{U}^{red}_{RF}$ in $E_1 \cup \cdots \cup E_{k+2}$ is decidable we need to show for each $1 \leq i \leq k+2$ that solvability of E_i-unification problem with linear constant restrictions is decidable.

For each $1 \leq i \leq k$, the decidability problem for E_i-unification with linear constant restrictions is equivalent to the decidability problem for word equations with linear constant restrictions and additional restrictions demanding that each variable of the equation must get a value of non-zero length in the solution. We can prove with the argument similar to the one used in the proof of Theorem 3.2.5 that the problem is decidable.

The decidability problem for E_{k+1}-unification with linear constant restrictions is equivalent to the decidability problem for word equations with linear constant restrictions and individual variable restrictions on length. Then, by Theorem 3.2.5 we have that solvability of the E_{k+1}-unification problem with linear constant restrictions is decidable.

The E_{k+2}-unification problem is a Robinson unification problem, solvability of which with linear constant restrictions is decidable (see [BS91]).

Thus, solvability of $\mathfrak{U}^{red}_{RF}$ is decidable, which, by Theorem 5.3.1, Theorem 5.3.2 and Theorem 5.3.3, implies that solvability of a general unification problem in the *RF*-theory is decidable.

5.4 General Unification in the Restricted Flat Theory – Unification Procedure

In this section we design a unification procedure to solve general unification problem $\mathfrak{U}_{RF}$ in a restricted flat theory.

We design the unification procedure as a tree generation process based on three basic steps: projection, weakly restricted flattening and transformation.

5.4.1 Projection

Projection in the restricted flat theory coincides with the projection in the free theory (see Figure 3.1). The only difference is that we change the subscript $\emptyset$ to RF.

5.4.2 Weakly Restricted Flattening

Weakly restricted flattening rule is defined on the Figure 5.7 below:

Weakly Restricted Flattening:	$s \stackrel{?}{=}_F t \rightsquigarrow \langle\langle wrestflt(s) \stackrel{?}{=}_F wrestflt(t), \varepsilon\rangle\rangle$

Fig. 5.7. Weakly restricted flattening rule.

5.4.3 Transformation

Each of the transformation rules for unification have one of the following forms: $\mathfrak{U} \rightsquigarrow \bot$ or $\mathfrak{U} \rightsquigarrow \langle\langle \mathfrak{S}_1, \sigma_1\rangle, \ldots, \langle \mathfrak{S}_n, \sigma_n\rangle\rangle$ where each of the successors $\mathfrak{S}_i$ is either $\top$ or a new unification problem.

Let $\mathfrak{U}$ have a form $s \stackrel{?}{=}_{RF} t$, where s and t are either

– identical terms,
– terms with different heads, or
– terms whose heads are not restricted flat symbols

then $\mathfrak{U}$ is transformed by one of the transformation rules on Figure 3.2 (changing the subscript $\emptyset$ to RF). Otherwise we define transformation rules for $\mathfrak{U}$ on Figure 5.8, Figure 5.9 and Figure 5.10.

$$
\begin{array}{lll}
\textsf{SuccessRF}: & x \stackrel{?}{\doteq}_{RF} h(x) \rightsquigarrow \langle\langle \top,\ \{x \leftarrow h(x)\}\rangle\rangle. & \\
& x \stackrel{?}{\doteq}_{RF} t \rightsquigarrow \langle\langle \top,\ \{x \leftarrow t\}\rangle\rangle, & \text{if } x \notin ivars(t) \text{ and } t \neq h(x). \\
& h(x) \stackrel{?}{\doteq}_{RF} x \rightsquigarrow \langle\langle \top,\ \{x \leftarrow h(x)\}\rangle\rangle. & \\
& t \stackrel{?}{\doteq}_{RF} x \rightsquigarrow \langle\langle \top,\ \{x \leftarrow t\}\rangle\rangle, & \text{if } x \notin ivars(t) \text{ and } t \neq h(x). \\
& & \\
\textsf{FailureRF}: & x \stackrel{?}{\doteq}_{RF} t \rightsquigarrow \bot, & \text{if } t \neq x,\ t \neq g(x) \\
& \quad \text{and } x \in ivars(t). & \\
& t \stackrel{?}{\doteq}_{RF} x \rightsquigarrow \bot, & \text{if } t \neq x,\ t \neq g(x) \\
& \quad \text{and } x \in ivars(t). & \\
& h() \stackrel{?}{\doteq}_{RF} h(t_1, \tilde{t}) \rightsquigarrow \bot. & \\
& h(t_1, \tilde{t}) \stackrel{?}{\doteq}_{RF} h() \rightsquigarrow \bot. & \\
& h(\overline{x}, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(s_1, \tilde{s}) \rightsquigarrow \bot, & \text{if } s_1 \neq \overline{x},\ s_1 \neq g(\overline{x}) \\
& \quad \text{and } \overline{x} \in svars(s_1). & \\
& h(s_1, \tilde{s}) \stackrel{?}{\doteq}_{RF} h(\overline{x}, \tilde{t}) \rightsquigarrow \bot, & \text{if } s_1 \neq \overline{x},\ s_1 \neq g(\overline{x}) \\
& \quad \text{and } \overline{x} \in svars(s_1). & \\
& h(t_1, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(s_1, \tilde{s}) \rightsquigarrow \bot, & \text{if } t_1 \stackrel{?}{\doteq}_{RF} s_1 \rightsquigarrow \bot. \\
& & \\
\textsf{EliminateRF}: & h(t_1, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(s_1, \tilde{s}) \rightsquigarrow & \text{if } t_1, s_1 \notin \mathcal{IV} \cup \mathcal{SV} \text{ and} \\
& \quad \langle\langle h(\tilde{t}\sigma) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma),\ \sigma\rangle\rangle, & \quad t_1 \stackrel{?}{\doteq}_{RF} s_1 \rightsquigarrow \langle\langle \top,\ \sigma\rangle\rangle. \\
& h(v, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(v, \tilde{s}) \rightsquigarrow & \text{where } v \in \mathcal{IV} \cup \mathcal{SV}. \\
& \quad \langle\langle h(\tilde{t}) \stackrel{?}{\doteq}_{RF} h(\tilde{s}),\ \varepsilon\rangle\rangle. & \\
& h(x, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(y, \tilde{s}) \rightsquigarrow & \text{where } x \neq y \text{ and} \\
& \quad \langle\langle h(\tilde{t}\sigma_1) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_1),\ \sigma_1\rangle, & \quad \sigma_1 = \{x \leftarrow y\}, \\
& \quad \langle h(\tilde{t}\sigma_2) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_2),\ \sigma_2\rangle, & \quad \sigma_2 = \{x \leftarrow h(y)\}, \\
& \quad \langle h(x, \tilde{t}\sigma_3) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_3),\ \sigma_3\rangle, & \quad \sigma_3 = \{x \leftarrow h(y, x)\}, \\
& \quad \langle h(\tilde{t}\sigma_4) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_4),\ \sigma_4\rangle, & \quad \sigma_4 = \{y \leftarrow h(x)\}, \\
& \quad \langle h(\tilde{t}\sigma_5) \stackrel{?}{\doteq}_{RF} h(y, \tilde{s}\sigma_5),\ \sigma_5\rangle\rangle, & \quad \sigma_5 = \{y \leftarrow h(x, y)\}. \\
& h(x, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(s_1, \tilde{s}) \rightsquigarrow & \text{where } s_1 \notin \mathcal{IV} \cup \mathcal{SV} \text{ and} \\
& \quad \langle\langle h(\tilde{t}\sigma_1) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_1),\ \sigma_1\rangle, & \quad \sigma_1 = \{x \leftarrow s_1\}, \\
& \quad \langle h(\tilde{t}\sigma_2) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_2),\ \sigma_2\rangle, & \quad \sigma_2 = \{x \leftarrow h(s_1)\}, \\
& \quad \langle h(x, \tilde{t}\sigma_3) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_3),\ \sigma_3\rangle\rangle, & \quad \sigma_3 = \{x \leftarrow h(s_1, x)\}. \\
& h(t_1, \tilde{t}) \stackrel{?}{\doteq}_{RF} h(x, \tilde{s}) \rightsquigarrow & \text{where } t_1 \notin \mathcal{IV} \cup \mathcal{SV} \text{ and} \\
& \quad \langle\langle h(\tilde{t}\sigma_1) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_1),\ \sigma_1\rangle, & \quad \sigma_1 = \{x \leftarrow t_1\}, \\
& \quad \langle h(\tilde{t}\sigma_2) \stackrel{?}{\doteq}_{RF} h(\tilde{s}\sigma_2),\ \sigma_2\rangle, & \quad \sigma_2 = \{x \leftarrow h(t_1)\}, \\
& \quad \langle h(\tilde{t}\sigma_3) \stackrel{?}{\doteq}_{RF} h(x, \tilde{s}\sigma_3),\ \sigma_3\rangle\rangle, & \quad \sigma_3 = \{x \leftarrow h(t_1, x)\}.
\end{array}
$$

Fig. 5.8. Transformation rules for the RF-unification. $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h and g are restricted flat flexible arity symbols.

EliminateRF (continued):

$$h(\overline{x},\tilde{t})\stackrel{?}{=}_{RF}h(y,\tilde{s}) \rightsquigarrow \quad \text{where}$$
$$\begin{array}{ll}
\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_1),\ \sigma_1\rangle, & \sigma_1=\{\overline{x}\leftarrow y\},\\
\langle h(\tilde{t}\sigma_2)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_2),\ \sigma_2\rangle, & \sigma_2=\{\overline{x}\leftarrow h(y)\},\\
\langle h(\overline{x},\tilde{t}\sigma_3)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_3),\ \sigma_3\rangle, & \sigma_3=\{\overline{x}\leftarrow h(y,\overline{x})\},\\
\langle h(\overline{x},\tilde{t}\sigma_4)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_4),\ \sigma_4\rangle, & \sigma_4=\{\overline{x}\leftarrow y,\overline{x}\},\\
\langle h(\overline{x},\tilde{t}\sigma_5)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_5),\ \sigma_5\rangle, & \sigma_5=\{\overline{x}\leftarrow y,h(\overline{x})\},\\
\langle h(\overline{x},\tilde{t}\sigma_6)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_6),\ \sigma_6\rangle, & \sigma_6=\{\overline{x}\leftarrow h(y),\overline{x}\},\\
\langle h(\overline{x},\tilde{t}\sigma_7)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_7),\ \sigma_7\rangle, & \sigma_7=\{\overline{x}\leftarrow h(y,\overline{x})\},\\
\langle h(\tilde{t}\sigma_8)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_8),\ \sigma_8\rangle, & \sigma_8=\{y\leftarrow h(\overline{x})\},\\
\langle h(\tilde{t}\sigma_9)\stackrel{?}{=}_{RF}h(y,\tilde{s}\sigma_9),\ \sigma_9\rangle\rangle, & \sigma_9=\{y\leftarrow h(\overline{x},y)\}.
\end{array}$$

$$h(y,\tilde{t})\stackrel{?}{=}_{RF}h(\overline{x},\tilde{s}) \rightsquigarrow \quad \text{where}$$
$$\begin{array}{ll}
\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_1),\ \sigma_1\rangle, & \sigma_1=\{\overline{x}\leftarrow y\},\\
\langle h(\tilde{t}\sigma_2)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_2),\ \sigma_2\rangle, & \sigma_2=\{\overline{x}\leftarrow h(y)\},\\
\langle h(\tilde{t}\sigma_3)\stackrel{?}{=}_{RF}h(\overline{x},\tilde{s}\sigma_3),\ \sigma_3\rangle, & \sigma_3=\{\overline{x}\leftarrow h(y,\overline{x})\},\\
\langle h(\tilde{t}\sigma_4)\stackrel{?}{=}_{RF}h(\overline{x},\tilde{s}\sigma_4),\ \sigma_4\rangle, & \sigma_4=\{\overline{x}\leftarrow y,\overline{x}\},\\
\langle h(\tilde{t}\sigma_5)\stackrel{?}{=}_{RF}h(\overline{x},\tilde{s}\sigma_5),\ \sigma_5\rangle, & \sigma_5=\{\overline{x}\leftarrow y,h(\overline{x})\},\\
\langle h(\tilde{t}\sigma_6)\stackrel{?}{=}_{RF}h(\overline{x},\tilde{s}\sigma_6),\ \sigma_6\rangle, & \sigma_6=\{\overline{x}\leftarrow h(y),\overline{x}\},\\
\langle h(\tilde{t}\sigma_7)\stackrel{?}{=}_{RF}h(\overline{x},\tilde{s}\sigma_7),\ \sigma_7\rangle, & \sigma_7=\{\overline{x}\leftarrow h(y,\overline{x})\},\\
\langle h(\tilde{t}\sigma_8)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_8),\ \sigma_8\rangle, & \sigma_8=\{y\leftarrow h(\overline{x})\},\\
\langle h(y,\tilde{t}\sigma_9)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_9),\ \sigma_9\rangle\rangle, & \sigma_9=\{y\leftarrow h(\overline{x},y)\}.
\end{array}$$

$$h(\overline{x},\tilde{t})\stackrel{?}{=}_{RF}h(\overline{y},\tilde{s}) \rightsquigarrow \quad \text{where } \overline{x}\neq\overline{y} \text{ and}$$
$$\begin{array}{ll}
\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_1),\ \sigma_1\rangle, & \sigma_1=\{\overline{x}\leftarrow \overline{y}\},\\
\langle h(\tilde{t}\sigma_2)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_2),\ \sigma_2\rangle, & \sigma_2=\{\overline{x}\leftarrow h(\overline{y})\},\\
\langle h(\overline{x},\tilde{t}\sigma_3)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_3)\rangle,\ \sigma_3\rangle, & \sigma_3=\{\overline{x}\leftarrow h(\overline{y},\overline{x})\},\\
\langle h(\overline{x},\tilde{t}\sigma_4)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_4),\ \sigma_4\rangle, & \sigma_4=\{\overline{x}\leftarrow \overline{y},\overline{x}\},\\
\langle h(\overline{x},\tilde{t}\sigma_5)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_5),\ \sigma_5\rangle, & \sigma_5=\{\overline{x}\leftarrow \overline{y},h(\overline{x})\},\\
\langle h(\overline{x},\tilde{t}\sigma_6)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_6),\ \sigma_6\rangle, & \sigma_6=\{\overline{x}\leftarrow h(\overline{y}),\overline{x}\},\\
\langle h(\overline{x},\tilde{t}\sigma_7)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_7),\ \sigma_7\rangle, & \sigma_7=\{\overline{x}\leftarrow h(\overline{y},\overline{x})\},\\
\langle h(\tilde{t}\sigma_8)\stackrel{?}{=}_{RF}h(\tilde{s}\sigma_8),\ \sigma_8\rangle, & \sigma_8=\{\overline{y}\leftarrow h(\overline{x})\},\\
\langle h(\tilde{t}\sigma_9)\stackrel{?}{=}_{RF}h(\overline{y},\tilde{s}\sigma_9),\ \sigma_9\rangle, & \sigma_9=\{\overline{y}\leftarrow h(\overline{x},\overline{y})\},\\
\langle h(\tilde{t}\sigma_{10})\stackrel{?}{=}_{RF}h(\overline{y},\tilde{s}\sigma_{10}),\ \sigma_{10}\rangle, & \sigma_{10}=\{\overline{y}\leftarrow \overline{x},\overline{y}\},\\
\langle h(\tilde{t}\sigma_{11})\stackrel{?}{=}_{RF}h(\overline{y},\tilde{s}\sigma_{11}),\ \sigma_{11}\rangle, & \sigma_{11}=\{\overline{y}\leftarrow \overline{x},h(\overline{y})\},\\
\langle h(\tilde{t}\sigma_{12})\stackrel{?}{=}_{RF}h(\overline{y},\tilde{s}\sigma_{12}),\ \sigma_{12}\rangle, & \sigma_{12}=\{\overline{y}\leftarrow h(\overline{x}),\overline{y}\},\\
\langle h(\tilde{t}\sigma_{13})\stackrel{?}{=}_{RF}h(\overline{y},\tilde{s}\sigma_{13}),\ \sigma_{13}\rangle\rangle, & \sigma_{13}=\{\overline{y}\leftarrow h(\overline{x}),h(\overline{y})\}.
\end{array}$$

Fig. 5.9. Transformation rules for the RF-unification (continued). $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h is a restricted flat symbol.

EliminateRF (continued):	$h(\overline{x},\tilde{t})\stackrel{?}{\doteq}_{RF}h(s_1,\tilde{s})$	$s_1 \notin \mathcal{IV}\cup\mathcal{SV}$, $\overline{x}\notin svars(s_1)$
	$\rightsquigarrow$	or $s_1 = g(\overline{x})$ and
	$\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{\overline{x}\leftarrow s_1\},$
	$\langle h(\tilde{t}\sigma_2)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{\overline{x}\leftarrow h(s_1)\},$
	$\langle h(\overline{x},\tilde{t}\sigma_3)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_3))\rangle,\ \sigma_3\rangle,$	$\sigma_3 = \{\overline{x}\leftarrow h(s_1,\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_4)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{\overline{x}\leftarrow s_1,\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_5)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_5),\ \sigma_5\rangle,$	$\sigma_5 = \{\overline{x}\leftarrow s_1,h(\overline{x})\},$
	$\langle h(\overline{x},\tilde{t}\sigma_6)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{\overline{x}\leftarrow h(s_1),\overline{x}\},$
	$\langle h(\overline{x},\tilde{t}\sigma_7)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_7),\ \sigma_7\rangle\rangle,$	$\sigma_7 = \{\overline{x}\leftarrow h(s_1,\overline{x})\}.$
	$h(t_1,\tilde{t})\stackrel{?}{\doteq}_{RF}h(\overline{x},\tilde{s})$	$t_1 \notin \mathcal{IV}\cup\mathcal{SV}$, $\overline{x}\notin svars(t_1)$
	$\rightsquigarrow$	or $t_1 = g(\overline{x})$ and
	$\langle\langle h(\tilde{t}\sigma_1)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$\sigma_1 = \{\overline{x}\leftarrow t_1\},$
	$\langle h(\tilde{t}\sigma_2)\stackrel{?}{\doteq}_{RF}h(\tilde{s}\sigma_2),\ \sigma_2\rangle,$	$\sigma_2 = \{\overline{x}\leftarrow h(t_1)\},$
	$\langle h(\tilde{t}\sigma_3)\stackrel{?}{\doteq}_{RF}h(\overline{x},\tilde{s}\sigma_3))\rangle,\ \sigma_3\rangle,$	$\sigma_3 = \{\overline{x}\leftarrow h(t_1,\overline{x})\},$
	$\langle h(\tilde{t}\sigma_4)\stackrel{?}{\doteq}_{RF}h(\overline{x},\tilde{s}\sigma_4),\ \sigma_4\rangle,$	$\sigma_4 = \{\overline{x}\leftarrow t_1,\overline{x}\},$
	$\langle h(\tilde{t}\sigma_5)\stackrel{?}{\doteq}_{RF}h(\overline{x},\tilde{s}\sigma_5),\ \sigma_5\rangle,$	$\sigma_5 = \{\overline{x}\leftarrow t_1,h(\overline{x})\},$
	$\langle h(\tilde{t}\sigma_6)\stackrel{?}{\doteq}_{RF}h(\overline{x},\tilde{s}\sigma_6),\ \sigma_6\rangle,$	$\sigma_6 = \{\overline{x}\leftarrow h(t_1),\overline{x}\},$
	$\langle h(\tilde{t}\sigma_7)\stackrel{?}{\doteq}_{RF}h(\overline{x},\tilde{s}\sigma_7),\ \sigma_7\rangle\rangle,$	$\sigma_7 = \{\overline{x}\leftarrow h(t_1,\overline{x})\}.$
SplitRF:	$h(t_1,\tilde{t})\stackrel{?}{\doteq}_{RF}h(s_1,\tilde{s})\rightsquigarrow$	if $t_1,s_1 \notin \mathcal{IV}\cup\mathcal{SV}$ and
	$\langle\langle h(r_1,\tilde{t}\sigma_1)\stackrel{?}{\doteq}_{RF}h(q_1,\tilde{s}\sigma_1),\ \sigma_1\rangle,$	$t_1\stackrel{?}{\doteq}_{RF}s_1\rightsquigarrow$
	$\ldots,$	$\langle\langle r_1\stackrel{?}{\doteq}_{RF}q_1,\ \sigma_1\rangle,\ldots$
	$\langle h(r_k,\tilde{t}\sigma_k)\stackrel{?}{\doteq}_{RF}h(q_k,\tilde{s}\sigma_k),\ \sigma_k\rangle\rangle$	$\langle r_k\stackrel{?}{\doteq}_{RF}q_k,\ \sigma_k\rangle\rangle.$

Fig. 5.10. Transformation rules for the RF-unification (continued). $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms. h and g are restricted flat symbols.

5.4.4 Tree Generation

We design the unification procedure for the restricted flat theory with sequence variables and flexible arity symbols as a tree generation process. The single steps in this process are projection, weakly restricted flattening and transformation for the restricted flat theory.

Each node of the tree is labeled either with a unification problem, $\top$ or $\bot$. The edges of the tree are labeled by substitutions. The nodes labeled with $\top$ or $\bot$ are terminal nodes. The nodes labeled with unification problems are non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Given a nonterminal node, let $\mathfrak{U}$ be a unification problem attached to it. First, we decide whether $\mathfrak{U}$ is unifiable. If the answer is negative, we replace $\mathfrak{U}$ with the new label $\bot$. If $\mathfrak{U}$ is unifiable, we apply projection, weakly restricted flattening or transformation on $\mathfrak{U}$ and get $\langle\langle \mathfrak{S}_1, \sigma_1\rangle, \ldots, \langle \mathfrak{S}_n, \sigma_n\rangle\rangle$. Then the node $\mathfrak{U}$ has n children, labeled respectively with $\mathfrak{S}_1, \ldots, \mathfrak{S}_n$ and the edge to the $\mathfrak{S}_i$ node is labeled with σ_i $(1 \leq i \leq n)$. The set $\{\sigma_1, \ldots, \sigma_n\}$ is denoted by $sub(\mathfrak{U})$.

We design the general unification procedure for $\mathfrak{U}_{RF}$ as a breadth first (level by level) tree generation process. We label the root of the tree with $\mathfrak{U}_{RF}$ (zero level). First level nodes of the tree are obtained from the original problem by projection. Starting from the second level, we apply only weakly restricted flattening and transformation steps to a unification problem of each node, first flattening it and then transforming the flattened problem, thus getting new successor nodes. The branch which ends with a node labeled by $\top$ is called a successful branch. The branch which ends with a node labeled by $\bot$ is a failed branch. For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node, flatten all the terms in the codomain of the composition by the weakly restricted flattening operation and attach the obtained substitution to the node together with the unification problem the node was labeled with. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N. Let $\Sigma(\mathfrak{U}_{RF})$ be the set of all substitutions associated with the $\top$ nodes. We call the tree a unification tree for $\mathfrak{U}_{RF}$ and denote it $utree(\mathfrak{U}_{RF})$.

Our goal is to prove that $\Sigma(\mathfrak{U}_{RF})$ is a complete set of RF-unifiers of $\mathfrak{U}_{RF}$.

Lemma 5.4.1. *For every RF-unifier ϕ of a general unification problem $\mathfrak{U}_{RF}$ the corresponding branch β in $utree(\mathfrak{U}_{RF})$ has the following property: if $\mathfrak{P}$ is a unification problem occurring in β with the associated substitution θ, then $\theta \ll_{RF}^{vars(\mathfrak{U}_{RF})} \phi$.*

Proof. Similar to the proof of Lemma 5.2.1. □

Theorem 5.4.1. *$\Sigma(\mathfrak{U}_{RF})$ is a complete set of RF-unifiers for $\mathfrak{U}_{RF}$.*

Proof. The theorem follows from Lemma 5.2.1 by the definition of Σ. $\square$

The following example shows that $\Sigma(\mathfrak{U}_{RF})$ is not minimal:

Example 5.4.1. Let f be a RF-symbol and $\mathfrak{U}_{RF}$ be $f(a,\overline{x})\stackrel{?}{=}_{RF}f(\overline{y},b)$. Then among the other unifiers $\Sigma(\mathfrak{U}_{RF})$ contains the following two substitutions:

$$\sigma = \{\overline{y} \leftarrow a, \overline{y},\ \overline{x} \leftarrow \overline{y}, b\},$$

$$\lambda = \{\overline{y} \leftarrow a, f(\overline{y}),\ \overline{x} \leftarrow f(\overline{y}), b\}.$$

Clearly, $\sigma \ll_{RF}^{vars(\mathfrak{U}_{RF})} \lambda$, because $\lambda = \sigma \circ \{\overline{y} \leftarrow f(\overline{y})\}$.

Now we show that the minimal complete set of RF-unifiers of $\mathfrak{U}_{RF}$ exists. Let t be a weakly restricted flattened term and $t\theta$ be an instance of t, also in the weakly restricted flattened form. Then

– if $t\theta$ is a term, then $depth(t\theta) \geq depth(t)$,
– if $t\theta$ is a sequence of terms $s_1, \ldots, s_n$, then $max(depth(s_1), \ldots, depth(s_n)) \geq depth(t)$.

Lemma 5.4.2. *Let $\theta_1 >_{RF}^{vars(\mathfrak{U}_{RF})} \theta_2 >_{RF}^{vars(\mathfrak{U}_{RF})} \cdots$ be a strictly decreasing chain of unifiers in $\Sigma(\mathfrak{U}_{RF})$. Then the chain is finite.*

Proof. Analogous to the proof of Lemma 5.2.2. $\square$

This lemma implies the existence of a minimal subset of $\Sigma(\mathfrak{U}_{RF})$. Now we can refine the unification procedure exactly in the same way as we did for the flat theory, to compute the minimal complete set of unifiers. Then, if $\Sigma_{min}(\mathfrak{U}_{RF})$ denotes the set of all substitutions which are associated with $\top$ nodes in the refined RF-unification tree, we have the following theorem:

Theorem 5.4.2. *Let $\mathfrak{U}_{RF}$ be a general RF-unification problem. Then*

$$\Sigma_{min}(\mathfrak{U}_{RF}) = mcu_{RF}(\mathfrak{U}_{RF}).$$

One can easily check by inspecting the transformation rules for RF-unification that matching modulo RF is finitary. Thus, for instance, if we try to unify terms $f(\overline{x})$ and $f(a)$ modulo RF, f being a restricted flat symbol, we get a solution $\{\{\overline{x} \leftarrow a\}, \{\overline{x} \leftarrow f(a)\}\}$, while unifying them modulo F with flat f leads to a infinite minimal complete set of unifiers (see Example 5.2.2).

But restriction has a drawback as well. For instance, the unification problem $f(\overline{x}, g(\overline{x}))\stackrel{?}{=}_{RF}f(a, b, g(a, f(), f(), b))$ with restricted flat f does not have a solution, while $f(\overline{x}, g(\overline{x}))\stackrel{?}{=}_{F}f(a, b, g(a, f(), f(), b))$ with flat f has (see Example 5.2.2).

5.5 General Unification in the Orderless Theory – Decidability

In this section we consider decidability of general unification problem in the orderless theory with sequence variables and flexible arity symbols. Recall that the axiom set of the theory has the following form:

$$E = \{f(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq f(\overline{x}, y, \overline{y}, x, \overline{z})\}.$$

The problem has a form $t_1 \stackrel{?}{\doteq}_O t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, orderless flexible arity function symbols, free constants and free fixed arity function symbols. We denote it by $\mathfrak{U}_O$ and prove that $\mathfrak{U}_O$ is decidable.

First we define the notion of a permuted version of a term as follows: Let $f_1(s_1^1, \ldots, s_{n_1}^1), \ldots, f_k(s_1^k, \ldots, s_{n_k}^k)$ be a sequence of all subterms of a term t with orderless flexible arity heads $f_1, \ldots, f_k$ and let for all $1 \leq i \leq k$, the sequence π_i be a permutation of the sequence $1, \ldots, n_i$. Then the *permuted version of a term t with respect to the permutations* $\pi_1, \ldots, \pi_k$, denoted by $perm(t, f_1, \pi_1, \ldots, f_k, \pi_k)$, is called the term obtained from t by replacing all the terms $f_1(s_1^1, \ldots, s_{n_1}^1), \ldots, f_k(s_1^k, \ldots, s_{n_k}^k)$ in t simultaneously with the terms $f_1(s_{\pi_1(1)}^1, \ldots, s_{\pi_1(n_1)}^1), \ldots, f_k(s_{\pi_k(1)}^k, \ldots, s_{\pi_k(n_k)}^k)$.

Let $perm(t)$ be the set of all permuted versions of a term t.

Let $\mathfrak{U}_O$ be $t_1 \stackrel{?}{\doteq}_O t_2$. The set of all intermediate $\emptyset$-unification problems for $\mathfrak{U}_O$, denoted as $S_\emptyset^{int}$, is defined as follows:

$$S_\emptyset^{int} = \{s_1 \stackrel{?}{\doteq}_\emptyset s_2 \mid s_1 \in perm(t_1) \text{ and } s_2 \in perm(t_2)\}.$$

The set $S_\emptyset^{int}$ is finite. These definitions lead to the following result:

Theorem 5.5.1. *$\mathfrak{U}_O$ is solvable iff there exists an $\mathfrak{U}_\emptyset^{int} \in S_\emptyset^{int}$ which is solvable.*

This theorem reduces the solvability problem for an orderless theory with sequence variables and flexible arity symbols to the solvability problem for a free theory with sequence variables and flexible arity symbols. As we showed in Section 3.2, the latter problem is decidable. Thus, solvability of general unification problem in an O-theory with sequence variables and flexible arity symbols is decidable.

5.6 General Unification in the Orderless Theory – Unification Procedure

The straightforward way to design a unification procedure to solve general unification problem $\mathfrak{U}_O$ in an orderless theory with sequence variables and flexible arity symbols would follow the schema we used in the previous section to prove decidability: apply $\emptyset$-inification procedure on all possible permuted versions of $\mathfrak{U}_O$. Such a procedure would guarantee correctness, completeness and, after corresponding postprocessing, minimality of the solution set. However, it would be highly inefficient. In this section we design the unification procedure in a different manner, based on ideas of Stickel's method for AC-unification [Sti81]. The procedure is a tree generation process based on three basic steps: projection, transformation and substitution processing.

5.6.1 Projection

Projection for a general unification problem in an orderless theory with sequence variables and flexible arity symbols coincides with the projection for the general unification problem in the free theory with sequence variables and flexible arity symbols.

5.6.2 Transformation

If $\mathfrak{U}_O$ has a form $s \stackrel{?}{=}_O t$, where s and t are either

- identical terms,
- terms with different heads, or
- terms such that at least one of them has non-orderless head

then $\mathfrak{U}_O$ is transformed by one of the transformation rules on Figure 3.2 (changing the subscript $\emptyset$ to O). Otherwise we define transformation rules for O-unification on Figure 5.11.

5.6.3 Solving Unification problem for Terms with the Same Orderless Head

In this subsection we design a procedure $\mathsf{OrdUnif}$ to solve a unification problem of the form $f(t_1, \tilde{t}) \stackrel{?}{=}_O f(s_1, \tilde{s})$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, Orderless flexible arity symbols, free constants and free fixed arity function symbols. The function symbol f is an orderless flexible arity function symbol. If $t_1 \in \mathcal{SV}$, then $\tilde{t}$ is not an empty sequence and if $s_1 \in \mathcal{SV}$, then $\tilde{s}$ is not an empty sequence. The procedure $\mathsf{OrdUnif}$ is very similar to the one described by Stickel [Sti81] for AC-unification and consists of the following steps below. Let $\mathfrak{U}_O^0$ be $f(t_1, \tilde{t}) \stackrel{?}{=}_O f(s_1, \tilde{s})$.

1. Eliminate all pairs of common arguments from $f(t_1, \tilde{t})$ and $f(s_1, \tilde{s})$, removing them a pair at a time, one from each of the argument list. Denote the obtained unification problem with $\mathfrak{U}_O^1$. Let it have a form $f(\tilde{r}) \stackrel{?}{=}_O f(\tilde{q})$. Then:

FailureOrd:	$f() \stackrel{?}{=}_O f(s_1, \tilde{s}) \rightsquigarrow \bot$.	
	$f(t_1, \tilde{t}) \stackrel{?}{=}_O f() \rightsquigarrow \bot$.	
	$f(\overline{x}) \stackrel{?}{=}_O f(s_1, \ldots, s_n) \rightsquigarrow \bot$	if $\overline{x} \in svars(f(s_1, \ldots, s_n))$ and either $n > 1$ or $n = 1$, $s_1 \neq \overline{x}$.
	$f(t_1, \ldots, t_n) \stackrel{?}{=}_O f(\overline{x}) \rightsquigarrow \bot$	if $\overline{x} \in svars(f(t_1, \ldots, t_n))$ and either $n > 1$ or $n = 1$, $t_1 \neq \overline{x}$.
EliminateOrd:	$f(\overline{x}) \stackrel{?}{=}_O f(s_1, \ldots, s_n) \rightsquigarrow$	if $\overline{x} \notin svars(f(s_1, \ldots, s_n))$
	$\langle\langle f() \stackrel{?}{=}_O f(s_2, \ldots, s_n)\sigma_1,\ \sigma_1\rangle$,	$\sigma_1 = \{\overline{x} \leftarrow s_1\}$
	$\langle f(\overline{x}) \stackrel{?}{=}_O f(s_2, \ldots, s_n)\sigma_2,\ \sigma_2\rangle$,	$\sigma_2 = \{\overline{x} \leftarrow s_1, \overline{x}\}$
	$\langle f(\overline{x}) \stackrel{?}{=}_O f(s_1, s_3 \ldots, s_n)\sigma_3, \sigma_3\rangle$,	$\sigma_3 = \{\overline{x} \leftarrow s_2, \overline{x}\}$
	$\ldots$	$\ldots$
	$\langle f(\overline{x}) \stackrel{?}{=}_O f(s_1, \ldots, s_{n-1})\sigma_{n+1}, \sigma_{n+1}\rangle$,	$\sigma_{n+1} = \{\overline{x} \leftarrow s_n, \overline{x}\}$.
	$f(t_1, \ldots, t_n) \stackrel{?}{=}_O f(\overline{x}) \rightsquigarrow$	if $\overline{x} \notin svars(f(t_1, \ldots, t_n))$
	$\langle\langle f(t_2, \ldots, t_n) \stackrel{?}{=}_O f()\sigma_1,\ \sigma_1\rangle$,	$\sigma_1 = \{\overline{x} \leftarrow t_1\}$
	$\langle f(t_2, \ldots, t_n) \stackrel{?}{=}_O f(\overline{x})\sigma_2,\ \sigma_2\rangle$,	$\sigma_2 = \{\overline{x} \leftarrow t_1, \overline{x}\}$
	$\langle f(t_1, t_3 \ldots, t_n) \stackrel{?}{=}_O f(\overline{x})\sigma_3, \sigma_3\rangle$,	$\sigma_3 = \{\overline{x} \leftarrow t_2, \overline{x}\}$
	$\ldots$	$\ldots$
	$\langle f(t_1, \ldots, t_{n-1}) \stackrel{?}{=}_O f(\overline{x})\sigma_{n+1}, \sigma_{n+1}\rangle$,	$\sigma_{n+1} = \{\overline{x} \leftarrow t_n, \overline{x}\}$.
SplitOrd:	$f(t_1, \ldots, t_n) \stackrel{?}{=}_O f(s_1, \ldots, s_m) \rightsquigarrow$	where $n \geq 1$, $m \geq 1$,
	$\mathsf{OrdUnif}(f(t_1, \ldots, t_n) \stackrel{?}{=}_O f(t_1, \ldots, t_n))$,	if $t_1 \in \mathcal{SV}$, then $n > 1$, if $s_1 \in \mathcal{SV}$, then $m > 1$, and OrdUnif is defined in Subsection 5.6.3

Fig. 5.11. Transformation rules for O-unification. $\tilde{t}$ and $\tilde{s}$ are possibly empty sequences of terms, f is an orderless flexible arity function symbol.

- If both $\tilde{r}$ and $\tilde{q}$ are empty sequences of terms, then $\mathsf{OrdUnif}(\mathfrak{U}_O) = \langle\langle\top, \varepsilon\rangle\rangle$.
- If only one of $\tilde{r}$ and $\tilde{q}$ is the empty sequence, then $\mathsf{OrdUnif}(\mathfrak{U}_O) = \bot$.
- If none of $\tilde{r}$ and $\tilde{q}$ is the empty sequence of terms, then go to Step 2.

2. If $\mathfrak{U}_O^1$ contains no sequence variables, then generate all possible permuted versions $\mathfrak{P}_1, \dots, \mathfrak{P}_n$ of $\mathfrak{U}_O^1$ and go to Step 7. Otherwise generate a generalization of $\mathfrak{U}_O^1$ replacing each a non-variable term occurring among $\tilde{r}$ or $\tilde{q}$ by a new individual variable. This step is called *variable abstraction*. We denote the unification problem obtained from $\mathfrak{U}_O^1$ by the variable abstraction by $\mathfrak{U}_O^2$. The terms in $\mathfrak{U}_O^2$ to be unified have only sequence or individual variables as their arguments.
3. Associate to $\mathfrak{U}_O^2$ a linear Diophantine equation *dio*, representing the number and multiplicity of (sequence and individual) variables in each term in the following way: Let us represent $\mathfrak{U}_O^2$ in the form

$$f(v_1^{N_1}, \dots, v_k^{N_k}) \stackrel{?}{=}_O f(w_1^{M_1}, \dots, w_l^{M_l}),$$

where
- all N_i-s, $1 \le i \le k$, are positive integers;
- all M_i-s, $1 \le i \le l$, are positive integers;
- v^N is a abbreviation for the sequence $v, \dots, v$ of length N;
- for all i, $1 \le i \le k$, $v_i \in \mathcal{IV} \cup \mathcal{SV}$;
- for all i, $1 \le i \le l$, $w_i \in \mathcal{IV} \cup \mathcal{SV}$;
- all v_i-s, $1 \le i \le k$, are distinct;
- all w_i-s, $1 \le i \le l$, are distinct;

Then *dio* is the following linear Diophantine equation with positive integer coefficients

$$N_1V_1 + \dots + N_kV_k = M_1W_1 + \dots + M_lW_l.$$

We say that for all i, $1 \le i \le k$, the variable V_i corresponds to v_i and for all i, $1 \le i \le l$, the variable W_i corresponds to w_i.
4. We are interested in solving *dio*. An efficient algorithm for solving systems of linear Diophantine Equations is described in [CD94]. Applying the algorithm on *dio* we obtain a finite basis B in the set S of non-negative solutions of *dio*. B is a minimal subset of S such that every solution in S can be written as a linear combination of the solutions in B, where the coefficients are natural numbers. We associate a new individual variable, called basis variable, with each element of B. We denote the set of basis variables by $\mathcal{BV}$. The basis for *dio* can be represented as it is shown on Figure 5.12, where $\{\overline{Z_1}, \dots, \overline{Z_n}\} \subseteq \{V_1, \dots, V_k, W_1, \dots, W_l\}$ are those variables that correspond to sequence variables from $\mathfrak{U}_O^2$ and $\{Z_1, \dots, Z_m\} \subseteq \{V_1, \dots, V_k, W_1, \dots, W_l\}$ is are those variables that correspond to individual variables from $\mathfrak{U}_O^2$.

Basis variable	$\overline{Z_1}$	$\ldots$	$\overline{Z_n}$	Z_1	$\ldots$	Z_m
u_1	c_{11}	$\ldots$	c_{1n}	d_{11}	$\ldots$	d_{1m}
$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$
u_r	c_{r1}	$\ldots$	c_{rn}	d_{r1}	$\ldots$	d_{rm}

Fig. 5.12. Basis for *dio*.

5. All the steps made so far basically coincide with the ones from Stickel algorithm. But now we constrain the solution of *dio* in a slightly different way. Namely, instead of asking each variable to have the value at least 1, we require each variable $\overline{Z_1}, \ldots, \overline{Z_n}$ to have the value at least 1 and each variable $Z_1, \ldots, Z_m$ to have the value 1. The first constraint comes from the consideration that sequence variables in the unification problem should not vanish at the transformation step. It is done only on the projection step. The second constraint is imposed by the requirement for each individual variable to be instantiated by a single term. To meet these requirements, we compose sums of the elements of the basis B in such a way that no elements occur more than once in each sum, the value of each variable $\overline{Z_1}, \ldots, \overline{Z_n}$ is at least 1 and the value of each variable $Z_1, \ldots, Z_m$ is 1.
6. For each sum s composed at Step 5 we generate a new unification problem $g(u_1, \ldots, u_m, f(\tilde{u_1}), \ldots, f(\tilde{u_n})) \stackrel{?}{=}_O g(t_1, \ldots, t_m, f(\overline{x_1}), \ldots, f(\overline{x_n}))$, such that
   - g is a new free flexible arity function symbol,
   - for all i, $1 \leq i \leq n$, $\overline{x_i}$ is the sequence variable corresponding to $\overline{Z_i}$,
   - for all i, $1 \leq i \leq m$, t_i is the term replaced at the Step 2 by the individual variable corresponding to Z_i,
   - for all i, $1 \leq i \leq m$, u_i is the basis variable associated to the solution element whose coefficient in the column Z_i is 1 in s,
   - for all i, $1 \leq i \leq n$, $\tilde{u_i}$ is the sequence of basis variables. Each basis variable occurs in $\tilde{u_i}$ as many times as specified in the column $\overline{Z_i}$ in s.

   Let $\mathfrak{P}_1, \ldots, \mathfrak{P}_n$ be the unification problems generated in this way.
7. $\mathsf{OrdUnif}(\mathfrak{U}_O^0) = \langle\langle\mathfrak{P}_1, \varepsilon_1\rangle, \ldots, \langle\mathfrak{P}_n, \varepsilon_n\rangle\rangle$.

Example 5.6.1. Let $\mathfrak{U}_O$ be a $f(\overline{x}, b, \overline{x}, y, a) \stackrel{?}{=}_O f(b, b, \overline{z}, b)$, where f is an orderless flexible arity symbol. We show step by step the application of the procedure $\mathsf{OrdUnif}$ on $\mathfrak{U}_O$.

1. We eliminate one occurrence of b from both sides of $\mathfrak{U}_O$ and get $\mathfrak{U}_O^1$ of the form $f(\overline{x}, \overline{x}, y, a) \stackrel{?}{=}_O f(b, \overline{z}, b)$.
2. Variable abstraction gives the new unification problem $\mathfrak{U}_O^1$ of the form $f(\overline{x}, \overline{x}, y, w) \stackrel{?}{=}_O f(v, \overline{z}, v)$.

3. Associate to $\mathfrak{U}_O^2$ a linear Diophantine equation *dio*: $2\overline{X}+Y+W = 2V+\overline{Z}$.
4. The basis B in the natural solution space of *dio* is shown on the Figure 5.13.

Basis variable	$\overline{X}$	Y	W	V	$\overline{Z}$	$2\overline{X}+Y+W$	$2V+\overline{Z}$
u_1	0	0	1	0	1	1	1
u_2	0	1	0	0	1	1	1
u_3	0	0	2	1	0	2	2
u_4	0	1	1	1	0	2	2
u_5	0	2	0	1	0	2	2
u_6	1	0	0	0	2	2	2
u_7	1	0	0	1	0	2	2

Fig. 5.13. Basis B in the the natural solution space of the equation $2\overline{X}+Y+W = 2V+\overline{Z}$. New variables $u_1, \ldots, u_7$ are associated with the elements of the basis.

5. We compose sums of the elements of the basis B in such a way that no elements occur more than once in each sum, the value of the variables $\overline{X}$ and $\overline{Z}$ is at least 1 and the value of the variables Y, W and V is 1. There are two such sums: $u_1 + u_2 + u_7$ and $u_1 + u_2 + u_6 + u_7$.
6. We generate two new unification problems:
   $g(u_1, u_7, u_2, f(u_7), f(u_1, u_2)) \overset{?}{=}_O g(a, b, y, f(\overline{x}), f(\overline{z}))$
   $g(u_1, u_7, u_2, f(u_6, u_7), f(u_1, u_2, u_6, u_6)) \overset{?}{=}_O g(a, b, y, f(\overline{x}), f(\overline{z}))$.
7. $\mathsf{OrdUnif}(\mathfrak{U}_O^0) =$
   $\langle\langle g(u_1, u_7, u_2, f(u_7), f(u_1, u_2)) \overset{?}{=}_O g(a, b, y, f(\overline{x}), f(\overline{z})), \varepsilon\rangle,$
   $\langle g(u_1, u_7, u_2, f(u_6, u_7), f(u_1, u_2, u_6, u_6)) \overset{?}{=}_O g(a, b, y, f(\overline{x}), f(\overline{z})), \varepsilon\rangle\rangle$.

5.6.4 Substitution Processing

The purpose of substitution processing is to replace uninstantiated basis variables in a substitution by sequence variables. Let

$$\sigma = \{x_1 \leftarrow s_1, \ldots, x_n \leftarrow s_n, \overline{x}_1 \leftarrow t_1^1, \ldots, t_{k_1}^1, \ldots, \overline{x}_m \leftarrow t_1^m, \ldots, t_{k_m}^m\}$$

be a substitution. Then we define $\mathsf{SubstProc}$ as follows:

– If for some i, $1 \le i \le n$, s_i is a basis variable, then $\mathsf{SubstProc}(\sigma) = \bot$.
– If for no i, $1 \le i \le n$, s_i is a basis variable, then $\mathsf{SubstProc}(\sigma) = \theta$, where θ is the substitution obtained from σ by replacing each basis variable in $t_1^1, \ldots, t_{k_1}^1, \ldots, t_1^m, \ldots, t_{k_m}^m$ with a new sequence variable.

Example 5.6.2. Let σ be a substitution

$$\sigma = \{u_1 \leftarrow a, u_7 \leftarrow b, u_2 \leftarrow y, \overline{x} \leftarrow u_6, b, \ \ \overline{z} \leftarrow a, y, u_6, u_6\}.$$

Then

$$\mathsf{SubstProc}(\sigma) = \{u_1 \leftarrow a, u_7 \leftarrow b, u_2 \leftarrow y, \overline{x} \leftarrow \overline{w}, b, \ \ \overline{z} \leftarrow a, y, \overline{w}, \overline{w}\},$$

where $\overline{w}$ is a new sequence variable which replaces u_6 in σ.

5.6.5 Tree Generation

We design the unification procedure for the orderless theory with sequence variables and flexible arity symbols as a tree generation process. The single steps in this process are projection, transformation and substitution processing.

Each node of the tree is labeled either with a unification problem, $\top$ or $\bot$. The edges of the tree are labeled by substitutions. The nodes labeled with $\top$ or $\bot$ are terminal nodes. The nodes labeled with unification problems are non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Given a nonterminal node, let $\mathfrak{U}$ be a unification problem attached to it. First, we decide whether $\mathfrak{U}$ is unifiable. If the answer is negative, we replace $\mathfrak{U}$ with the new label $\bot$. If $\mathfrak{U}$ is unifiable, we apply projection or transformation on $\mathfrak{U}$ and get $\langle\langle\mathfrak{S}_1, \sigma_1\rangle, \ldots, \langle\mathfrak{S}_n, \sigma_n\rangle\rangle$. Then the node $\mathfrak{U}$ has n children, labeled respectively with $\mathfrak{S}_1, \ldots, \mathfrak{S}_n$ and the edge to the $\mathfrak{S}_i$ node is labeled with σ_i $(1 \leq i \leq n)$. The set $\{\sigma_1, \ldots, \sigma_n\}$ is denoted by $sub(\mathfrak{U})$.

We design the general unification procedure for $\mathfrak{U}_O$ as a breadth first (level by level) tree generation process. We label the root of the tree with $\mathfrak{U}_O$ (zero level). First level nodes of the tree are obtained from the original problem by projection. Starting from the second level, we apply only transformation steps to a unification problem of each node, thus getting new successor nodes. The branch which ends with a node labeled by $\bot$ is a failed branch. For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node and attach the obtained substitution to the node together with the unification problem the node was labeled with. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N. For a node $\top$, we apply the substitution processing operation on the substitution σ associated with the node. If $\mathsf{SubstProc}(\sigma) = \bot$, we replace the label of the node with $\bot$. If $\mathsf{SubstProc}(\sigma) = \theta$, we replace the substitution σ with θ. Let $\Sigma(\mathfrak{U}_O)$ be the set of all substitutions associated with the $\top$ nodes. We call the tree a unification tree for $\mathfrak{U}_O$ and denote it $utree(\mathfrak{U}_O)$.

Theorem 5.6.1. *Let $s \stackrel{?}{=}_O t$ be a general O-unification problem. Then the set $\Sigma(\mathfrak{U}_O)$ is a complete set of O-unifiers for $\mathfrak{U}_O$.*

Proof. The idea is similar to Lemma 3.3.2 and Fages' proof of completeness of AC-unification [Fag87]. $\square$

The set $\Sigma(\mathfrak{U}_O)$ is not minimal. For the O-unification problem

$$f(f(a,x),z) \stackrel{?}{=}_O f(f(y,b),f(a,b)),$$

where f is orderless, it contains the substitutions $\sigma = \{y \leftarrow a,\ x \leftarrow b,\ z \leftarrow f(a,b)\}$ and $\lambda = \{x \leftarrow b,\ z \leftarrow f(y,b)\}$ and $\lambda \ll_O^{\{x,y,z\}} \sigma$.

However, it can be easily shown that the minimal set exists. If we apply during the tree generation the same minimization procedure as we used for the flat theory, we can achieve that the procedure generates the minimal complete set of O-unifiers. Let $\Sigma_{min}(\mathfrak{U}_O)$ be the set of substitutions attached to the successful leaves of the minimized tree $utree_{min}(\mathfrak{U}_O)$. The we have the following theorem:

Theorem 5.6.2. *Let $\mathfrak{U}_O$ be a general O-unification problem. Then*

$$\Sigma_{min}(\mathfrak{U}_O) = mcu_O(\mathfrak{U}_O).$$

Another interesting result about orderless unification is the following:

Theorem 5.6.3. *Let $\mathfrak{U}_O$ be a O-unification problem without non-orderless flexible arity symbols. Then $mcu_O(\mathfrak{U}_O)$ is finite.*

Proof. After finitely many steps in the tree generation process of $\mathfrak{U}_O$ we get a tree with the leaves labeled either by $\top$, $\bot$ or unification problems built over individual variables, constants, free fixed and free flexible arity symbols, in which sequence variables occur only at the last argument positions. Then by Theorem 3.3.8 the tree generation process terminates, which implies that $mcu_O(\mathfrak{U}_O)$ is finite. $\square$

Example 5.6.3. Let $\mathfrak{U}_O$ a unification problem $f(\overline{x},b,\overline{x},y,a) \stackrel{?}{=}_O f(b,b,\overline{z},b)$, where f is an orderless flexible arity symbol. In Example 5.6.1 it was shown that the problem can be reduced to two new unification problems:

1. $g(u_1,u_7,u_2,f(u_7),f(u_1,u_2)) \stackrel{?}{=}_O g(a,b,y,f(\overline{x}),f(\overline{z}))$
2. $g(u_1,u_7,u_2,f(u_6,u_7),f(u_1,u_2,u_6,u_6)) \stackrel{?}{=}_O g(a,b,y,f(\overline{x}),f(\overline{z}))$.

The first problem has two solutions:

$$\theta_1 = \{u_1 \leftarrow a, u_7 \leftarrow b, u_2 \leftarrow y, \overline{x} \leftarrow b,\ \overline{z} \leftarrow a, y\},$$
$$\theta_2 = \{u_1 \leftarrow a, u_7 \leftarrow b, u_2 \leftarrow y, \overline{x} \leftarrow b,\ \overline{z} \leftarrow y, a\}.$$

The second problem has 24 solutions:

$$
\begin{aligned}
\sigma_1 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow a, y, \overline{y}, \overline{y}\},\\
\sigma_2 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow a, \overline{y}, y, \overline{y}\},\\
\sigma_3 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow a, \overline{y}, \overline{y}, y\},\\
\sigma_4 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow \overline{y}, a, \overline{y}, y\},\\
\sigma_5 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow \overline{y}, \overline{y}, a, y\},\\
\sigma_6 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow y, a, \overline{y}, \overline{y}\},\\
\sigma_7 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow y, \overline{y}, a, \overline{y}\},\\
\sigma_8 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow y, \overline{y}, \overline{y}, a\},\\
\sigma_9 &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow \overline{y}, y, \overline{y}, a\},\\
\sigma_{10} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow \overline{y}, \overline{y}, y, a\},\\
\sigma_{11} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow \overline{y}, a, y, \overline{y}\},\\
\sigma_{12} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow \overline{y}, b,\ \overline{z} \leftarrow \overline{y}, y, a, \overline{y}\},\\
\sigma_{13} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow a, y, \overline{y}, \overline{y}\},\\
\sigma_{14} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow a, \overline{y}, y, \overline{y}\},\\
\sigma_{15} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow a, \overline{y}, \overline{y}, y\},\\
\sigma_{16} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow \overline{y}, a, \overline{y}, y\},\\
\sigma_{17} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow \overline{y}, \overline{y}, a, y\},\\
\sigma_{18} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow y, a, \overline{y}, \overline{y}\},\\
\sigma_{19} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow y, \overline{y}, a, \overline{y}\},\\
\sigma_{20} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow y, \overline{y}, \overline{y}, a\},\\
\sigma_{21} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow \overline{y}, y, \overline{y}, a\},\\
\sigma_{22} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow \overline{y}, \overline{y}, y, a\},\\
\sigma_{23} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow \overline{y}, a, y, \overline{y}\},\\
\sigma_{24} &= \{u_1 \leftarrow a,\ u_7 \leftarrow b,\ u_2 \leftarrow y,\ \overline{x} \leftarrow b, \overline{y},\ \overline{z} \leftarrow \overline{y}, y, a, \overline{y}\}.
\end{aligned}
$$

The union of these two sets form the minimal complete set of unifiers of $\mathfrak{U}_O$.

5.7 Some Classes of Equational Theories

Bürckert et al [BHSS90] investigated classes of equational theories which are important in unification theory: permutative, finite, Noetherian, simple, almost collapse free, collapse free, regular and Ω-free theories. In this section we adapt the definitions of these notions for the theories with sequence variables and flexible arity symbols and classify the theories with respect to these notions.

Definition 5.7.1 (Permutation Equation). *An equation $s \doteq t$ is called permutation equation iff for every symbol $l \in \mathcal{CONST} \cup \mathcal{FFIX} \cup \mathcal{FFLEX} \cup \mathcal{IV} \cup \mathcal{SV}$, the number of occurrences of l in s is the same as the number of occurrences of l in t.*

Definition 5.7.2 (Regular Equation). *An equation $s \doteq t$ is called regular equation iff $vars(s) = vars(t)$.*

Definition 5.7.3 (Collapse Equation). *An equation $s \doteq t$ is called collapse equation iff $s \in \mathcal{IV}$ and $t \notin \mathcal{IV} \cup \mathcal{SV}$, or $t \in \mathcal{IV}$ and $s \notin \mathcal{IV} \cup \mathcal{SV}$.*

Definition 5.7.4 (Subterm Collapsing Equation). *An equation $s \doteq t$ is called subterm collapsing equation iff s is a proper subterm of t or t is a proper subterm of s.*

Definition 5.7.5 (Projection Equation). *A collapse equation of the form $f(x_1, \dots, x_i, \dots, x_n) \doteq x_i$ (for $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$ and for some i with $1 \leq i \leq n$) with pairwise different individual variables $x_1, \dots, x_n$ is called projection equation for f.*

Definition 5.7.6 (Permutative Theory). *An equational theory E is called permutative iff every equation in E is a permutation equation.*

Definition 5.7.7 (Regular Theory). *An equational theory E is called regular iff every equation in E is a regular equation.*

Definition 5.7.8 (Collapse-free Theory). *An equational theory E is called collapse-free iff E does not contain any collapse equation.*

Definition 5.7.9 (Simple Theory). *An equational theory E is called simple iff E does not contain any subterm-collapsing equation.*

Definition 5.7.10 (Almost Collapse-Free Theory). *An equational theory E is called almost collapse-free iff there exists a projection equation in E for the leading function symbol of every collapse equation in E:*

$$f(t_1, \dots, t_n) \doteq_E y \text{ implies } f(x_1, \dots, x_i, \dots, x_n) \doteq_E x_i.$$

Definition 5.7.11 (Finite Theory). *An equational theory E is called finite iff every equivalence class of the corresponding congruence relation $\doteq_E$ is finite.*

Definition 5.7.12 (Noetherian Theory). *An equational theory E is called Noetherian iff every strictly descending chain of substitutions $\sigma_1 >^V_E \sigma_2 >^V_E \dots$ is finite for all finite sets of variables V.*

Definition 5.7.13 (Ω-Free Theory). *An equational theory E is called Ω-Free iff every function symbol $f \in \mathcal{FFIX} \cup \mathcal{FFLEX}$ is decomposable:*

$$f(s_1, \dots, s_n) \doteq_E f(t_1, \dots, t_n) \text{ implies } s_i \doteq_E t_i$$

for every i, $1 \leq i \leq n$.

Definition 5.7.14 (Strongly Complete Theory). *An equational theory E is called strongly complete iff for every E-solvable unification problem $x \overset{?}{\doteq}_E t$ there is a complete set of unifiers, where $\{x\}$ is the domain of all these unifiers.*

Table 5.1 characterizes free, flat, restricted flat and orderless theories and summarizes the unification results of the thesis.

	Free	Flat	Restricted flat	Orderless
Decidability of unification	yes	yes	yes	yes
Unification type	∞	∞	∞	∞
Matching type	ω	∞	ω	ω
Permutative	yes	no	no	yes
Regular	yes	yes	yes	yes
Collapse free	yes	yes	yes	yes
Simple	yes	no	no	yes
Almost collapse free	yes	yes	yes	yes
Finite	yes	no	no	yes
Noetherian	yes	yes	yes	yes
Ω-free	no	no	no	no
Strongly complete	yes	yes	yes	yes

Table 5.1. Classification of equational theories with sequence variables and flexible arity symbols.

6. Conclusion

This thesis defines syntax and semantics of equational theories with sequence variables and flexible arity symbols and introduces solving and proving methods. The main results are the following:

1. Syntax and semantics:
   a) Description of a Birkhoff-style calculus for pure equational logic with sequence variables and flexible arity symbols;
   b) Proving soundness and completeness of the calculus.
2. Solving:
   a) Designing unification procedures for free, flat, restricted flat and orderless equational theories with sequence variables and flexible arity symbols;
   b) Proving decidability of the unification problem in these theories;
   c) Proving minimality and completeness of the unification procedures;
   d) Designing a minimal and complete unification procedure for an extension of the free theory with patterns.
3. Proving:
   a) Defining ground total reduction orderings for terms and pattern-terms with sequence variables and flexible arity symbols.
   b) Proving that the basic equational superposition with ordering and equational constraints is a refutationally complete proving method for the free theory with sequence variables and flexible arity symbols.
   c) Proving that for a special class of free equations with sequence variables and flexible arity symbols unfailing completion can be used as a refutationally complete proving method.
4. Implementation:
   a) Implementation of the unification procedure for the free theory as a *Mathematica* package;
   b) Integration of the package into the *Theorema* system.

We want to emphasize that these results are by no means the end of the story. There are open questions and new challenging problems which could be subjects of future work:

– Decidability of the pattern unification problem.
– Solving ordering and equational constraints efficiently.

- Designing flat and restricted flat unification procedures that ensure minimality of the computed solution set without further postprocessing.
- Designing complete (and, preferably, minimal) unification procedures for flat-orderless and restricted flat-orderless theories.
- Development of appropriate data structures for indexing terms with sequence variables and flexible arity symbols.
- Investigating full first-order theories with sequence variables and flexible arity symbols.
- Inductive theorem proving with sequence variables and flexible arity symbols.

References

[AC95] F. Ajili and E. Contejean. Complete solving of linear Diophantine equations and inequations without adding variables. In U. Montanari and F. Rossi, editors, *Proceedings of the First International Conference on Principles and Practice of Constraint Programming, CP'95*, volume 1949 of *Lecture Notes in Computer Science*, pages 1–17, Cassis, France, 19–22 September 1995. Springer Verlag.

[AD93] J. Avenhaus and J. Denzinger. Distributing equational theorem proving. In C. Kirchner, editor, *Proceedings of the 5th International Conference on Rewriting Techniques and Applications, RTA'93*, volume 690 of *Lecture Notes in Computer Science*, pages 62–76, Montreal, Canada, 16–18June 1993. Springer Verlag.

[AP90] H. Abdulrab and J.-P. Pécuchet. Solving word equations. *Journal of Symbolic Computation*, 8(5):499–522, 1990.

[APR01] G. E. Andrews, P. Paule, and A. Riese. Macmahon's partition analysis III: The Omega package. *European Journal of Combinatorics*, 22:887–904, 2001.

[Baa86] F. Baader. Unification in idempotent semigroups is of type zero. *Journal of Automated Reasoning*, 2(3):283–286, 1986.

[Bac91] L. Bachmair. *Canonical Equational Proofs.* Birkhäuser, Boston, 1991.

[BD94] L. Bachmair and N. Dershowitz. Equational inference, canonical proofs, and proof orderings. *Journal of the ACM*, 41(2):236–276, 1994.

[BDH86] L. Bachmair, N. Dershowitz, and J. Hsiang. Orderings for equational proofs. In *Proceedings of the 1st IEEE Symposium on Logic in Computer Science, LICS'86*, pages 346–357, Cambridge, Massachusetts, US, 16–18 June 1986. IEEE Computer Society Press.

[BDJ+00a] B. Buchberger, C. Dupré, T. Jebelean, B. Konev, F. Kriftner, T. Kutsia, K. Nakagawa, F. Piroi, D. Vasaru, and W. Windsteiger. The Theorema system: Proving, solving and computing for the working mathematician. Technical Report 00-38, Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria, 2000.

[BDJ+00b] B. Buchberger, C. Dupré, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. The Theorema project: A progress report. In M. Kerber and M. Kohlhase, editors, *Symbolic Computation and Automated Reasoning. Proceedings of Calculemus'2000 Conference*, pages 98–113, St. Andrews, UK, 6–7 August 2000.

[BDP89] L. Bachmair, N. Dershowitz, and D. Plaisted. Completion without failure. In H. Aït-Kaci and M. Nivat, editors, *Resolution of*

Equations in Algebraic Structures, volume 2, pages 1–30. Elsevier Science, 1989.

[BGLS95] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic paramodulation. *Information and Computation*, 121(2):172–192, 1995.

[BHSS90] H.-J. Bürckert, A. Herold, and M. Schmidt-Schauß. On equational theories, unification and (un)decidability. In C. Kirchner, editor, *Unification*, pages 69–116. Academic Press, 1990.

[Bir35] G. Birkhoff. On the structure of abstract algebras. In *Proceedings of the Cambridge Philosophical Society 31(4)*, pages 433–454, 1935.

[BJK+97] B. Buchberger, T. Jebelean, F. Kriftner, M. Marin, E. Tomuta, and D. Vasaru. A survey of the Theorema project. In W. Küchlin, editor, *Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC'97*, pages 384–391, Maui, Hawaii, US, 21–23July 1997. ACM Press.

[BLSS01] M. Benedikt, L. Libkin, T. Schwentick, and L. Segoufin. String operations in query languages. In *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 183–194, Santa Barbara, California, US, 21–23 May 2001.

[BN98] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, New York, 1998.

[BS91] F. Baader and K. U. Schulz. General A- and AX-unification via optimized combination procedures. In H. Abdulrab and J.-P. Pécuchet, editors, *Proceedings of the Second International Workshop on Word Equations and Related Topics, IWWERT-91*, volume 677 of *Lecture Notes in Computer Science*, pages 23–42, Rouen, France, 7–9 October 1991. Springer Verlag.

[BS93] F. Baader and J. Siekmann. Unification theory. In D. M. Gabbay, C. J. Hogger, and A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 2, Deduction Methodologies, pages 41–125. Oxford University Press, Oxford, UK, 1993.

[BS96] F. Baader and K. U. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. *Journal of Symbolic Computation*, 21(2):211–244, 1996.

[BS01] F. Baader and W. Snyder. Unification theory. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 8, pages 445–532. Elsevier Science, 2001.

[Buc96] B. Buchberger. Mathematica as a rewrite language. In T. Ida, A. Ohori, and M. Takeichi, editors, *Proceedings of the 2nd Fuji International Workshop on Functional and Logic Programming*, pages 1–13, Shonan Village Center, Japan, 1–4 November 1996. World Scientific.

[Buc97] B. Buchberger. Computing, solving, proving: A report of the Theorema project. In J. Dix, U. Furbach, and A. Nerode, editors, *Proceedings of the 4th International Conference on Logic Programing and Nonmonotonic Reasoning, LPNMR'97*, volume 1265 of *Lecture Notes in Artificial Intelligence*, pages 220–221, Dagstuhl, Germany, 28–31July 1997. Springer Verlag.

[Buc01] B. Buchberger. Personal communication, 2001.

[CD94] E. Contejean and H. Devie. An efficient incremental algorithm for solving systems of linear Diophantine equations. *Information and Computation*, 113(1):143–172, 1994.

[CNNR98] H. Comon, P. Narendran, R. Nieuwenhuis, and M. Rusinowitch. Decision problems in ordered rewriting. In *Proceedings of the 13st IEEE Symposium on Logic in Computer Science, LICS'98*, pages 276–286, Indianapolis, Indiana, US, 21–24 June 1998. IEEE Computer Society Press.

[Col90] A. Colmerauer. An introduction to Prolog III. *Communications of ACM*, 33(7):69–91, 1990.

[Der82] N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

[DJ90] N. Dershowitz and J.-P. Jouannaud. Rewriting systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 243–320. Elsevier Science, Amsterdam, 1990.

[DP01] N. Dershowitz and D. A. Plaisted. Rewriting. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 9, pages 535–610. Elsevier Science, 2001.

[Fag87] F. Fages. Associative-commutative unification. *Journal of Symbolic Computation*, 3:257–275, 1987.

[FFR97] A. Farquhar, R. Fikes, and J. Rice. The Ontolingua Server: A tool for collaborative ontology construction. *Int. Journal of Human-Computer Studies*, 46(6):707–727, 1997.

[FH86] F. Fages and G. Huet. Complete sets of unifiers and matchers in equational theories. *Theoretical Computer Science*, 43(2-3):189–200, 1986.

[Gen95] M. R. Genesereth. Epilog for Lisp 2.0 Manual. Technical report, Epistemics Inc., Palo Alto, California, US, 1995.

[GF92] M. R. Genesereth and R. E. Fikes. Knowledge Interchange Format, Version 3.0 Reference Manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, Stanford, California, US, June 1992.

[GNU94] G. Grahne, M. Nykänen, and E. Ukkonen. Reasoning about strings in databases. In *Proceedings of the 13th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Sys-*

tems, pages 303–312, Minneapolis, Minnesota, US, 24–26 May 1994.

[GW92] S. Ginsburg and X. S. Wang. Pattern matching by Rs-operations: Toward a unified approach to querying sequenced data. In *Proceedings of the 11th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 293–300, San Diego, California, US, 2–4 June 1992.

[GW00] G. Grahne and E. Waller. How to make SQL stand for string query language. In R. Connor and A. Mendelzon, editors, *Research Issues in Structured and Semistructured Database Programming. Proceedings of the 7th International Workshop on Database Programming Languages, DBPL'99*, volume 1949 of *Lecture Notes in Computer Science*, pages 61–79, Kinloch Rannoch, UK, 1–3 September 2000. Springer Verlag.

[Ham97] M. Hamana. Term rewriting with sequences. In *Proceedings of the First International Theorema Workshop, RISC-Linz Technical Report 97-20*, Hagenberg, Austria, 9–10 June 1997.

[HM01] P. Hayes and C. Menzel. Semantics of knowledge interchange format. http://reliant.teknowledge.com/IJCAI01/HayesMenzel-SKIF-IJCAI2001.pdf, 2001.

[Jaf90] J. Jaffar. Minimal and complete word unification. *Journal of ACM*, 37(1):47–85, 1990.

[JK91] J.-P. Jouannaud and C. Kirchner. Solving equations in abstract algebras: A rule-based survey of unification. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of A. Robinson*, pages 257–321. The MIT Press, Cambridge, Massachusetts, US, 1991.

[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.

[KKR90] C. Kirchner, H. Kirchner, and M. Rusinowitch. Deduction with symbolic constraints. *Revue Française d'Intelligence Artificielle*, 4(3):9–52, 1990.

[KL80] S. Kamin and J.-J. Lévy. Two generalizations of the recursive path ordering. Unpublished note, Department of Computer Science, University of Illinois, Urbana, Illinois, US, 1980.

[Klo92] J. W. Klop. Term rewriting systems. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, chapter 1, pages 1–117. Oxford University Press, 1992.

[KMN88] D. Kapur, D. R. Musser, and P. Narendran. Only prime superpositions need to be considered in the Knuth-Bendix completion procedure. *Journal of Symbolic Computation*, 6(2):19–36, 1988.

[KR99] S. Kepser and J. Richts. UniMoK: A system for combining equational unification algorithms. In P. Narendran and M. Rusinowitch, editors, *Proceedings of the 10th International Conference on Rewriting Techniques and Applications, RTA'99*, volume 1631 of *Lecture Notes in Computer Science*, pages 248–251. Springer Verlag, 2–4 July 1999.

[Kut02a] T. Kutsia. Pattern unification with sequence variables and flexible arity symbols. In M. Ojeda-Asiego, editor, *Proceedings of the Workshop on Unification in Non-Classical Logics*, volume 66, number 5 of *Electronic Notes on Theoretical Computer Science*, Málaga, Spain, 12–13 July 2002. Elsevier Science. To appear.

[Kut02b] T. Kutsia. Unification in a free theory with sequence variables and flexible arity symbols and its extensions. SFB Report 02-6, Johannes Kepler University, Linz, Austria, 2002.

[Kut02c] T. Kutsia. Unification with sequence variables and flexible arity symbols and its extension with pattern-terms. In J. Calmet, B. Benhamou, O. Caprotti, L. Henocque, and V. Sorge, editors, *Artificial Intelligence, Automated Reasoning and Symbolic Computation. Proceedings of Joint AICS'2002 – Calculemus'2002 conference*, volume 2385 of *Lecture Notes in Artificial Intelligence*, Marseille, France, 1–5 July 2002. Springer Verlag. To appear.

[Len72] A. Lentin. Equations in free monoids. In M. Nivat, editor, *Automata, Languages and Programming*, pages 67–85, Paris, France, 3–7 July 1972. North-Holland, Amsterdam.

[Mak77] G. S. Makanin. The problem of solvability of equations on a free semigroup. *Math. USSR Sbornik*, 32(2), 1977.

[Mar02] M. Marin. Personal communication, 2002.

[MB95] G. Mecca and A. J. Bonner. Sequences, Datalog and transducers. In *Proceedings of the 14th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 23–35, San Jose, California, US, 22–25 May 1995.

[MN90] U. Martin and T. Nipkow. Ordered rewriting and confluence. In M. E. Stickel, editor, *Proceedings of the 10th International Conference on Automated Deduction, CADE'90*, volume 449 of *Lecture Notes in Artificial Intelligence*, pages 366–380, Kaiserslautern, Germany, 24–27 July 1990. Springer Verlag.

[Mor85] F. Mora. Gröbner bases for non-commutative polynomial rings. In J. Calmet, editor, *Proceedings of the 3rd International Conference on Algebraic Algorithms and Error-Correcting Codes, AAECC'85*, volume 229 of *Lecture Notes in Computer Science*, pages 353–362, Grenoble, France, 15–19 July 1985. Springer Verlag.

[NR92a] R. Nieuwenhuis and A. Rubio. Basic superposition is complete. In B. Krieg-Brückner, editor, *Proceedings of the European Symposium of Programming, ESOP'92*, volume 582 of *Lecture Notes in Computer Science*, Rennes, France, 26–28February 1992. Springer Verlag.

[NR92b] R. Nieuwenhuis and A. Rubio. Theorem proving with ordering constrained clauses. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction, CADE'92*, volume 814 of *Lecture Notes in Artificial Intelligence*, Saratoga Springs, New York, US, 15–18 June 1992. Springer Verlag.

[NR95] R. Nieuwenhuis and A. Rubio. Theorem proving with ordering and equality constrained clauses. *Journal of Symbolic Computation*, 19:321–351, 1995.

[NR01] R. Nieuwenhuis and A. Rubio. Paramodulation-based theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 7, pages 371–443. Elsevier Science, 2001.

[Pet90] G. E. Peterson. Complete sets of reductions with constraints. In M. E. Stickel, editor, *Proceedings of the 10th International Conference on Automated Deduction, CADE'90*, volume 449 of *Lecture Notes in Artificial Intelligence*, pages 381–395, Kaiserslautern, Germany, 24–27 July 1990. Springer Verlag.

[Plo72] G. Plotkin. Building in equational theories. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 7, pages 73–90, Edinburgh, UK, 1972. Edinburgh University Press.

[Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[Rub95] A. Rubio. Theorem proving modulo associativity. In H. Kleine Büning, editor, *Proceedings of the Conference of European Association for Computer Science Logic, CSL'95*, volume 1092 of *Lecture Notes in Computer Science*, pages 452–467, Paderborn, Germany, 22–29 September 1995. Springer Verlag.

[Sch90] K. U. Schulz. Makanin's algorithm for word equations – two improvements and a generalization. In K. Schulz, editor, *Proceedings of Word Equations and Related Topics, IWWERT'90*, number 572 in Lecture Notes in Computer Science, pages 85–150, Tübingen, Germany, 1–3 October 1990. Springer Verlag.

[Sch93] K. U. Schulz. Word unification and transformation of generalized equations. *Journal of Automated Reasoning*, 11(2):149–184, 1993.

[Sie75] J. Siekmann. String unification. Research paper, Essex University, 1975.

[Sie78] J. Siekmann. Unification and matching problems. Memo, Essex University, 1978.

[SS86] M. Schmidt-Schauß. Unification under associativity and idempotence is of type nullary. *Journal of Automated Reasoning*, 2(3):277–281, 1986.

[Sti81] M. E. Stickel. A unification algorithm for associative-commutative functions. *Journal of the ACM*, 28(2):423–434, 1981.

[WB83] F. Winkler and B. Buchberger. A criterion for eliminating unnecessary reduction in the Knuth-Bendix algorithm. In *Proceedings of the Colloquium on Algebra, Combinatorics and Logic in Computer Science*, volume II, pages 849–869, Gyor, Hungary, 1983.

[WB01] M. Widera and C. Beierle. A term rewriting scheme for function symbols with variable arity. Technical Report 280, Praktische Informatik VIII, FernUniversität Hagen, Germany, 2001.

[Win01] W. Windsteiger. A set theory prover in Theorema: Implementation and practival applications. Technical Report 01-14, PhD Thesis. Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria, 2001.

[Wol99] S. Wolfram. *The Mathematica Book*. Cambridge University Press and Wolfram Research, Inc., fourth edition, 1999.

Index

CURRICULUM VITAE

Teimuraz Kutsia

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University, A–4040 Linz, Austria
Tel: +43 (0)732 2468-9962 Fax: +43 (0)732 2468 9930
kutsia@risc.uni-linz.ac.at
http:/www.risc.uni-linz.ac.at/people/tkutsia/

Date and Place of Birth: November 30, 1968, Abasha, Georgia, USSR

Nationality: Georgian

Education:

1974 – 1985	School education (finished with the gold award for the best student), Abasha, Georgia
1985 – 1987, 1989–92	Studies at the Dept. of Mechanics and Mathematics, Tbilisi State University, Tbilisi, Georgia
1987 – 1989	Military servise
1992	Diploma degree in Mathematics, Diploma with Honours
1993 – 1997	Research Assistant at Vekua Institute of Applied Mathematics, Tbilisi State University, Tbilisi, Georgia
1998	Degree of Candidate of Science in Physics and Mathematics
1998 –	Ph.D. study at Research Institute for Symbolic Computation, Johannes Kepler University, Linz
2000 –	Member of scientific staff, Software Competence Center Hagenberg

Grants:

Upper Austrian Government Scholarship (November 1998 – January 2001)

SFB grant F1302 of the Austrian "Fonds zur Förderung der wissenschaftlichen Forschung" (February 2001 –)

Publications:

T. Kutsia. Pattern unification with sequence variables and flexible arity symbols. In: M. Ojeda-Asiego, editor, *Proceedings of the Workshop on Unification in Non-Classical Logics*, July 12-13, 2002, Málaga, Spain.

Electronic Notes on Theoretical Computer Science, volume 66, issue 5, Elsevier Science, 2002 (to appear).

T. Kutsia. Unification with Sequence Variables and Flexible Arity Symbols and its Extension with Pattern-Terms. In J. Calmet, B. Benhamou, O. Caprotti, L. Henocque, and V. Sorge, editors, *Artificial Intelligence, Automated Reasoning and Symbolic Computation. Proceedings of Joint AICS'2002 – Calculemus'2002 conference*, July 1-5, 2002, Marseille, France. Volume 2385 of Lecture Notes in Artificial Intelligence, Springer Verlag, 2002 (to appear).

T. Kutsia. Unification in a Free Theory with Sequence Variables and Flexible Arity Symbols and its Extensions SFB Report 02-6, March 2002, Johannes Kepler University, Linz.

T. Kutsia. Unification in the Empty and Flat Theories with Sequence Variables and Flexible Arity Symbols. In: F. Baader, V. Diekert, C. Tinelli, R. Treinen (eds.), *Proceedings of 15th International Workshop on Unification*, June 18-19, 2001, Siena, Italy.

T. Kutsia, K. Nakagawa. An Interface between Theorema and External Automated Deduction Systems. In: S. Linton and R. Sebastiani (eds.), *Proceedings of 9th Symposium on the Integration of Symbolic Computation and Mechanized Reasoning*, June 21-23, 2001, Siena, Italy.

T.Kutsia. Semantics and Proof Theory of Disjunctive Logic Programs with Implicative Goals. *Proceedings of the 2nd International Symposium on Language, Logic and Computation*, Tbilisi, September 15-20, 1997, Tbilisi University Press, 160-178, 1997.

T.Kutsia. On Semantics and Proof Theory of General Logic Programs. *Journal of Applied Mathematics and Informatics*, 1. 96-110, 1996, Tbilisi University Press.