Johannes Kepler University Linz

Research Institute for Symbolic Computation
RISC - Linz

Volume of Parametrised Polyhedra

Manfred Minimair

Diplomarbeit
zur Erlangung des akademischen Grades
Diplom-Ingenieur
in der Studienrichtung
Technische Mathematik

submitted to
Univ.-Doz. Dr. Hong

January 1997

Dedicated to
my parents
and to
my brothers
Gerald and *Hannes*

Preface

Deutsch

Thema der Diplomarbeit. Darstellung der Volumensberechnung parametrisierter, konvexer Polyhedra.

Themenauswahl. Die Auswahl dieses Themas erfolgte wegen der im *praktischen* und *wissenschaftlichen Bereich* gegebenen *Aktualität* und Präferenz für die anstehende Problematik bei Volumensberechnung parametrisierter Polyhedra.

Im praktischen Bereich ist vor allem die Bestimmung des Volumens geometrischer Körper, wie der *Polyhedra*, von besonderer differenzierter Relevanz. Viele technische Bereiche, angefangen von der *Herstellung von Massenartikeln* des täglichen Bedarfes, bis zur *Produzierung von Gütern mit hohem technischen Standard*, erfordern im Fertigungsbereich die Bestimmung verschiedenster Volumensgrößen. Beispielsweise seien zum Letzteren dazu die Bereiche der Auto-, Schiffbau- oder Flugzeugindustrie genannt. Aber auch der Architektur- und Baubereich ist von dieser Thematik betroffen.

Die Frage der Volumensbestimmung ist jedoch nicht nur für Produktionsgüter, sondern vor allem auch für die Entwicklung von technischen *Produktionsmittel*, wie Maschinen oder Werkzeuge, wichtig. Von Bedeutung ist dabei allein aus *Kostengründen Form, Größe und Gewicht* eines Produktes oder eines technischen Produktionsmittels. Die Industrie ist daher aus *Wettbewerbsgründen* interessiert, eine Optimierung dieser Kriterien zu erreichen. Ein *optimales und probates Hilfsmittel* dafür ist die Volumensbestimmung nicht nur eines einzelnen Polyhedrons, sondern vielmehr die Volumina aller Polyhedra aus einer gegebenen parametrisierten Familie.

Im wissenschaftlichen Bereich tritt die Frage nach dem Volumen von parametrisierten Polyhedra auch in *aktuellen Forschungsgebieten* der

Informatik auf. Dazu gehören beispielsweise die automatische *Performancevorhersage* von sequentiellen und parallelen Programmen oder die automatische *Parallelisierung* sequentieller Programme. Im speziellen stößt man auf die Frage nach dem Volumen parametrisierter Polyhedra bei der Analyse und Parallelisierung von verschachtelten Zählschleifen.

Konkrete Aufgabenstellung. Als Aufgabenstellung war ursprünglich folgender Umfang vorgegeben:

- *Beschreibung* von wissenschaftlich relevanten algorithmischen Verfahren, die der Volumensberechnung von parametrisierten konvexen Polyhedra dienen.
- *Untersuchung* der theoretischen Grundlagen im Hinblick auf Vollständigkeit und Korrektheit.
- *Implementierung* dieser algorithmischen Verfahren.

Im Zuge der Bearbeitung zeigte sich, daß dieser Aufgabenstellung nur dann entsprochen werden kann, wenn vorher verschiedene *Komplettierungen* der theoretischen Grundlagen vorgenommen werden. Aus diesem Grunde wurden, wie später ausführlicher beschrieben wird, die in den untersuchten Verfahren festgestellten lückenhaften Verfahrensschritte ergänzt und für die Beschreibungsdarstellung *homogene mathematische Rahmenbedingungen* definiert.

Darüber hinaus eröffnete sich jedoch auch noch die interessante Möglichkeit, in wissenschaftlich noch nicht abgeklärten Bereichen die theoretischen Grundlagen unter dem *Aspekt der Praxisbezogenheit fortzuentwickeln.*

Strukturierung der Diplomarbeit. Die Diplomarbeit ist in fünf große Kapitel gegliedert. Diese werden wie folgt beschrieben.

Problem Statement. Hier wird das Problem der Volumensberechnung von parametrisierten Polyhedra ausführlich motiviert und beschrieben.

Basic Theory. Dieses Kapitel enthält Definitionen, Sätze und Techniken, die für die Beschreibung, Komplettierung und Verallgemeinerung der Algorithmen maßgeblich sind.

Abstract Treatment of the Problem. In diesem Kapitel wird die Verallgemeinerung der untersuchten Verfahren innerhalb der Maßtheorie präsentiert.

State of the Art. Hier werden die aktuellen Verfahren als Spezialisierungen ihrer maßtheoretischen Verallgemeinerungen dargestellt. Sie werden bereits komplettiert beschrieben. Weiters wird ihre Korrektheit bewiesen.

Improvements. Dieses Kapitel enthält Vorschläge zur Verbesserung des Laufzeitverhaltens der Algorithmen und die Dokumentation deren Timingergebnisse.

Komplettierung der untersuchten Algorithmen. Bei der Dokumentierung der algorithmischen Verfahren hat sich hinsichtlich der Verfahrensschritte gezeigt, daß diese sehr unterschiedlich in den Publikationen dargelegt werden. Die Verfahren selbst sind zwar teilweise ident oder ähnlich. Offen bleibt jedoch ein *homogener Darstellungs- bzw. Beschreibungsrahmen.*

Die Untersuchung der Verfahren hat ergeben, daß die Verfahrensschritte plausibel erläutert wurden, jedoch diese Beschreibungen zu keinen *korrekten Beweis* führen. Auch ist ein solcher nicht unmittelbar erbringbar.

Abgesehen davon, waren im Quellenmaterial keinerlei *Timingergebnisse* dieser Verfahren dokumentiert worden.

Aufgrund dieses Status der untersuchten Algorithmen wurde für die Verfahrensbeschreibung auf Basis der Maßtheorie, insbesondere des klassischen Satzes von Fubini, ein homogener mathematischer Rahmen definiert und danach die Fälle inkorrekter Volumensberechnungen, sowohl theoretisch als auch unter Zuhilfenahme der Implementierung der unvollständigen Algorithmen, analysiert. Basierend auf diesen Erkenntnissen wurden die untersuchten Algorithmen und die Implementierung komplettiert.

Fortentwicklung der Algorithmen und Ergebnis der Diplomarbeit. Aufgrund der veränderten Voraussetzungen, unter denen die Arbeit stattfand, war es im speziellen möglich, nicht nur die *Korrektheit* der untersuchten Algorithmen zu *beweisen*, sondern diese auch in Teilbereichen *weiterzuentwickeln.*

Im konkreten ergaben sich folgende Ergebnisse:

1. *Richtigstellung* der inkorrekten Bearbeitung von *Polyhedra mit offenen Seiten* in Zwischenergebnisse der Volumensberechungen im Zuge der Komplettierung der lückenhaften Algorithmen.

2. Entwicklung eines Verfahrens zur *Projektion von Gitterpunkten*, die in flachen Polyhedra enthalten sind.
3. Ermöglichung der *Volumensberechnung von parametrisierten Polyhedra mit offenen Seiten.*
4. *Verallgemeinerung* der Algorithmen innerhalb der *Maßtheorie*, insbesondere Zurückführung auf den klassischen Satz von Fubini.
5. Volumensberechnung bezüglich *beliebiger Maße*, sowie insbesondere Volumensberechnung nach dem üblichen *Lebesgue-Maß* des reellen Vektorraumes und nach einem gewissem *punktezählenden Maß*.
6. Dokumentierung der *Timingergebnisse*.
7. *Kontrolle der Explosion* der Größe der Zwischenergebnisse und des Ergebnisses.

   Anmerkung. Die Timingergebnisse brachten eine Bestätigung der theoretisch hohen Zeit- und Speicherkomplexität der hier behandelten Algorithmen. Daher wurden im Rahmen dieser Arbeit auch Versuche angestellt, die Explosion der Größe der Zwischenergebnisse und des Ergebnisses zu kontrollieren. Die dabei erzielten Verbesserungen basieren auf einem inkrementellen Konsistenztest parametrisierter Polyhedra aus dem Gebiet der linearen Optimierung. Dieser erweist sich teils als sehr effektiv, wie Testergebnisse der Implementierung zeigen.

Grenzen und Ausblick. Die Testergebnisse der Implementierung ergaben eine Explosion der Größe der Zwischenergebnisse und des Ergebnisses der Verfahren in zweifacher Hinsicht. Nämlich einerseits findet ein explosionsartiges Anwachsen der Größe der berechneten stückweisen Polynome statt. Andererseits explodiert aber auch die *Größe der Zähler und Nenner* der rationalen Zahlen, die die stückweisen Polynome beschreiben. Die Kontrolle der ersteren Explosion wird in den folgenden Seiten behandelt, nicht aber die der zweiteren. Dies schränkt die allgemeine Verwendbarkeit der Algorithmen noch manchmal ein.

Deshalb ist bereits geplant, mittels *Techniken eingeführt von Bareiss* auch die zweite Art von Explosion zu kontrollieren. Weiters soll noch die *Speicherverwaltung* der Verfahren verbessert werden.

English

Subject of the diploma thesis. Description of how the volume of parametrised convex polyhedra can be computed.

Motivation of subject. We chose this subject because the computation of the volume of parametrised polyhedra is of great *importance in practice and in science at present.*

In practice it is relevant to derive the volumes of geometric objects, like *polyhedra*. In many technical areas, ranging from the *production of mass articles* needed in daily life to the *production of goods on a very high technical level*, one needs to know various volumes. Consider, for instance, the areas of car, shipping and aviation industries. Moreover, architecture and building are concerned by this, too.

But apart from the goods of production the computation of volumes also concerns the *means of production*, like machines and tools. Because of *costs considerations*, the *shapes, volumes and the weights* of products or technical means of production are very important. *Competition* increases the industry interest in optimising these criteria. The computation of the volume of not only one polyhedron but of all polyhedra out of a parametrised family is an *optimal and needed* tool for doing this.

Additionally one encounters the question about the volume of parametrised polyhedra in *recent areas of research* in *computer science.* Take automatic performance prediction of sequential and parallel programs or automatic parallelisation of sequential programs, for example. In particular one encounters this question when analysing and parallelising nested counting loops.

Concrete terms of assignment. At the beginning of the project the following tasks were specified:

- *Describing* several algorithmic methods that are scientifically relevant and that can be used for the computation of the volumes of parametrised convex polyhedra.
- *Checking* whether the theoretical sources are correct and complete.
- *Implementing* the algorithms.

In checking the theoretical sources we found out that the above tasks could only be carried out after having *completed* the methods in diverse ways. For this reason, the steps in the algorithms that were full of gaps had to be supplemented, as is described later in detail. Moreover we defined a *homogenous mathematical framework* for the exhibition of the descriptions.

Apart from this we found some interesting opportunities to further develop the theoretical sources. This was done with *practical applications* in mind and in directions that hadn't been considered scientifically previously.

Structure of the thesis. The diploma thesis contains five main chapters. We describe these as follows.

Problem Statement. Here the problem of computing the volumes of parametrised polyhedra is motivated in detail and described.

Basic Theory. This chapter contains definitions, theorems and techniques that are needed for the descriptions, completions and generalisations of the algorithms.

Abstract Treatment of the Problem. In this chapter the measure theoretic generalisations of the considered methods are presented.

State of the Art. Here the recent methods are described as being specialisations of the measure theoretic generalisations of the previous chapter. They are presented already completed. Moreover, we prove their correctness.

Improvements. This chapter contains proposals for improving the running of the algorithms. Descriptions of the timing results are also included.

Completion of the considered algorithms. When documenting the algorithmic methods we found out that they are described very differently in the original publications. A *homogenous framework for describing* was lacking.

When evaluating the methods we found out that the steps of the algorithms are explained in a plausible way. Nevertheless, these descriptions don't lead to a *correctness proof*. Furthermore, this proof doesn't follow from them immediately.

Apart from this, one can't find any documentation of timing results in the scientific sources.

Because of the state of the considered algorithms we defined a homogenous mathematical framework for describing the methods. This framework is based on measure theory and in particular on the classical Fubini's theorem. Afterwards, we analysed the cases of incorrect volume computations theoretically as well as using the implementations of the incomplete algorithms. Based on this we completed the algorithms and the implementations.

Additional developments of the algorithms and results of the work. The changed working conditions explain why we were able to *further develop* the algorithms along with proving their *correctness*.

The following results were obtained:

1. *Correction* of the incorrect processing of *polyhedra with open faces* in intermediate results when we completed the algorithms.
2. Development of a technique for *projecting the grid points* contained in flat polyhedra.
3. Possibility of computing the *volumes of parametrised polyhedra with open faces*.
4. *Generalisations* of the algorithms within *measure theory*; in particular reduction to the classical Fubini's theorem.
5. Volume computation with respect to *arbitrary measures*, as well as to the well known *Lebesgue measure* of the real vector space and with respect to a certain *counting measure*.
6. Documentation of the results of the *timing*.
7. *Control of the explosion* of the sizes of the intermediate results and of the result.

   Remark. The timings confirmed the theoretical high time and space complexity of the considered algorithms. Thus we tried to control the explosion of the sizes of the intermediate results and of the result. The achieved improvements are based on an incremental consistency test for parametrised polyhedra out of the area of linear optimisation. This test is very effective in part which is proven by the timings.

Restrictions and future directions. The testing results revealed two different kinds of explosions of the sizes of intermediate results and of the result. On the one hand explosive growths of the sizes of the computed piecewise polynomials happen. On the other hand *the sizes of the numerators and denominators* of the rational numbers representing the piecewise polynomials explode dramatically. We treated the first kind of explosion but didn't tackle the second one. Because of this, the general applicability of the algorithms is sometimes restricted.

Therefore we already plan to control the second kind of explosion using *techniques introduced by Bareiss*. Moreover, we want to improve the *memory management* of the algorithms.

Contents

Notation and Terminology

Throughout this diploma thesis the standard notation and terminology of first order predicate logics and set theory is used. Apart from this we fix the following notations.

1 Sets and Algebraic Structures

Let

$$\begin{aligned}
\mathbb{N} &:= \text{the positive integers, i.e. natural numbers,}\\
\mathbb{N}_0 &:= \mathbb{N} \cup \{0\},\\
(\mathbb{Z}, +, \cdot) &:= \text{the ring of integers; abbreviated by } \mathbb{Z},\\
(\mathbb{Q}, +, \cdot) &:= \text{the field of rational numbers; abbreviated by } \mathbb{Q},\\
(\mathbb{R}, +, \cdot) &:= \text{the field of real numbers; abbreviated by } \mathbb{R}.
\end{aligned}$$

Obviously the above sets are related in the following way:

$$\mathbb{N} \subseteq \mathbb{N}_0 \subseteq \mathbb{Z} \subseteq \mathbb{Q} \subseteq \mathbb{R}.$$

Moreover $\overline{\mathbb{R}} := \mathbb{R} \cup \{-\infty, +\infty\}$ denotes the two point compactifyication of $\mathbb{R}$ (cf. [Ash72]). Mostly we will abbreviate $+\infty$ by ∞.

Let $r \in \mathbb{R}$ and $\leq$ be the usual linear order defined on $\mathbb{R}$. Then we extend $\leq$ to $\overline{\mathbb{R}}$ in the following way:

$$-\infty < r < \infty.$$

For $r \in \mathbb{R}$ and $d \in \overline{\mathbb{R}}$ the following arithmetic operations are valid in $\overline{\mathbb{R}}$:

$$\begin{aligned}
\frac{r}{\infty} = \frac{r}{-\infty} &= 0,\\
d + \infty = \infty + d &= \infty, \quad \text{for } d \neq -\infty,\\
d - \infty = -\infty + d &= -\infty, \quad \text{for } d \neq \infty,\\
d \cdot \infty = \infty \cdot d &= \begin{cases} \infty & \text{if } d > 0,\\ 0 & \text{if } d = 0,\\ -\infty & \text{if } d < 0. \end{cases}
\end{aligned}$$

$\infty - \infty$, $-\infty + \infty$, $\frac{\infty}{\infty}$ and $\frac{-\infty}{-\infty}$ are not defined.

Apart from this let

$$\overline{\mathbb{Q}}_0^+ := \{q \in \mathbb{Q} \,|\, 0 \leq q\} \cup \{+\infty\},$$
$$\overline{\mathbb{R}}_0^+ := \{d \in \overline{\mathbb{R}} \,|\, 0 \leq d\}.$$

For $d, e \in \overline{\mathbb{R}}$ open intervals are defined by

$$]d, e[:= \left\{f \in \overline{\mathbb{R}} \,|\, d < f < e\right\}.$$

Closed intervals $[d, e]$ and half-open ones, as $]d, e]$ and $[d, e[$, are defined similarly.

Furthermore let $\mathbb{L} := \{<, \leq\}$[1] be the set containing the relations $<$ and $\leq$. By the following table we impose a monoid structure on this set. The relation

$\cdot$	$<$	$\leq$
$<$	$<$	$<$
$\leq$	$<$	$\leq$

Table i: Operation on $\mathbb{L}$

$\leq$ plays the role of the unit in the monoid $(\mathbb{L}, \cdot)$. Note that the monoid $(\mathbb{L}, \cdot)$ is isomorphic to the multiplicative Abelian monoid of the integers modulo 2 via the isomorphism $< \mapsto 0, \leq \mapsto 1$.

2 Finite Sequences of Natural Numbers

For any $k \in \mathbb{N}_0$ and $Y \subseteq \mathbb{N}$ we call a function $I : \{1, \ldots, k\} \to Y$ a finite sequence of natural numbers.

Let $i \in \{1, \ldots, k\}$. Then the function application of I to i will be denoted either by $I(i)$ or by I_i.

Mostly we will specify a finite sequence I by enumerating its elements, i.e. its image. That means we will write I as $(I_1, I_2, \ldots, I_k)$.

Moreover let $X := \{1, \ldots, k\}, Y$ be finite and let $I : X \to Y$ be strictly monotonous increasing. Then we define its complement

$$\overline{I} : \{1, \ldots, |Y| - |X|\} \to Y \backslash I(X)$$

recursively:

$$\overline{I}(1) := \min(Y \backslash I(X)),$$
$$\overline{I}(i+1) := \min\left\{y \in Y \backslash I(X) \,|\, \overline{I}_i < y\right\}.$$

[1] $\mathbb{L}$ stands for "less".

Example i
Let $I := (3,4) : \{1,2\} \to \{1,2,3,4,5\}$. Then

$$\overline{I} = (1,2,5) : \{1,2,3\} \to \{1,2,3,4,5\}\,.$$

$\square$

3 Typed Variables

The variables d, e will be in $\overline{\mathbb{R}}$. We always let i, j, k, l, m, n range over $\mathbb{N}_0$. The letters p, q will denote rational numbers. The variables r, s, t will always be real numbers. Whereas the variables $u, v, v_1, v_2, \ldots,\ w, w_1, w_2, \ldots$ will range over the integers.

Moreover I, J, K will be strictly monotonously increasing finite sequences of natural numbers. Further $X, X_1, X_2, \ldots,\ Y, Y_1, Y_2, \ldots,\ Z, Z_1, Z_2, \ldots$ will always be sets.

By $\lessdot, \lhd, \blacktriangleleft$ we denote variables ranging over $\mathbb{L}$.

All typed variables[2] introduced in this thesis are contained in tables ii and iii.

Alphabet	Variable	Domain	Page
non-letter	$\lessdot$, $\lhd$, $\blacktriangleleft$	$\mathbb{L}, \mathbb{L}^n, \mathbb{L}^{(\mathbb{N})}$	xii, xv, xv
Greek	μ	measures on $\mathcal{A}$ and $\mathcal{B}\left(\mathbb{R}^n\right)$, σ-finite,resp.	12 and 59, resp.
	μ_i	measures on $\mathcal{A}_i$	12
	Ω, Ω_i	sets	11
Calligraphic	$\mathcal{A}$	σ-algebra over Ω	11
	$\mathcal{A}_i$	σ-algebra over Ω_i	

Table ii: List of typed non Latin variables

[2]In a few cases we will use these symbols denoting variables as constant symbols for auxiliary constants within the same domain as the corresponding variables, too. But these constants will only be used within a limited scope, for instance in proofs when we specialise universal quantified expressions or in the statement of theorems in order to denote the values of auxiliary terms. Therefore no misunderstandings should occur because of this. We also believe that the introduction of new symbols for these rare cases would turn the reading of the text more difficult.

4 Operators and Functions

Let

$$
\begin{aligned}
\neg &:= \text{the logical negation,}\\
\vee &:= \text{the logical disjunction,}\\
\wedge &:= \text{the logical conjunction,}\\
\lfloor r \rfloor &:= \max\{v \in \mathbb{Z} \mid v \leq r\},\\
\lceil r \rceil &:= \min\{v \in \mathbb{Z} \mid v \geq r\},\\
\leq &:= \text{the usual linear order on } \overline{\mathbb{R}} \text{ as defined above,}\\
\mathbf{1}_X &:= \text{the indicator function (cf. [Ash72]), for any set } X.\\
\min &:= \text{the minimum of a tuple or a set,}\\
\max &:= \text{the maximum of a tuple or a set,}\\
f^{-1} &:= \text{the inverse of the function } f \text{ if } f \text{ is injective,}\\
X^C &:= \text{the set complement of } X,\\
|X| &:= \text{the number of elements of the set } X, \text{ i.e,}\\
&\quad \begin{cases} n & \text{iff there exists a bijection from } X \text{ onto } \{1,\ldots,n\}\\ & \text{for some (unique) } n,\\ \infty & \text{else.}\end{cases}
\end{aligned}
$$

We denote the function composition of two functions g, h by $g \circ h$ whenever it is defined.

5 Vector Spaces

Let F be a field and V, W be vector spaces over F. Any function $A : V \to W$ will be called linear iff $A(f_1 o_1 + f_2 o_2) = f_1 A(o_1) + f_2 A(o_2)$, for all $f_1, f_2 \in F$ and $o_1, o_2 \in V$. As V will be finite dimensional in all cases and the dimension of W at least countable we will represent such linear maps by matrices in examples. Definitions and theorems will be formulated without making use of any specific representation of linear functions.[3]

We call a function $B : V \to W$ affine linear iff the function $o \mapsto B(o) - B(0)$ is linear. Often we will abbreviate the function application of an affine linear function B to o by Bo.

The direct product of n copies of F (as a vector space over F) will be denoted by F^n. Elements in these kinds of vector spaces will be denoted by lower case letters with an arrow $\vec{\ }$ on top. We will represent the elements

[3] We believe that this leads to a formalism that is more succinct.

$\vec{x} \in F^n$ by column vectors $\begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix}$, where $x_j \in F$, for $j = 1, \dots, n$. The x_i are called the components of $\vec{x}$.

Matrices of finite dimension will be denoted by upper case letters with an arrow $\vec{\ }$ on top. We will represent these matrices $\vec{M}$ by tables of elements of F delimited by square brackets, as for example

$$\begin{bmatrix} M_{11} & \dots & M_{1n} \\ \dots & \dots & \dots \\ M_{m1} & \dots & M_{mn} \end{bmatrix}.$$

We call the M_{ij} the elements of $\vec{M}$. If all M_{ij} are zero we denote this matrix by 0. As usual $\vec{M}$ will be identified with the corresponding linear mapping from F^n to F^m. Moreover affine linear mappings from F^n to F^m will be denoted[4] by $\vec{M} - \vec{z}$.

In the case of $n = 0$ the only element of the null space F^0 will be the empty column vector (). We will identify () with $0 \in F$. Hence F^0 is just $\{0\}$.

The special vectors $\begin{pmatrix} 0 \\ \dots \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ \dots \\ 1 \end{pmatrix} \in F^n$ will be abbreviated by 0 and 1, resp.

The outer direct sum of $\aleph_0$, i.e. the cardinality of $\mathbb{N}$, copies of F will be denoted by $F^{(\mathbb{N})}$. Elements in these kinds of vector spaces will be denoted by lower case letters with the following accent on top: $\breve{\ }$. The elements $\breve{x} \in F^{(\mathbb{N})}$ will be represented by infinite column vectors $\begin{pmatrix} x_1 \\ x_2 \\ \dots \end{pmatrix}$, where $x_i \in F$, for $i \in \mathbb{N}$, with the property:

$$x_i = 0, \text{ for all } i \geq m, \text{ for some } m.$$

The x_i are called the components of $\breve{x}$. We will ensure that there won't be any confusion possible between components of $\vec{x}$ and $\breve{x}$.

The infinite column vector $\begin{pmatrix} 0 \\ 0 \\ \dots \end{pmatrix}$ will be abbreviated by 0.

Matrices with infinitely many rows but finitely many columns will be denoted by upper case letters with the same accent $\breve{\ }$ on top. We will represent these matrices $\breve{M}$ by tables of elements of F with infinitely many rows delimited by square brackets, as for example

$$\begin{bmatrix} M_{11} & \dots & M_{1n} \\ M_{21} & \dots & M_{2n} \\ \dots & \dots & \dots \end{bmatrix}.$$

We call the M_{ij} the elements of $\vec{M}$. We will ensure that there won't be any confusion possible between elements of $\vec{M}$ and $\breve{M}$. If all M_{ij} are zero we denote this matrix by 0.

[4] Compare page 31, too.

Mostly F will be $\mathbb{R}$ in this thesis. We will regard $\mathbb{R}^n, \mathbb{R}^{(\mathbb{N})}$ as vector spaces over the real numbers $\mathbb{R}$. For this case we fix some special conventions. Let

$$\begin{aligned} \mathbb{Z}^n &:= \{\vec{x} \in \mathbb{R}^n \,|\, x_j \in \mathbb{Z}, \text{ for all } j = 1, \ldots, n\}, \\ \mathbb{Q}^n &:= \{\vec{x} \in \mathbb{R}^n \,|\, x_j \in \mathbb{Q}, \text{ for all } j = 1, \ldots, n\}. \end{aligned}$$

Note that this means that $\mathbb{Z}^0 = \mathbb{Q}^0 = \mathbb{R}^0 = \{0\}$.

Apart from this let

$$\begin{aligned} \mathbb{Z}^{(\mathbb{N})} &:= \left\{\breve{x} \in \mathbb{R}^{(\mathbb{N})} \,|\, x_i \in \mathbb{Z}, \text{ for all } i \in \mathbb{N}\right\}, \\ \mathbb{Q}^{(\mathbb{N})} &:= \left\{\breve{x} \in \mathbb{R}^{(\mathbb{N})} \,|\, x_i \in \mathbb{Q}, \text{ for all } i \in \mathbb{N}\right\}. \end{aligned}$$

Let $X \subseteq \mathbb{R}^n$. Then we say X is convex iff $\vec{x}, \vec{y} \in X$ implies $s\vec{x} + (1-s)\vec{y} \in X$, for all $s \in [0, 1]$.

As the only spaces we deal with are vector spaces - apart from measure spaces in sections 2.1, Measure Theory, 2.2, Lebesgue Measure and 2.3, Integer Measure - we will leave out the word "vector" when we speak of vector spaces and if no misunderstandings are possible.

6 Comparison in Vector Spaces

Next we introduce some notation for component wise comparison of vectors. That means we will define vectors of comparison relations. We use a similar notation as for vector spaces.

By $\mathbb{L}^n$ we denote the outer direct product of n copies of the monoid $\mathbb{L}$. We will represent the elements of $\mathbb{L}^n$ by stacks, i.e. column vectors without enclosing parentheses, with n elements in $\mathbb{L}$, e.g.

$$\mathbb{L}^2 = \left\{ \begin{matrix} < \\ < \end{matrix}, \begin{matrix} < \\ \leq \end{matrix}, \begin{matrix} \leq \\ < \end{matrix}, \begin{matrix} \leq \\ \leq \end{matrix} \right\}.$$

The trivial monoid $\mathbb{L}^0$ is just the set $\{\leq\}$. Variables in $\mathbb{L}^n$ will be denoted by $\lessdot, \lhd, \blacktriangleleft$. Components of $\lessdot$ will be denoted by $\lessdot_1, \lessdot_2$, etc. The special stacks $\begin{smallmatrix} \leq \\ \cdots \\ \leq \end{smallmatrix}, \begin{smallmatrix} < \\ \cdots \\ < \end{smallmatrix} \in \mathbb{L}^n$ will be abbreviated by $\leq$ and $<$, resp.

Moreover let $\mathbb{L}^{(\mathbb{N})}$ denote the outer direct sum of $\aleph_0$ copies of $\mathbb{L}$. The elements of this monoids will be represented by infinite stacks $\lessdot$ with elements in $\mathbb{L}$ having the property:

$$\lessdot_i \ = \leq, \text{ for all } i \geq m, \text{ for some } m.$$

Therfore we have, for instance, $\begin{smallmatrix} \leq \\ < \\ \leq \\ \leq \\ \cdots \end{smallmatrix} \in \mathbb{L}^{(\mathbb{N})}$. Apart form $\lessdot$ we will denote variables ranging over $\mathbb{L}^{(\mathbb{N})}$ by $\lhd$ and $\blacktriangleleft$. We will always ensure that it is clear from the context over which domain among $\mathbb{L} \times \mathbb{L}, \mathbb{L}^n \times \mathbb{L}^n$ and $\mathbb{L}^{(\mathbb{N})} \times \mathbb{L}^{(\mathbb{N})}$ a variable $\lessdot, \lhd$ or $\blacktriangleleft$ ranges. The infinite stack $\underset{\vdots}{\overset{\leq}{\leq}}$ will be abbreviated by $\leq$.

Moreover let $\vec{y}, \vec{z} \in \mathbb{R}^n$ and $\breve{y}, \breve{z} \in \mathbb{R}^{(\mathbb{N})}$. Then

$$\vec{y} \lessdot \vec{z} \text{ iff } y_j \lessdot_j z_j, \text{ for } j = 1, \dots, n$$

and

$$\breve{y} \lessdot \breve{z} \text{ iff } y_i \lessdot_j z_i, \text{ for } i \in \mathbb{N}.$$

Intervals in $\mathbb{R}^n$ are defined in the same way as in $\overline{\mathbb{R}}$. Namely, For $\vec{y}, \vec{z} \in \mathbb{R}^n$ open intervals have the form

$$]\vec{y}, \vec{z}[:= \{\vec{x} \in \mathbb{R}^n \mid \vec{y} < \vec{x} < \vec{z}\}.$$

Closed intervals $[\vec{y}, \vec{z}]$ and half-open ones, as $]\vec{y}, \vec{z}]$ and $[\vec{y}, \vec{z}[$, are defined similarly.

7 Polynomials

We denote the ring of polynomials over $\mathbb{R}$ in the "variables" $\chi_1, \dots, \chi_n$. by $(\mathbb{R}[\chi_1, \dots, \chi_n], +, \cdot)$. We will abbreviate $\mathbb{R}[\chi_1, \dots, \chi_n]$ by $\mathbb{R}[\vec{\chi}]$ if n is obvious from the context. In such cases we will call elements of these rings simply polynomials.

If $\iota_{\vec{r}} : \mathbb{R}[\vec{\chi}] \to \mathbb{R}$ denotes the evaluation homomorphism that evaluates a polynomial by replacing χ_i by $r_i \in \mathbb{R}$, for $i = 1, \dots, n$, then we will abbreviate the function $\vec{x} \mapsto \iota_{\vec{x}}(P)$ by P. It will always be clear from the context if either the polynomial P or the function P is meant. Functions of this type are called polynomial functions (in $\mathbb{R}[\vec{\chi}]$).

Taking into account that we are dealing with polynomials over $\mathbb{R}$ we also know that the mapping

$$P \mapsto (\vec{x} \mapsto \iota_{\vec{x}}(P))$$

is an embedding of the ring of polynomials into the ring of real functions in n variables.

Apart from this we will see later that the integrals of polynomial function w.r.t. certain measures (cf. definition 2.11) exist (cf. lemmas 2.22, λ-integration, and 2.26, γ-integration) and can be computed using only algebraic ring operations.

Thus there is no need to distinguish between polynomials and polynomial functions in this thesis.

We want to point out that we regard the rings of polynomials over the rationals $\mathbb{Q}[\vec{\chi}]$ and over the integers $\mathbb{Z}[\vec{\chi}]$, resp., as subrings [5] of $\mathbb{R}[\vec{\chi}]$.

[5] Hence $P \in \mathbb{Q}[\vec{\chi}], \mathbb{Z}[\vec{\chi}]$ corresponds to a function defined on the vector space over the reals.

Later we will only deal with functions that are either polynomial functions corresponding to rational polynomials or being of the form $\vec{x} \mapsto \infty$. Therefore we arrange that the variables P, Q will range over $\bigcup_n \mathbb{Q}[\chi_1, \ldots, \chi_n] \cup \{\infty\}$ or over the corresponding set of functions, resp.

Polynomials of the form $\chi_1^{i_1}, \ldots, \chi_n^{i_n}$ are called monomials and the corresponding functions monomial functions/monomials (in $\mathbb{R}[\vec{\chi}]$).

8 Algorithms

The syntax and semantics of the algorithms should be clear from the context. We only want to point out that we use the same typed variable symbols in algorithms as in tables ii and iii.

Moreover variable assignments are denoted by $\leftarrow$.

Alphabet	Variable	Domain	Page
Latin lower case	d e	$\overline{\mathbb{R}}$	xii
	g h	$\Omega \to \overline{\mathbb{R}}$	13
	i j k l m n	$\mathbb{N}_0$	xii
	q p	$\mathbb{Q}$	xii
	r s t	$\mathbb{R}$	xii
	u $v, v_1, v_2, \ldots$ $w, w_1, w_2, \ldots$	$\mathbb{Z}$	xii
Latin upper case	A B C	$\mathcal{AL}(n)$	28
	D E	piecewise polynomials	58
	I J K	strictly monotonously increasing sequences of natural numbers	xii
	P Q	$\bigcup_n \mathbb{Q}[\chi_1, \ldots, \chi_n] \cup \{\infty\}$	xvii
	R S T	$\mathcal{LS}(n)$	28
	$X, X_1, X_2, \ldots$ $Y, Y_1, Y_2, \ldots$ $Z, Z_1, Z_2, \ldots$	sets	xii

Table iii: List of typed Latin variables

Chapter 1

Problem Statement

We present algorithms used to compute the volumes of families of convex polyhedra (cf. [Zie94]). We consider *two different* notions of volume, namely the volumes w.r.t.

1. the well known *Lebesgue* measure (cf. definition 2.21)
2. and a certain *integer counting* measure (cf. definition 2.23).

Methods that can be used in order to compute the first kind of volume of a *single* convex polyhedron are presented, for instance, in [Fah93], [HKSF94], [Ber95], [Fah96]. In contrast to this we won't compute a *number* that represents the volume of a *single* polyhedron but a *function* depending on some parameters that yields the volume of a polyhedron that is member of a *parametrised* family of polyhedra.

The computation of the volume of a geometric object of the real space is an ancient mathematical problem already treated by Greek mathematicians, as Eudoxus (400 - 347 B.C., cf. [AW89], [GIS90]), that arises in many areas of engineering even today. To know the volume of a given object is an essential need in, for example, computer aided manufacturing of custom-built products as in interior décor as well as in series production as it is usual in car industry. For if one knows the volume of a certain workpiece or of an approximation by polyhedra one can determine its cost of material very accurately. Apart from aesthetic and practical objectives a producer wants to design the shape of a product such that its production costs are minimal. As the production costs are heavily influenced by the cost of material the producer usually wants to have an overview about the volumes of alternative workpieces at the same time in order to solve this optimisation problem. Therefore he needs a function yielding the volume of a family of polyhedra and not only a single object.

Mathematically we describe this function in the following way.

Problem 1.1 (Finite real volume)
Let

$$S := \{A_i \leq a_i \mid i \in \{1, \ldots, m\}\},$$

$$\mathcal{P}(S, \vec{y}) := \left\{\vec{x} \in \mathbb{R}^k \mid A_i(\vec{x}, \vec{y}) \leq a_i, \text{ for } i = 1, \ldots, m\right\}, \text{ for } \vec{y} \in \mathbb{R}^{n-k},$$

and

$$Y := \{\vec{y} \mid \mathcal{P}(S, \vec{y}) \text{ is bounded.}\},$$

where $A_i : \mathbb{R}^n \to \mathbb{R}$ is linear and $a_i \in \mathbb{R}$, for $i = 1, \ldots, m$ and $k \leq n$.

We want to compute the function $\mathcal{R}_{S,k}$ [2] defined by

$$\begin{array}{rccl} \mathcal{R}_{S,k} & : & Y & \to & \mathbb{R}_0^+ \\ & & \vec{y} & \mapsto & \lambda_k(\mathcal{P}(S, \vec{y})), \end{array}$$

where λ_k denotes the so-called Lebesgue measure[1] (cf. definition 2.21).

□

As the functions A_i are linear the solution set $\mathcal{P}(S, \vec{y})$ of the system S of linear inequalities parametrised by $\vec{y}$ is a convex polyhedron (cf. [Zie94]).

Later we will see that the cases of $\mathcal{P}(S, \vec{y})$ being unbounded are handled by the volume algorithm presented by us, too. Therefore we generalise the problem 1.1, Finite real volume, to the problem 1.2, Real volume, for the sake of completeness and in order to simplify the statement of the problem.

Even though in practice the given system S of linear inequalities might not contain strict inequalities we will also allow strict inequalities in the input because they will arise during the application of the volume algorithm anyway. This kind of generalisation can also be found in the problem 1.2, Real volume.

Now the corresponding solution set might not be a convex polyhedron in the classical sense any more. But we still know that the closure of this set is a convex polyhedron. Therefor we will call these geometric objects "convex polyhedra with open faces" (cf. definition 2.33). Usually we will leave out the word "convex" because these solution sets are always convex. We will also omit the phrase "with open faces" because all theorems for "classical" polyhedra will be formulated or generalised such that they will hold for ones "with open faces", too.

Problem 1.2 (Real volume)
Let

$$S := \{A_i \lessdot_i a_i \mid i \in \{1, \ldots, m\}\},$$

$$\mathcal{P}(S, \vec{y}) := \left\{\vec{x} \in \mathbb{R}^k \mid A_i(\vec{x}, \vec{y}) \lessdot_i a_i, \text{ for } i = 1, \ldots, m\right\}, \text{ for } \vec{y} \in \mathbb{R}^{n-k},$$

[1] The Lebesgue measure measures the *volume* of a subset of the vector space $\mathbb{R}^k$.

where $A_i : \mathbb{R}^n \to \mathbb{R}$ is linear, $a_i \in \mathbb{R}$ and $\lessdot_i \in \mathbb{L}$, for $i = 1, \ldots, m$ and $k \leq n$.

We want to compute the function $\mathcal{R}_{S,k}$ [2] defined by

$$\begin{array}{rcccl} \mathcal{R}_{S,k} & : & \mathbb{R}^{n-k} & \to & \overline{\mathbb{R}}_0^+ \\ & & \vec{y} & \mapsto & \lambda_k \left(\mathcal{P} \left(S, \vec{y} \right) \right), \end{array}$$

where λ_k denotes the so-called Lebesgue measure (cf. definition 2.21).

□

One two dimensional instance of the problem 1.2, Real volume, without parameters can be found in the following example.

Example 1.3
Let the system S of linear inequalities given by

$$\begin{array}{rrrrrr} - & x_1 & - & 4x_2 & \leq & -9, \\ - & 2x_1 & - & x_2 & \leq & -4, \\ & x_1 & - & 2x_2 & \leq & 0, \\ & x_1 & & & \leq & 4, \\ & 2x_1 & + & x_2 & \leq & 11, \\ - & 2x_1 & + & 6x_2 & \leq & 17, \\ - & 6x_1 & - & x_2 & \leq & -6. \end{array}$$

The solution set $\mathcal{P}(S, 0)$ of S is the interior of the hexagon in figure 1.1 including its boundary.

Here the area (in general the volume) $\mathcal{R}_{S,2}$ is a number, namely $\frac{23}{4}$.

□

The next example shows a parametrised polyhedron.

Example 1.4
Let the system S of linear inequalities given by

$$\begin{array}{rrrrrrrr} - & x_1 & - & 4x_2 & & & \leq & -9 \\ - & 2x_1 & - & x_2 & & & \leq & -4 \\ & x_1 & - & 2x_2 & & & \leq & 0 \\ & x_1 & & & - & \boldsymbol{x_3} & \leq & 0 \\ & 2x_1 & + & x_2 & & & \leq & 11 \\ - & 2x_1 & + & 6x_2 & & & \leq & 17 \\ - & 6x_1 & - & x_2 & & & \leq & -6 \end{array}$$

[2]Actually we will show that $\mathcal{R}_{S,k}$ can be represented by a piecewise polynomial (cf. definition 3.3).

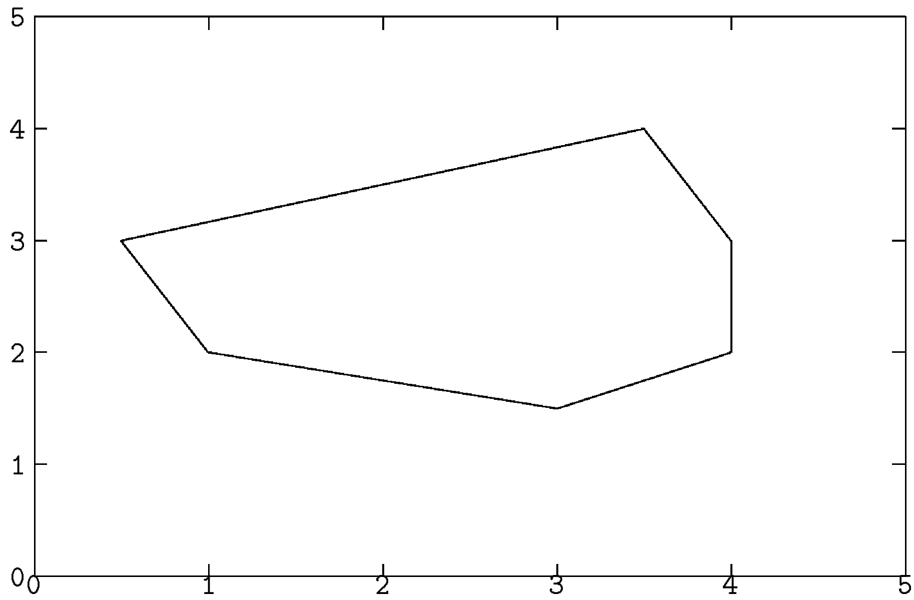

Figure 1.1: Hexagon [Zie94]

This is the system of the previous example 1.3 augmented by the parameter x_3. For $x_3 = 4$ the set $\mathcal{P}(S,4)$ exactly is the hexagon in figure 1.1. There x_3 is the distance of the small vertical edge from the vertical axis. The family of polygons that is generated by different values of x_3 is illustrated by figure 1.2. Let

$$\begin{aligned}
P^1 &:= \frac{7}{24}y^2 + \frac{7}{12}y - \frac{7}{8},\\
P^2 &:= \frac{7}{2},\\
P^3 &:= \frac{7}{6}y^2 - \frac{7}{6}y + \frac{7}{24},\\
P^4 &:= \frac{7}{24},\\
P^5 &:= -\frac{5}{4}y^2 + 11y - \frac{371}{16},\\
P^6 &:= \frac{81}{80},\\
P^7 &:= -\frac{1}{12}y^2 + \frac{17}{6}y - \frac{31}{4},\\
P^8 &:= \frac{55}{48}
\end{aligned}$$

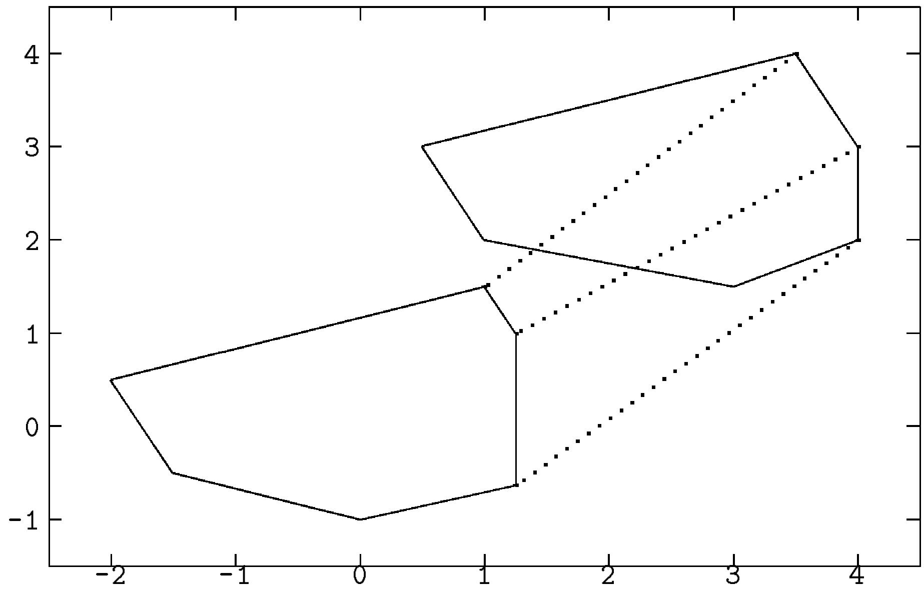

Figure 1.2: Parametrised polygon

and

$$\begin{aligned}
T^1 &:= \{x_1 \leq 3, -x_1 \leq -1\}, \\
T^2 &:= \{-x_1 < -3\}, \\
T^3 &:= \left\{x_1 < 1, -x_1 \leq -\frac{1}{2}\right\}, \\
T^4 &:= \{-x_1 \leq -1\}, \\
T^5 &:= \left\{x_1 \leq \frac{22}{5}, -x_1 \leq -\frac{7}{2}\right\}, \\
T^6 &:= \left\{-x_1 < -\frac{22}{5}\right\}, \\
T^7 &:= \left\{x_1 < \frac{7}{2}, -x_1 < -3\right\}, \\
T^8 &:= \left\{-x_1 \leq -\frac{7}{2}\right\}.
\end{aligned}$$

Then $\mathcal{R}_{S,2}$ is a function in one variable, namely

$$\mathcal{R}_{S,2}(x_3) = \sum_{i=1}^{8} P^i \mathbf{1}_{\mathcal{P}(T^i,0)}(x_3).$$

This is the result of the implementation of the volume algorithm.

□

Apart from this the real volume can be used in order to approximate the second kind of volume mentioned above. Informally spoken, computing the "integer volume" means counting the number of points located within a given polyhedron that have integer coordinates.

Although this problem might sound very simple at the first glance it is by far not trivial in the general case. It arises in many areas of applied mathematics. One of them is symbolic performance analysis of computer programs which is stimulated by the recent development of parallelising Fortran compilers. We illustrate this as follows.

Example 1.5

Figure 1.3 contains a part of a Fortran 77 program that numerically computes the twice continously differentiable solution u of the 2-dimensional boundary value problem $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f$ defined on a rectangular area parallel to the coordinate axes.

```
        DO 1 L = 2,M
           DO 1 I = L,MIN(M,L+S-1)
              C = A(I,L-1)/A(L-1,L-1)
              DO 2 J = L,MIN(M,L+R-1)
2                A(I,J) = A(I,J) - C*A(L-1,J)
1             B(I) = B(I) - C*B(L-1)
```

Figure 1.3: [Mar89, poisson1.f77]

This part of the program consists of three nested `DO` loops. The `DO` variables that are incremented by 1 at each repetition of the corresponding loops are `L`, `I` and `J`. They are of type `INTEGER`. The `INTEGER` variables `M`, `S` and `R` are unbounded parameters in the program code segment 1.3. The actual computations are performed in lines `1`, `2` and in the line preceding the innermost `DO` loop. In this case these chiefly are elementary operations on `REAL` numbers.

A compiler that automatically parallelises sequential Fortran programs must be able to measure the work load that is imposed on a single processor by nested `DO` loops as in figure 1.3. After having parallelised such a program part it must also be able to decide whether the work load is equally distributed among the available processors. These tasks result in analysing the complexity of nested `DO` loops as in 1.3, i.e. counting how often the commands in the body of the innermost loop are executed. In the example discussed here this means counting the number of executions of line `2`.

Formally we describe this in the following way in the case of program 1.3. Let $x_1, x_2, x_3, x_4, x_5, x_6 \in \mathbb{Z}$ such that $(\mathtt{L}, \mathtt{I}, \mathtt{J}) = (x_1, x_2, x_3)$ at one execution of line `2` and $(\mathtt{M}, \mathtt{S}, \mathtt{R}) = (x_4, x_5, x_6)$. Obviously this is the case iff $x_1, x_2, x_3, x_4, x_5, x_6$ satisfy system (1.1) of inequalities.

$$\begin{aligned} x_1 &\geq 2 \\ x_4 &\geq x_1 \\ x_2 &\geq x_1 \\ x_4 &\geq x_2 \\ x_1 + x_5 - 1 &\geq x_2 \\ x_3 &\geq x_1 \\ x_4 &\geq x_3 \\ x_1 + x_6 - 1 &\geq x_3 \end{aligned} \tag{1.1}$$

Counting how often line `2` is executed is equivalent to counting the number of solutions (x_1, x_2, x_3) of the system (1.1) for each possible assignment (x_4, x_5, x_6) of $(\mathtt{M}, \mathtt{S}, \mathtt{R})$ because the triple $(\mathtt{L}, \mathtt{I}, \mathtt{J})$ gets assigned a new value at each loop iteration that it hasn't hold during any previous iteration and $(\mathtt{M}, \mathtt{S}, \mathtt{R})$ remains constant during the whole computation.

$$\begin{aligned} -x_1 &\leq -2 \\ -x_1 + x_4 &\leq 0 \\ x_1 - x_2 &\leq 0 \\ x_2 - x_4 &\leq 0 \\ -x_1 + x_2 - x_5 &\leq -1 \\ x_1 - x_3 &\leq 0 \\ x_3 - x_4 &\leq 0 \\ -x_1 + x_3 - x_6 &\leq -1 \end{aligned} \tag{1.2}$$

After having transformed (1.1) into the equivalent system (1.2) of inequalities the left hand sides of each inequality are linear in the tuple

$$(x_1, x_2, x_3, x_4, x_5, x_6)$$

that is formed out of all variables occurring in (1.2). Whereas the right hand sides only consist of constants.

Therefore we arrive at the problem of counting the number of integer solutions of a parametrised system of linear inequalities.

□

The next example shows one situation when this problem arises in data parallelisation.

Example 1.6 ([HP94])
Let's have a look at the Fortran 77 program in figure 1.4 that computes the adjoint-convolution. Obviously the outermost loop could be executed com-

```
        N = 50
        DO 1 I = 1,N*N
           DO 1 K = I,N*N
1             A(I) = A(I) + X*B(K)*C(K-1)
```

Figure 1.4: [HP94, adjoint-convolution]

pletely in parallel. But let's assume that there are only four free processors available on the computer for which this program must be compiled. Therefore the work of this loop must be divided among this processors equally. That means the set $\{1, \ldots, 2500\}$, i.e. the range of the loop variable `I`, must divided in four parts $[y_1, y_2[\cap\mathbb{Z}$, $[y_2, y_3[\cap\mathbb{Z}$, $[y_3, y_4[\cap\mathbb{Z}$ and $[y_4, y_5[\cap\mathbb{Z}$, where $y_1 := 1$ and $y_5 := 2500$. Then for the j-th processor we let `I` range from y_j to $y^{j+1} - 1$.

Therefore a parallel version for a four processor machine would have the structure shown by figure 1.5. In this program the first line means that this

```
        DOPAR 1 P = 1,4
           DO 1 I = ITER(P),ITER(P+1)-1
              DO 1 K = I,2500
1                A(I) = A(I) + X*B(K)*C(K-1)
```

Figure 1.5: [HP94, parallel adjoint-convolution]

outermost loop is performed in parallel on the four available processors. The identifier `ITER` in the second line is an array of dimension 5. It plays the role of the above $y_1, \ldots, y_5$.

In order to determine the desired values of the y_j we must know how often the two inner loops are executed for a given value of `P`. The number of executions, i.e. possible assignments of the pair (`I,K`), can be described as the number of solutions of a parametrised system of linear inequalities similar to the previous example.

The values of `K` and `I` will be denoted by x_1 and x_2, resp. Whereas the values of `ITER(P)` and `ITER(P+1)` will be denoted by x_3 and x_4, resp. Then we have

$$\begin{aligned} -x_1 + x_2 &\leq 0, \\ x_1 &\leq 2500, \\ -x_2 + x_3 &\leq 0, \\ x_2 - x_4 &\leq -1. \end{aligned}$$

Again we want to know the number of integer solutions of the previous linear system of inequalities for given parameter x_3 and x_4.

□

Abstracting from examples 1.5, 1.6 and taking into account the formulation of problem 1.2, Real volume, we arrive at the mathematical problem 1.7, Number of integer solutions.

Problem 1.7 (Number of integer solutions)
Let S be as in problem 1.2, Real volume.

We want to compute the function $\mathcal{C}_{S,k}$ [3] defined by

$$\begin{array}{rccl} \mathcal{C}_{S,k} & : & \mathbb{Z}^{n-k} & \rightarrow & \overline{\mathbb{R}}_0^+ \\ & & \vec{y} & \mapsto & |\,\mathcal{P}(S,\vec{y}) \cap \mathbb{Z}^k|. \end{array}$$

□

Therefore this problem is just the problem of counting the number of integer points within the polyhedron $\mathcal{P}(S,\vec{y})$.

The reader should note that the vector $\vec{y}$ is restricted to $\mathbb{Z}^{n-k}$ in problem 1.7, Number of integer solutions, on the contrary to problem 1.2, Real volume. Later we will see that this is their only basic difference.

Example 1.8 (Ad example 1.3)
Here the number of integer points $\mathcal{C}_{S,2}$ equals 8.

□

For example 1.4 we don't include the parametrised number of integer points because the corresponding formula consists of 39 summands.

[3] We will show that $\mathcal{C}_{S,k}$ can be represented by a piecewise polynomial (cf. definition 3.3).

Chapter 2

Basic Theory

Here we provide the basic definitions and unifying theory for handling the problems 1.2, "Real volume", and 1.7, "Number of integer solutions".

Section 2.1, Measure Theory, sets up the basic ideas of measure theory which will be used subsequently. Later we will show that all known methods are certain refinements of the classical theorem 2.20 of Fubini's w.r.t. the Lebesgue and the integer measure (cf. definition 2.23), resp. Fubini's theorem is known to be a fundamental result of measure theory.

Moreover we believe that the notation of measure theory and particularly Fubini's theorem lead to a clearer unifying description of the problems "Real volume" and "Number of integer solutions" from the geometric viewpoint. Namely, Fubini's theorem says that a multiple higher dimensional integral can be computed by a reduction to lower dimensional ones. In our cases this means that either the real volume of a polyhedron or the number of integer points within a polyhedron can be determined by decomposing the polyhedron into lower dimensional ones and either by measuring the real volume of these lower dimensional polyhedra or by summing up the number of integer points within these lower dimensional polyhedra.

The basic properties of the Lebesgue measure are reviewed in section 2.2, Lebesgue Measure.

The integer measure is defined in section 2.3, Integer Measure.

Apart from this section 2.4, Polyhedra with Open Faces, introduces the notation we use for denoting polyhedra and linear systems of inequalities.

In section 2.5, Projections of Polyhedra, we show how we handle projections of polyhedra.

Finally in section 2.6, Flat Polyhedra, we will investigate polyhedra embedded in a hyperplane. We will show that the integer points of a flat polyhedron are the affine linear image of the integer points of a non flat one. Moreover we will project these integer points.

2.1 Measure Theory

Here we review the well-known definitions of measure theory, a few corollaries we will use later and a fundamental result, namely the classical Fubini's theorem 2.20.

Measure theory formalises and generalises one's intuitive idea of length, area and volume in the real space, as well as the notion of probability. This generalisation is called a measure that can be defined on more general domains than the measures already mentioned. This abstraction is achieved by specifying the unifying properties of classical and "well-known" measures as the ones above.

By using some abstract measure one can define a more general process of integration than, for example, Riemann integration of real integrable functions.

For these abstract integrals one can prove many abstract versions of useful theorems of real integration. We will state two of these results, namely the classical product measure theorem 2.19 and the classical Fubini's theorem 2.20.

Let us begin by fixing the following notation. Let Ω and Ω_i always be arbitrary sets. These ones are the universes whose "nice" subsets we want to measure. These "nice" subsets are collected in a set called σ-algebra.

Definition 2.1 (σ-algebra [Ash72])
Let $\mathcal{A}$ be a collection of subsets of Ω. Then $\mathcal{A}$ is called a σ-algebra over Ω iff $\Omega \in \mathcal{A}$ and $\mathcal{A}$ is closed under complementation and countably infinite union, that is,

1. $\Omega \in \mathcal{A}$.
2. *If* $X \in \mathcal{A}$ *then* $X^c \in \mathcal{A}$.
3. *If* $X_i \in \mathcal{A}$ *then* $\bigcup_{i=0}^{\infty} X_i \in \mathcal{A}$.

$\square$

From now on let $\mathcal{A}$ and $\mathcal{A}_i$, resp., be arbitrary σ-algebras over Ω and Ω_i.

An important special σ-algebra is the class of Borel sets. Roughly spoken, it contains all types of intervals, open and closed subsets of $\overline{\mathbb{R}}$ among other subsets that are constructed from these ones by using complementation and countably infinite union. Subsets that aren't measurable w.r.t. the Lebesgue measure (cf. definition 2.21) aren't contained in the class of Borel sets yet.

Definition 2.2 (Borel sets [Ash72]**)**
The class of Borel sets of $\overline{\mathbb{R}}$, denoted by $\mathcal{B}\left(\overline{\mathbb{R}}\right)$, is defined as the smallest σ-algebra of subsets of $\overline{\mathbb{R}}$ containing all intervals $]d, e]$.

The class of Borel sets of $\mathbb{R}^n$, denoted by $\mathcal{B}(\mathbb{R}^n)$, is defined in the same way.

□

Definition 2.3 (Measurable space [Ash72]**)**
$(\Omega, \mathcal{A})$ is called a measurable space.

□

In this thesis we will have to consider the measurable spaces $\left(\overline{\mathbb{R}}, \mathcal{B}\left(\overline{\mathbb{R}}\right)\right)$ and $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$.

Definition 2.4 (Measure [Ash72]**)**
Let (X_i) be an arbitrary infinite or finite sequence of pairwise disjoint sets in $\mathcal{A}$, then $\mu : \mathcal{A} \to \overline{\mathbb{R}}_0^+$ is a measure on $\mathcal{A}$ iff

$$\mu\left(\bigcup_i X_i\right) = \sum_i \mu(X_i).$$

□

We exclude useless measures of the form $X \mapsto \infty$. Such measures would require the consideration of many unnecessary and trivial special cases in the following definitions and proofs.

Now let μ and μ_i always be arbitrary measures defined on $\mathcal{A}$ and $\mathcal{A}_i$, resp.

Given the above ingredients we build up the measure space which can be considered as a proper generalisation of the measures already mentioned and their domains of definition.

Definition 2.5 (Measure space [Ash72]**)**
A measure space is a triple $(\Omega, \mathcal{A}, \mu)$.

□

An immediate consequence of the defining properties of a measure are the following fundamental inequalities.

Corollary 2.6 ([Ash72])
Let $X, Y \in \mathcal{A}$. Then

1. $\mu(X \cup Y) \leq \mu(X) + \mu(Y)$,
2. *if* $X \subseteq Y$, *we have* $\mu(X) \leq \mu(Y)$.

□

A certain regularity condition on a measure is the so-called σ-finiteness that the classical measures as length or area satisfy. We will only have to deal with σ-finite measures.

Definition 2.7 (σ-finite measure [Ash72])
μ is said to be σ-finite on $\mathcal{A}$ iff there exist $X_i \in \mathcal{A}$, for $i \in \mathbb{N}_0$, such that

$$\Omega = \bigcup_{i=0}^{\infty} X_i$$

and

$$\mu\left(X_i\right) < \infty \text{ for all } i \in \mathbb{N}_0.$$

□

A necessary precondition for a function to be integrable is to be measurable and in particular Borel measurable.

Definition 2.8 (Measurable [Ash72])
The map $h : \Omega_1 \to \Omega_2$ is measurable relative to the σ-algebras $\mathcal{A}_1$ and $\mathcal{A}_2$ iff

$$h^{-1}(Y) \in \mathcal{A}_1 \text{ for each } Y \in \mathcal{A}_2.$$

□

From now on g, h will always denote arbitrary functions from Ω to $\overline{\mathbb{R}}$.

Definition 2.9 (Borel measurable [Ash72])
The map h is said to be Borel measurable on $(\Omega, \mathcal{A})$ iff h is measurable relative to the σ-algebras $\mathcal{A}$ and $\mathcal{B}\left(\overline{\mathbb{R}}\right)$.

□

Now the definition of the integral is performed stepwise. First it is defined on so-called simple functions. Then this definition is extended to nonnegative and to arbitrary ones, in turn.

Definition 2.10 (Simple functions [Ash72])
The map h is said to be simple iff

$$h = \sum_{i=0}^{l} y_i \mathbf{1}_{X_i},$$

where the X_i are pairwise disjoint sets in $\mathcal{A}$ and $y_i \in \overline{\mathbb{R}}$, for $i = 0, \dots, l$.

□

Obviously simple functions are Borel measurable.

Definition 2.11 (Integral [Ash72]**)**
Let h be Borel measurable.

Case h **simple:** *Let $h = \sum_{i=0}^{l} y_i \mathbf{1}_{X_i}$ as in definition 2.10. We define*

$$\int_\Omega h\, d\mu := \sum_{i=0}^{l} y_i\, \mu(X_i)\,. \tag{2.1}$$

if both $+\infty$ and $-\infty$ don't occur in the above sum. Otherwise we say the integral doesn't exist.

Case h **nonnegative:**

$$\int_\Omega h\, d\mu := \sup\left\{\int_\Omega g\, d\mu \mid g \text{ simple, for } 0 \le g \le h\right\}.$$

Otherwise:

$$\int_\Omega h\, d\mu := \int_\Omega h^+\, d\mu - \int_\Omega h^-\, d\mu \tag{2.2}$$

if the right hand side of 2.2 is not of the form $\infty - \infty$. Otherwise we say the integral doesn't exist.

If $X \in \mathcal{A}$ we define

$$\int_X h\, d\mu := \int_\Omega h \mathbf{1}_X\, d\mu.$$

We say that $\int_X h\, d\mu$ exists iff $\int_\Omega h\mathbf{1}_X\, d\mu$ exists. We also use the notation $\int_X h(x)\, d\mu(x)$ for $\int_X h\, d\mu$. If $\Omega = \mathbb{R}$ and $X = [s,t]$ we define

$$\int_s^t h\, d\mu := \begin{cases} \int_{[s,t]} h\, d\mu & \text{if } s \le t, \\ -\int_{[t,s]} h\, d\mu & \text{else } (s > t). \end{cases}$$

□

Corollary 2.12 ([Ash72])
The integral in definition 2.11 is well defined.

□

We review some properties of the integral now.

The one of the most elementary properties of the integral is its linearity, i.e. its additivity and homogeneity. We state the additivity theorem explicitely because the following corollary that we will use in section 2.3, Integer Measure, is based on it.

Theorem 2.13 (Additivity theorem [Ash72])
Let g, h be Borel measurable functions, and assume that $f + g$ is well-defined. If $\int_\Omega g\,d\mu$ and $\int_\Omega h\,d\mu$ exist and $\int_\Omega g\,d\mu + \int_\Omega h\,d\mu$ is well-defined, then

$$\int_\Omega g + h\,d\mu = \int_\Omega g\,d\mu + \int_\Omega h\,d\mu.$$

□

Corollary 2.14
Let $X, Y \in \mathcal{A}$ with $X \cap Y = \emptyset$ and h be Borel measurable such that $\int_X h\,d\mu$ and $\int_Y h\,d\mu$ exist. Then

$$\int_{X \cup Y} h\,d\mu = \int_X h\,d\mu + \int_Y h\,d\mu.$$

□

Proof:

Note that $\mathbf{1}_{X \cup Y} = \mathbf{1}_X + \mathbf{1}_Y$ and apply the additivity theorem 2.13.

□

The next theorem says that integrating over functions that differ only on a set of measure zero always yields the same result.

Theorem 2.15 ([Ash72])
Let g, h be Borel measurable functions. If there exists a set $X \in \mathcal{A}$ with $\mu(X) = 0$ such that $g = h$ on $\Omega \backslash X$ and $\int_\Omega g\,d\mu$ exists, then so does $\int_\Omega h\,d\mu$, and

$$\int_\Omega g\,d\mu = \int_\Omega h\,d\mu.$$

□

We won't use theorem 2.15 directly but a certain special case of this theorem. Namely, we want to ensure that integrating one function over sets that differ only up to a set of measure zero yields always the same result. This is formulated in the following corollary. But first we will introduce a standard notation for this kind of set difference.

Definition 2.16 (Set difference)
We define

$$X \bigtriangleup Y := (X \cup Y) \setminus (X \cap Y).$$

□

Corollary 2.17
Let $X, Y \in \mathcal{A}$ with $\mu(X \bigtriangleup Y) = 0$ and h be Borel measurable such that $\int_X h \, d\mu$ exists. Then

$$\int_X h \, d\mu = \int_Y h \, d\mu.$$

In particular $\int_Y h \, d\mu$ exists.

□

Proof:

Note that

$$X \bigtriangleup Y \in \mathcal{A}$$

and

$$h\mathbf{1}_X = h\mathbf{1}_Y$$

on

$$\Omega \setminus (X \bigtriangleup Y).$$

Therefore the corollary is an immediate consequence of theorem 2.15.

□

The next lemma says if X, Y and X, Z differ up to a set of measure zero, resp., then so do Y and Z. We will use this lemma in section 2.5, Projections of Polyhedra, when we project polyhedra to lower dimensional subspaces.

Lemma 2.18
Let $X, Y, Z \in \mathcal{A}$ and

$$\mu(X \bigtriangleup Y) = 0,$$
$$\mu(X \bigtriangleup Z) = 0.$$

Then

$$\mu(Y \bigtriangleup Z) = 0.$$

□

Proof:

We show that

$$Y \triangle Z \subseteq (X \triangle Y) \cup (X \triangle Z). \tag{2.3}$$

We have

$$\begin{aligned} X \triangle Y \quad &\cup \quad X \triangle Z = \\ X \backslash (X \cap Y) \cup Y \backslash (X \cap Y) \quad &\cup \quad X \backslash (X \cap Z) \cup Z \backslash (X \cap Z) = \\ X \backslash (X \cap Y) \cup X \backslash (X \cap Z) \quad &\cup \quad Y \backslash (X \cap Y) \cup Z \backslash (X \cap Z) = \\ X \backslash (Y \cap Z) \quad &\cup \quad (Y \cup Z) \backslash X. \end{aligned}$$

Take $x \in Y \triangle Z$ arbitrarily. Assume that

1. $x \in X$. Then we know that $x \in X \backslash (Y \cap Z)$.
2. $x \notin X$. Then we have $x \in (Y \cup Z) \backslash X$.

This shows line (2.3).

Now the application of the two inequalities of corollary 2.6 to line (2.3) proves the lemma.

□

Now we state the main theorems of this subsection, the classical product measure theorem and the classical Fubini's theorem.

Theorem 2.19 (Classical product measure theorem [Ash72]**)**
Let μ_1, μ_2 be σ-finite on $\mathcal{A}_1, \mathcal{A}_2$, $\Omega := \Omega_1 \times \Omega_2$, $\mathcal{A} := \mathcal{A}_1 \times \mathcal{A}_2$ and

$$\begin{aligned} Z_1(y) &:= \{x \in \Omega_1 \,|\, (x, y) \in Z\}, \\ Z_2(x) &:= \{y \in \Omega_2 \,|\, (x, y) \in Z\}, \text{ for } Z \in \mathcal{A}. \end{aligned}$$

The function given by

$$\begin{array}{rcll} \mu & : & \mathcal{A} & \to \overline{\mathbb{R}}_0^+ \\ & & Z & \mapsto \int_{\Omega_1} \mu_2(Z_2(x))\, d\mu_1(x) = \int_{\Omega_2} \mu_1(Z_1(y))\, d\mu_2(y) \end{array}$$

is the unique measure on $\mathcal{A}$ such that

$$\mu(X \times Y) = \mu_1(X)\, \mu_2(Y) \text{ for all } X \in \mathcal{A}_1, Y \in \mathcal{A}_2.$$

Furthermore, μ is σ-finite on $\mathcal{A}$.

The measure μ is called the product of μ_1 and μ_2, written $\mu_1 \otimes \mu_2$.

□

Theorem 2.20 (Classical Fubini's Theorem [Ash72]**)**
Let $\Omega := \Omega_1 \times \Omega_2$, $\mathcal{A} := \mathcal{A}_1 \times \mathcal{A}_2$, $\mu := \mu_1 \otimes \mu_2$, where μ_1, μ_2 are σ-finite measures, and h be Borel measurable and such that $\int_\Omega h\, d\mu$ exists, then

$$\int_\Omega h\, d\mu = \int_{\Omega_1} \int_{\Omega_2} h\, d\mu_2\, d\mu_1 = \int_{\Omega_2} \int_{\Omega_1} h\, d\mu_1\, d\mu_2.$$

□

Fubini's theorem says that when a lower dimensional decomposition of the space Ω is known then the integral on this domain can be reduced to integrals on these lower dimensional domains. Obviously the way of this decomposition strongly depends on the nature of the space Ω and the corresponding measure.

2.2 Lebesgue Measure

Now we state the definition of the Lebesgue measure and some of its basic properties that are used subsequently. The Lebesgue Measure is used in order to formulate the problem 1.2 "Real volume" in the language of measure theory (cf. problem 3.1 "Lebesgue volume").

Definition 2.21 (Lebesgue measure on $\mathbb{R}^n$ [Ash72]**)**
Let

$$\lambda_n\left(]\vec{s}, \vec{t}]\right) := \prod_{i=1}^{n} (t_i - s_i).$$

The unique (cf. [Ash72]) extension of λ_n to $\mathcal{B}(\mathbb{R}^n)$ is called Lebesgue measure on $\mathbb{R}^n$. The extension is also denoted by λ_n.

□

It is a well known fact ([Ash72]) that $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n), \lambda_n)$ is a measure space, where λ_n is σ-finite. Moreover $\lambda_{i+j} = \lambda_i \otimes \lambda_j$.

For the rest of this work we fix the following abbreviation. We will denote λ_i by λ because it will always be clear from the context which λ_i is meant.

The methods solving the problem "Real volume" require to integrate polynomials w.r.t. Lebesgue measure. Because of the linearity of the integral this task can be reduced to the integration of monomial functions x^j.

Elementary analysis (cf. [Ash72], [Heu91]) tells us how this can be done. We state the corresponding formulae in the following lemma.

Lemma 2.22 (λ-integration)
We have[1]

$$\int_{\{u \mid s \lessdot u \lhd t\}} x^j \, d\lambda(x) = \begin{cases} 0 & \text{if } s > t, \\ \frac{1}{j+1}\left(t^{j+1} - s^{j+1}\right) & \text{else}^2 (s \leq t). \end{cases}$$

□

2.3 Integer Measure

The following definition lays the ground for formulating the problem "Number of integer solutions" in the language of measure theory. This can be seen in the definition of problem 3.2 "Integer volume" that is equivalent to the original problem "Number of integer solutions".

We will also exhibit how integrals of polynomials with respect to this measure can be computed.

Definition 2.23 (Integer measure on $\mathbb{R}^n$)
The integer measure on $\mathbb{R}^n$ has the following form:

$$\begin{array}{rcccl} \gamma_n & : & \mathcal{B}(\mathbb{R}^n) & \to & \overline{\mathbb{N}}_0 \\ & & X & \mapsto & |X \cap \mathbb{Z}^n|. \end{array}$$

□

In order to apply the results of section 2.1, Measure Theory, we have to show that $\mathbb{R}^n$ together with the integer measure defined above set up a measure space. This is done in the proof of the following corollary.

Corollary 2.24
$(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n), \gamma_n)$ *is a measure space, where γ_n is σ-finite.*

□

Proof:

[1] Note that this lemma states that the definite integral can be computed by algebraic operations.

[2] We want to remark that the treatment of this case division heavily influences the structure of the algorithms solving the problem "Real volume". This division leads to a certain decomposition of the given polyhedron.

We already know that $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ is a measurable space.

Let (X_i) be a sequence of pairwise disjoint sets in $\mathcal{B}(\mathbb{R}^n)$. Then $(X_i \cap \mathbb{Z}^n)$ is a pairwise disjoint sequence, too. Hence

$$\gamma_n\left(\bigcup_i (X_i \cap \mathbb{Z}^n)\right) = \left|\bigcup_i (X_i \cap \mathbb{Z}^n)\right| = \sum_i |X_i \cap \mathbb{Z}^n| = \sum_i \gamma_n(X_i).$$

Let $Y_i := \{r \mid -i \leq r < i\}$, $Z_i := (Y_i)^n$. Then $\mathbb{R}^n = \bigcup_i Z_i$ and $\gamma_n(Z_i) = (2i)^n < \infty$.

□

The following corollary is the basis for the proper application of Fubini's theorem in the situation of problem 1.7, "Number of integer solutions".

Corollary 2.25
$\gamma_{i+j} = \gamma_i \otimes \gamma_j$.

□

Proof:
We have to show that $\gamma_{i+j}(X \times Y) = \gamma_i(X)\gamma_j(Y)$, for all $X \in \mathcal{B}(\mathbb{R}^i), Y \in \mathcal{B}(\mathbb{R}^j)$. This holds because

$$\begin{aligned}
\gamma_{i+j}(X \times Y) &= \\
&= |(X \times Y) \cap \mathbb{Z}^{i+j}| = \\
&= |(X \cap \mathbb{Z}^i) \times (Y \cap \mathbb{Z}^j)| = \\
&= |X \cap \mathbb{Z}^i||Y \cap \mathbb{Z}^j| = \\
&= \gamma_i(X)\gamma_j(Y).
\end{aligned}$$

□

From now on we will abbreviate γ_i by γ because it will always be clear from the context which γ_i is meant.

The methods solving the problem "Number of integer solutions" require to integrate polynomials w.r.t. the measure γ. Because of the linearity of the integral this task can be reduced to the integration of monomials x^j. The following lemma indicates how this can be done.

Lemma 2.26 (γ-integration)
We have[3]

$$\int_{\{u \mid s \lessdot u \lhd t\}} x^j \, d\gamma(x) = \sum_{x=v}^{w} x^j,$$

[3]Note that this lemma states that the definite integral can be computed by algebraic operations.

where

$$v := \begin{cases} \lceil s \rceil & \textit{if} \lessdot = \leq, \\ \lfloor s \rfloor + 1 & \textit{else} \ (\lessdot = <), \end{cases}$$
$$w := \begin{cases} \lfloor t \rfloor & \textit{if} \ \lhd = \leq, \\ \lceil t \rceil - 1 & \textit{else} \ (\lhd = <). \end{cases}$$

□

Proof:

As the integral $\int_{[v,w]} x^j \, d\lambda$ exists (cf. lemma 2.22) we know that $x^j \mathbf{1}_{[v,w]}$ is Borel measurable. As obviously

$$\gamma\left([v,w] \bigtriangleup \{r \,|\, s \lessdot r \lhd t\}\right) = 0$$

it suffices to show that

$$\int_{[v,w]} x^j \, d\gamma = \sum_{x=v}^{w} x^j$$

by the application of corollary 2.17. Now, let

1. $v > w$. In this case we get $[v,w] = \emptyset$. Therefore

$$\int_{[v,w]} x^j \, d\gamma = 0 = \sum_{x=v}^{w} x^j.$$

2. $v \leq w$ and let

   (a) $j = 0$. Then we have

$$\begin{aligned} \int_{[v,w]} 1 \, d\gamma &= \gamma\left([v,w]\right) \\ &= v - w + 1 \\ &= \sum_{x=v}^{w} 1. \end{aligned}$$

   (b) $j \geq 1$. We prove this case by induction on w.

      i. Let $w = v$. Then $\int_{[v,v]} x^j \, d\gamma = v^j \gamma\left(\{v\}\right) = v^j = \sum_{x=v}^{v} x^j$.

      ii. We assume that $\int_{[v,w]} x^j \, d\gamma = \sum_{x=v}^{w} x^j$.

      iii. $w \to w + 1$: We must show that $\int_{[v,w+1]} x^j \, d\gamma = \sum_{x=v}^{w+1} x^j$.

iv. Proof: First we show that

$$\int_{]w,w+1]} x^j \, d\gamma = (w+1)^j \,. \tag{2.4}$$

Obviously $x^j \mathbf{1}_{[w+1,w+1]}$ is Borel measurable and

$$\int_{[w+1,w+1]} x^j \, d\gamma = (w+1)^j \,.$$

Then corollary 2.17 yields equation (2.4).
By the use of this equation and the induction hypothesis we derive that

$$\begin{aligned} \int_{[v,w+1]} x^j \, d\gamma &= \int_{[v,w]} x^j \, d\gamma \; + \; \int_{]w,w+1]} x^j \, d\gamma \\ &= \sum_{x=v}^{w} x^j \; + \; (w+1)^j \\ &= \sum_{x=v}^{w+1} x^j \end{aligned}$$

□

In order to compute the above integrals we must be able to evaluate sums of the type

$$\sum_{x=v}^{w} x^j, \tag{2.5}$$

where j is a given non negative integer constant.

In order to perform the summation we use the following well known formulae.

Theorem 2.27 (Symbolic summation [Haz88], [Taw95]**)**

1. Let $v > w+1$. Then

$$\sum_{x=v}^{w} x^j = 0.$$

2. *Let* $v \leq w + 1$[4]. *Then*[5] *we have*

$$\sum_{x=1}^{n} x^j = \frac{1}{j+1} \sum_{k=0}^{j} \binom{j}{k} \beta_k \, (n+1)^{j+1-k}\,, \tag{2.6}$$

where the β_k *are the so-called Bernoulli numbers, defined as follows (cf. [Kno64], [Haz88]).*

$$\beta_0 := 1$$
$$\sum_{i=0}^{k} \binom{k+1}{i} \beta_i = 0, \ \textit{for } k \geq 1.$$

□

Now, if $v > w + 1$ we know that the sum is zero, anyway. Therefore assume $v \leq w + 1$.

If $j = 0$ the sum in line (2.5) simply has the form

$$\sum_{x=v}^{w} x^j = w - v + 1.$$

For $j \geq 1$, we compute the sum

1. by index shifting, namely $\sum_{x=v}^{w} x^j = \sum_{x=1}^{w-v+1} (x + v - 1)^j$,
2. by an expansion of $(x + v - 1)^j$ and
3. by a repeated application of theorem 2.27 to the monomials x^l occurring in the expansion, where $0 \leq l \leq j$.

Later we will iteratively apply the γ-integration lemma 2.26 in order to solve the problem "Number of integer solutions". This results in the computation of nested sums as $\sum_{x_k=v_k}^{w_k} \cdots \sum_{x_1=v_1}^{w_1}$. But note that the v_i and w_i will contain floor- and ceiling functions of the forms $\lfloor \frac{P}{u} \rfloor$ and $\lceil \frac{P}{u} \rceil$, resp., where $u > 0$. To be completely precise P will be a linear polynomial with integer coefficients. Regarded as function P will also be restricted to $\mathbb{Z}^k$, for some k.

As we can't sum over expressions containing the floor and ceiling functions symbolically we will have to remove them. For this there are two different approaches:

---

[4] This case division is essential because only in this case the application of the following summation formulae yields the correct results in general. How to handle this case division is one of the main problems when integrating w.r.t. the measure γ.

This motivates Tawbi's definition 4.25 of a *regular* system of inequalities, too.

[5] The reader can easily check that the formula for symbolic summation found in [Taw95] that is also cited in [Pug94] is wrong. The correct version can be found in [Haz88].

1. approximate summation,
2. disjoint splintering.

For approximate summation itself there are known two basic techniques. Namely,

1. estimation of the average values of the floor- and ceiling-errors,
2. computation of lower, upper bounds and best-guess answers.

Let's start:

Estimation of the average values. This technique is proposed by Tawbi (cf. [Taw95]). She observes that

1. the floor error E_f equals $\lfloor \frac{v}{w} \rfloor - \frac{v}{w} = -\frac{v \mod w}{w}$ and
2. the ceiling error E_c equals $\lceil \frac{v}{w} \rceil - \frac{v}{w} = 1 - \frac{v \mod w}{w}$.

Assuming that $v \mod w$ is distributed uniformly in $\{0, \ldots, w-1\}$ we can compute the average floor error $\bar{E}_f$ and the average ceiling error $\bar{E}_c$. We derive that

1. $\bar{E}_f = \frac{1}{w}\left(\sum_{i=0}^{w-1} -\frac{i}{w}\right) = -\frac{w-1}{2w}$,
2. $\bar{E}_c = \frac{1}{w}\sum_{i=0}^{w-1}\left(1 - \frac{i}{w}\right) = \frac{w+1}{2w}$.

Lower, upper bounds and best-guess answers. This approximation technique was introduced by Pugh (cf. [Pug94]). He uses the following simple inequalities[6]:

$$\frac{v-(w-1)}{w} \leq \left\lfloor \frac{v}{w} \right\rfloor \leq \frac{v}{w} \leq \left\lceil \frac{v}{w} \right\rceil \leq \frac{v+(w-1)}{w}.$$

As the sum $\sum_{x=v}^{w} x^j$ is monotonous decreasing in v and increasing in w we can replace the floor and ceiling functions occurring in v and w by their corresponding bounds. We might also replace them by the best-guess answer $\frac{v}{w}$ or by the mean value of the upper and lower bound.

Apart from approximate summation we can do the following.

[6] Pugh (cf. [Pug94]) writes that both v and w must be positive in order to ensure the validity of the inequalities. But obviously they are also valid for non positive v

Disjoint splintering. This method is also due to Pugh's (cf. [Pug94]).

1. Assuming that $w|v-i$ we get that $\left\lfloor \frac{v}{w} \right\rfloor = \frac{v-i}{w}$ and
2. assuming that $w|v+i$ we have that $\left\lceil \frac{v}{w} \right\rceil = \frac{v+i}{w}$,

where $i = 0, \ldots, w-1$. We illustrate disjoint splintering in the following example.

Example 2.28 ([Pug94])

$$\sum_{i=1}^{\lfloor \frac{n}{3} \rfloor} i = \begin{cases} \sum_{i=1}^{\frac{n}{3}} i = \frac{n(n+33)}{18} & \text{if } 3|n, \\ \sum_{i=1}^{\frac{n-1}{3}} i = \frac{(n-1)(n+2)}{18} & \text{if } 3|\left(n-1\right), \\ \sum_{i=1}^{\frac{n-2}{3}} i = \frac{(n-2)(n+1)}{18} & \text{if } 3|\left(n-2\right). \end{cases}$$

$\square$

Later we will equivalently write[7]

$$-x + \frac{1}{w}v - \frac{i}{w} \;=\; 0, \text{ for some } x \in \mathbb{Z},$$

and

$$y + \frac{1}{w}v + \frac{i}{w} \;=\; 0, \text{ for some } y \in \mathbb{Z},$$

resp., instead of

$$\frac{1}{w}v - \frac{i}{w} \;\in\; \mathbb{Z}$$

and

$$\frac{1}{w}v + \frac{i}{w} \;\in\; \mathbb{Z},$$

resp., which means

$$w|v-i$$

and

$$w|v+i,$$

resp.

[7]This is the motivation for the definition of so-called flat polyhedra in section 2.6.

2.4 Convex Polyhedra with Open Faces

Here our main goal is to clarify what we understand by a "convex polyhedron with open faces".

The objects with which we have to deal are the solution sets of linear systems of inequalities with rational coefficients. Intuitively, we regard such a system as a finite sequence of single strict or non strict inequalities that are linear in all occurring variables. Traditionally, these convex solution sets of linear systems of non strict inequalities are called convex polyhedra (cf. [Zie94]). As we are also dealing with strict inequalities which means that the corresponding polyhedron might be open in direction to some face we call these objects "convex polyhedra with open faces".

Now, out of these inequalities one can build an affine linear mapping as can be seen in the next example.

Example 2.29
We equivalently transform the linear system of inequalities S given by

$$\frac{1}{2}x_1 + \frac{2}{3}x_2 < 4, \tag{2.7}$$

$$-7x_1 + \frac{4}{9}x_2 \leq \frac{11}{7}, \tag{2.8}$$

$$x_2 < 2 \tag{2.9}$$

into

$$A\vec{x} := \begin{bmatrix} \frac{1}{2} & \frac{2}{3} \\ -7 & \frac{4}{9} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} 4 \\ \frac{11}{7} \\ 2 \end{pmatrix} \begin{matrix} < \\ \leq \\ < \end{matrix} 0.$$

We see that the main ingredient of the above linear system is the affine linear function A and the stack of comparison relations $\substack{< \\ \leq \\ <}$. Moreover we remark that all occurring numbers are rational.

□

Apart from this we will show how to select "parts" of affine linear functions and linear systems of inequalities and how to combine them again. If a linear system of inequalities is given in a representation as above this means how to choose certain rows and columns of the matrix and vector involved and how to build a new affine linear mapping and linear system of inequalities out of them. We need to do this because this allows us to decompose the corresponding polyhedra as we will see in chapters 3, Abstract Treatment of the Problem, and 4, State of the Art.

Moreover we will define a function that allows us to compute so-called sections of affine linear functions. We need them in order to reduce a polyhedron to a family of lower dimensional ones according to the classical Fubini's theorem.

Let's start with affine linear functions, the basic ingredients of linear systems of inequalities.

Definition 2.30 (Affine linear functions with rational coefficients) *Let $\vec{x} \in \mathbb{R}^n$, $A, B : \mathbb{R}^n \to \mathbb{R}^{(\mathbb{N})}$. Then we define the sum of A and B, as well as multiplication of A by a real number as usual, namely:*

$$(A + B)\,\vec{x} := A\vec{x} + B\vec{x},$$
$$(r \cdot A)\,\vec{x} := r \cdot (A\vec{x})\,.$$

Then we denote the set of affine linear functions from $\mathbb{R}^n$ to $\mathbb{R}^{(\mathbb{N})}$[8] with rational coefficients[9] by

$$\mathcal{AL}\,(n) := \left\{A : \mathbb{R}^n \to \mathbb{R}^{(\mathbb{N})} \,|\, A \text{ is affine linear and } A\,(\mathbb{Q}^n) \subseteq \mathbb{Q}^{(\mathbb{N})}.\right\}$$

We call

$$(\mathcal{AL}\,(n), +, \cdot)$$

the vector space of affine linear functions from $\mathbb{R}^n$ to $\mathbb{R}^{(\mathbb{N})}$ with rational coefficients. It will always be clear from the context if the vector space or the set $\mathcal{AL}\,(n)$ is meant.

Moreover for $\breve{a} \in \mathbb{R}^{(\mathbb{N})}$ we define:

$$(A + \breve{a})\,\vec{x} := (\breve{a} + A)\,\vec{x} := A\vec{x} + \breve{a}.$$

□

[8]The reader might wonder why we consider affine linear maps whose image lie in an infinite dimensional vector space. At the first glance this seems to be unnecessary. For the image itself is of finite dimension because the pre-image is finite dimensional. Nevertheless we use this kind of affine linear functions because later in chapter 4, State of the Art, we will have to intersect the solution sets of linear systems of inequalities, i.e. polyhedra, very often. This means combining different linear systems. The images of the resulting affine linear maps that represent the resulting systems will usually be of different dimensions than the images of the original mappings. As it is unnecessary to keep track of these dimensions we let our affine linear functions map into a space that can contain all images of any finite dimension.

[9]We say with "with rational coefficients" because these functions can be fully specified by rational numbers as we will see in lemma 2.43

As we will only deal with affine linear functions *with rational coefficients*, we will leave out "with rational coefficients" when we speak about this functions and no misunderstandings are possible.

For the rest of this thesis let us fix the following notation. The maps A, B, C will always be in $\mathcal{AL}(n)$.

Corollary 2.31
In fact, $\mathcal{AL}(n)$ is a vector space over the real numbers.

□

Proof:

The verification of the vector space structure of $\mathcal{AL}(n)$ is trivial and left to the reader.

□

Given an affine linear function and a stack in $\mathbb{L}^{(\mathbb{N})}$. we can form a linear system of inequalities now.

Definition 2.32 (Linear system of inequalities)
We call

$$\mathcal{LS}(n) := \mathcal{AL}(n) \times \mathbb{L}^{(\mathbb{N})}$$

the set of linear systems of inequalities. The linear system of inequalities $(0, \leq)$ *will be denoted by* 0.

By defining

$$(A, \lessdot) + (B, \lhd) := (A + B, \lessdot \cdot \lhd)$$

$(\mathcal{LS}(n), +)$ *forms a monoid with zero being* $(0, \leq)$.

Let $(A, \lessdot) \in \mathcal{LS}(n)$. *Then we say* $\vec{x}$ *satisfies* $(A, \lessdot)$ *iff* $A\vec{x} \lessdot 0$.

Moreover $\mathrm{Type}(A, \lessdot) := \lessdot$.

□

Let us fix the following notation. The letters R, S, T will always denote linear systems of inequalities in $\mathcal{LS}(n)$.

Often we will write "system" or "system of inequalities" instead of "linear system of inequalities" or "system of linear inequalities" because these are the only "systems" we are dealing with in this work.

We call the solution set of a linear system of inequalities a convex polyhedron with open faces. But mostly we will abbreviate this phrase simply by "polyhedron".

The word "convex" is left out because non convex polyhedra with open faces cannot be specified by linear systems of inequalities (compare Ziegler"s, [Zie94], proof for "traditional" polyhedra) and we only deal with such polyhedra.

The phrase "with open faces" is omitted because there won't be any need to distinguish between "classical" polyhedra and polyhedra "with open faces" in most cases (compare also chapter 1, Problem Statement).

Definition 2.33 (Polyhedron with open faces)

1. *Then*
$$\mathcal{P}(S) := \{\vec{x} \in \mathbb{R}^n \mid \vec{x} \text{ satisfies } S\}$$
is called the polyhedron with open faces defined by S.

2. *We denote the set of integer points of $\mathcal{P}(S)$ by $\mathcal{Z}(S)$, i.e.*
$$\mathcal{Z}(S) := \mathcal{P}(S) \cap \mathbb{Z}^n.$$

□

Note that $(\mathcal{P}(\mathcal{LS}(n)), \cap)$ forms an Abelian monoid whose zero is the empty set $\emptyset$. Unfortunately, this monoid isn't a homomorphic image of the monoid $(\mathcal{LS}(n), +)$. But later in this section, we will define an alternative operation $\uplus$ on $\mathcal{LS}(n)$ such that $(\mathcal{P}(\mathcal{LS}(n)), \cap)$ is a homomorphic image of this monoid (cf. definition 2.47).

We need to construct such an operation $\uplus$ because we want to compute with elements in $\mathcal{LS}(n)$ such that this computations are compatible with the intersection in the monoid $(\mathcal{P}(\mathcal{LS}(n)), \cap)$.

Next we define a function that yields a bound on the number of components of the elements of the image of an affine linear function. Using this bound we can derive a certain finite representation of these mappings.

We also extend this bound to linear systems of inequalities. Later, this will allow us to take the "union" of linear systems.

Definition 2.34 (Size)
Let

$$\begin{array}{rcll} \|.\|_n & : & \mathcal{AL}(n) & \to \mathbb{N}_0 \\ & & A & \mapsto \begin{cases} 0 & \text{if } A = 0, \\ \max\{i \mid a_i \neq 0, \text{ for some } \check{a} \in A(\mathbb{R}^n)\} & \text{else.} \end{cases} \end{array}$$

Then we call $\|A\|_n$ *the size*[10] *of* A*. If* $\breve{a} \in \mathbb{R}^{(\mathbb{N})}$ *and* $\vec{x} \mapsto \breve{a} \in \mathcal{AL}(n)$ *we say*

$$\|\breve{a}\|_n := \|\vec{x} \mapsto \breve{a}\|_n$$

is the size of $\breve{a}$*.*

For $\lessdot \in \mathbb{L}^{(\mathbb{N})}$ *we define*

$$\|\lessdot\|_n := \begin{cases} 0 & \text{iff } \lessdot = \leq \\ \max\{i \mid \lessdot_i \neq \leq\} & \text{else.} \end{cases}$$

We extend the size to linear systems of inequalities in the following way. Let $(A, \lessdot) := S$*. Then*

$$\|S\|_n := \max\left\{\|A\|_n, \|\lessdot\|_n\right\}.$$

Usually we will abbreviate $\|.\|_n$ *by* $\|.\|$*.*

□

Corollary 2.35
The size $\|.\|$ *is well defined.*

□

Proof:

The size of a linear system is well defined if the size of an affine linear function is well defined.

For a constant map A the size $\|.\|$ is obviously well defined. Therefore we assume that A isn't constant.

We show that the set

$$I := \{i \mid a_i \neq 0, \text{ for some } \breve{a} \in A(\mathbb{R}^n)\}$$

is finite which implies that its maximum exists.

We know that $\mathbb{R}^n$ is spanned by the set $\{\vec{y_j} \mid j = 1, \ldots, n\}$, where

$$y_{j_l} := \begin{cases} 1 & \text{if } j = l, \\ 0 & \text{else.} \end{cases}$$

[10] Especially in the area of functional analysis the symbol $||.||$ is used to denote a function having the property of a norm (cf. [Heu92]). Here we use this symbol because the size yields an upper bound of the dimension of the image space of an affine linear mapping. In other words, it roughly measures the number of non zero components of vectors in the image of an affine linear function.

Although the size satisfies the triangle inequality $||A+B|| \leq ||A||+||B||$ and the property, $||A|| = 0$ iff $A = 0$, which can be checked easily it is not a norm. For it is not multiplicative, i.e. it doesn't satisfy the relation $||r \cdot A|| = |r| \cdot ||A||$.

Therefore $(A - A(0))(\mathbb{R}^n)$ is spanned by the finite set

$$Z := \{(A - A(0))\,\vec{y_j} \mid j = 1, \ldots, n\}.$$

Now let $k := \max\{i \mid a_i \neq 0, \text{ for some } \breve{a} \in Z\}$.

As all elements of $(A - A(0))(\mathbb{R}^n)$ are linear combinations of elements of Z and the elements of $A(\mathbb{R}^n)$ are these linear combinations plus $A(0)$ we conclude that $I \subseteq \{1, \ldots, \max\{k, \|A(0)\|\}\}$ and

$$\|A\| = \max\{k, \|A(0)\|\}. \tag{2.10}$$

□

It is a well known fact of linear algebra that an affine linear function can be decomposed into a matrix and a vector. Here this means

$$A\vec{x} = \breve{M}\vec{x} - \breve{a}.$$

Note that $\breve{M}$ has infinitely many rows.

Therefore we will represent an affine linear function in the form "matrix minus[11] vector", i.e. $\breve{M} - \breve{a}$, in examples.

Because of equation 2.10 we also know that the last non zero row of $\breve{M}$ can be at most the $\|S\|$-th one, if $S = (A, \lessdot)$. Moreover $\breve{a} = -A(0)$, where $a_i = 0$, for $i > \|S\|$.

Therefore in examples we will omit these vanishing rows and vector elements and represent a system $(A, \lessdot)$ by a matrix $\vec{M}$ of dimension $\|S\| \times n$ minus a vector $\vec{a}$ of length $\|S\|$, i.e. $\vec{M} - \vec{a}$ and an element $\lhd \in \mathbb{L}^{\|S\|}$. If A is constant then $\vec{M}$ consists only of zeros. Later (cf. lemma 2.43) we will see that $\vec{M}$ and $\vec{a}$ only contain rational numbers.

Moreover we will also write linear systems of inequalities in the form of the lines (2.7) to (2.9) in examples.

Next we define two linear mappings which we use in order to select "parts" of affine linear functions. Then this selection will be extended to linear systems of inequalities.

The I-th selection chooses the components of a vector in $\mathbb{R}^{(\mathbb{N})}$ specified by the sequence I and builds a new vector with these components at the top followed by zeros. This function is linear obviously. It can be factored into a projection to the subspace $\left\{\breve{b} \mid b_i = 0 \text{ if } i \notin I(\mathbb{N})\right\}$ of $\mathbb{R}^{(\mathbb{N})}$ followed by a rotation of the space $\mathbb{R}^{(\mathbb{N})}$.

We also define this selection on the monoid $\mathbb{L}^{(\mathbb{N})}$ in the same way.

[11] We decide to write A in the form "matrix minus a vector" and not "matrix plus a vector" because we consider this more natural in this context. In this work we are dealing with polyhedra, i.e. the solution sets of inequalities of the form $A\vec{x} \lessdot 0$ which correspond to $\breve{M}\vec{x} - \breve{a} \lessdot 0$. So we know that $\breve{M}\vec{x}$ is bounded by $\breve{a}$ from above.

Definition 2.36 (I-th selection)

1. *The I-th selection σ_I is defined as follows:*

$$\begin{array}{rcll} \sigma_I & : & \mathbb{R}^{(\mathbb{N})} & \rightarrow \mathbb{R}^{(\mathbb{N})} \\ & & \breve{a} & \mapsto \breve{b}, \text{ where } b_i := \begin{cases} a_{I_i} & \text{if } i \in I^{-1}(\mathbb{N}), \\ 0 & \text{else.} \end{cases} \end{array}$$

   If $I : \{1, \ldots, m\} \rightarrow \mathbb{N}$ is a sequence of consecutive integers we denote σ_I by $\sigma_{I_1}^{I_m}$. In the case of $I = (i)$ we leave out the right upper index and write σ_i. We call σ_i the i-th selection function.

2. *In the same way we also define:*

$$\sigma_I : \mathbb{L}^{(\mathbb{N})} \rightarrow \mathbb{L}^{(\mathbb{N})}.$$

□

Example 2.37
Let $I := (2, 4, 5)$. Then

$$\sigma_I \begin{pmatrix} -1 \\ 7 \\ 4 \\ 3 \\ 2 \\ 6 \\ 0 \\ 0 \\ \ldots \end{pmatrix} = \begin{pmatrix} 7 \\ 3 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \ldots \end{pmatrix}.$$

□

The next linear function, the (J, n)-th embedding, maps the space $\mathbb{R}^k$ onto the components of the vectors in $\mathbb{R}^n$ that are given by the sequence J, where $k = |J^{-1}(\{1, \ldots, n\})|$.

Definition 2.38 ((J, n)-th embedding)
Let $J : \{1, \ldots, k\} \rightarrow \{1, \ldots, n\}$, for some $k \leq n$.

The (J, n)-th embedding $\eta_{J,n}$ is defined as follows:

$$\begin{array}{rcll} \eta_{J,n} & : & \mathbb{R}^k & \rightarrow \mathbb{R}^n \\ & & \vec{x} & \mapsto \vec{y}, \text{ where } y_j := \begin{cases} x_{J^{-1}(j)} & \text{if } j \in J(\{1, \ldots, k\}), \\ 0 & \text{else.} \end{cases} \end{array}$$

If J is a sequence of consecutive integers we denote $\eta_{J,n}$ by $\eta_{J_1,n}^{J_k}$. In the case of $J = (j)$ we leave out the right upper index and write $\eta_{j,n}$. We call $\eta_{j,n}$ the

(j,n)-th embedding. If n is clear from the context we will leave it out and we will write η_J, $\eta_{J_1}^{J_k}$ and η_j.

$\square$

Example 2.39
Let $J := (2,4,5)$. Then

$$\eta_{J,6}\begin{pmatrix}2\\5\\-1\end{pmatrix} = \begin{pmatrix}0\\2\\0\\5\\-1\\0\end{pmatrix}.$$

$\square$

In the following definition we apply the selection and embedding functions to an affine linear mapping. This allows us to compute its "coefficients" (cf. definition 2.42) and its "sections" (cf. definition 2.51).

Definition 2.40 (Selection)
We define:

$$A_I := \sigma_I \circ A$$

and let $J : \{1,\ldots,k\} \to \{1,\ldots,n\}$, for some $k \le n$, then[12]

$$_JA := A \circ \eta_{J,n}.$$

If $I : \{1,\ldots,m\} \to \mathbb{N}$ is a sequence of consecutive integers we denote A_I by $A_{I_1}^{I_m}$. In the case of $I = (i)$ we leave out the right upper index and write A_i.

If J is a sequence of consecutive integers we denote $_JA$ by $^{j_k}_{j_1}A$. In the case of $J = (j)$ we leave out the left upper index and write $_jA$.

$\square$

As function composition is associative we have that $_J(A_I) = ({}_JA)_I$. Hence we will leave out the parentheses and will write $_JA_I$.

As σ_I and η_J are both linear we know that any $_JA_I : \mathbb{R}^{|J|} \to \mathbb{R}^{|I|}$ is affine linear. Moreover the definitions of σ_I and η_J imply that $\sigma_I\left(\mathbb{Q}^{(\mathbb{N})}\right) \subseteq \mathbb{Q}^{(\mathbb{N})}$ as well as $\eta_J\left(\mathbb{Q}^k\right) \subseteq \mathbb{Q}^n$. Therefore we have that $_JA_I \in \mathcal{AL}(k)$.

[12]The notation we introduce might look slightly unusual at the first sight because left indices are used by us more often than usual. Nevertheless the reader will see that this notation is very convenient for our purposes after a short phase of getting used to it.

When the affine linear function A is given in "matrix minus vector" representation, i.e. $\vec{M} - \vec{a}$, as part of a linear system then ${}_JA_I$ denotes just the map

$$[M_{ij}]_{i \in I(\mathbb{N}), j \in J(\mathbb{N})} - (z_i)_{i \in I(\mathbb{N})}.$$

Example 2.41
Let

$$A := \begin{bmatrix} 3 & 2 & 1 & 0 \\ -1 & -5 & 7 & 8 \\ 2 & -3 & 4 & 6 \\ -9 & 12 & 3 & 4 \end{bmatrix} - \begin{pmatrix} 1 \\ 4 \\ 3 \\ 2 \end{pmatrix}$$

and

$$I := (2, 3), J := (1, 4).$$

Then

$${}_JA_I = \begin{bmatrix} -1 & 8 \\ 2 & 6 \end{bmatrix} - \begin{pmatrix} 4 \\ 3 \end{pmatrix}.$$

□

Definition 2.42 (*ij*-th coefficient)
Let $\breve{b} := {}_j(A - A(0))(1)$ *then*

$$A_{ij} := b_i$$

be the ij*-th coefficient of* A *and let* $\breve{a} := -A(0)$ *then*

$$A_{i\infty} := a_i$$

the $i\infty$*-th coefficient of* A*.*

□

Again in "matrix minus vector" representation the above definition means that $A_{ij} = M_{ij}$ and $A_{i\infty} = a_i$, for $i, j \leq \|S\|$.

Lemma 2.43
For all i, j *we have*

$$A_{ij}, A_{i\infty} \in \mathbb{Q}.$$

□

Proof:

We use the notation of definition 2.42.

As ${}_j(A - A(0)) \in \mathcal{AL}(1)$ and $-A \in \mathcal{AL}(n)$ we conclude that $\breve{b}, \breve{a} \in \mathbb{Q}^{(\mathbb{N})}$.

□

The last lemma tells us that an affine linear function in $\mathcal{AL}(n)$ can be represented by a matrix with rational elements minus a vector with rational components.

Now we extend this selection to linear systems.

Definition 2.44 (Selection of parts of linear systems)

1. $(A, \lessdot)_I := (A_I, \sigma_I \lessdot)$,
2. ${}_J(A, \lessdot) := ({}_J A, \lessdot)$,
3. $(A, \lessdot)_{ij} := A_{ij}$,
4. $(A, \lessdot)_{i\infty} := A_{i\infty}$.

□

Obviously this extension is well defined.

Now after being able to select some parts of linear systems we want to combine them again. If one thinks of a system of inequalities as a sequence, i.e. a list in the words of a computer scientist, of single inequalities the natural combination operation would be appending of two lists. The set of all lists together with the appending operation forms a monoid. Thus the following definitions are designed in a way such that we arrive at this monoid in our framework.

Definition 2.45 (Shifting function)

1. *The shifting function θ is defined as follows:*

$$\begin{array}{rcccl} \theta & : & \mathbb{R}^{(\mathbb{N})} & \to & \mathbb{R}^{(\mathbb{N})} \\ & & \breve{a} & \mapsto & \breve{b} \text{ where } b_1 := 0 \text{ and } b_i := a_{i-1}, \text{ for } i \geq 2. \end{array}$$

2. *In the same way we also define*

$$\theta : \mathbb{L}^{(\mathbb{N})} \to \mathbb{L}^{(\mathbb{N})}.$$

3. *We extend θ to $\mathcal{LS}(n)$ in the following way:*

$$\theta(A, \lessdot) := (\theta \circ A, \theta(\lessdot)).$$

□

Example 2.46

$$\theta \begin{pmatrix} 2 \\ 7 \\ 6 \\ 0 \\ 0 \\ 0 \\ \dots \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 7 \\ 6 \\ 0 \\ 0 \\ \dots \end{pmatrix}.$$

□

The shifting function puts a zero at the top of a vector in $\mathbb{R}^{(\mathbb{N})}$. This corresponds to reserving some space at the top of the second of the two lists of inequalities that we want to append. Having as much space available as the length of the first list we can copy this list there. We do this in the next definition.

Definition 2.47 (Union of linear systems)
We define the union $S \uplus T$ of two linear systems S, T in the following way:

$$S + \theta^{\|S\|} T.$$

Let $S^u \in \mathcal{LS}(n)$, for $u \in \{v, \dots, w\}$ and some $v, w \in \mathbb{Z}$. We also define:

$$\biguplus_{u=v}^{w} S^u := \begin{cases} 0 & \text{if } w < v, \\ S^u \uplus \left(\biguplus_{u=v+1}^{w} S^u \right) & \text{else } (v \leq w). \end{cases}$$

□

Corollary 2.48
The union $\uplus$ is well defined.

□

Proof:

Obvious.

□

Now the length of the resulting list must be exactly the sum of the lengths of the two original lists. Hence we have the following corollary.

Corollary 2.49

$$\|S \uplus T\| = \|S\| + \|T\|.$$

□

Proof:

In the case $T = 0$ nothing needs to be shown. Therefore assume $T \neq 0$. Obviously the corollary is implied by the following two equalities.

$$\begin{aligned} \left\| A + \theta^k \circ B \right\| &= k + \|B\|, \text{ if } \|A\| \leq k, \\ \left\| \lessdot \cdot \theta^k (\lhd) \right\| &= k + \|\lhd\|, \text{ if } \|\lessdot\| \leq k, \end{aligned}$$

where $B \neq 0$ and $\lhd \neq \leq$. The first equality can easily be proven by using the representation of $\|.\|$ derived in the proof of corollary 2.35. To fill in the details is left to the reader.

The second equality is obvious.

□

The next corollary finishes the construction of the monoid of linear systems of inequalities.

Corollary 2.50

1. *The binary operator* $\uplus : \mathcal{LS}(n) \times \mathcal{LS}(n) \to \mathcal{LS}(n)$ *is associative.*
2. $S \uplus 0 = 0 \uplus S = S$.
3. $(\mathcal{LS}(n), \uplus)$ *is a monoid whose zero is* 0.

□

Proof:

We prove point 1. The other ones are obvious given point 1.

We have

$$\begin{aligned} (S \uplus T) \uplus U &= (S \uplus T) + \theta^{\|S \uplus T\|} U \\ &= S + \theta^{\|S\|} T + \theta^{\|S\| + \|T\|} U \\ &= S + \theta^{\|S\|} \left(T + \theta^{\|T\|} U \right) \\ &= S + \theta^{\|S\|} (T \uplus U) \\ &= S \uplus (T \uplus U) \end{aligned}$$

□

Given the operator $\uplus$ we can formalise the fact that a linear system of inequalities is composed of a sequence of single inequalities. Namely, we have

$$S = \biguplus_{i=1}^{\|S\|} S_i.$$

The S_i correspond to the inequalities that form a system of linear inequalities given as in example 2.29.

Moreover note that

$$\mathcal{P}\left(S \uplus T\right) = \mathcal{P}\left(S\right) \cap \mathcal{P}\left(T\right).$$

Note that this implies that the monoid $\left(\mathcal{P}\left(\mathcal{LS}\left(n\right)\right), \cap\right)$ is a homomorphic image of the monoid $\left(\mathcal{LS}\left(n\right), \uplus\right)$.

This fact and the previous equation imply that $\mathcal{P}\left(S\right) \in \mathcal{B}\left(\mathbb{R}^n\right)$, for all S.

The homomorphism $\mathcal{P}$ divides the set $\mathcal{LS}\left(n\right)$ in classes whose members have the same solution set. For instance, this mapping is invariant under reordering and repetition of inequalities. More precisely we have

$$\mathcal{P}\left(S\right) = \mathcal{P}\left(\biguplus_{i=1}^{m} S_{f(i)}\right),$$

where $f : \{1, \ldots, m\} \to \{1, \ldots, \|S\|\}$ is onto. This means that duplicated inequalities are redundant (cf. definition 2.53).

Next we define how to denote "sections" of linear systems of inequalities informally spoken. This means, we want to describe the appearance of the corresponding affine linear map and the resulting linear system when restricted to a hyperplane

$$\{\vec{x} \in \mathbb{R}^n \,|\, x_{\bar{J}(j)} = y_j, \text{ for } j = 1, \ldots, n-k\},$$

for given $J : \{1, \ldots, k\} \to \{1, \ldots, n\}$ and y_j. Hence we state the following definition.

Definition 2.51 ((J, n)-th section)
Let $J : \{1, \ldots, k\} \to \{1, \ldots, n\}$, for some $k \leq n$.

The (J, n)-th section $\mathcal{F}_{J,n}$ is defined as follows:

$$\begin{array}{rrcl} \mathcal{F}_{J,n} & : \mathcal{LS}\left(n\right) & \to & \left(\mathbb{Q}^{n-k} \to \mathcal{LS}\left(k\right)\right) \\ & \left(A, \lessdot\right) & \mapsto & \left(\vec{y} \mapsto \left({}_{J}A + {}_{\bar{J}}A\vec{y} + A\left(0\right), \lessdot\right)\right). \end{array}$$

If J is a sequence of consecutive integers we denote $\mathcal{F}_{J,n}$ by $\mathcal{F}_{J_1,n}^{J_k}$. In the case of $J = (j)$ we leave out the right upper index and write $\mathcal{F}_{j,n}$. We call $\mathcal{F}_{j,n}$ the

(j,n)-th section. If the value of n is clear from the context we will leave it out and we will write $\mathcal{F}_J, \mathcal{F}_{J_1}^{J_k}$ and $\mathcal{F}_j$ and especially say the j-th section in the case of $\mathcal{F}_j$.

□

We illustrate the above definition in the following example.

Example 2.52 (Ad example 2.29)
The second section of S is

$$\mathcal{F}_2 S(\vec{y}) = \left(\begin{bmatrix} \frac{2}{3} \\ \frac{4}{9} \\ 1 \end{bmatrix} - \left(- \begin{bmatrix} \frac{1}{2} \\ -7 \\ 0 \end{bmatrix} \vec{y} + \begin{pmatrix} 4 \\ \frac{11}{7} \\ 2 \end{pmatrix} \right), \begin{matrix} < \\ \leq \\ < \end{matrix} \right).$$

□

Sometimes a linear system might contain too many inequalities. That means if we remove a specific inequality the solution set of the resulting system remains the same as the original system. We call such inequalities redundant.

Definition 2.53 (Redundant)
We say T is redundant for the linear system S iff[13]

$$\mathcal{P}(T \uplus S) = \mathcal{P}(S).$$

Equivalently if $S = S^1 \uplus S^2$ we also say T is redundant in[14] $S^1 \uplus T \uplus S^2$.

□

In other words the above definition means

$$\vec{x} \in \mathcal{P}(S) \text{ implies } \vec{x} \in \mathcal{P}(T).$$

Example 2.54 (Ad example 2.29)
The inequality

$$-7x_1 + \frac{4}{9}x_2 \leq \frac{201}{10}$$

is redundant for S.

---

[13]We should rather define

$$T \text{ is redundant for } S \text{ iff } \mu(\mathcal{P}(T \uplus S) \triangle \mathcal{P}(S)) = 0.$$

In the case of $\mu = \gamma$ this condition is implied by the above one.

We don't use the latter characterisation of redundancy because we will only detect the above stricter version in chapter 5, Improvements.

[14]Note that $\mathcal{P}(T \uplus S) = \mathcal{P}\left(S^1 \uplus T \uplus S^2\right)$.

□

Sometimes we will also encounter linear systems of inequalities having the empty set as solution set. We will call them inconsistent. Hence the following definition.

Definition 2.55 (Inconsistent)
We call S inconsistent iff[15]

$$\mathcal{P}(S) = \emptyset,$$

whereas we call it consistent iff

$$\mathcal{P}(S) \neq \emptyset,$$

□

2.5 Projections of Polyhedra

In chapters 3, Abstract Treatment of the Problem, and 4, State of the Art, we will have to project a polyhedron $\mathcal{P}(S) \subseteq \mathbb{R}^n$ to the lower dimensional subspace $\{\vec{x} \in \mathbb{R}^n \,|\, x_j = 0, \text{ if } j \notin J(\{1,\dots,k\})\}$, where $J : \{1,\dots,k\} \to \{1,\dots,n\}$. This subspace is isomorphic to $\mathbb{R}^k$.

Therefore we define a linear function $\pi_{J,n}$ that performs this projection and embeds the result into the space $\mathbb{R}^k$. Because of this isomorphism we call this mapping the (J,n)-th projection although one might argue that this is a slight abuse of language.

Definition 2.56 ((J,n)-th projection)
Let $J : \{1,\dots,k\} \to \{1,\dots,n\}$, for some $k \leq n$.

The (J,n)-th projection $\pi_{J,n}$ is defined as follows:

$$\begin{array}{rccl} \pi_{J,n} & : & \mathbb{R}^n & \to & \mathbb{R}^k \\ & & \vec{x} & \mapsto & \vec{y}, \text{ where } y_j := x_{J_j}, \text{ for } j \in \{1,\dots,k\}. \end{array}$$

If J is a sequence of consecutive integers we denote $\pi_{J,n}$ by $\pi_{J_1,n}^{J_k}$. In the case of $J = (j)$ we leave out the right upper index and write $\pi_{j,n}$. We call $\pi_{j,n}$ the (j,n)-th projection. If n is clear from the context we will leave it out and we will write $\pi_J, \pi_{J_1}^{J_k}$ and π_j.

[15] A more general notion of inconsistency would be

$$\mu(\mathcal{P}(S)) = 0.$$

But we don't use it like the version of redundancy involving μ.

We also define for linear systems of inequalities in $\mathcal{LS}(k)$

$$(B, <) \circ \pi_{J,n} := (B \circ \pi_{J,n}, <).$$

□

Such a projection is described in the right hand side of the following commutative diagram. Note that we are only interested in a set that differs from the projection by a set of measure zero w.r.t. the measure μ (cf. section 3.2, Abstract Solution)!

We will do this by projecting the original polyhedron $\mathcal{P}(S)$ in principle. Such a projection is a polyhedron in turn (cf. theorem 2.62). This is indicated in the left hand side of the diagram. But there is no need for computing this projection from $\mathbb{R}^n$ to $\mathbb{R}^k$ exactly because we only need an approximate polyhedron that differs from the exact projection by a set of measure zero. Therefore we can arbitrarily choose any polyhedron, i.e. a linear system of inequalities defining this polyhedron, with this property. The set of all such linear systems of inequalities is denoted by ${}_{\mu}\mathrm{Proj}_J(S)$. This is shown in the right hand side of the diagram.

$$\begin{tikzcd}
\mathcal{P}(S) \arrow[r, equal] \arrow[d] & \mathcal{P}(S) \arrow[d, "T \in {}_{\mu}\mathrm{Proj}_J(S)"] \\
\pi_J(\mathcal{P}(S)) \arrow[r, "\neq"] \arrow[d] & \mathcal{P}(T) \arrow[d] \\
\mu(\pi_J(\mathcal{P}(S))) \arrow[r, equal] & \mu(\mathcal{P}(T))
\end{tikzcd}$$

The above is made precise in the following definition.

Definition 2.57 (Approximate projections)
Let $J : \{1, \ldots, k\} \to \{1, \ldots, n\}$, for some $k \leq n$.

$${}_{\mu}\mathrm{Proj}_{J,n}(S) := \{T \in \mathcal{LS}(k) \mid \gamma(\pi_J(\mathcal{P}(S)) \bigtriangleup \mathcal{P}(T)) = 0\}.$$

If J is a sequence of consecutive integers we denote ${}_{\mu}\mathrm{Proj}_{J,n}$ by ${}_{\mu}\mathrm{Proj}^{J_k}_{J_1,n}$. In the case of $J = (j)$ we leave out the right upper index and write ${}_{\mu}\mathrm{Proj}_{j,n}$. We call ${}_{\mu}\mathrm{Proj}_{j,n}$ the (j,n)-th projection. If n is clear from the context we will leave it out and we will write ${}_{\mu}\mathrm{Proj}_J$, ${}_{\mu}\mathrm{Proj}^{J_k}_{J_1}$ and ${}_{\mu}\mathrm{Proj}_j$.

□

Later we will use the obvious equality

$$\pi_{i+1}^{n} = \pi_{2}^{n-i+1} \circ \pi_{i}^{n} \tag{2.11}$$

that allows us to reduce the projection along a certain multi dimensional subspace to a projection along a subspace of lower dimension than the original one and along a one dimensional subspace.

In the following we will state a well known method in order to compute one representative out of the set ${}_{\mu}\mathrm{Proj}_{\overline{(k)}}(S)$ that yields an exact projection of the corresponding polyhedron. This scheme is called the Fourier-Motzkin projection. First we give a motivation for the following definitions.

Given a linear system of inequalities we can find certain restrictions on any specific component of vectors in the domain of an affine linear function. These restrictions are conditions on the range in which this component, say the k-th one, lie, namely

1. strict/non strict lower bounds,
2. strict/non strict upper bounds and
3. inequalities not containing this component, that is, the corresponding affine linear functions are constant in this component.

We want to remark that inequalities not containing a certain component must also be regarded as restriction on this component in some sense. Namely, it is possible that we find a vector whose k-th component lies between its lower and upper bounds but whose other components don't satisfy the remaining inequalities that do not contain this component.

We illustrate this in the following example.

Example 2.58 (Ad example 2.29)

1. Inequality (2.8) is equivalent to the fact that x_1 has the strict lower bound
$$\frac{4}{63}x_2 - \frac{11}{49}.$$
2. Inequality (2.7) is equivalent to the fact that x_1 has the non strict upper bound
$$-\frac{4}{3}x_2 - (-8)\,.$$
3. Inequality (2.9) doesn't contain x_1:
$$x_2 < 2. \tag{2.12}$$

Now let $x_1 := 0, x_2 := 3$. Obviously x_1 lies between its lower and upper bounds but x_2 doesn't satisfy inequality (2.12).

□

As we can see in the above example the upper and the lower bound are again affine linear functions.

In more general systems as in the above one one single component usually has more than one lower and upper bound. We combine all these bounds into one affine linear function A for each type of bound. In order to remember which bound is strict or non strict we attach an element $\lessdot \in \mathbb{L}^{(\mathbb{N})}$ to this function and represent such bounds by pairs $(A, \lessdot) \in \mathcal{LS}(n)$, where $\lessdot_i = \leq$ means that A_i is a non strict bound and $\lessdot_i = <$ means that A_i is a strict bound.

Moreover we will also combine the restrictions not containing this component into one system. This can be seen in the next definitions.

Definition 2.59 (Diagonal function)
Let $\breve{c} \in \mathbb{R}^{(\mathbb{N})}$. We call

$$\begin{array}{rcccl} \delta_{\breve{c}} & : & \mathbb{R}^{(\mathbb{N})} & \to & \mathbb{R}^{(\mathbb{N})} \\ & & \breve{a} & \mapsto & \breve{b}, \ \textit{where } b_i := c_i a_i, \ \textit{for } i \in \mathbb{N} \end{array}$$

a diagonal function.

We extend $\delta_{\breve{c}}$ to linear systems in the following way.

$$\delta_{\breve{c}}(A, \lessdot) := (\delta_{\breve{c}} \circ A, \lessdot).$$

□

Obviously any diagonal function is linear.

Definition 2.60 (Restrictions)
Let $l \leq n$, $(l) : \{1\} \to \{1, \ldots, n\}$ be a finite sequence of natural numbers and $L := \overline{(l)}$. Apart from this

1. *let $I : \{1, \ldots, k\} \to \mathbb{N}$ such that*

$$I(\{1, \ldots, k\}) = \{i \mid S_{il} < 0\},$$

for some k. Further let

$$b_i := \begin{cases} \frac{1}{S_{I_i l}} & \textit{if } i \leq k, \\ 0 & \textit{else.} \end{cases}$$

Then

$$\begin{array}{rcccl} \mathcal{LOW}_{l,n} & : & \mathcal{LS}(n) & \to & \mathcal{LS}(n-1) \\ & & S & \mapsto & -\delta_{\breve{b}}({}_L S_I). \end{array}$$

is called the lower bounds of the l-th component.

2. *let* $J : \{1, \ldots, k\} \to \mathbb{N}$ *such that*

$$J(\{1, \ldots, k\}) = \{j \mid S_{jl} > 0\},$$

for some k. *Further let*

$$c_i := \begin{cases} \frac{1}{S_{J_j l}} & \text{if } j \leq k, \\ 0 & \text{else.} \end{cases}$$

Then

$$\begin{array}{rcrcl} \mathcal{UPP}_{l,n} & : & \mathcal{LS}(n) & \to & \mathcal{LS}(n-1) \\ & & S & \mapsto & -\delta_{\breve{c}}({}_L S_J). \end{array}$$

is called the upper bounds of the l*-th component.*

3. *let* $K : \{1, \ldots, k\} \to \{1, \ldots, \|S\|\}$ *such that*

$$K(\{1, \ldots, k\}) = \{i \mid S_{il} = 0\},$$

for some K. *Then*

$$\begin{array}{rcrcl} \mathcal{NOT}_{l,n} & : & \mathcal{LS}(n) & \to & \mathcal{LS}(n-1) \\ & & S & \mapsto & {}_L S_K. \end{array}$$

is called the restriction not containing the l*-th component.*

If n *is clear from the context we will abbreviate* $\mathcal{LOW}_{l,n}, \mathcal{UPP}_{l,n}$ *and* $\mathcal{NOT}_{l,n}$ *by* $\mathcal{LOW}_l, \mathcal{UPP}_l$ *and* $\mathcal{NOT}_l$.

□

As the above sequences I, J, K in the definitions of $\mathcal{LOW}_{l,n}, \mathcal{UPP}_{l,n}$ and $\mathcal{NOT}_{l,n}$ are unique obviously the functions $\mathcal{LOW}_{l,n}, \mathcal{UPP}_{l,n}$ and $\mathcal{NOT}_{l,n}$ are well defined.

In the Fourier-Motzkin projection we will have to compare some lower bound $(A, \lessdot)$ with some upper bound $(B, \lhd)$. That means we want to investigate the set of vectors $\vec{x}$ with the property $A\vec{x} \blacktriangleleft B\vec{x}$ or, equivalently, $(A - B)\vec{x} \blacktriangleleft 0$, where $\blacktriangleleft := \lessdot \cdot \lhd$. Sometimes we will also have to form strict and non-strict inequalities out of such bounds independently from the types of these bounds.

Therefore we introduce a suggestive notation for these comparisons (cf. chapter 4, State of the Art).

Definition 2.61 (Comparison of bounds)

1. *Let*

$$(A, \lessdot) \propto (B, \lhd) := (A - B, \lessdot \cdot \lhd),$$

2. *Let*

$$(A, \lessdot) \preceq (B, \lhd) := (A - B, \leq),$$

3. *and*

$$(A, \lessdot) \prec (B, \lhd) := (A - B, \blacktriangleleft),$$

where $\blacktriangleleft_i := \begin{cases} < & \text{if } i \leq \max\{\|A\|, \|B\|\}, \\ \leq & \text{else.} \end{cases}$

4. *We arrange that*

$$R \uplus S \propto T := R \uplus (S \propto T),$$
$$R \propto S \uplus T := (R \propto S) \uplus T,$$

5.

$$R \uplus S \preceq T := R \uplus (S \preceq T),$$
$$R \preceq S \uplus T := (R \preceq S) \uplus T,$$

6. *and*

$$R \uplus S \prec T := R \uplus (S \prec T),$$
$$R \prec S \uplus T := (R \prec S) \uplus T.$$

□

Now we can formulate the Fourier-Motzkin projection.

Theorem 2.62 (Fourier-Motzkin projection [Zie94]**)**
Let

$$S := \mathcal{LOW}_k(R),$$
$$T := \mathcal{UPP}_k(R),$$
$$U := \mathcal{NOT}_k(R) \uplus \biguplus_{i=1}^{\|S\|} \biguplus_{j=1}^{\|T\|} S_i \propto T_j.$$

Then we have[16]

$$\mathcal{P}(U) = \pi_{\overline{(k)}}(\mathcal{P}(R)). \tag{2.13}$$

[16]We use a notation and formulation that is quite different to Ziegler's (cf. [Zie94]). The main difference is the following:

1. We allow strict inequalities, too.

In particular it is valid that

$$U \in {}_{\mu}\mathrm{Proj}_{\overline{(k)}}(R)\,. \tag{2.14}$$

□

Example 2.63 (Ad example 2.58)
Here U from above is given by

$$\begin{aligned} \frac{4}{63}x_2 - \frac{11}{49} &< -\frac{4}{3}x_2 + 8, \\ x_2 &< 2, \end{aligned}$$

i.e. by

$$\begin{aligned} \frac{88}{63}x_2 - \frac{403}{49} &< 0, \\ x_2 &< 2. \end{aligned}$$

□

2.6 Flat Polyhedra

Later in sections 4, State of the Art, and 5, Improvements, we will encounter so-called “flat polyhedra”. These are polyhedra with open faces that lie within some affine hyperplane. They can be treated differently to arbitrary ones in the course of volume computation[17]:

- In the case of the real volume we simply know that their volumes are zero. This leads to a short cut in the computation of the volume, cf. section 5, Improvements.

2. Ziegler computes $R_{I_i k} R_{J_j k} T_j \preceq R_{J_j k} R_{I_i k} S_i$. Whereas we compute $S_i \preceq T_j$, where I, J are as in definition 2.60. The latter doesn't yield the same affine linear function but a linear system of inequalities having the same solution set.

   We use this formulation because Haghighat and Polychronopoulos (cf. [HP94]), Tawbi (cf. [Taw95]) whose solutions of the problem “Number of integer solutions” are described in chapter 4, State of the Art, use this kind of projection.

Although this kind of projection is a bit more general than Ziegler's the verification of line (2.13) proceeds as it is done in [Zie94].

[17]We want to emphasise that the different treatment of flat polyhedra distinguishes the volume computations w.r.t. different measures in this work. This occurs because it can happen that flat polyhedra might be of non-zero integer measure even though they always are zero sets w.r.t. Lebesgue measure.

- In the case of the integer volume we can eliminate some variables without integrating, cf. sections 4, State of the Art, and 5, Improvements.

Let's start with flat polyhedra.

Definition 2.64 (Flat polyhedron)
The polyhedron $\mathcal{P}(R) \subseteq \mathbb{R}^n$ is called a flat polyhedron[18] *(embedded in $\mathcal{P}(T)$) iff*

$$\mathcal{P}(R) = \pi_{\overline{(1,\ldots,m)}} \left(\mathcal{P}\left(R \circ \pi_{\overline{(1,\ldots,m)}} \right) \cap \mathcal{P}(T) \right),$$

where

$$T := (A, \leq) \uplus (-A, \leq) \in \mathcal{LS}(n+m)$$

and

$$\mathcal{P}(T) \neq \mathbb{R}^{n+m}.$$

□

Example 2.65
Let R be given by the following inequalities.

$$\left(\begin{bmatrix} 5 & 3 \\ -6 & 7 \end{bmatrix} - \begin{pmatrix} -1 \\ 2 \end{pmatrix}, \leq \right). \tag{2.15}$$

Further let T be given by

$$\left(\begin{bmatrix} 1 & 2 & 3 \\ -1 & -2 & -3 \end{bmatrix} - \begin{pmatrix} 3 \\ -3 \end{pmatrix}, \leq \right). \tag{2.16}$$

and $m := 1$. Then we have $\mathcal{P}(R) = \pi_{(2,3)} \left(\mathcal{P}\left(R \circ \pi_{(2,3)} \right) \cap \mathcal{P}(T) \right)$, i.e. $\mathcal{P}(R)$ is a flat polyhedron embedded in $\mathcal{P}(T)$.

□

In other words this definition says that $\mathcal{P}(R)$ can be embedded into an affine hyperplane $\mathcal{P}(T)$. For $m > 0$ this is nothing interesting geometrically. We can always find T as in definition 2.64. But if $m = 0$ we know that $\lambda(\mathcal{P}(R)) = 0$.

When we will compute the integer volume we will be forced to compute it of

$$\pi_{\overline{(1,\ldots,m)}} \left(\mathcal{Z}\left(R \circ \pi_{\overline{(1,\ldots,m)}} \right) \cap \mathcal{Z}(T) \right)$$

[18] Pugh calls such polyhedra "clauses in projected form" in [Pug94]. He briefly sketches a method for dealing with these clauses. It is based on Smith normal form computations. Alas, we don't see a way how to apply this method under the general circumstances occurring later. Therefore we present an alternative way of dealing with flat polyhedra.

Moreover we only compute matrices in Hermite normal form, i.e. triangular ones, instead of ones in Smith normal form that are diagonal matrices.

instead of

$$\pi_{\overline{(1,\ldots,m)}}\left(\mathcal{P}\left(R \circ \pi_{\overline{(1,\ldots,m)}}\right) \cap \mathcal{P}(T)\right)$$

for some flat polyhedron R. Observe that this isn't the same! Therefore we also allow $m > 0$ in the definition of a flat polyhedron.

Moreover we will need an alternative way of representing the set

$$\pi_{\overline{(1,\ldots,m)}}\left(\mathcal{Z}\left(R \circ \pi_{\overline{(1,\ldots,m)}}\right) \cap \mathcal{Z}(T)\right)$$

without the usage of the projection function $\pi_{\overline{(1,\ldots,m)}}$ and the system T later. This representation will be developed on the following pages. For the rest of this section let R, T, A be as in definition 2.64. As we will apply algorithms of number theory to A we assume w.l.o.g. that $A_{ij} \in \mathbb{Z}$, for all $i, j \in \mathbb{N}_0 \cup \{\infty\}$.

First we will compute a parametrisation of

$$\mathcal{Z}(T)$$

by an affine linear function

$$F : \mathbb{R}^l \to \mathbb{R}^{n+m}, \text{ for some } l,$$

where

$$\mathcal{Z}(T) = F\left(\mathbb{Z}^l\right).$$

Moreover the matrix describing this function has a certain upper triangular shape and only integer elements.

Such non quadratic upper triangular matrices with integer coefficients are also said to be in Hermite normal form.

Definition 2.66 (Hermite normal form [Coh93])
Let $\vec{M}$ be an integer matrix with m rows and n columns. If

1. $M_{(j+m-\min(m,n))\,j} \geq 1$,
2. $M_{ij} = 0$, *if* $i > j + m - \min(m,n)$, *and*
3. $0 \leq M_{(k+m-\min(m,n))\,j} < M_{(k+m-\min(m,n))\,k}$, *if* $k < j$,

then $\vec{M}$ is in Hermite normal form[19].

□

[19] This is a special case of the general definition found in [Coh93].

Example 2.67
The matrix

$$\begin{bmatrix} 11 & 12 & 13 & 14 & 15 \\ 15 & 16 & 17 & 18 & 19 \\ 5 & 4 & 3 & 2 & 1 \\ 0 & 7 & 6 & 5 & 4 \\ 0 & 0 & 10 & 9 & 8 \\ 0 & 0 & 0 & 11 & 10 \\ 0 & 0 & 0 & 0 & 11 \end{bmatrix}$$

is in Hermite normal form. So is

$$\begin{bmatrix} 0 & 0 & 10 & 9 & 8 \\ 0 & 0 & 0 & 11 & 10 \\ 0 & 0 & 0 & 0 & 11 \end{bmatrix}.$$

□

Hermite normal forms can always be computed. This states the following theorem.

Theorem 2.68 ([Coh93])
Let $\vec{N}$ be an integer matrix with l columns. Then there exists a unique integer matrix $\vec{M}$ with l columns in Hermite normal form such that

$$\vec{M} = \vec{N}\vec{U},$$

where $\vec{U}$ is an $l \times l$ integer matrix whose determinant equals ± 1.

□

Definition 2.69
Let $\vec{M}$ be as in theorem 2.68. Then the matrix formed by the non zero columns of $\vec{M}$ is called the Hermite normal form of $\vec{N}$.

□

Note that *the* Hermite normal form of $\vec{N}$ is *in* Hermite normal form.

Cohen proves this theorem by formulating several algorithms. They can be summarised in algorithm 2.70, HermiteNormalForm. In order to state[20] this algorithm we need the following four subroutines:

[20] Our statement of the algorithm will be slightly different to Cohen's one because we don't want to use conditional jumps.

$$\begin{aligned}
\mathrm{Col}(\vec{M}, i)\ &, \text{equals the } i\text{-th column vector of } \vec{M}, \\
\mathrm{SetCol}(\vec{M}, i, \vec{c})\ &, \text{where the } i\text{-th column of } \vec{M} \text{ gets assigned} \\
& \text{the vector } \vec{c}, \\
\mathrm{ColOp}(\vec{M}, u, i, v, j, \vec{U})\ &, \text{which is equivalent to} \\
& \mathrm{SetCol}\Big(\vec{M}, i, u\,\mathrm{Col}(\vec{M}, i)\ - v\,\mathrm{Col}(\vec{M}, j)\ \Big) \\
& \text{and} \\
& \mathrm{SetCol}\Big(\vec{U}, i, u\,\mathrm{Col}(\vec{U}, i)\ - v\,\mathrm{Col}(\vec{U}, j)\ \Big)\ , \\
\mathrm{GCDEx}(u, v, w, a, b)\ &, \text{where } w \text{ is the greatest common divisor of } u \text{ and } v, \\
& \text{whereas } a \text{ and } b \text{ are the corresponding cofactors,}
\end{aligned}$$

where $\vec{M}, \vec{U}$ are integer matrices, $\vec{c}$ is an integer vector and a, b, u, v, w are integers.

Algorithm 2.70
$(\vec{U}, \vec{W}, \vec{M}) \leftarrow \mathrm{HermiteNormalForm}(\vec{A})$

Input:
$\vec{A}$ .... *an integer matrix having m rows and n columns.*

Output:
$\vec{U}$ .... *an integer matrix such that $\vec{A}\vec{U}$ is in Hermite normal form.*
$\vec{W}$ ...... *the Hermite normal form of $\vec{A}$.*
$\vec{M}$ ...... *an integer matrix whose columns form a basis of the kernel of $\vec{A}$.*

[1] Initialise.
$i \leftarrow m$
$j \leftarrow n$
$k \leftarrow n$
$\vec{U} \leftarrow$ *the n dimensional identity square matrix*
If $m \leq n$ **then**
  $l \leftarrow 1$
else
  $l \leftarrow m - n + 1$

[2] Main iteration.
Do

[2.1] Check zero.

If $j \neq 1$ **then**

$j \leftarrow j - 1$

If $A_{ij} \neq 0$ **then**

[2.1.1.1] Euclidean step.

$\mathrm{GCDEx}(A_{ik}, A_{ij}, d, u, v)$

$\vec{a} \leftarrow u\,\mathrm{Col}(\vec{A}, k) + v\,\mathrm{Col}(\vec{A}, j)$

$\vec{u} \leftarrow u\,\mathrm{Col}(\vec{U}, k) + v\,\mathrm{Col}(\vec{U}, j)$

$\mathrm{ColOp}(\vec{A}, \frac{A_{ik}}{d}, j, \frac{A_{ij}}{d}, k, \vec{U})$

$\mathrm{SetCol}(\vec{A}, k, \vec{a})$

$\mathrm{SetCol}(\vec{U}, k, \vec{u})$

else

[2.1.2] Final reductions.

$b \leftarrow A_{ik}$

If $b < 0$ **then**

$\mathrm{SetCol}\Big(\vec{A}, k, -\mathrm{Col}(\vec{A}, k)\Big)$

$\mathrm{SetCol}\Big(\vec{U}, k, -\mathrm{Col}(\vec{U}, k)\Big)$

$b \leftarrow -b$

else

If $b = 0$ **then**

$k \leftarrow k + 1$

else

For $j = k + 1$**,** n

$q \leftarrow \left\lfloor \frac{A_{ij}}{b} \right\rfloor$

$\mathrm{ColOp}(\vec{A}, 1, j, q, k, \vec{U})$

[2.1.3] Not yet finished.

$i \leftarrow i - 1$

$k \leftarrow k - 1$

$j \leftarrow k$

while $i + 1 \neq l$

[3] Finished.

For $j = 1, \quad k$
 SetCol$\Big(\vec{M}, j, \mathrm{Col}(\vec{U}, j)\ \Big)$
For $j = 1, \quad n-k$
 SetCol$\Big(\vec{W}, j, \mathrm{Col}(\vec{A}, j+k)\ \Big)$

□

The Hermite normal form algorithm will now be used in order to compute the parametrisation of

$$\mathcal{Z}\,(T)\,.$$

Using the notation of definition 2.64 let $\vec{N} := (A_{ij})_{\substack{i=1,\dots,n+m,\\ j=1,\dots,\|A\|}}$ and $a_i := A\,(0)_i$, for $i \leq \|A\|$. Then

$$\mathcal{Z}\,(T) = \Big\{\vec{x} \in \mathbb{Z}^{n+m} \mid \vec{N}\vec{x} = \vec{a}\Big\}\,. \tag{2.17}$$

We compute

$$\Big(\vec{U^1}, \vec{W^1}, \vec{H^1}\Big) \leftarrow \mathrm{HermiteNormalForm}(\vec{N})\ .$$

Then we execute

$$\Big(\vec{U^2}, \vec{W^2}, \vec{H}\Big) \leftarrow \mathrm{HermiteNormalForm}\Big(\vec{H^1}\Big)\ .$$

Note that $\vec{H}$ is a representation of the kernel of $\vec{N}$, too, because

$$\vec{H} = \vec{H^1}\vec{U^2}.$$

The latter equality is valid because $\vec{H^1}$ has a trivial kernel. This also means that the number of its rows is greater or equal than the number of its columns. Moreover $\vec{U^2}$ induces a bijection from $\mathbb{Z}^l$ to $\mathbb{Z}^l$ because its determinant is ± 1 (cf. [Coh93]).

In order to describe the integer points of the affine hyperplane $\mathcal{P}\,(T)$ we need to compute one vector $\vec{x}$ satisfying $\vec{N}\vec{x} = \vec{a}$ additionally (compare [Bun88], [PS95]). If $\mathcal{Z}\,(T) \neq \emptyset$ this can always be done (cf. [Bun88], [PS95]) and it can always be decided if there exists such an $\vec{x}$. As the theory of solving systems of Diophantine equations is a traditional branch of number theory there exist several methods for doing this (cf. [Bun88], [PS95],[GK84]). Here we describe one possibility by using Hermite normal forms.

First we solve

$$\vec{N}\vec{U^1}\vec{b} = \vec{a}$$

for $\vec{b}$ over the integers. This can be done easily because

$$\vec{N}\vec{U}^1 = \left[\vec{0} \;\; \vec{W}^1\right],$$

where $\vec{0}$ is a zero matrix having as many columns as $\vec{H}^1$. As $\vec{W}^1$ is in Hermite normal form we can compute each component of the vector $\vec{b}$ one after the other. Moreover note that we might not be able to compute an *integer* vector $\vec{b}$ if some pivot element at the diagonal of $\vec{W}$ doesn't divide the right hand side of the above equation. Then $\mathcal{Z}(T) = \emptyset$.

Given $\vec{b}$ we know that

$$\vec{x} = \vec{U}^1\vec{b}.$$

Therefore we have

$$\mathcal{Z}(T) = \left\{\vec{H}\vec{y} + \vec{U}^1\vec{b} \,|\, \vec{y} \in \mathbb{Z}^l\right\},$$

for some l.

Moreover we conclude that the number of rows of $\vec{H}$ is greater or equal the number of its columns, i.e. $n + m \geq l$, because $\vec{H}$ has a trivial kernel.

Given this representation of $\mathcal{Z}(T)$ we compute

$$\pi_{\overline{(1,\ldots,m)}}\left(\mathcal{Z}\left(R \circ \pi_{\overline{(1,\ldots,m)}}\right) \cap \mathcal{Z}(T)\right)$$

now. Later we will only have to do this if the row rank of $\vec{N}$ is $\geq m$. Hence we assume the following.

Assumption 2.71
For the rest of this section let the row rank of $\vec{N}$ be $\geq m$.

$\square$

Note that assumption 2.71 implies that $l \leq n$. In other words solving the system of equations in the right hand side of line 2.17 means eliminating the components $x_1, \ldots, x_m$ and some components out of $x_{m+1}, \ldots, x_{m+n}$. The number of the latter components is exactly $m + n - l$.

Let

$$\begin{aligned}
\vec{C} &:= [H_{ij}]_{\substack{i=m+1,\ldots,m+n,\\ j=1,\ldots,l,}}\\
\vec{D} &:= [H_{ij}]_{\substack{i=m+n-l+1,\ldots,m+n,\\ j=1,\ldots,l,}}\\
\vec{y} &:= \left(\vec{D}\right)^{-1}\left(\pi^n_{n-l+1}\left(\vec{x} + \vec{U}^1\vec{b}\right)\right).
\end{aligned}$$

We really can compute $\left(\vec{D}\right)^{-1}$ because $\left(\vec{D}\right)$ is an upper triangular square matrix. In general its inverse will have rational coefficients. Then

$$\vec{x} \in \pi_{\overline{(1,\ldots,m)}}\left(\mathcal{Z}\left(R \circ \pi_{\overline{(1,\ldots,m)}}\right) \cap \mathcal{Z}(T)\right)$$

iff

$$\begin{gathered}\vec{x} = \vec{C}\vec{y} + \pi_{m+1}^{m+n}\left(\vec{U^1}\vec{b}\right), \\ \vec{y} \in \mathbb{Z}^l \\ \text{and} \\ \vec{x} \in \mathcal{Z}(R).\end{gathered}$$

Summarising we write as follows.
Let

$$\begin{aligned} R &:= (B, \lessdot), \\ F &:= \vec{y} \mapsto \vec{C}\vec{y} + \pi_{m+1}^{m+n}\left(\vec{U^1}\vec{b}\right), \\ S &:= (B \circ F, \lessdot). \end{aligned}$$

Then[21]

$$\pi_{\overline{(1,\ldots,m)}}\left(\mathcal{Z}\left(R \circ \pi_{\overline{(1,\ldots,m)}}\right) \cap \mathcal{Z}(T)\right) = \{F(\vec{y}) \mid \vec{y} \in \mathcal{Z}(S)\}.$$

Later we will also have to project this set. Using this representation we can easily do this. Namely we have

$$\pi_{k+1}^{n}\{F(\vec{y}) \mid \vec{y} \in \mathcal{Z}(S)\} = \left\{G\left(\pi_{k-\min(k,n-l)+1}^{l}\vec{y}\right) \mid \vec{y} \in \mathcal{Z}(S)\right\}, \tag{2.18}$$

where $n \geq k$ and

$$\begin{aligned} \vec{E} &:= [H_{ij}]_{\substack{i=m+k+1,\ldots,m+n, \\ j=k-\min(k,n-l)+1,\ldots,l,}} \\ G &:= \vec{z} \mapsto \vec{E}\vec{z} + \pi_{m+k+1}^{m+n}\left(\vec{U^1}\vec{b}\right). \end{aligned}$$

Later k will be exactly the number of integration variables. Whereas the rest of the rows of this matrix will correspond to the parameters. The following figure 2.1 shows the above partitioning of the integer kernel $\vec{H}$.

[21]Note that this says that the set of integer points of a flat polyhedron is the affine linear image of the integer points of some non flat one.

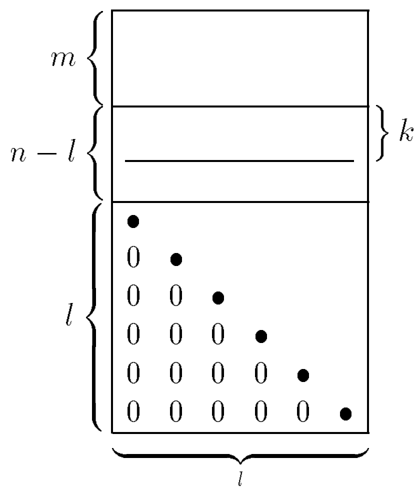

Figure 2.1: Kernel $\vec{H}$ of $\vec{N}$ in Hermite normal form

Note that the Hermite normal form structure of $\vec{H}$ implies that the map G has a left inverse. The existence of this left inverse is needed in the specification of the integer volume algorithm. In general we have that $G^{-1}\mathbb{Z}^{n-k} \cap \mathbb{Q}^{\min(l,n-k)} \neq \emptyset$. But if $k \geq n-l$ we know that G is bijective. In this case we can decide even more easily than in general whether a $\vec{y} \in \mathbb{Z}^{n-k}$ lies within the integer image of G. This is done in the next lemma.

Lemma 2.72
Let $f : \mathbb{R}^{n-k} \to \mathbb{R}^{n-k}$ *with* $f\left(\mathbb{Z}^{n-k}\right) \subseteq \mathbb{Z}^{n-k}$ *be bijective. Then we have*

$$\vec{y} \in f\left(\mathbb{Z}^{n-k}\right)$$

iff

$$f^{-1}\vec{y} \in \mathbb{Z}^{n-k}.$$

□

Proof:

"⇒": There exists a $\vec{x} \in \mathbb{Z}^{n-k}$ such that $\vec{y} = f\vec{x}$. Therefore $f^{-1}\vec{y} = \vec{x} \in \mathbb{Z}^{n-k}$.
"⇐": We have $\vec{y} = ff^{-1}\vec{y} \in f\left(\mathbb{Z}^{n-k}\right)$.

□

For an application of these formulae see chapters 4, State of the Art, and 5, Improvements.

Chapter 3

Abstract Treatment of the Problem

Now we couch the problems 1.2, Real volume, and 1.7, Number of integer solutions, into the language introduced in the previous chapter.

Moreover we show how one can apply Fubini's theorem 2.20 in order to derive an "abstract" algorithm for solving the "general" problem of volume computation. Here "abstract" and "general" means that the algorithm 3.9, Fubini$_2$, is formulated for an arbitrary measure and not all steps in this algorithm are refined so far that they can be efficiently implemented on a computing device. The actual solution methods of chapter 4, State of the Art, are refinements of algorithm Fubini$_2$ which is itself a refinement of algorithm 3.6, Fubini$_1$.

3.1 Reformulation of the Problem

We reformulate the problems 1.2, Real volume, and 1.7, Number of integer solutions, by applying the definitions of chapter 2, Basic Theory. There are no changes. We only use the integral notation for denoting $\mathcal{R}_{S,k}$ and $\mathcal{C}_{S,k}$. This allows us to apply the techniques of measure theory to our problems.

Problem 3.1 (Lebesgue volume)
Let $k \leq n$. We want to compute a representation of the function $\mathcal{R}_{S,k}$ defined by

$$\begin{array}{rcrcl} \mathcal{R}_{S,k} & : & \mathbb{R}^{n-k} & \to & \overline{\mathbb{R}}_0^+ \\ & & \vec{y} & \mapsto & \int_{\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)} 1 \, d\lambda. \end{array}$$

□

Problem 3.2 (Integer volume)
Let $k \leq n$. We want to compute a representation of the function $\mathcal{C}_{S,k}$ defined by

$$\begin{array}{rccl} \mathcal{C}_{S,k} & : & \mathbb{Z}^{n-k} & \to & \overline{\mathbb{R}}_0^+ \\ & & \vec{y} & \mapsto & \int_{\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)} 1 \, d\gamma \Big|_{\mathbb{Z}^{n-k}}. \end{array}$$

□

Obviously the problems 1.2, Real volume, and 1.7, Number of integer solutions, are the same[1] as the problems 3.1, Lebesgue volume, and 3.2, Integer volume. From now on we will refer to the latter ones.

Note that apart from the replacement of λ by γ the main obvious difference in the two problem statements is the restriction of $\vec{y}$ to $\mathbb{Z}^{n-k}$ in the problem of the Integer volume. This is motivated by this occurrence of the problem in practical applications as already described in chapter 1, Problem Statement. But we also want to point out that without this restriction the problem wouldn't be immediately solvable with the methods presented later.

We want to emphasise that the above two problems are really "interesting" ones from the point of view of complexity theory.

It is well known (cf. [HU93]) that the problem of deciding whether a polyhedron contains any integer points is NP-complete (cf. [HU93]). This is the problem of *integer programming*. Therefore it isn't surprising at all that counting the number of integer solution of a system of linear inequalities *without parameters* is "at least" as difficult as NP-complete problems. More precisely one can show that it is a member of the class of ♯P-complete problems (cf. [Val79]). These problems form a class of equivalent counting problems. Typical members of this class are the problems of counting the number of solutions of NP-complete problems. Therefore one knows that the class of ♯P-complete problems contains members *computationally very difficult*.

In the case of the Lebesgue volume *without parameters* we are in a similar situation. This problem is ♯P-hard.

These few facts surely reveal the "interestingness" of *parametrised* volume computation.

As already mentioned in chapter 1, Problem Statement, we will represent the functions $\mathcal{R}_{S,k}$, $\mathcal{C}_{S,k}$ by so-called piecewise polynomials that are defined next.

[1] Because of this the reader will apologise that we define the symbols $\mathcal{C}$ and $\mathcal{R}$ twice, namely here and in chapter 1, Problem Statement. In both cases the two defining expressions denote the same function anyway.

Definition 3.3 (Piecewise polynomial)
Let $D : \mathbb{R}^n \to \overline{\mathbb{R}}$. We call D a piecewise polynomial (in n variables) iff

$$D = \sum_{(Q,Y)\in Z} Q \cdot \mathbf{1}_Y,$$

where Z is a finite set[2] *of pairs*[3] *(Q, Y), for some Q and Y.*

□

One piecewise polynomial can be found in example 1.4.

We arrange that D and E will range over the set of piecewise polynomials.

In fact the output of the volume algorithm in chapter 4, State of the Art, will be a special piecewise polynomial.

In the case of the Lebesgue volume it will be represented by a finite set Z of pairs (Q, T), for some Q and T. This data represents the piecewise polynomial

$$P = \sum_{(Q,T)\,\in\, Z} Q \cdot \mathbf{1}_{\mathcal{P}(T)}.$$

In the case of the Integer volume it will have the form

$$P = \sum_{\left(Q,T,\vec{A}-\vec{a},X\right)\,\in\, Z} Q \cdot \mathbf{1}_{\vec{A}(\mathcal{P}(T))-\vec{a}\,\cap\, X},$$

where Z is a finite set of quadruples $\left(Q, T, \vec{A} - \vec{a}, X\right)$, for some Q, T, $X \subseteq \mathbb{Z}^{n-k}$ and $\vec{A} - \vec{a}$ being affine linear.

We will see in sections 4.2, Regular Decomposition, and 4.3, Fubini and Decomposition Combined, that this is possible at all.

3.2 Abstract Solution

Now we present a first approach to solve the problems Lebesgue volume and Integer volume. It is still very abstract in the sense that it is independent of the measure used to measure the volume. By the application of Fubini's theorem we will show how to compute the following function $\mathcal{V}_{\mu,S,k}$.

[2]Note that Z isn't unique!

[3]This means that some "pieces" of these piecewise polynomials might not be polynomials but ∞.

Problem 3.4 (Volume)
Let $k \leq n$. We want to compute the function[4] $\mathcal{V}_{\mu,S,k}$ *defined by*

$$\begin{array}{cccl} \mathcal{V}_{\mu,S,k} & : & \mathbb{R}^{n-k} & \to & \overline{\mathbb{R}} \\ & & \vec{y} & \mapsto & \int_{\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)} 1 \, d\mu, \end{array}$$

$(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n), \mu)$ *is a measure space and* μ *is* σ*-finite.*

□

For the rest of this thesis let's assume that μ is as in problem 3.4, Volume, such that the volume function is well defined.

In section 4, State of the Art, we will specialise the measure μ and we will compute the maps $\mathcal{R}_{S,k}$ and $\mathcal{C}_{S,k}$ related to $\mathcal{V}_{\mu,S,k}$ in the following way:

$$\begin{aligned} \mathcal{R}_{S,k} &= \mathcal{V}_{\lambda,S,k}, \\ \mathcal{C}_{S,k} &= \mathcal{V}_{\gamma,S,k}|_{\mathbb{Z}^{n-k}} . \end{aligned}$$

Now we compute $\mathcal{V}_{\mu,S,k}$ recursively. That means: given $\mathcal{V}_{\mu,S,i}$ for some $i \leq k$ we integrate $\mathcal{V}_{\mu,S,i}(., \vec{y})$ over the projection of $\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)$ to $\pi_{i+1}^k\left(\mathbb{R}^k\right)$. This is made exact in the following lemma.

Lemma 3.5
Let $i \leq k \leq n$, $S \in \mathcal{LS}(n)$. Then

$$\mathcal{V}_{\mu,S,k}(\vec{y}) = \int_{\pi_{i+1}^k\left(\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)\right)} \mathcal{V}_{\mu,S,i}(\vec{x}, \vec{y}) \, d\mu(\vec{x}) .$$

□

Proof:

Apply Fubini's theorem 2.20 and the following formula:

$$\mathbf{1}_{\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)}(\vec{x}) = \mathbf{1}_{\pi_{i+1}^k\left(\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)\right)}(x_{i+1}, \ldots, x_k) \, \mathbf{1}_{\mathcal{P}\left(\mathcal{F}_1^i S(x_{i+1},\ldots,x_k,\vec{y})\right)}(x_1, \ldots, x_i) .$$

[4]For an arbitrary measure μ we cannot say how to represent this function. We are particularly interested in the measures λ and γ for which the analogous functions $\mathcal{R}_{S,k}$ and $\mathcal{C}_{S,k}$, resp., are piecewise polynomials. Therefore it is satisfying to compute $\mathcal{V}_{\mu,S,k}$ for such measures μ such that it is a piecewise polynomial. We don't state this restriction explicitely because we don't refine all steps in the algorithms in this section so far that the representation of the map $\mathcal{V}_{\mu,S,k}$ influences the presentation of the algorithms. The reader just might imagine that we are working with an abstract computable data type (cf. [vAHU83]) "function" on which all operations occurring there are defined. More precisely, the algorithms in this section are recursive relative to this data type in the sense of computability theory.

□

Based on lemma 3.5 we formulate the following general skeletal recursive algorithm for solving the problem 3.4, Volume, by setting $i := k - 1$, i.e. we only integrate over a one dimensional projection.

Algorithm 3.6

$h \leftarrow \text{Fubini}_1\big(k, n, S\big)$

Input:

$k, n \ldots\ldots k \leq n.$

$S \ldots\ldots\ldots S \in \mathcal{LS}\,(n).$

Output:

$h \ldots\ldots\ldots$ *equals* $\mathcal{V}_{\mu,S,k}$.

[1] Recursion base.

(3.1) **If** $k = 0$ **then**

(3.2) $h \leftarrow \mathbf{1}_{\mathcal{P}(S)}$

[2] Recursion body.

else

(3.3) $g \leftarrow \text{Fubini}_1\big(k - 1, n, S\big)$

(3.4) $h \leftarrow \int_{\pi_k\left(\mathcal{P}\left(\mathcal{F}_1^k S(.)\right)\right)} g\,(\vec{x}, .)\, d\mu\,(\vec{x})$

Return

□

Corollary 3.7

Algorithm 3.6, Fubini$_1$, *is correct.*

□

Proof:

Partial Correctness: In the proof we apply the techniques of formal program verification (cf. [Fut89]).

We have to show that

$$\{k \leq n \text{ and } S \in \mathcal{LS}\,(n)\}\ \text{Fubini}_1\big(k, n, S\big)\ \ \{h = \mathcal{V}_{\mu,S,k}\}\,.$$

This means

1. $\{k = 0$ and $k \leq n$ and $S \in \mathcal{LS}\,(n)\}$ (3.2) $\{h = \mathcal{V}_{\mu,S,k}\}$ that is valid by the definition of $\mathcal{V}_{\mu,S,0}$.

2. $\{k \neq 0 \text{ and } k \leq n \text{ and } S \in \mathcal{LS}(n)\}$ (3.3)(3.4) $\{h = \mathcal{V}_{\mu,S,k}\}$.

   Applying the rule of backward substitution we get:

$$\{k \neq 0 \text{ and } k \leq n \text{ and } S \in \mathcal{LS}(n)\}$$
$$(3.3)$$
$$\left\{\int_{\pi_k\left(\mathcal{P}\left(\mathcal{F}_1^k S\right)\right)} g(\vec{x},.)\, d\mu(\vec{x}) = \mathcal{V}_{\mu,S,k}.\right\}$$

   By induction on the tree of recursive function calls we have to show:

   (a)
$$(k \neq 0 \text{ and } k \leq n \text{ and } S \in \mathcal{LS}(n))$$
   implies
$$(k-1 \leq n \text{ and } S \in \mathcal{LS}(n))$$
   which is trivial.

   (b)
$$g = \mathcal{V}_{\mu,S,k-1}$$
   implies
$$\int_{\pi_k\left(\mathcal{P}\left(\mathcal{F}_1^k S\right)\right)} g(\vec{x},.)\, d\mu(\vec{x}) = \mathcal{V}_{\mu,S,k}$$
   which is valid because of lemma 3.5 by choosing $i := k-1$.

Termination: Obvious.

□

Now we refine lemma 3.5.

1. In order to compute $\pi_{i+1}^k\left(\mathcal{P}\left(\mathcal{F}_1^k S(\vec{y})\right)\right)$ we must only know a linear system of inequalities defining $\pi_{i+1}^n(\mathcal{P}(S))$.

2. It suffices to compute a linear system corresponding to $\pi_{i+1}^n(\mathcal{P}(S))$ approximately in the sense of section 2.5, Projections of Polyhedra.

Lemma 3.8
Let $i \leq k \leq n$, $S \in \mathcal{LS}(n)$. Then

$$\mathcal{V}_{\mu,S,k}(\vec{y}) = \int_{\mathcal{P}\left(\mathcal{F}_1^{k-i} T(\vec{y})\right)} \mathcal{V}_{\mu,S,i}(\vec{x},\vec{y})\, d\mu(\vec{x}), \text{ where } T \in {}_{\mu}\mathrm{Proj}_{i+1}^n(S).$$

□

Proof:

Let

$$T \in {}_{\mu}\mathrm{Proj}^{n}_{i+1}(S) \tag{3.5}$$

and

$$Y := \pi^{k}_{i+1}\left(\mathcal{P}\left(\mathcal{F}^{k}_{1} S(\vec{y})\right)\right) \triangle \mathcal{P}\left(\mathcal{F}^{k-i}_{1} T(\vec{y})\right).$$

In order to apply lemma 2.17 on lemma 3.5 we have to show that

$$\mu(Y) = 0.$$

Assumption (3.5) implies that

$$\mu(X) = 0,$$

where

$$X := \pi^{n}_{i+1}(\mathcal{P}(S)) \triangle \mathcal{P}(T).$$

Now,

$$\begin{array}{rcccl} \mu(X) & = & & & \\ & = & \mu(X)\cdot 1 & = & \\ & = & \mu(X)\,\mu(\{\vec{y}\}) & = & \\ & = & \mu(X\times\{\vec{y}\}) & = & \\ & = & \mu(Y) & = & \\ & = & & = & 0. \end{array}$$

□

Again we turn the previous lemma into a recursive algorithm by setting $i := k-1$.

Algorithm 3.9

$(h, U) \leftarrow \mathrm{Fubini}_2(k, n, S)$

Input:

$k, n \ldots\ldots k \leq n.$

$S \ldots\ldots\ldots \in \mathcal{LS}(n).$

Output:

$h, U \ldots\ldots$ *such that*

$$\mathcal{V}_{\mu,S,k}(\vec{y}) = \begin{cases} h(\vec{y}) & \textit{if } \vec{y} \in \mathcal{P}(U), \\ 0 & \textit{else}, \end{cases} \tag{3.6}$$

where $\mathcal{P}(U) = \pi^{n}_{k+1}(\mathcal{P}(S))$

[1] Recursion base.

(3.7) **If** $k = 0$ **then**

(3.8) $(h, U) \leftarrow (1, S)$

[2] Recursion body.

else

(3.9) $(g, T) \leftarrow \mathrm{Fubini}_2\big(k-1, n, S\big)$

(3.10) $U \leftarrow$ *such that*[5] $\mathcal{P}\,(U) = \pi_2^{n-k+1}\,(\mathcal{P}\,(T))$

(3.11) $h \leftarrow \left(\int_{\mathcal{P}\left(\mathcal{F}_1 T(.)\right)} g\,(\vec{x}, .)\, d\mu\,(\vec{x})\right)\Big|_{\mathcal{P}(U)}$

Return

□

Corollary 3.10
Algorithm 3.9, Fubini$_2$*, is correct.*

□

Proof:

Partial correctness: In the proof we apply the techniques of formal program verification (cf. [Fut89]).

We have to show that

$$\{k \le n \text{ and } S \in \mathcal{LS}\,(n)\}$$
$$\mathrm{Fubini}_2\big(k, n, S\big)$$
$$\left\{ \mathcal{V}_{\mu,S,k}\,(\vec{y}) = \begin{cases} h\,(\vec{y}) & \text{if } \vec{y} \in \mathcal{P}\,(U), \\ 0 & \text{else,} \end{cases} \text{ and } \mathcal{P}\,(U) = \pi_{k+1}^{n}\,\mathcal{P}\,(S)\,. \right\}$$

This means

1.

$$\{k = 0 \text{ and } k \le n \text{ and } S \in \mathcal{LS}\,(n)\}$$
$$(3.8)$$
$$\left\{ \mathcal{V}_{\mu,S,k}\,(\vec{y}) = \begin{cases} h\,(\vec{y}) & \text{if } \vec{y} \in \mathcal{P}\,(U), \\ 0 & \text{else,} \end{cases} \text{ and } \mathcal{P}\,(U) = \pi_{k+1}^{n}\,\mathcal{P}\,(S) \right\}$$

that is valid by the definition of $\mathcal{V}_{\mu,S,0}$.

[5] U can be computed performing Fourier-Motzkin projection.

2.

$$\{k \neq 0 \text{ and } k \leq n \text{ and } S \in \mathcal{LS}(n)\}$$
$$(3.9)(3.10)(3.11)$$
$$\left\{ \mathcal{V}_{\mu,S,k}(\vec{y}) = \begin{cases} h(\vec{y}) & \text{if } \vec{y} \in \mathcal{P}(U), \\ 0 & \text{else,} \end{cases} \text{ and } \mathcal{P}(U) = \pi^n_{k+1}\,\mathcal{P}(S) \right\}$$

Applying the rule of backward substitution we get:

$$\{k \neq 0 \text{ and } k \leq n \text{ and } S \in \mathcal{LS}(n)\}$$
$$(3.9)$$
$$\left\{ \begin{array}{l} \mathcal{V}_{\mu,S,k}(\vec{y}) = \begin{cases} \int_{\mathcal{P}(\mathcal{F}_1 T(\vec{y}))} g(\vec{x},\vec{y})\, d\mu(\vec{x}) & \text{if } \vec{y} \in \mathcal{P}(U), \\ 0 & \text{else,} \end{cases} \\ \qquad\qquad \text{and } \mathcal{P}(U) = \pi^n_{k+1}\,\mathcal{P}(S). \end{array} \right\}$$

By induction on the tree of recursive function calls we have to show:

(a)

$$(k \neq 0 \text{ and } k \leq n \text{ and } S \in \mathcal{LS}(n))$$

implies

$$(k-1 \leq n \text{ and } S \in \mathcal{LS}(n))$$

which is trivial.

(b)

$$\mathcal{V}_{\mu,S,k-1}(\vec{y}) = \begin{cases} g(\vec{y}) & \text{if } \vec{y} \in \mathcal{P}(T), \\ 0 & \text{else,} \end{cases}$$
$$\text{where } \mathcal{P}(T) = \pi^n_k\,\mathcal{P}(S),$$

implies

$$\mathcal{V}_{\mu,S,k}(\vec{y}) = \begin{cases} \int_{\mathcal{P}(\mathcal{F}_1 T(\vec{y}))} g(\vec{x},\vec{y})\, d\mu(\vec{x}) & \text{if } \vec{y} \in \mathcal{P}(U), \\ 0 & \text{else,} \end{cases}$$
$$\text{and } \mathcal{P}(U) = \pi^n_{k+1}\,\mathcal{P}(S).$$

which is valid because of lemma 3.8 by choosing $i := k-1$ and equation 2.11.

Termination: Obvious.

□

In the next chapter we will refine algorithm 3.9. The challenge will be to compute the integral in line (3.11). This will be done by a proper decomposition of the polyhedron.

Chapter 4

State of the Art

Here we describe all known approaches for solving the problem 3.2, Integer volume, or certain instances of this problem. We also use these methods in order to solve the problem 3.1, Lebesgue volume, applying the generalisation techniques introduced in chapter 3, Abstract Treatment of the Problem.

Section 4.1, Unit Step Calculus, reviews a method invented by Haghighat and Polychronopoulos (cf. [HP94]) treating "nearly" triangular (cf. problem 4.3) systems of linear inequalities. We will show that this method isn't capable enough for solving this instance of problem 3.2, Integer volume, and because of the same reasons the problem 3.1, Lebesgue volume, either.

In section 4.2, Regular Decomposition, we present an algorithm developed by Tawbi that can be completed such that it solves the problems, Integer volume, (cf. [Taw95]) if the linear system is triangular (cf. definition 4.1) and also the problem, Lebesgue volume.

Section 4.3, Fubini and Decomposition Combined, treats Pugh's algorithm (cf. [Pug94]) that solves the problem, Integer volume, completely. After a slight modification it also solves the problem, Lebesgue volume. This method can be regarded as an extension and completion of the algorithm of Tawbi's' ([Taw95]).

All three of them don't allow systems of linear inequalities containing strict inequalities as inputs to the algorithms. But in the case of Tawbi's' and Pugh's methods this generalisation to non strict inequalities is really obvious such that we will present the algorithms already in this generalised form.

4.1 Unit Step Calculus

Now we review a method for computing $\mathcal{C}_{S,k}$ if the given system S is triangular or "nearly triangular" (cf. problem 4.3) published in [HP94] which is shown by us to be incomplete. This approach is based on a calculus involving the so-called unit step function (cf. definition 4.9).

First we introduce the notion of triangular linear system of depth k. A system S is triangular of depth k iff the matrices $[S_{2i-1,j}]_{i,j=1,\ldots,k}$ as well as $[S_{2i,j}]_{i,j=1,\ldots,k}$ are both upper triangular matrices[1] where the first one consists of negative diagonal elements and the second one of positive diagonal elements. Moreover we require the matrix $[S_{i,j}]_{i=2k+1,\ldots,||S||,j=1,\ldots,k}$ to be 0.

Such triangular systems arise when the complexity of nested loops is measured as described in chapter 1, Problem Statement.

Definition 4.1 (Triangular linear system of inequalities)
Let $k \leq n$. Then $S \in \mathcal{LS}(n)$ is said to be triangular of depth k iff

One of the following conditions is satisfied:

1. $k = 0$.
2. $S_{11} < 0, S_{21} > 0, S_{i1} = 0$, *for* $i \geq 3$, *and* ${}_2^nS_3^{||S||}$ *is triangular of depth* $k - 1$.

□

We illustrate this definition in the following example.

Example 4.2
Let

$$S := \left(\begin{bmatrix} -3 & 2 & 0 & 7 \\ 2 & 0 & 8 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 6 & 1 & 5 \\ 0 & 0 & -4 & 0 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 \end{bmatrix} - \begin{pmatrix} 8 \\ 7 \\ 6 \\ 5 \\ 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}, \leq \right).$$

We see that $S \in \mathcal{LS}(4)$ and it is triangular of depth 3.

□

[1] We caution the reader against mixing the meanings of triangular linear system of inequalities defined in definition 4.1 with the meaning of triangular matrix of linear algebra.

Haghighat and Polychronopoulos consider the following instance of the problem 3.2, Integer volume.

Problem 4.3 (Nearly triangular)
Let $S \in \mathcal{LS}(n)$. We want to compute $\mathcal{C}_{S,k}$ if

$$S \text{ or } S_2^{\|S\|} \text{ is triangular}$$

and

$$S_{ij}, S_{i\infty} \in \mathbb{Z}, \text{ for } i = 1, \ldots, \|S\|, j = 1, \ldots, n.$$

□

In order to illustrate this instance of problem 3.2, Integer volume, we show two simple examples of linear systems of inequalities that could serve as inputs to an algorithm solving problem 4.3, Nearly triangular.

Example 4.4 ([HP94])
Let

$$S := \left(\begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} - \begin{pmatrix} 0 \\ 5 \\ -3 \\ 0 \\ -1 \\ 0 \end{pmatrix}, \leq \right) \tag{4.1}$$

Obviously $S \in \mathcal{LS}(4)$ and S is triangular of depth 3.

□

Example 4.5 ([HP94])
Let

$$S := \left(\begin{bmatrix} 1 & 1 & -2 \\ -1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & -2 \end{bmatrix} - \begin{pmatrix} 0 \\ -1 \\ 0 \\ -1 \\ 0 \end{pmatrix}, \leq \right) \tag{4.2}$$

Here $S \in \mathcal{LS}(3)$ and S_2^5 is triangular of depth 2.

□

Now we show how $\mathcal{C}_{S,k}$ can be expressed in terms of sums, $\sum$, for these two examples by using the special structure of S, Fubini's theorem 2.20 and the γ-integration lemma 2.26.

Exactly these sums are computed by Haghighat and Polychronopoulos.

Example 4.6 (Ad example 4.4)
We rewrite the system S in the following way:

$$\begin{array}{rcccl} x_2 & \leq & x_1 & \leq & 5, \\ 3 & \leq & x_2 & \leq & x_3, \\ 1 & \leq & x_3 & \leq & x_4. \end{array}$$

One can see that the lower and upper bounds on each variable x_i, for $i \leq 3$, depend only on variables x_j with $j > i$. This is a consequence of the special structure of S. Apart from this note that the system was chosen by Haghighat and Polychronopoulos such that no rational numbers occur in the affine linear expressions bounding each variable!

Now we have

$$\begin{aligned} \mathcal{C}_{S,3}(x_4) &= \int_{\mathcal{P}\left(S_1^3(x_4)\right)} 1 \, d\gamma \\ &= \int_1^{x_4} \int_3^{x_3} \int_{x_2}^5 1 \, d\gamma(x_1) \, d\gamma(x_2) \, d\gamma(x_3) \quad \text{(Fubini's theorem 2.20)} \\ &= \sum_{x_3=1}^{x_4} \sum_{x_2=3}^{x_3} \sum_{x_1=x_2}^{5} 1 \quad \text{(lemma 2.26).} \end{aligned}$$

□

Example 4.7 (Ad example 4.5)
We rewrite the system S in the following way:

$$\begin{array}{rcccl} & & x_1 & \leq & -x_2 + 2x_3, \\ 1 & \leq & x_1 & \leq & x_2, \\ 1 & \leq & x_2 & \leq & 2x_3. \end{array}$$

Again the bounds on each variable depend only on variables with greater indices. Moreover there only occur integers as above.

Now we have

$$\mathcal{C}_{S,2}(x_3) = \int_{\mathcal{P}\left(S_1^2(x_3)\right)} 1\, d\gamma \tag{4.3}$$

$$= \int_1^{2x_3} \int_1^{\min(x_2,-x_2+2x_3)} 1\, d\gamma(x_1)\, d\gamma(x_2) \quad \text{(Fubini's theorem 2.20)} \tag{4.4}$$

$$= \sum_{x_2=1}^{2x_3} \sum_{x_1=1}^{\min(x_2,-x_2+2x_3)} 1 \quad \text{(lemma 2.26)}. \tag{4.5}$$

The inner sum can be rewritten further, namely:

$$\mathcal{C}_{S,2}(x_3) = \sum_{x_2=1}^{2x_3} \left(\sum_{x_1=1}^{x2} 1 - \sum_{x_1=\max(1,-x_2+2x_3)}^{x2} 1 \right). \tag{4.6}$$

This last step performed by Haghighat and Polychronopoulos will be discussed in detail later.

□

The authors of [HP94] introduce the truth function and the unit step function in order to compute the above sums.

Definition 4.8 (Truth function [HP94]**)**
Let X_i be a set, for $i = 1, \ldots, l$, V an l-ary relation defined on $\prod_{i=1}^{l} X_i$. Then the truth function[2] $\mathcal{T}$ is defined in the following way[3]:

$$\mathcal{T}(V(x_1, \ldots, x_l)) := \begin{cases} 1 & \textit{iff } V(x_1, \ldots, x_l), \\ 0 & \textit{else.} \end{cases}$$

□

Definition 4.9 (Unit step function [HP94]**)**
The unit step function[2] $\mathcal{U}$ is defined as:

$$\begin{array}{rccl} \mathcal{U} & : & \mathbb{R} & \rightarrow & \{0,1\} \\ & & x & \mapsto & \mathcal{T}(x > 0). \end{array}$$

□

To put it another way, the unit step function is the indicator function of the interval $]0,\infty[$ restricted to $\mathbb{R}$, i.e. $\mathcal{U} = \mathbf{1}_{]0,\infty[}|_{\mathbb{R}}$.

Now we state a few properties of these two functions. The first corollary shows how $\mathcal{T}$ and the logical operations are related. The second corollary states some rules how expressions involving $\mathcal{T}$ and $\mathcal{U}$ can be rewritten.

Corollary 4.10 ([HP94])
Let X_i, Y_j be sets, for $i = 1, \dots, l$ and $j = 1, \dots, l'$,resp., V and W be relations defined on $\prod_{i=1}^{l} X_i$ and $\prod_{j=1}^{l'} Y_j$, resp., $x \in \prod_{i=1}^{l} X_i$ and $y \in \prod_{j=1}^{l'} Y_j$. Then

1. $\mathcal{T}(V(x)) + \mathcal{T}(\neg V(x)) = 1,$
2. $\mathcal{T}(V(x))\mathcal{T}(\neg V(x)) = 0,$
3. $\mathcal{T}(V(x) \wedge W(y)) = \mathcal{T}(V(x))\mathcal{T}(W(y)),$
4. $\mathcal{T}(V(x) \vee W(y)) = \mathcal{T}(V(x)) + \mathcal{T}(W(y)) - \mathcal{T}(V(x) \wedge W(y)).$

□

Corollary 4.11 ([HP94])
For any $x, \alpha \in \mathbb{R}$, the following hold:

1. $\mathcal{U}(\alpha x) = \begin{cases} \mathcal{U}(x) & \text{iff } \alpha \geq 0, \\ \mathcal{U}(-x) & \text{else.} \end{cases}$
2. $\mathcal{U}(xy) = \mathcal{U}(x)\mathcal{U}(y) + \mathcal{U}(-x)\mathcal{U}(-y).$
3. $\mathcal{U}^n(x) = \mathcal{U}(x)$, *where $n \geq 1$ and*
$$\mathcal{U}^k(x) := \begin{cases} x & \text{iff } k = 0, \\ \mathcal{U}\left(\mathcal{U}^{k-1}(x)\right) & \text{else.} \end{cases}$$

[2]We use a notation slightly different to [HP94]. There the truth and unit step function is denoted by the symbols τ and μ. However we use the symbols $\mathcal{T}$ and $\mathcal{U}$ because in section 2.1, Measure Theory, μ is a variable ranging over all measures. This is the standard notation of measure theory.

For consistency we also change τ to $\mathcal{T}$.

[3]Haghighat and Polychronopoulos define the truth function $\mathcal{T}$ a bit differently. Namely,

$$\begin{aligned} \mathcal{T} \ : \ \{\text{FALSE}, \text{TRUE}\} &\rightarrow \{0, 1\} \\ c &\mapsto \begin{cases} 0 & \text{iff } c = \text{FALSE}, \\ 1 & \text{else } (c = \text{TRUE}), \end{cases} \end{aligned}$$

where FALSE and TRUE are so-called conditional values. We don't adopt this definition directly because the language of first order logics we use doesn't contain any symbols whose semantics is "false" and "true".

4. $\mathcal{U}(x)\mathcal{U}(-x) = 0$.
5. $\mathcal{U}(x)\, x = \max(0, x)$.
6. $\mathcal{T}(x > 0) = \mathcal{U}(x)$.
7. $\mathcal{T}(x < 0) = \mathcal{U}(-x)$.
8. $\mathcal{T}(x = 0) = 1 - \mathcal{U}(x) - \mathcal{U}(-x)$.
9. $\mathcal{T}(x \geq 0) = 1 - \mathcal{U}(-x)$.
10. $\mathcal{T}(x \leq 0) = 1 - \mathcal{U}(x)$.
11. $\mathcal{T}(x \neq 0) = \mathcal{U}(x) + \mathcal{U}(-x)$.

□

Based on corollary 4.11 Haghighat and Polychronopoulos introduce a calculus we call unit step calculus.

Definition 4.12 (Unit step calculus)
The set of rewrite rules given by corollary 4.11 form the unit step calculus.

□

In order to evaluate a sum of the form

$$\sum_{x=v}^{w} P(x),$$

where $P \in \mathbb{Z}[x]$, they apply the unit step function in the following way

$$\sum_{x=v}^{w} P(x) = \mathcal{U}(w - v + 1) \sum_{x=v}^{w} P(x). \tag{4.7}$$

This means that the right hand side of equality 4.7 is zero iff $w < v$, i.e. the sum on the left hand side is defined to be zero. In the other case the left hand sum and right hand side are the same. This application of the unit step function is motivated by the following facts.

In the case of $v \leq w$ we know that $\sum_{x=v}^{w} P(x)$ equals the symbolic sum whose computation is described in section 2.3, Integer Measure. As the right hand side of the above equality is zero if $b > a$ we can replace $\sum_{x=v}^{w} P(x)$ by the symbolic sum independently from how v and w are related.

The authors of [HP94] claim that by applying the unit step calculus one can solve "reasonable simple" instances of problem 4.3 and compute all occurring sums but they don't explain what "reasonable simple" is and they don't prove the completeness of this calculus. They illustrate their ideas via the two examples we have already introduced above but don't state a general algorithm that solves problem 4.3.

Next we will describe their strategy and explain how this coincides with algorithm 3.9, Fubini_2.

The main difference to this algorithm is the restriction of the integer volume function to the integers during the computation. That means instead of $\mathcal{P}(U)$ we have to write $\mathcal{Z}(U)$ in equation 3.6 and line 3.11. This is justified because while integrating we need not know how a function acts on a set of measure zero (cf. theorem 2.13). But exactly the sets without integer points are of γ-measure zero.

The results of these replacements are summarised in the following algorithm.

Algorithm 4.13

$(h, U) \leftarrow \text{Fubini}_{\mathbb{Z}}\big(k, n, S\big)$

Input:

$k, n \ldots\ldots k \leq n.$

$S \ldots\ldots\ldots \in \mathcal{LS}(n).$

Output:

h, U .... *such that*

$$\mathcal{C}_{S,k}(\vec{y}) = \begin{cases} h(\vec{y}) & \textit{iff } \vec{y} \in \mathcal{Z}(U), \\ 0 & \textit{else,} \end{cases}$$

where $\mathcal{P}(U) = \pi^n_{k+1}(\mathcal{P}(S))$

[1] Recursion base.

(4.8) **If** k = 0 **then**

(4.9) $(h, U) \leftarrow (1, S)$

[2] Recursion body.

else

(4.10) $(g, T) \leftarrow \text{Fubini}_{\mathbb{Z}}\big(k-1, n, S\big)$

(4.11) $U \leftarrow$ *such that* $\mathcal{P}(U) = \pi_2^{n-k+1}(\mathcal{P}(T))$

(4.12) $h \leftarrow \left(\int_{\mathcal{P}(\mathcal{F}_1 T(.))} g(\vec{x}, .)\, d\gamma(\vec{x})\right)\Big|_{\mathcal{Z}(U)}$

Return

$\square$

Example 4.14 (Ad example 4.6)
We give a trace of algorithm Fubini$_{\mathbb{Z}}$ and show in which extent Haghighat and Polychronopoulos follow this algorithm. We will label each recursion step by equalities of the form $k = l$ where l is the value of k at this recursion step as well as the output variables h, U with upper indices of the form h^l, U^l.

In each recursion step one summation is computed starting from the innermost sum.

$\boldsymbol{k = 0}$**:**

$$\left(h^0, U^0\right) = (1, S).$$

$\boldsymbol{k = 1}$**:**

$$(g, T) = \left(1, U^0\right).$$

As U^1 with $\mathcal{P}\left(U^1\right) = \pi_2^4 \mathcal{P}(T)$ we choose the Fourier-Motzkin projection. We get the following system of inequalities:

$$U^1 := \left(\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 1 \end{bmatrix} - \begin{pmatrix} 5 \\ -3 \\ 0 \\ -1 \\ 0 \end{pmatrix}, \leq \right)$$

We compute the integral (4.12). Assuming that $(x_2, x_3, x_4) \in \mathcal{Z}\left(U^1\right)$ we get:

$$\begin{aligned}\left.\left(\int_{\mathcal{P}\left(\mathcal{F}_1 T(x_2,x_3,x_4)\right)} 1 \, d\gamma\left(x_1\right)\right)\right|_{\mathcal{Z}(U^1)} &= \sum_{x_1=x_2}^{5} 1 \\ &= \mathcal{U}\left(6 - x_2\right) \sum_{x_1=x_2}^{5} 1 \\ &= \begin{cases} 6 - x_2 & \text{if } \mathcal{U}\left(6 - x_2\right) = 1, \\ 0 & \text{else.} \end{cases}\end{aligned}$$

Now Haghighat and Polychronopoulos say that:

$$\mathcal{U}\left(6 - x_2\right) = 1 \text{ if } x_2 \leq 5.$$

This inequality replaces the inequalities T_1 and T_2 of T in the projection U^1. Only these two ones are affected because of the triangular structure of T. The other ones remain the same. Therefore the above else-case where the integral vanishes is redundant. So we get

$$h^1 = 6 - x_2.$$

From this we conclude that Haghighat and Polychronopoulos encode the Fourier-Motzkin projection of the given polyhedron by the function $\mathcal{U}$. We will observe the same in the following steps of the algorithm.

$\boldsymbol{k = 2}$**:**

$$(g, T) = \left(h^1, U^1\right).$$

Again we perform the Fourier-Motzkin projection and get:

$$R := \left(\begin{bmatrix} 0 & 0 \\ -1 & 0 \\ -1 & 0 \\ 1 & -1 \end{bmatrix} - \begin{pmatrix} 2 \\ -3 \\ -1 \\ 0 \end{pmatrix}, \leq \right)$$

The inequalities R_1 and R_3 are redundant obviously. Therefore we choose the simplified system:

$$U^2 := \left(\begin{bmatrix} -1 & 0 \\ 1 & -1 \end{bmatrix} - \begin{pmatrix} -3 \\ 0 \end{pmatrix}, \leq \right)$$

We compute the integral

$$\left(\int_{\mathcal{P}\left(\mathcal{F}_1 T(x_3, x_4)\right)} 6 - x_2 \, d\gamma\left(x_2\right) \right) \Bigg|_{\mathcal{Z}(U^2)}.$$

As the system T contains the two upper bounds x_3 and 5 on x_2 the integral equals

$$\begin{aligned} \sum_{x_2=3}^{\min(x_3,5)} 6 - x_2 &= \mathcal{U}\left(\min\left(x_3, 5\right) - 2\right) \sum_{x_2=3}^{\min(x_3,5)} 6 - x_2 \\ &= \begin{cases} \frac{-\min^2(x_3,5) + 11\min(x_3,5) - 18}{2} & \text{if}\, \mathcal{U}\left(\min\left(x_3, 5\right) - 2\right) \\ & \quad = 1, \\ 0 & \text{else.} \end{cases} \end{aligned}$$

Haghighat and Polychronopoulos conclude that

$$\mathcal{U}\left(\min\left(x_3, 5\right) - 2\right) = 1 \text{ if } x_3 \geq 3 \text{ and } 5 \geq 3.$$

Because of the same reasons as above we get

$$h^2 := \frac{-\min^2(x_3, 5) + 11\min(x_3, 5) - 18}{2}.$$

***k* = 3:**

$$(g, T) = \left(h^2, U^2\right).$$

The Fourier-Motzkin elimination of T is:

$$U^3 := \left(\left[-1\right] - (-3),\ \leq\right)$$

In order to compute the integral

$$\left(\int_{\mathcal{P}\left(\mathcal{F}_1 T(x_4)\right)} \frac{-\min^2(x_3, 5) + 11\min(x_3, 5) - 18}{2}\, d\gamma(x_3)\right)\Bigg|_{\mathcal{Z}(U^3)}$$

Haghighat and Polychronopoulos split the polyhedra $\mathcal{P}(\mathcal{F}_1 T(x_4))$ and the projection U^3 because the following symbolic summation over the integrand requires a case distinction because of the min-functions occurring there.

We have

$$\min(x_3, 5) = \begin{cases} x_3 & \text{if } x_3 \leq 5, \\ 5 & \text{else } (6 \leq x_3). \end{cases}$$

First part: We label the output variables h, U, the linear system determining the splitted polyhedron and the projection by $'$.

We add the inequality $x_3 \leq 5$ to T and get

$$T' := \left(\begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 1 & -1 \end{bmatrix} - \begin{pmatrix} 5 \\ -3 \\ 0 \end{pmatrix},\ \leq\right)$$

The simplified projection U' of T' remains the same in this case, namely

$$U' := U^3.$$

Assuming that $x_4 \in \mathcal{Z}(U')$ we get

$$\left(\int_{\mathcal{P}\left(\mathcal{F}_1 T'(x_4)\right)} \frac{-\left(x_3\right)^2+11\,x_3-18}{2}\,d\gamma\left(x_3\right)\right)\Bigg|_{\mathcal{Z}(U')} =$$

$$\sum_{x_3=3}^{\min(x_4,5)} \frac{-\left(x_3\right)^2+11\,x_3-18}{2} =$$

$$\mathcal{U}\left(\min\left(x_4,5\right)-2\right)\sum_{x_3=3}^{\min(x_4,5)} \frac{-\left(x_3\right)^2+11\,x_3-18}{2} =$$

$$\begin{cases} \frac{-\min^3(x_4,5)+15\min^2(x_4,5)-38\min(x_4,5)+24}{6} & \text{if}\,\mathcal{U}\left(\min\left(x_4,5\right)-2\right)=1,\\ 0 & \text{else.}\end{cases}$$

Again we have

$$h' := \frac{-\min^3\left(x_4,5\right)+15\min^2\left(x_4,5\right)-38\min\left(x_4,5\right)+24}{6}.$$

Second part: We label the output variables h, U and the constants determining this part of the polyhedron by $''$.

We add the inequality $-x_3 \leq -6$ to T. As this inequality makes the inequality T_1 redundant we get the simplified system:

$$T'' := \left(\begin{bmatrix}-1 & 0\\ 1 & -1\end{bmatrix} - \begin{pmatrix}-6\\ 0\end{pmatrix}, \leq\right)$$

The projection U'' of T'' is:

$$U''\left(x_4\right) := \left(\left[-1\right]-\left(-6\right),0\right)$$

Assuming that $x_4 \in \mathcal{Z}\left(U''\right)$ we get

$$\left(\int_{\mathcal{P}\left(\mathcal{F}_1 T''(x_4)\right)} \frac{-25+55-18}{2}\,d\gamma\left(x_3\right)\right)\Bigg|_{\mathcal{Z}(U'')} =$$

$$\sum_{x_3=6}^{x_4} \frac{-25+55-18}{2} =$$

$$\mathcal{U}\left(x_4-5\right)\sum_{x_3=6}^{x_4} \frac{-25+55-18}{2} =$$

$$\begin{cases} 6\left(x_4-5\right) & \text{if}\,\mathcal{U}\left(x_4-5\right)=1,\\ 0 & \text{else.}\end{cases}$$

Again we have
$$h'' := 6\,(x_4 - 5)\,.$$

When looking at the two parts of the polyhedron,
$$-x_4 \leq -3 \text{ and } -x_4 \leq -6,$$

we see that this splitting isn't disjoint! Alas, this isn't a specific property of this approach. This will also be with the two other methods that just are more general and complete versions of this one. Moreover we want to remark that the introduction of the min-function is unnecessary if the splitting is already performed at step $k = 2$.

Now, combining h' and h'' we get:
$$\begin{aligned}\mathcal{C}_{S,3}\,(x_4) := {} & \mathcal{U}\,(\min\,(x_4, 5) - 2) \cdot \\ & \quad \frac{-\min^3(x_4, 5) + 15\min^2(x_4, 5) - 38\min(x_4, 5) + 24}{6} + \\ & \mathcal{U}\,(x_4 - 5)\, 6\,(x_4 - 5)\,.\end{aligned}$$

By the application of corollary 4.11 this formula can be transformed into:

$$\begin{aligned}& \mathcal{U}\,(\min\,(x_4, 5) - 2) \cdot \\ & \quad \frac{-\min^3(x_4, 5) + 15\min^2(x_4, 5) - 38\min(x_4, 5) + 24}{6} + \\ & 6\max\,(x_4 - 5, 0)\,.\end{aligned}$$

The above formula is the result of Haghighat's and Polychronopoulos' computation.

Using this formula we derive a piecewise polynomial that represents $\mathcal{C}_{S,3}$ by a further splitting of the polyhedron. We find that
$$\mathcal{C}_{S,3}\,(x_4) = \begin{cases} 0 & \text{if } x_4 \leq 2, \\ \frac{-(x_4)^3 + 15\,(x_4)^2 - 38\,x_4 + 24}{6} & \text{if } 3 \leq x_4 \leq 5, \\ 14 + 6\,(x_4 - 5) & \text{else } (6 \leq x_4). \end{cases}$$

So
$$h = \mathcal{C}_{S,3}\big|_{\{x_4\,|\,3 \leq x_4\}}\,.$$

□

Example 4.15 (Ad example 4.7)
Apart from the first step the computation of the integer volume function is straightforward and can be done as in the previous example. Therefore we will only discuss this step in detail. It corresponds to the last rewriting of the volume function in example 4.7.

We use the same way of presentation as in the previous example.

***k* = 0:**

$$\left(h^0, U^0\right) = (1, S).$$

In contrast to the previous example S_2^5 is triangular but not S.

***k* = 1:**

$$(g, T) = \left(1, U^0\right).$$

Let U^1 such that $\mathcal{P}\left(U^1\right) = \pi_2^3 \mathcal{P}(T)$.

As only T_2^5 is triangular Haghighat and Polychronopoulos apply the Unit Step Calculus already at this moment in order to compute the integral (4.12). Assuming that $(x_2, x_3) \in \mathcal{Z}\left(U^1\right)$ they say:

$$\begin{aligned}
\left.\left(\int_{\mathcal{P}\left(\mathcal{F}_1 T(x_2,x_3)\right)} 1 \, d\gamma(x_1)\right)\right|_{\mathcal{Z}(U^1)} &= \sum_{x_1=1}^{x_2} \mathcal{T}(x_1 + x_2 - 2x_3 \leq 0) \\
&= \sum_{x_1=1}^{x_2} \left(1 - \mathcal{U}(x_1 + x_2 - 2x_3)\right) \\
&= \sum_{x_1=1}^{x_2} 1 - \sum_{x_1=1}^{x_2} \mathcal{U}(x_1 + x_2 - 2x_3).
\end{aligned}$$

Geometrically this operation means that they count the integer points within the polyhedron $\mathcal{P}(S)$ by counting the integer points within a polyhedron that contains $\mathcal{P}(S)$. Afterwards they subtract the number of integer points that lie within the bigger polyhedron but not within $\mathcal{P}(S)$. Formally this can be expressed in the following way:

$$
\begin{aligned}
\mathcal{C}_{S,2}(x_3) &= \int_{\mathcal{P}\left(\mathcal{F}_1^2 S(x_3)\right)} 1\, d\gamma \\
&= \int_{\mathcal{P}\left(\mathcal{F}_1^2\left(S_2^5\right)(x_3)\right)\setminus\mathcal{P}\left(\mathcal{F}_1^2 R(x_3)\right)} 1\, d\gamma \\
&= \int_{\mathcal{P}\left(\mathcal{F}_1^2\left(S_2^5\right)(x_3)\right)} 1\, d\gamma - \int_{\mathcal{P}\left(\mathcal{F}_1^2 R(x_3)\right)} 1\, d\gamma \\
&= \mathcal{C}_{S_2^5,2}(x_3) - \mathcal{C}_{R,2}(x_3),
\end{aligned}
$$

where R equals S with S_1 replaced by $-x_1 - x_2 + 2x_3 < 0$. The above equalities are only valid because the subtrahend is finite!

We want to point out that Haghighat and Polychronopoulos give no reason why they apply the Unit Step Calculus in this way at this moment. But if the reader would try to compute the number of integer points by following their strategy without doing this he would have to sum over an expression of the form $\min(x_2, -x_2 + 2x_3 + 1)$, where the summation variable x_2 is contained in both arguments of the min-function. But they don't mention any technique that can be used to handle such situations where not only one argument of a min- or max-function contains the summation variable.

Moreover we remark that such a substraction operation is not covered by the algorithm Fubini$_{\mathbb{Z}}$. Later we will see that this operation is unnecessary anyway if one avoids the introduction of min- and max-functions.

Applying algorithm Fubini$_{\mathbb{Z}}$ to $\mathcal{C}_{S_2^5,2}(x_3)$ and $\mathcal{C}_{R,2}(x_3)$ separately and then combining both results yields

$$
\mathcal{C}_{S,2}(x_3) = \begin{cases} 0 & \text{if } x_3 \le 0, \\ x_3^2 & \text{else } (x_3 > 0). \end{cases}
$$

□

We summarise the main weaknesses of the Unit Step Calculus method.

1. It is not possible to sum up terms containing the min- and max-functions that have more than one argument that contains the summation variable.

2. The introduction of min- and max-functions isn't necessary at all.

3. The projection of a polyhedron that is defined by a triangular system of inequalities of depth k to $\pi_i^n(\mathbb{R}^n)$, where $i \le k \le n$, might not be

representable by a triangular system of inequalities. Thus the method can get stuck.

4. We must integrate w.r.t. a fixed variable order imposed by the structure of the input.

5. Rational coefficients that might occur when computing the lower and upper bound for summation over a certain variable cannot be handled.

In the next section we will see that the problem of point 3 can be solved by a certain decomposition of the polyhedron. This will also eliminate the weaknesses of points 1 and 2. But before passing on let's verify in an example that Fourier-Motzkin projection is not closed on the set of inputs to problem 4.3, Nearly triangular. This means that Haghighat's and Polychronopoulos' method can get stuck as already mentioned.

Example 4.16
Consider S given by

$$\left(\begin{bmatrix} 1 & -1 & -1 & +1 \\ -1 & 1 & -1 & +1 \\ 0 & 1 & +1 & +1 \\ 0 & -1 & +1 & +1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix} - \begin{pmatrix} 5 \\ -8 \\ -2 \\ 9 \\ 1 \\ -10 \end{pmatrix}, \leq \right).$$

Obviously S is triangular of depth 3. Eliminating the first variable with Fourier-Motzkin projection yields

$$\left(\begin{bmatrix} 0 & -2 & +2 \\ 1 & +1 & +1 \\ -1 & +1 & +1 \\ 0 & 1 & -1 \\ 0 & -1 & 0 \end{bmatrix} - \begin{pmatrix} -3 \\ -2 \\ 9 \\ 1 \\ -10 \end{pmatrix}, \leq \right).$$

We see that this projection is not triangular any more but still a valid input for problem 4.3, Nearly triangular. Thus let's again eliminate the first variable. We get

$$\left(\begin{bmatrix} -2 & +2 \\ 2 & +2 \\ 1 & -1 \\ -1 & 0 \end{bmatrix} - \begin{pmatrix} -3 \\ 7 \\ 1 \\ -10 \end{pmatrix}, \leq \right).$$

We see that already after two projections the system of linear inequalities isn't a valid input any more!

□

The disadvantage of point 4 will also vanish in section 4.3, Fubini and Decomposition Combined, when we merge the decomposition of the next section with Fubini's theorem.

The problem of point 5 is illustrated in the following example. It will be solved in the last section, Fubini and Decomposition Combined, by applying the techniques of Pugh's disjoint splintering (cf. page 25) and section 2.6, Flat Polyhedra.

Example 4.17
Take the system S

$$\begin{aligned} 3x_1 + 4x_2 &\leq 7 \\ -5x_1 + x_2 &\leq -1, \end{aligned}$$

for instance. This is equivalent to

$$\frac{1}{5}x_2 + \frac{1}{5} \leq x_1 \leq -\frac{4}{3}x_2 + \frac{7}{3}.$$

By applying algorithm Fubini$_{\mathbb{Z}}$ and taking in account lemma 2.26 we have to compute

$$\sum_{x_1=\left\lceil \frac{1}{5}x_2+\frac{1}{5}\right\rceil}^{\left\lfloor -\frac{4}{3}x_2+\frac{7}{3}\right\rfloor} 1.$$

The ceiling and the floor functions in the bounds of the sum are a consequence of the rational coefficients in the linear terms. We can't guarantee any more that the linear terms are integers when evaluated on integer parameters.

The above sum can be represented by the following formula

$$\left\lfloor -\frac{4}{3}x_2 + \frac{7}{3}\right\rfloor - \left\lceil \frac{1}{5}x_2 + \frac{1}{5}\right\rceil + 1.$$

If we still want to sum over x_2 we are stuck at this moment. Therefore we must find a way to remove the floor and ceiling functions.

□

Apart from the above we want to point out that one could also compute the Lebesgue volume for the previous examples in the same way as it is done for the integer volume. One just would have to apply algorithm Fubini$_2$ instead, i.e. replace the sums by Lebesgue integrals.

4.2 Regular Decomposition

Tawbi (cf. [Taw95]) overcomes the problems of points 1 to 3 on page 79 by disjointly decomposing the given polyhedron. Each splinter and the projection of each splinter can be represented by a triangular system of linear inequalities. The latter is made precise in the following definition.

Definition 4.18 (Triangular projection)
Let $S \in \mathcal{LS}(n)$ be triangular of depth k. Then S is said to have a triangular projection down to depth k iff

$$\mathcal{LOW}_i(S) \propto \mathcal{UPP}_i(S) \tag{4.13}$$

is redundant in T^i, where

$$\mathcal{P}\left(T^i\right) = \pi_{i+1}^{n} \mathcal{P}(S),$$

for $i \leq k$.

$\square$

Note that for such systems the Fourier-Motzkin projection can be computed very easily. The property of having a triangular projection implies

$$\mathcal{P}\left({}_{i+1}^{n}S_{2i+1}^{\|S\|}\right) = \pi_{i+1}^{n} \mathcal{P}(S), \tag{4.14}$$

for $i \leq k$. As ${}_{i+1}^{n}S_{2i+1}^{\|S\|}$ is triangular, for $i \leq k$, this also implies that the projection of a system having a triangular projection, for $i < k$, really can be represented by a triangular system of inequalities!

We illustrate the above definition in an example.

Example 4.19
Consider S given by

$$\left(\begin{bmatrix} 1 & -1 & -\frac{1}{2} & +\frac{1}{2} \\ -1 & 1 & -\frac{1}{2} & +\frac{1}{2} \\ 0 & 1 & +1 & +1 \\ 0 & -1 & +1 & +1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix} - \begin{pmatrix} 7 \\ -6 \\ -2 \\ 9 \\ 1 \\ -10 \end{pmatrix}, \leq \right).$$

Obviously S is triangular of depth 3. It has a triangular projection down to depth 1, too. For eliminating the first variable with Fourier-Motzkin projection yields

$$\left(\begin{bmatrix} 0 & -1 & +1 \\ 1 & +1 & +1 \\ -1 & +1 & +1 \\ 0 & 1 & -1 \\ 0 & -1 & 0 \end{bmatrix} - \begin{pmatrix} -1 \\ -2 \\ 9 \\ 1 \\ -10 \end{pmatrix}, \leq \right).$$

We see that the first line that is identical with line 4.13 in definition 4.18 is redundant for the rest of the linear system.

□

Before describing how this decomposition can be achieved computationally let's see how we could decompose a polygon into several disjoint triangular ones graphically. At this point we won't require the resulting splinters to have triangular projections.

Example 4.20
Take the linear system S given by

$$\left(\begin{bmatrix} -1 & -1 \\ 1 & -1 \\ 1 & 1 \\ -1 & 1 \end{bmatrix} - \begin{pmatrix} 0 \\ 2 \\ 3 \\ 0 \end{pmatrix}, \leq \right), \tag{4.15}$$

for instance. Geometrically its solution set is a quadrilateral drawn in figure 4.1. One can see that each inequality forming the previous system of linear inequalities corresponds to an edge of this quadrilateral. Alas, the above system isn't triangular of depth 1. Graphically this means that the quadrilateral is bounded by *four* non horizontal lines.

Now Tawbi splits the plane along horizontal lines passing through each vertex. Obviously this also means decomposing the polygon. These lines are the dotted lines in figure 4.20.

Now each part of the polygon is bounded by at most *two* non horizontal lines. This means that the corresponding systems are triangular of depth 1. Take the system of the middle splinter, for instance. It can be described by the system

$$\left(\begin{bmatrix} -1 & -1 \\ 1 & -1 \\ 0 & -2 \\ 0 & 2 \end{bmatrix} - \begin{pmatrix} 0 \\ 2 \\ 0 \\ 1 \end{pmatrix}, \leq \right). \tag{4.16}$$

Note that lines three and four in the system 4.16 are the results of computing $\preceq$ of lines one, four and three, two, resp., in the system 4.15.

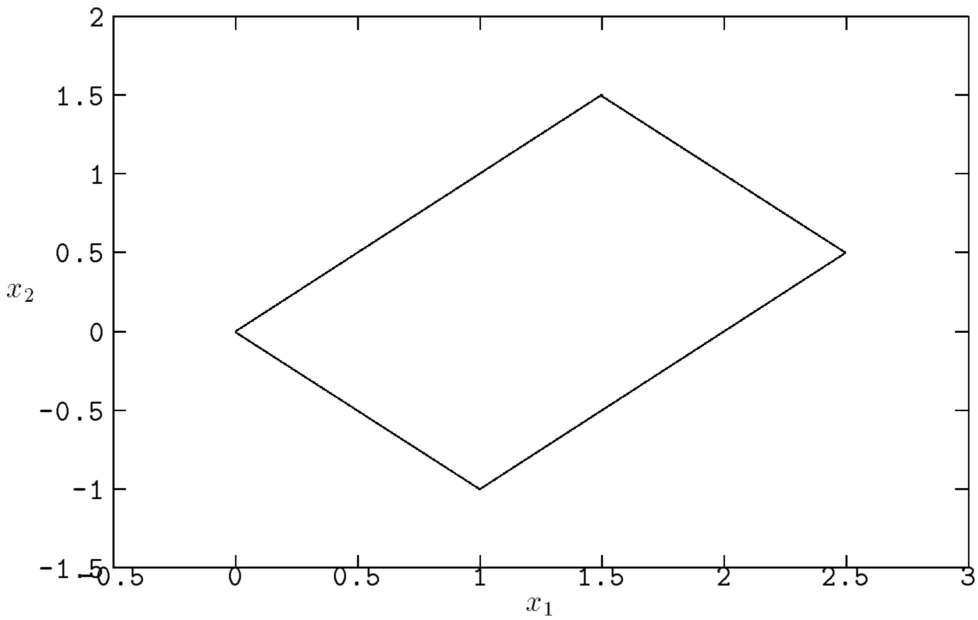

Figure 4.1: Quadrilateral

$\square$

The following theorem lies the basis for the above decomposition[4]. It transforms the intersection of interval like sets into a disjoint union of such sets.

Theorem 4.21
Let $s_i \in \mathbb{R}, \lessdot_i \in \mathbb{L}$.

1. *Let* X_i *be the set of numbers* r *having the properties*

$$
\begin{aligned}
& s_i \lessdot_i r, \\
\text{for } 1 \le l \le i-1, \quad & s_l \begin{cases} < s_i & \text{if } \lessdot_i = \le, \\ \le s_i & \text{else } (\lessdot_i = <), \end{cases} \\
\text{for } i+1 \le l \le n, \quad & s_l \begin{cases} \le s_i & \text{if } \lessdot_i = \le, \\ < s_i & \text{else } (\lessdot_i = <), \end{cases}
\end{aligned}
$$

   for $i = 1, \ldots, m$. *Then we have*

$$\bigcap_{i=1}^{m} \{r \mid s_i \lessdot_i r\} = \bigcup_{i=1}^{m} X_i$$

[4]We want to remark that our terminology and presentation of the splitting algorithm differs from Tawbi's very much. But the reader might note easily that our presentation is a completion of Tawbi's one and her ideas are completely covered by our description.

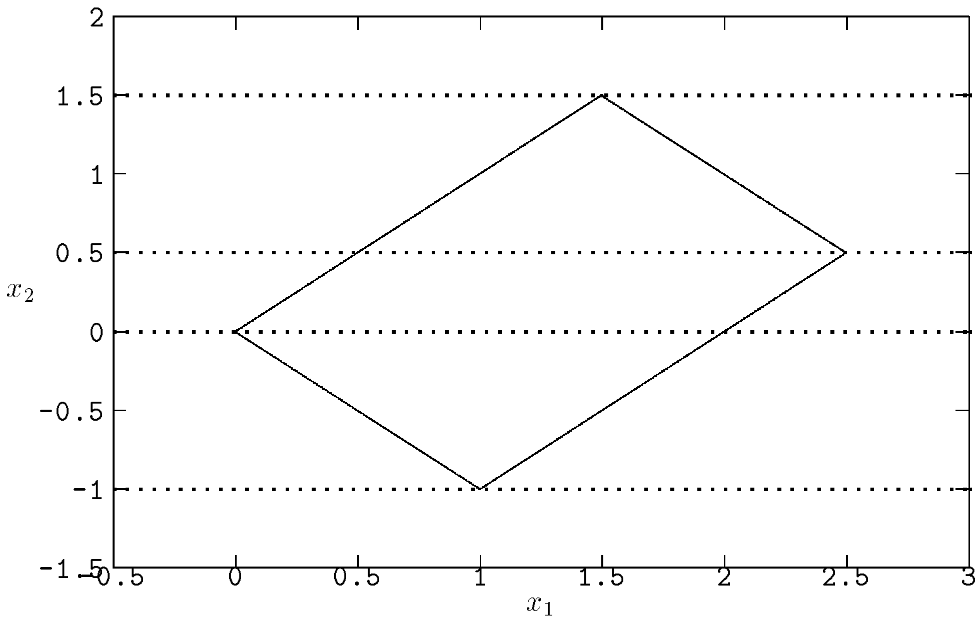

Figure 4.2: Decomposition

and the X_i are pairwise disjoint.

2. *Let Y_j be the set of numbers r having the properties*

$$r \lhd_j t_j, \tag{4.17}$$

$$\text{for } 1 \le l \le j-1, \quad t_j \begin{cases} < t_l & \text{if } \lhd_j = \le, \\ \le t_l & \text{else } (\lhd_j = <), \end{cases} \tag{4.18}$$

$$\text{for } j+1 \le l \le n, \quad t_j \begin{cases} \le t_l & \text{if } \lhd_j = \le, \\ < t_l & \text{else } (\lhd_j = <), \end{cases} \tag{4.19}$$

for $j = 1, \ldots, n$. Then we have

$$\bigcap_{j=1}^{n} \{r \mid r \lhd_j t_j\} = \bigcup_{j=1}^{n} Y_j$$

and the Y_j are pairwise disjoint.

3. *Hence combining points 1 and 2 we get*

$$\bigcap_{i=1}^{m} \bigcap_{j=1}^{n} \{r \mid s_i \lessdot_i r \lhd_j t_j\} = \bigcup_{i=1}^{n} \bigcup_{j=1}^{n} (X_i \cap Y_j),$$

for pairwise disjoint $X_i \cap Y_j$.

□

Proof:

We proof part 2 of the theorem. Part 1 can be shown similarly. Point 3 is a consequence of 1 and 2.

1. Let

$$M := \left\{ k \mid t_k = \min_{i=1}^{n} \{t_i\} \right\} \neq \emptyset,$$
$$N := \{ i \in M \mid \lhd_i = < \},$$

   and

$$k \in \begin{cases} N & \text{if } N \neq \emptyset, \\ M & \text{else.} \end{cases}$$

   Note that

$$\bigcap_{i=1}^{n} \{ r \mid r \lhd_i t_i \} = \{ r \mid r \lhd_k t_k \}.$$

   In particular if we choose

$$k := \begin{cases} \max(N) & \text{if } N \neq \emptyset, \\ \min(M) & \text{else} \end{cases}$$

   we have

$$\bigcap_{i=1}^{n} \{ r \mid r \lhd_i t_i \} = \{ r \mid r \lhd_k t_k \} = Y_k.$$

2. Assume that $r \in Y_j \cap Y_k$, for some $k < j$.

   (a) Consider $\lhd_j = \leq$. As $r \in Y_j$ and because of condition 4.18 we know that

$$t_j < t_k.$$

   But as $r \in Y_k$ condition 4.19 also tells us that $t_k \leq t_j$.

   (b) Thus assume $\lhd_j = <$. Now condition 4.18 implies that $t_j \leq t_k$. Moreover we know that $t_k < t_j$ because of line 4.19. So this leads to a contradiction, too.

□

Now we are ready to formulate Tawbi's[5] decomposition algorithm.

Algorithm 4.22

$Z \leftarrow \mathrm{Split}(S, k, n)$

Input:

S .... *a linear system[6] of inequalities in n variables*

k .... *the number of the last variable to whose respect the system is splitted*

n .... *the number of all occurring variables*

Output:

Z .... *a set of systems of inequalities having a triangular projection down to depth k, such that*

1. *the $\mathcal{P}(T)$, where $T \in Z$, are disjoint,*
2. *$\mathcal{P}(S) = \bigcup_{T \in T} \mathcal{P}(T)$.*

[1] Recursion base.

If $k = 0$ **then**

$Z \leftarrow \{S\}$

Return

[2] Recursion body.

$Z \leftarrow \emptyset$

For V **in** $\mathrm{Split}(k-1, n, S)$

[2.1] Restrictions on x_1.

$N \leftarrow \mathcal{NOT}_1(V) \circ \pi_{\overline{(1)}}$

$L \leftarrow \mathcal{LOW}_1(V) \circ \pi_{\overline{(1)}}$

$U \leftarrow \mathcal{UPP}_1(V) \circ \pi_{\overline{(1)}}$

[2.2] Decomposition.

For (i, j) **in** $\{1, \ldots, \|L\|\} \times \{1, \ldots, \|U\|\}$

[2.2.1] Add inequalities not involving x_1.

$T \leftarrow N$

---

[5]Tawbi doesn't care about the **else** cases in the algorithm. We have to add them in order to complete it.

Moreover the input doesn't contain strict comparisons according to Tawbi. But if the algorithm is completed the input can also contain strict ones.

[6]Tawbi assumes that S is triangular. We can't assume this because the algorithm will be applied recursively to non triangular systems of linear inequalities.

[2.2.2] Add bounds on x_1.

(4.20) $T \leftarrow T \uplus L_i \propto (\pi_1, \leq)$

(4.21) $T \leftarrow T \uplus (\pi_1, \leq) \propto U_j$

[2.2.3] L_i is maximum of L.

For $l = 1, \;\; i-1$
  If Type $(L_i) = \leq$ **then**
    $R \leftarrow L_l \prec L_i$
  else
    $R \leftarrow L_l \preceq L_i$
  $T \leftarrow T \uplus R$
For $l = i+1, \;\; \|L\|$
  If Type $(L_l) = \leq$ **then**
    $R \leftarrow L_l \preceq L_i$
  else
    $R \leftarrow L_l \prec L_i$
  $T \leftarrow T \uplus R$

[2.2.4] U_j is minimum of U.

For $l = 1, \;\; j-1$
  If Type $(U_j) = \leq$ **then**
    $R \leftarrow U_j \prec U_l$
  else
    $R \leftarrow U_j \preceq U_l$
  $T \leftarrow T \uplus R$
For $l = j+1, \;\; \|U\|$
  If Type $(U_l) = \leq$ **then**
    $R \leftarrow U_j \preceq U_l$
  else
    $R \leftarrow U_j \prec U_l$
  $T \leftarrow T \uplus R$

[2.2.5] Projection.

$R \leftarrow L_i \propto U_j$
$T \leftarrow T \uplus R$

[2.2.6] Add result.

$Z \leftarrow Z \cup \{T\}$

[2.3] Check if unbounded[7] *polyhedron.*

If $\|L\| = 0$ or $\|U\|$=0 **then**

Iterate the loop body either for j or for i

but leave out either line 4.20 or line 4.21.

Return

□

Corollary 4.23
Algorithm 4.22, Split, *is correct.*

□

Proof:

Disjoint splitting: Obviously the polyhedron is splitted disjointly because at the i-th recursion theorem 4.21 is applied, for $i \leq k$. The s_i and t_i in theorem 4.21 are the lower bounds and upper bounds, resp., on the i-th (variable) component.

Triangular projection down to depth k: In order to ensure that the inequalities that are generated by Fourier-Motzkin projection are redundant in each splinter they are added at step [2.2.5] to T.

This doesn't change $\mathcal{P}(T)$ because the system, i.e. single inequality, generated at step [2.2.5] is redundant in T because of the inequalities added to T at step [2.2.2].

Termination: Obvious.

□

Alas, there is no precise complexity analysis of the previous algorithm. But Tawbi computes an upper bound for the number of inequalities generated if the input system S is triangular and if $k = n$. It is $\frac{(n+1!)^2}{2^n}$.

Now let's see in an example what the algorithm Split does.

Example 4.24 (Ad example 4.20)
We get

$$\{T^1, T^2, T^3, T^4\} = \text{Split}(S, 1, 3) \ ,$$

[7] This step can't be found in Tawbi's algorithm because Tawbi doesn't allow unbounded polyhedra as input. We add this step because $\|L\| = 0$ or $\|U\| = 0$ implies that the input is unbounded obviously. (The reversion of this implication which is also true (cf. [Taw95]) is not used in the correctness proof of this algorithm!)

where

$$T^1 := \left(\begin{bmatrix} -1 & -1 \\ 1 & -1 \\ 0 & 2 \\ 0 & 2 \\ 0 & -2 \end{bmatrix} - \begin{pmatrix} 0 \\ 2 \\ 0 \\ 1 \\ 2 \end{pmatrix}, \begin{matrix} \leq \\ \leq \\ \leq \\ \leq \\ \leq \end{matrix} \right),$$

$$T^2 := \left(\begin{bmatrix} -1 & -1 \\ 1 & -1 \\ 0 & 2 \\ 0 & 2 \\ 0 & 0 \end{bmatrix} - \begin{pmatrix} 0 \\ 3 \\ 0 \\ -1 \\ 2 \end{pmatrix}, \begin{matrix} \leq \\ \leq \\ \leq \\ < \\ \leq \end{matrix} \right),$$

$$T^3 := \left(\begin{bmatrix} -1 & 1 \\ 1 & -1 \\ 0 & -2 \\ 0 & 2 \\ 0 & -2 \end{bmatrix} - \begin{pmatrix} 0 \\ 2 \\ 0 \\ 1 \\ 2 \end{pmatrix}, \begin{matrix} \leq \\ \leq \\ < \\ \leq \\ \leq \end{matrix} \right),$$

$$T^4 := \left(\begin{bmatrix} -1 & 1 \\ 1 & -1 \\ 0 & -2 \\ 0 & -2 \\ 0 & 2 \end{bmatrix} - \begin{pmatrix} 0 \\ 3 \\ 0 \\ -1 \\ 3 \end{pmatrix}, \begin{matrix} \leq \\ \leq \\ < \\ < \\ \leq \end{matrix} \right).$$

Note that T^2 is inconsistent and that T^3 basically corresponds to the middle splinter in figure 4.2.

□

Now in the case of $\boldsymbol{\mu} = \boldsymbol{\lambda}$ the algorithm Fubini$_2$ can be applied directly to each splinter in the output of algorithm Split. The occurring projections can be computed by means of formula 4.14. Moreover note that definition 4.18 also implies that the case where the Lebesgue integral is zero can't occur any more in lemma 2.22.

On the other hand in the case of $\boldsymbol{\mu} = \boldsymbol{\gamma}$ the property of having a triangular projection seems to be too weak in order to ensure that the γ-integral isn't identical to zero (cf. lemma 2.26). In fact we have to ensure that the result of the splitting algorithm satisfies the property of being regular[8] additionally.

[8]The reader already might have wondered why we call this section "*Regular* Decomposition". The reason is the definition 4.25. As Tawbi is only considering γ-integration in [Taw95] she starts from the definition of regular. We present the definition of regular at the end of this section because we want to make clear that this property is only needed in the case of γ-integration.

Definition 4.25 (Regular [Taw95]**)**
Let $S \in \mathcal{LS}(n)$ be triangular of depth k and

$$(A_i, \lessdot_i) := \mathcal{LOW}_i(S)_1,$$
$$(B_i, \lhd_i) := \mathcal{UPP}_i(S)_1.$$

Then S is said to be regular [9] *of depth k iff*

$$\lceil s \rceil \leq \lfloor t \rfloor + 1, \text{ if } \lessdot_i = \leq \text{ and } \lhd_i = \leq, \tag{4.22}$$
$$\lceil s \rceil \leq \lceil t \rceil, \quad \text{ if } \lessdot_i = \leq \text{ and } \lhd_i = <, \tag{4.23}$$
$$\lfloor s \rfloor + 1 \leq \lfloor t \rfloor + 1, \text{ if } \lessdot_i = < \text{ and } \lhd_i = \leq, \tag{4.24}$$
$$\lfloor s \rfloor + 1 \leq \lceil t \rceil, \quad \text{ if } \lessdot_i = < \text{ and } \lhd_i = <, \tag{4.25}$$

for

$$s := z_1, \; \vec{z} := A_i \vec{y},$$
$$t := z_1, \; \vec{z} := B_i \vec{y}$$

and for all $\vec{y} \in \pi^n_{i+1} \mathcal{P}(S)$ and $i \leq k$.

□

The above definition means that the smallest integer v being greater than the lower bound imposed on the i-th component is less, equal or greater at most by 1 than the greatest integer w being smaller than the corresponding upper bound, for all i. In these cases the difference of the right hand and the left hand side of each of the inequalities 4.22-4.25 is exactly the number of integers being located between the left hand and the right hand side.

We illustrate the definition of regular in the following example.

Example 4.26
First we give instances of the three possible cases of regular linear systems of inequalities. The last system is not regular.

1. Let $\frac{1}{2} \leq x \leq \frac{5}{2}$. Then $\lfloor \frac{5}{2} \rfloor - \lceil \frac{1}{2} \rceil + 1 = 2 \geq 0$ and $v = 1, w = 2$. Here we have $v < w$.

2. Let $\frac{1}{2} \leq x \leq \frac{3}{2}$. Then $\lfloor \frac{3}{2} \rfloor - \lceil \frac{1}{2} \rceil + 1 = 1 \geq 0$ and $v = 1, w = 1$. Here we have $v = w$.

3. Let $\frac{3}{4} \leq x \leq \frac{1}{2}$. Then $\lfloor \frac{1}{2} \rfloor - \lceil \frac{3}{4} \rceil + 1 = 0 \geq 0$ and $v = 1, w = 0$. Here we have $l > u$ and $v = w + 1$.

[9] Tawbi only cares about the case of line 4.22. But in order to give the notion of regular its intended meaning we also consider the cases of lines 4.23, 4.24, 4.25.

4. Let $\frac{3}{2} \leq x \leq \frac{1}{2}$. Then $\lfloor \frac{1}{2} \rfloor - \lceil \frac{3}{2} \rceil + 1 = -1 < 0$ and $v = 2, w = 0$. This linear system is not regular. We have $v > w$ and $v = w + 2$.

□

Now we state a sufficient criterion for regularity.

Lemma 4.27
Let $r, s, t \in \mathbb{R}$ such that

$$s \lessdot r \lhd t.$$

Then

$$\begin{aligned}
\lceil s \rceil &\leq \lfloor t \rfloor + 1, && \text{if } \lessdot = \leq \text{ and } \lhd = \leq, \\
\lceil s \rceil &\leq \lceil t \rceil, && \text{if } \lessdot = \leq \text{ and } \lhd = <, \qquad (4.26)\\
\lfloor s \rfloor + 1 &\leq \lfloor t \rfloor + 1, && \text{if } \lessdot = < \text{ and } \lhd = \leq, \qquad (4.27)\\
\lfloor s \rfloor + 1 &\leq \lceil t \rceil, && \text{if } \lessdot = < \text{ and } \lhd = <.
\end{aligned}$$

□

Proof:

The cases of lines 4.26 and 4.27 are obvious. We only must show the two other ones.

Case $\lessdot = \leq$ and $\lhd = \leq$**:** Note that $s \leq t$ implies $\lfloor s \rfloor \leq \lfloor t \rfloor$. We also know that $\lceil s \rceil \leq \lfloor s \rfloor + 1$. Therefore

$$\lfloor t \rfloor - \lceil s \rceil + 1 \geq \lfloor s \rfloor - \lfloor s \rfloor - 1 + 1 = 0.$$

Case $\lessdot = <$ and $\lhd = <$**:** Assume

1. $t \in \mathbb{Z}$. Then $s < t$ implies $\lfloor s \rfloor < t = \lceil t \rceil$. Hence $\lfloor s \rfloor + 1 \leq \lceil t \rceil$.
2. $t \notin \mathbb{Z}$. Then $s < t$ implies $\lfloor s \rfloor \leq \lfloor t \rfloor < \lfloor t \rfloor + 1 = \lceil t \rceil$. Hence $\lfloor s \rfloor + 1 \leq \lceil t \rceil$.

□

The last lemma just says that we can also use the decomposition algorithm of this section for the problem of the integer volume in the same handy way as for the real volume[10].

[10]Note that we do not yet know how to avoid the floor and ceiling functions from point 5 on page 80.

There is one refinement of the algorithm Split that we still have to point out.

Before adding the newly generated inequality R to T in the algorithm Tawbi checks if R is so-called "trivially false". If this is the case one needs not go on decomposing. Because the resulting splinter is empty in any case.

She considers the following linear systems of inequalities as trivially false.

Definition 4.28 (Trivially false)
The system $(A, \lessdot)$ is trivially false iff

$$A - A(0) = 0 \text{ and } \neg\left(A(0) \lessdot 0\right).$$

□

We don't include this short cut into the algorithm because later in chapter 5, Improvements, we will show more powerful ways to speed up the computation which contain the previous one as a special case.

4.3 Fubini and Decomposition Combined

Here we present an algorithm that solves the problems 3.1, Lebesgue volume, and 3.2, Integer volume, completely. Its specialisation w.r.t. the integer measure is originated by Pugh (cf. [Pug94]) and it was published in 1994.

Pugh combines the algorithm $\text{Fubini}_{\mathbb{Z}}$, Fubini_2, resp., and Tawbi's regular decomposition[11].

1. By means of this combination it is possible to integrate w.r.t. any order of variables we want. We can choose the component w.r.t. which we want to integrate during the application of the algorithm. This solves point 4 on page 80.

2. He also states a strategy how to overcome the problem of the floor and ceiling functions (cf. point 5 on page 80).

The first point is the easier one. So let's start with this one.

First of all we know that we needn't fix the order of integration in the algorithms Fubini_2 and $\text{Fubini}_{\mathbb{Z}}$ because of Fubini's theorem.

We need not fix the order of splitting in the algorithm Split either. We might formulate this algorithm indeterministically such that steps [2.2.1] to [2.2.5] are applied to all components out of the set $\{1, \ldots, k\}$ but in an arbitrary order. If we do this the output will only have triangular projections up to a rotation of the variable space.

[11]As in Tawbi's paper strict inequalities aren't treated correctly in [Pug94].

Now we combine both algorithms. This means that in each recursion of the algorithm Split we also integrate w.r.t. to the component to whose respect the system of inequalities is splitted. Instead of computing systems that correspond to the disjoint splinters of the polyhedron we only compute systems that correspond to their projections. This is just the information that we need in the algorithms Fubini_2 and $\text{Fubini}_{\mathbb{Z}}$. Because of line 4.14 computing the projection only means that we can leave out step [2.2.2] in algorithm Split.

Having the previous facts in mind we formulate[12] the algorithm Volume. In this final statement of this algorithm we also make assumptions about the appearance of the representation of the output. Obviously this is highly dependent on properties of the measure μ. So for each different measure the algorithm still must be refined in some way. We will show this after the statement of the algorithm for $\mu \in \{\lambda, \gamma\}$.

In this sense lines 4.30, 4.28 and 4.31 are to be understood.

Algorithm 4.29

$$(Z, p) \quad \leftarrow \text{Volume}\big(k, p, n, S\big)$$

Input:

k .... *the index of the last summation variable of* S.

p .... *index of the first parameter of* S, $p > k$.

n .... $S \in \mathcal{LS}\,(n)$.

S .... *a linear system of inequalities*

Output:

Z .... *a set of quadruples* $\Big(P, T, \vec{A} - \vec{a}, X\Big)$, *where*

- $P \in \mathbb{Q}[x_1, \ldots, x_{n-k}]$
- T *is a linear system of inequalities in* $n - k$ *variables*
- $\vec{A}$ *is a matrix having* $n - p$ *rows*
- $X \subseteq \mathbb{R}^{n-p}$

such that the sets $\mathcal{P}\,(T)$ *are disjoint and*[13]

$$\mathcal{V}_{\mu,S,k} = \int_{\mathbb{R}^{p-k-1}} \sum_{\left(P,T,\vec{A}-\vec{a},X\right)\in Z} \mathbf{1}_{\left(\vec{A}\,\mathcal{P}(T)-\vec{a}\right)\cap X} P \, d\mu\,(x_1, \ldots, x_{p-k-1})\,.$$

[12] Our presentation of the algorithm Volume is quite different to Pugh's one. But the reader can easily see that they are equivalent.

[13] Note that this formula only makes sense computationally if $\vec{A} - \vec{a}$ really has a left inverse!

p .... *index of the first parameter in the result.*

[1] Recursion base.

If $k = 0$ **then**

(4.28) Initialise $\vec{A} - \vec{a}, X$ as required by the measure μ.

$Z \leftarrow \left\{\left(1, S, \vec{A} - \vec{a}, X\right)\right\}$

Return

[2] Recursion body.

$(Z, p) \quad \leftarrow \text{Volume}\big(k-1, p, n, S\big)$

For $\left(Q, V, \vec{A} - \vec{a}, X\right)$ **in** Z

[2.1] Chose elimination variable.

$r \leftarrow$ *arbitrarily chosen from* $\{1, \ldots, p-1\}$

[2.2] Restrictions on x_r.

$N \leftarrow \mathcal{NOT}_r\,(V)$

$L \leftarrow \mathcal{LOW}_r\,(V)$

$U \leftarrow \mathcal{UPP}_r\,(V)$

[2.3] Decomposition.

For (i, j) **in** $\{1, \ldots, \|L\|\} \times \{1, \ldots, \|U\|\}$

[2.3.1] Add inequalities not involving x_r.

$T \leftarrow N$

[2.3.2] L_i *is maximum of* L.

For $l = 1,\ \ i-1$

If $\text{Type}\,(L_i) = \leq$ **then**

$R \leftarrow L_l \prec L_i$

else

$R \leftarrow L_l \preceq L_i$

$T \leftarrow T \uplus R$

For $l = i+1,\ \ \|L\|$

If $\text{Type}\,(L_l) = \leq$ **then**

$R \leftarrow L_l \preceq L_i$

else

$R \leftarrow L_l \prec L_i$

$T \leftarrow T \uplus R$

[2.3.3] U_j *is minimum of* U.

For $l = 1, \;\; j-1$

If Type $(U_j) = \leq$ **then**

$R \leftarrow U_j \prec U_l$

else

$R \leftarrow U_j \preceq U_l$

$T \leftarrow T \uplus R$

For $l = j+1, \;\; \|U\|$

If Type $(U_l) = \leq$ **then**

$R \leftarrow U_j \preceq U_l$

else

$R \leftarrow U_j \prec U_l$

$T \leftarrow T \uplus R$

[2.3.4] Projection.

$R \leftarrow L_i \propto U_j$

$T \leftarrow T \uplus R$

[2.3.5] Integration.

$\breve{s} \leftarrow L_i \vec{x}$

$\breve{t} \leftarrow U_j \vec{x}$

(4.29) $P \leftarrow \int_{s_1}^{t_1} Q \, d\mu\,(x_r)$

(4.30) Modify $T, \vec{A} - \vec{a}, X$ as required by the measure μ.

[2.3.6] Add result.

$Z \leftarrow Z \cup \left\{ \left(P, T, \vec{A} - \vec{a}, X \right) \right\}$

[2.4] Check if unbounded polyhedron.

If $\|L\| = 0$ or $\|U\| = 0$ **then**

(4.31) Modify $\vec{A} - \vec{a}, X$ as required by the measure μ.

$Z \leftarrow Z \cup \left\{ \left(\infty, N, \vec{A} - \vec{a}, X \right) \right\}$

$p \leftarrow p - 1$

Return

$\square$

Corollary 4.30
Algorithm 4.29, Volume, *is correct in the case of* $\mu = \lambda$.

In the specification of the algorithm $\vec{A} - \vec{a}$ *is the identity function on* $\mathbb{R}^{n-k}$ *and* $X = \mathbb{R}^{n-k}$. *In lines 4.30, 4.28 and 4.31 nothing needs to be done.*

□

Proof:

The partial correctness is an obvious consequence of the previous discussion. The termination is obvious.

□

Thus the result of $\text{Volume}\big(k, k+1, n, S\big)$ is a representation of $\mathcal{V}_{\lambda,S,k}$. We see that it is a piecewise polynomial.

In order to apply the algorithm Volume in the case of $\mu = \gamma$ we must specialise lines 4.30 and 4.28 such the problem of point 5 on page 80 concerning the floor and ceiling functions is solved.

Before formulating the solution strategy in general let's have a look at the following example.

Example 4.31 (Ad example 4.17)
Let's treat x_2 as it were a parameter.

First let's apply Pugh's disjoint splintering on page 25. For example, let's compute the sum if we assume that $x_2 \in \mathbb{Z}$ and

$$\begin{aligned} -\frac{4}{3}x_2 + \frac{7}{3} &\in \mathbb{Z}, \\ \frac{1}{5}x_2 + \frac{1}{5} &\in \mathbb{Z}, \end{aligned}$$

i.e. this splinter is the one corresponding to $i, j = 0$ on page 25. The above is equivalent to the following condition.

$$\begin{aligned} -4x_2 + 7 &= 3y_1, \\ x_2 + 1 &= 5y_2, \text{ for some } y_1, y_2 \in \mathbb{Z}. \end{aligned}$$

Equivalently we have

$$\begin{bmatrix} -3 & 0 & -4 \\ 0 & -5 & 1 \end{bmatrix} \begin{pmatrix} y_1 \\ y_2 \\ x_2 \end{pmatrix} - \begin{pmatrix} -7 \\ -1 \end{pmatrix} = 0, \text{ for some } y_1, y_2 \in \mathbb{Z}.$$

Note that this equations mean that we want to count the integer points of

$$\pi_{\overline{(1,\ldots,2)}}\left(\mathcal{Z}\left(R \circ \pi_{\overline{(1,\ldots,2)}}\right) \cap \mathcal{Z}\left(T\right)\right),$$

where $\mathcal{P}(T)$ is the hyperplane given as the solution set of the above system of equations and R is given by the inequality

$$\frac{1}{5}x_2 + \frac{1}{5} \leq -\frac{4}{3}x_2 + \frac{7}{3}.$$

This inequality is the projection of the system S as computed by the volume algorithm. Therefore these formulae are instances of those ones presented in section 2.6, Flat Polyhedra.

Next we parametrise the integer points in $\mathcal{P}(T)$. They are given by

$$\begin{bmatrix} -20 \\ 3 \\ 15 \end{bmatrix} z_1 + \begin{pmatrix} 77 \\ -11 \\ -56 \end{pmatrix},$$

where $z_1 \in \mathbb{Z}$. Observe that this means $z_1 = \frac{1}{15}x_2 + \frac{56}{15}$. Thus we have to restrict the set of x_2 satisfying R by the condition $\frac{1}{15}x_2 + \frac{56}{15} \in \mathbb{Z}$ (compare also section, Flat Polyhedra). Now we can evaluate

$$\left\lfloor -\frac{4}{3}x_2 + \frac{7}{3} \right\rfloor - \left\lceil \frac{1}{5}x_2 + \frac{1}{5} \right\rceil + 1.$$

This equals

$$17z_1 + 89,$$

where

$$23z_1 \leq 88.$$

The latter system of inequalities is only substituted $15z_1 - 56$ for x_2 in R. Summarising we can represent the integer volume of the single splinter considered here by the following pieces of information.

1. It equals $17z_1 + 89$.
2. The variable $z_1 \in \mathbb{Z}$ satisfies $23z_1 \leq 88$.
3. From the original variable x_2 the new variable z_1 can be computed using the formula $z_1 = \frac{1}{15}x_2 + \frac{56}{15}$.
4. Additionally we must assume $\frac{1}{15}x_2 + \frac{56}{15} \in \mathbb{Z}$.

We encourage the reader to compare this result to the general one in section 2.6, Flat Polyhedra. This is exactly the same information as summarised in line 2.18.

For the other splinters the integer volume can be computed similarly.

Now let's assume x_2 isn't a parameter of the original input of some application of the volume algorithm but was obtained in the same way as z_1 by disjoint splintering and summation. Then x_2 must satisfy some conditions similar to those ones of z_1. Take the following conditions for, instance.

1. We have $x_2 = \frac{2}{11}y_1 - 6$ and $y_1 \in \mathbb{Z}$.
2. Moreover y_1 must be such that $\frac{2}{11}y_1 - 6 \in \mathbb{Z}$.

Then we conclude that

1. $z_1 = \frac{1}{15}\left(\frac{2}{11}y_1 - 6\right) + \frac{56}{15}$.
2. Additionally we must assume that

$$\frac{1}{15}\left(\frac{2}{11}y_1 - 6\right) + \frac{56}{15} \in \mathbb{Z}$$

and

$$\frac{2}{11}y_1 - 6 \in \mathbb{Z}.$$

This shall illustrate the we can compute the integer volume in the same way as the Lebesgue volume but we just have to keep track of a few additional restrictions of the parameter space.

□

Corollary 4.32
Algorithm 4.29, Volume, *is correct in the case of $\mu = \gamma$ if we assume that step [2.3.5] doesn't generate only one quadruple $P, T, \vec{A} - \vec{a}, X$ but a set Y of quadruples $P, T, \vec{A} - \vec{a}, X$ that is added to Z in step [2.3.6].*

In the specification of the algorithm we have

$$X = \left\{ \vec{x} \in \mathbb{Z}^{n-p} \mid \left(\vec{B} - \vec{b}\right)^{-1} \vec{x} \in \mathbb{Z}^l, \text{ for some } l, \text{ and } \vec{B} - \vec{b} \in W \right\},$$

for some set W containing only such mappings having left inverses. In line 4.28 $\vec{A} - \vec{a}$ is initialised to the identity mapping on $\mathbb{Z}^{n-p}$ and X to the space $\mathbb{Z}^{n-p}$. In line 4.31 nothing needs to be done.

□

Proof:

We must show that the result of γ-integration in step [2.3.5] is of the form of the set Y. Then the partial correctness is a consequence of the previous sections. The termination is obvious.

The application of Pugh's disjoint splintering of page 25 in order to eliminate the floor and ceiling functions in the integration bounds obviously splits the variable and parameter space disjointly.

For each single splinter we get a condition of the form $C\vec{x} = 0$, where $C \in \mathcal{AL}\,(n-k+2)$. Applying the techniques of section 2.6, Flat Polyhedra, we get an affine linear variable transformation $F : \mathbb{R}^{n-k-1} \to \mathbb{R}^{n-k-1}$ that will be substituted into the system T in the algorithm Volume. We also get a corresponding transformation of the parameters $G : \mathbb{R}^{n-p} \to \mathbb{R}^{n-p}$. We only have to partition the kernel matrix of C as in figure 2.1 appropriately. Using G we modify $\vec{A}-\vec{a}$ and the set W of X in step [2.3.5] in the following way.

$$\begin{aligned} \vec{A}-\vec{a} &\leftarrow \left(\vec{A}-\vec{a}\right) \circ G \\ W &\leftarrow W \cup \left\{ \left(\vec{A}-\vec{a}\right) \circ G \right\} \end{aligned}$$

Moreover we replace the result P of the integration of the polynomial Q by $P \circ F$.

□

Hence Volume$\big(k, k+1, n, S\big)$ is a representation of $\mathcal{V}_{\gamma,S,k}$ if $\mu = \gamma$. It is a piecewise polynomial as in the case $\mu = \gamma$.

Now we want to point out some ways of speeding up the volume algorithm that Pugh proposes.

As the ordering of the variables can be chosen arbitrarily when decomposing Pugh chooses a variable with as few lower and upper bounds as possible. We will investigate this proposal in section 5.4, Timing of the Implementation.

Apart from this he simplifies the systems of inequalities that are generated by the algorithm. Moreover he performs an inconsistency check at each recursion step. For this he uses a procedure described in [Pug92]. It is based on Fourier-Motzkin projection.

In chapter 5, Improvements, we will also detect inconsistencies and redundancies. As during the volume computation the inequalities are generated one after the other we will use an algorithm that is formulated incrementally in contrast to Pugh's one.

Chapter 5

Improvements

Here we exhibit and evaluate approaches to speed up the execution of the volume algorithm. Particularly we want to control the blow up of intermediate results as well as of the output.

Take the hexagon in example 1.3, for instance. When integrating over the first variable the (real) volume algorithm generates *twelve* different systems of linear inequalities. But it turns out that among these ones only *three* are consistent! Moreover the majority of the inequalities in these consistent systems is redundant.

This blow up is even more dramatic in the case of integer volume. Pugh's disjoint splintering divides each part of the polyhedron further into many splinters.

For controlling this blow up we apply techniques of linear optimisation and operations research summarised in [Stu91].

In the last section, Timing of the Implementation, we compare the effectiveness of these modifications.

5.1 Pruning the Recursion Tree

As the solution set of an inconsistent system of linear inequalities is empty by definition we know that its volume w.r.t. any measure is zero without doing any computations.

Therefore we'd like to check if the system in the base case of the volume algorithm is inconsistent as well as the system $T \uplus R$ each time a new inequality R is added to T. If this is the case the computation can be aborted in this branch of the recursion because the volume is zero. Moreover we'd like to do the same with the systems generated by Pugh's disjoint splintering.

In order to obtain such a procedure we will generalise an incremental consistency test formulated by Stuckey and found in [Stu91]. This one only checks if a system of *non* strict inequalities has a *positive* solution. Stuckey's consistency test is an application of the first phase of the well known simplex method of linear optimisation. Because of this he uses a representation of systems of linear inequalities that is quite different to ours. Namely, they are written as

$$A\vec{x} = 0 \text{ and } \vec{x} \geq 0. \tag{5.1}$$

We will show later how an arbitrary system T of linear inequalities can be transformed into this form.

In order to formulate the specification of Stuckey's test we need to define what's a null variable.

Definition 5.1 (Null variable)
Let $X \subseteq \mathbb{R}^n$.

Then i *is called the index of a null variable of* X *iff*[1]

$$\pi_i(X) = \{0\}.$$

□

The above definition means that the i-th component of $\vec{x} \in X$ is necessarily zero.

Now we can specify Stuckey's test[2].

$$(u, Y, A) \leftarrow \text{IsSimplexConsistent}(A, R)$$

Input:
$A \ldots\ldots\ldots \in \mathcal{AL}(n)^3$, as in line 5.1.
$R \ldots\ldots\ldots \in \mathcal{LS}(n)$ with $\|R\| = 1$.

Output:

$$u \ldots\ldots\ldots = \begin{cases} 0 & \text{if there doesn't exist a } \vec{x} \in \mathcal{P}(R) \text{ satisfying condition 5.1,} \\ 1 & \text{else.} \end{cases}$$

[1] Our definition is a bit more general than Stuckey's one because this is more appropriate for the notation we use. The reader will easily see that their geometric contents is the same. Moreover in our definition it would be more natural to write *null component* instead of null variable but we don't want to replace traditional phrases by new unusual ones.

[2] We don't describe explicitely how this test works because this is not the intended contents of this diploma thesis. This is already done by Werner Landerl. He writes a diploma thesis containing a detailed description of Stuckey's test at the same time this thesis about volume computation is written. The interested reader might contact Werner Landerl at RISC-Linz, Johannes Kepler University Linz, Austria.

$$Y \ldots\ldots = \begin{cases} \emptyset & \text{if } u = 0, \\ \text{the set of null variable indices of the set} & \\ \text{of all } \vec{x} \in \mathcal{P}(R) \text{ satisfying condition 5.1} & \text{else.} \end{cases}$$

A .... the input[3] A augmented by R.

In order to apply this test we must transform a system of inequalities

$$(B, \lessdot) \in \mathcal{LS}(n)$$

into the form 5.1. If $\lessdot = \leq$ this can be done by standard techniques of linear optimisation. In this case the next lemma just states this transformation.

Lemma 5.2
The linear system

$$(B, \lessdot) \in \mathcal{LS}(n) \textit{ is consistent}$$
$$\textit{iff}$$

there exist $\vec{x}, \vec{y} \in \mathbb{R}^n$ *and* $\vec{z} \in \mathbb{R}^{\|(B,\lessdot)\|}$ *such that*

$$B\vec{x} - B\vec{y} + \breve{b} + B(0) = 0, \tag{5.2}$$

where

$$b_i := \begin{cases} z_i & \textit{for } i \leq \|(B, \lessdot)\|, \\ 0 & \textit{else} \end{cases}$$

and

$$\vec{x}, \vec{y} \geq 0 \textit{ and } -\breve{b} \lessdot 0.$$

□

Proof:
"$\Rightarrow$": Let $\vec{c} \in \mathcal{P}((B, \lessdot))$. Let $\vec{x}, \vec{y} \geq 0$ be such that

$$\vec{c} = \vec{x} - \vec{y}$$

and $\vec{z}$ be such that

$$\breve{b} = -B(\vec{c}).$$

The latter is possible because $\|-B(\vec{c})\| \leq \|(B, \lessdot)\|$. Then we have

$$B\vec{x} - B\vec{y} + \breve{b} + B(0) = 0$$

and

$$\vec{x}, \vec{y} \geq 0 \text{ and } -\breve{b} = B(\vec{c}) \lessdot 0.$$

[3] Note that A is represented by a simplex tableaux in [Stu91]. How to transform A (and R) into a simplex tableaux is a technical detail of the realisation of this algorithm we don't care about here.

"$\Leftarrow$": We have $\vec{x} - \vec{y} \in \mathcal{P}((B, \lessdot))$.

$\square$

If we wouldn't have to deal with strict inequalities we would be done now. We'd just transform a system of the shape $(B, \leq)$ into the form of line 5.1 by the application of lemma 5.2 and then execute the algorithm IsSimplexConsistent.

But in the case $\lessdot \neq \leq$ one might imagine polyhedra such that for all $\vec{x}$ with $B\vec{x} \leq 0$ there exists a $j \leq \|B\|$ such that $y_j = 0$, where $\breve{y} := B\vec{x}$. We will see later that such inconvenient situations aren't possible concerning polyhedra. It might only happen that there exists a $j \leq \|B\|$ such that for all $\vec{x}$ with $B\vec{x} \leq 0$ we have $y_j = 0$, where $\breve{y} := B\vec{x}$. This can be regarded as a certain uniformity property of equality.

So let's make this precise in the following definition.

Definition 5.3 (Implicit equality)
We call $\lessdot_j$ an implicit equality of the linear system $(B, \lessdot)$ iff

$$y_j = 0, \; \breve{y} := B\vec{x},$$

for all $\vec{x} \in \mathcal{P}((B, \lessdot))$, and $j \leq \|(B, \lessdot)\|$.

The number j is called the index of the implicit equality $\lessdot_j$.

$\square$

First note that this means that the polyhedron $\mathcal{P}((B, \lessdot))$ is a flat polyhedron (with $m = 0$) in the sense of section 2.6, Flat Polyhedra, if $(B, \lessdot)$ has an implicit equality. Geometrically, one can really think of such a polyhedron as of a flat object, like a two dimensional plane in three dimensional space.

Secondly, observe that the above definition means that $2n + j$ is the index of a null variable of the set of vectors $\vec{d} \geq 0$ satisfying

$$C\vec{d} = 0, \tag{5.3}$$

where $C \in \mathcal{AL}(2n + \|(B, \lessdot)\|)$ is defined by (compare line 5.2)

$$(\vec{x}, \vec{y}, \vec{z}) \mapsto B\vec{x} - B\vec{y} + \vec{z} + B(0). \tag{5.4}$$

Now the corollary of the following lemma states the above uniformity property of equality. The lemma is proven by Lassez and McAloon using some short topological argument in [LM92].

Lemma 5.4 ([LM92])
The system

$$(B,<) \text{ is consistent}$$

iff the system

$$(B,\leq) \text{ is consistent}$$
$$\text{and}$$
$$\lessdot_j \text{ is no implicit equality of } (B,\leq)\,,\ \text{for } j \leq \|(B,\leq)\|\,.$$

□

Now we get the desired corollary.

Corollary 5.5
The system

$$(B,\lessdot) \text{ is consistent}$$

iff the system

$$(B,\leq) \text{ is consistent}$$
$$\text{and}$$
$$\lessdot_j \text{ is no implicit equality of } (B,\leq)\,,\ \text{for } j \leq \|(B,\leq)\| \text{ with } \lessdot_j = <.$$

□

The reader will surely agree with us that this corollary is a trivial consequence of the previous lemma. But as it is of such an importance here we'd like to prove it directly. The proof is similar to Lassez's and McAloon's one but doesn't use Topology.

Proof:

"⇒": Trivial.

"⇐": Assume $B\vec{x} \leq 0$ but $\neg\,(B\vec{x} \lessdot 0)$, for some given $\vec{x}$. This means that $Y \neq \emptyset$, where

$$Y := \{j \mid c_j = 0,\ \text{where } \breve{c} := B\vec{x},\ \text{and } \lessdot_j = <\}\,.$$

But as $\lessdot_j$ is no implicit equality of $(B,\leq)$, for all $j \in Y$, we get vectors $\vec{y^j}$ such that

$$B\vec{y^j} \leq 0 \text{ and } d_j < 0,\ \text{where } \breve{d} := B\vec{y^j},$$

for $j \in Y$.

Because of the affine linearity of B, i. e. of the convexity of $\mathcal{P}((B, \leq))$, we have

$$B\left(\frac{1}{1+|Y|}\left(\vec{x}+\sum_{j\in Y}\vec{y^j}\right)\right) \leq 0.$$

Moreover it is also valid that[4]

$$\begin{aligned}\check{e} &:= B\left(\frac{1}{1+|Y|}\left(\vec{x}+\sum_{j\in Y}\vec{y^j}\right)\right)\\ &= \frac{1}{1+|Y|}B\vec{x}+\frac{1}{1+|Y|}\sum_{j\in Y}B\vec{y^j}.\end{aligned}$$

Hence

$$e_j < 0,$$

for all j with $\lessdot_j = <$. Thus we have

$$\frac{1}{1+|Y|}\left(\vec{x}+\sum_{j\in Y}\vec{y^j}\right) \in \mathcal{P}((B, \lessdot)).$$

□

As we can use the algorithm IsSimplexConsistent for reporting implicit equalities, too, we can also check the consistency of linear systems defining polyhedra with open faces applying corollary 5.5.

Summarising we get a method having the following specification.

$$(u, Y, C) \leftarrow \text{IsConsistent}(C, R)$$

Input:

C .... $\in \mathcal{AL}(2n + \|B\|)$, where $B \in \mathcal{LS}(n)$ corresponds to C as in line 5.4.

R .... $\in \mathcal{LS}(n)$ with $\|R\| = 1$.

Output:

$$u \ldots\ldots = \begin{cases} 0 & \text{if there doesn't exist a } \vec{x} \in \mathcal{P}(R) \text{ satisfying} \\ & \text{the condition of line 5.3,} \\ 1 & \text{else.} \end{cases}$$

$$Y \ldots\ldots = \begin{cases} \emptyset & \text{if } u = 0, \\ \text{the set of indices of implicit equalities of} & \\ T \uplus R = (B, \lessdot) \uplus R & \text{else.} \end{cases}$$

C .... the input C augmented by R.

[4]Just don't forget that the equality is just some special case. The map B is *affine* linear!

5.2 Redundancy Detection

As already mentioned in the introduction to this chapter the volume algorithm generates many redundant inequalities. This inequalities are not needed in order to define the given polyhedron uniquely. Thus we want to eliminate them from the system T before it is added to Z in the algorithm Volume. We will reduce this redundancy elimination to consistency detection in the rest of this section.

This is done by the following lemma.

Lemma 5.6 (Redundancy detection)
Let $T \in \mathcal{LS}(n)$ and $(C, \lessdot) := T_i$, for some i. Moreover we define

$$S^i := \biguplus_{j \neq i} T_j,$$

for all i. Then

$$T_i \text{ is redundant in } T$$

iff

$$(-C, \lhd) \uplus S^i \text{ is inconsistent,}$$

where

$$\lhd_j := \begin{cases} \leq & \text{if } j = 1 \text{ and } \lessdot_1 = <. \\ < & \text{if } j = 1 \text{ and } \lessdot_1 = \leq. \\ \leq & \text{else.} \end{cases}$$

□

Proof:

We use the facts that

$$\mathcal{P}\left(S^i\right) = \mathcal{P}\left((C, \lessdot) \uplus S^i\right) \cup \mathcal{P}\left((-C, \lhd) \uplus S^i\right) \tag{5.5}$$

and

$$\mathcal{P}\left((C, \lessdot) \uplus S^i\right) \cap \mathcal{P}\left((-C, \lhd) \uplus S^i\right) = \emptyset. \tag{5.6}$$

"$\Rightarrow$": We have

$$\mathcal{P}\left((C, \lessdot) \uplus S^i\right) = \mathcal{P}\left(S^i\right).$$

Because of lines 5.5 and 5.6 this implies that

$$\mathcal{P}\left((-C, \lhd) \uplus S^i\right) = \emptyset.$$

"$\Leftarrow$": Obvious given line 5.5.

$\square$

Thus before adding T to Z in the volume algorithm we check if T_i is redundant in T, for all $i \leq \|T\|$, by applying the above lemma and using the method for consistency checking from the previous section.

5.3 Flatness Detection

In the case of the real volume we know that a polyhedron having an implicit equality has Lebesgue volume zero. Therefore we can treat such flat polyhedra (compare also section 2.6, Flat Polyhedra) as they were inconsistent. As the detection of implicit equalities is a side result of the consistency check in section 5.1, Pruning the Recursion Tree, nothing new needs to be done. So in this case we refer to section 5.1.

In the case of the integer volume we can only eliminate some variables without integration using the techniques described in section 2.6, Flat Polyhedra. This means shortening the recursion tree instead of pruning it. This elimination takes place in the base case of the algorithm Volume and just right after the integration before we add the newly generated T to Z.

More precisely, for eliminating some variables we will use line 2.18. Using the notation there we have $m = 0$. We can eliminate all variables but $l - \min(l, n-k) = k - \min(k, n-l)$ ones, where l is the dimension of the solution space defined by the implicit equalities and k is the number of integration variables.

5.4 Timing of the Implementation

Here we present the running time of the implementation of the algorithm 4.29, Volume, for various inputs. We also compare the improvements proposed in this chapter to the plain volume algorithm. Moreover we investigate two different strategies of choosing the next integration variable in step [2.1].

The algorithm was implemented in the programming language C++. Additionally we used some functions to be found in the library *Sturm*. This library contains sequential as well as parallel routines for algebraic constraint solving. Its development[5] is led by Hoon Hong at RISC-Linz. In our implementation we mainly used exact rational arithmetic and several data types

[5]Besides Hoon Hong and Manfred Minimair, the author, the following people contributed to this library at the time of the writing of this thesis: Olga Caprotti, Andreas Neubacher, Ovidiu Podisor, Stefan Ratschan, Rosario Salamone and Daniela Vasaru.

found in the Sturm-library. The implementation was tested on an SGI workstation out of the Iris Indigo series with an R4000 processor running with 50 MHz. It's main memory was 96 MB.

For timing the algorithm we made the following assumptions.

A random input to the volume algorithm mainly depends on *four* parameters, namely

1. the number of integration variables,

2. the number of parameters,

3. the size of the system of linear inequalities and

4. the size of the rational coefficients.

Moreover all rational numbers in the inputs were represented in lowest terms, i.e. the greatest common divisor of the numerator and the denominator of each rational number was 1.

Chiefly we wanted to know how the running time of the algorithm behaves when the number of integration variables and the number of parameters are increasing. Moreover we wanted to find this out for the number of summands[6] in the piecewise polynomial. We always assumed that the size of the system of linear inequalities is twice the number of integration variables. In practice, like nested loop analysis (cf. chapter 1), the size will always be of the order of the number of integration variables.

Besides to this we decided that the random input should contain only very small rational numbers for the final testing of the algorithm. This means that the numerators and the denominators of rationals in the input were out of the range $[-8, 8]$.

We also tested the volume algorithm on systems with bigger coefficients, e.g. out of $[-32, 32]$. But in the latter case we experienced such a memory consumption due to an enormous coefficient blow up in the intermediate linear systems of inequalities and polynomials as well as a blow up of the output in any version of the volume algorithm that no systematic testing was possible. We found out that both inequalities and polynomials have dramatically growing non trivial contents even for very simple inputs[7].

But we want to remark that already here one can illustrate the effectiveness of the proposed improvements. For computing the real volume of a non

[6]The sizes of the linear systems of inequalities in the output are irrelevant because they are bounded by the size of the input.

[7]We already plan to control this blow up in future by applying techniques introduced by Bareiss.

parametrised 4-dimensional polyhedron the plain volume algorithm took in average 4 minutes and the piecewise polynomial in the output consisted of about $200,000$ summands. The combination of all three improvements of this chapter together with Pugh's strategy of choosing an integration variable with as few splinters as possible took 0.2 minutes. This is a speed up of 20. The length of the output was only 1.300 because of pruning the recursion tree. This is a reduction by a factor of ca. 150. For bigger inputs the volume computation even didn't terminate. Therefore we reduced the size of the coefficients.

This reduction to $[-8, 8]$ also implies that random systems of inequalities are more likely to be sparse. This explains why the output isn't that big as it can be seen in figure 5.2 below.

In the figures below either the running time in seconds or the lengths of the ouputs are drawn in the direction of the y-axis. The marks on the x-axis are either the number of integration variables or parameters.

Let's start with the results of the Lebesgue version of the volume algorithm.

Figure 5.1 shows its running time on random inputs having zero parameters when the number of integration variables is increasing.

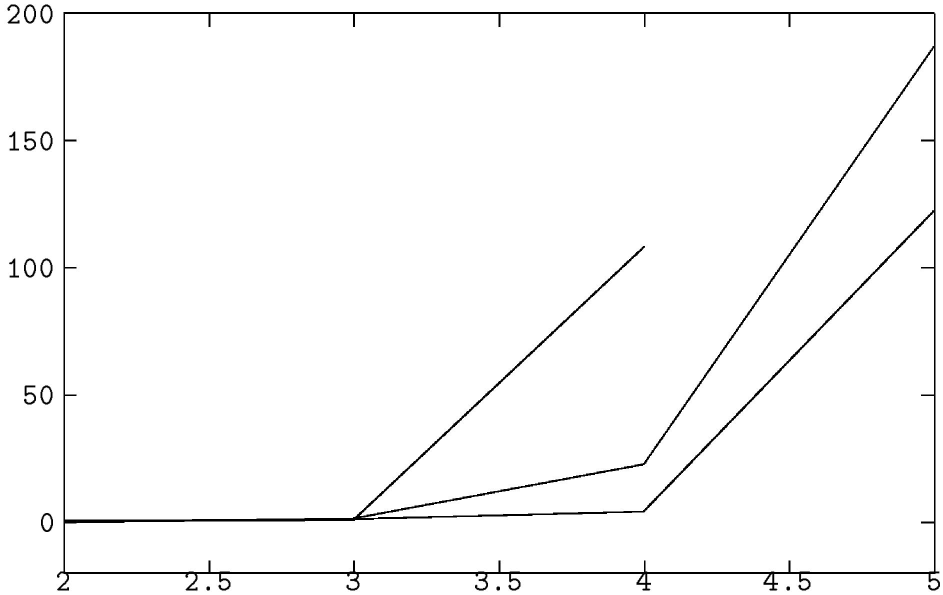

Figure 5.1: Running time: 0 parameters

One can see three different lines. One of them is shorter than the other ones. This is the running time of the plain volume algorithm without improvements. Alas, its running time could only be measured up to 4 integration variables. For more ones it didn't terminate within several hours.

The higer one of the two other lines represents the running time of the real volume algorithm combined with the three improvements proposed in this chapter. Moreover we chose an obdurate, i.e. fixed, integration order, namely just one variable after the other.

The lower line is the running time of the improved algorithm, too. But the integration order is smart, i.e. isn't fixed. Always the integration variable generating as few splinters as possible is chosen in step [2.1]. Alas, for none of the two improved versions we could obtain a result for more than 5 integration variables within several hours.

First of all one can see in figure 5.1 that at a certain number of integration variables an explosion of the running time occurs for all three versions. For the plain algorithm this already happens with 4 ones, whereas for the improved versions with 5 variables.

Moreover the improved versions achieve a significant speed up when integrating over 4 variables. It is approximately 5. But for 2 and 3 integration variables the running times are roughly the same.

Apart from this we see that the strategy of choosing the minimum splitting variable is slightly faster than integrating over one variable after the other. For 4 variables the speedup is ca. 4.4 and for 5 variables 1.5.

The next figure 5.2 shows the lengths of the outputs, i.e. the number of summands of the piecewise polynomials. The shapes of the curves are similar to the running times. The highest one is the plain algorithm followed by obdurate integration and smart one.

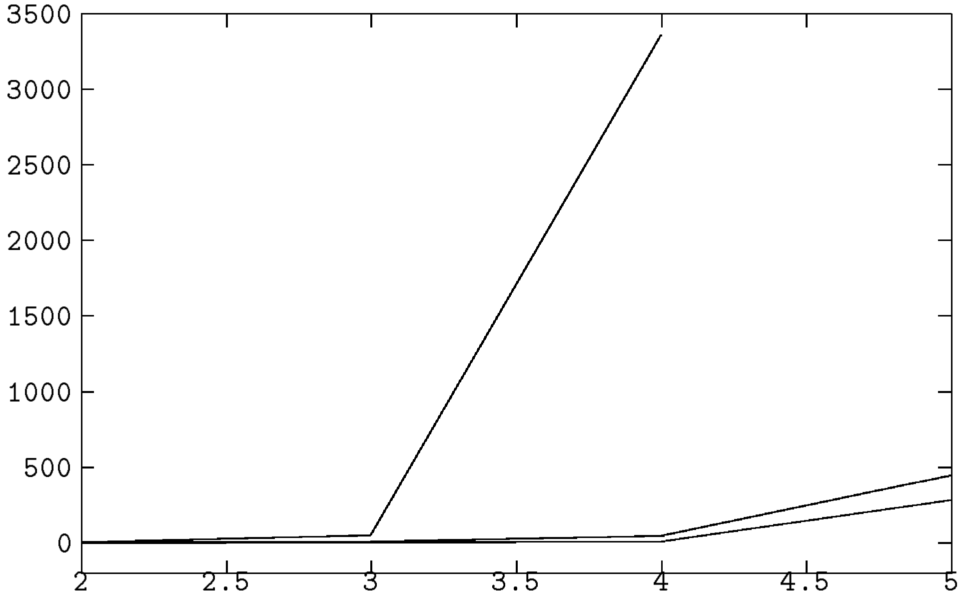

Figure 5.2: Length: 0 parameters

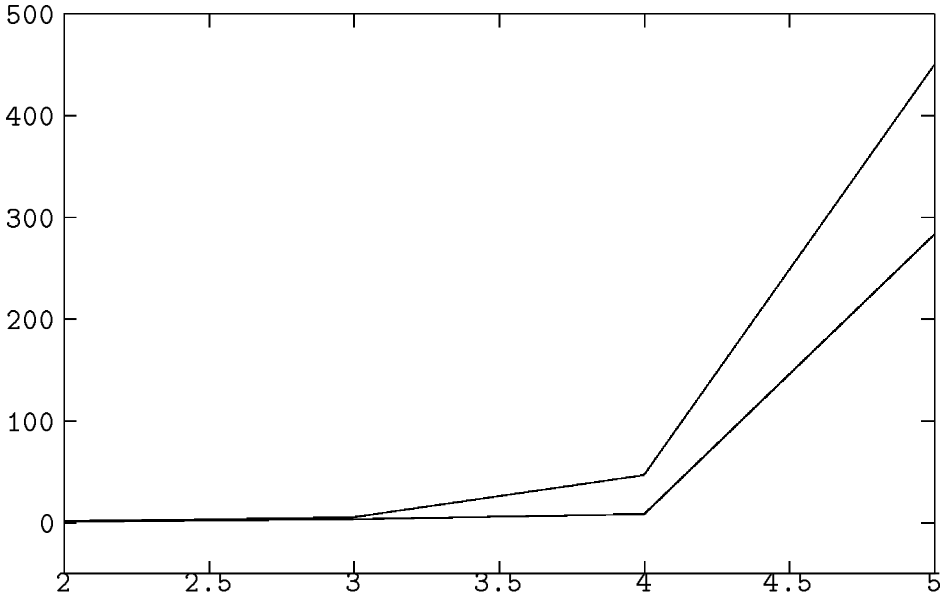

Figure 5.3: Improved length: 0 parameters

The scaling of figure 5.2 might leave the impression that smart integration isn't better in reducing the size of the output than obdurate one. Therefore figure 5.3 only contains the lengths of the ouputs of the two latter ones.

Summarizing the two figures we conclude that the blow up of intermediate and final results is effectively controlled by the improvements. Namely, for 4 variables the result is smaller by a factor of ca. 380. This effect also occurs for 3 variables. There the factor is 5.5. Moreover with 4 variables smart integration is better than obdurate one by a factor of 5.2 and with 5 variables by a factor of 1.5. This superiority of smart integration to obdurate one also occured with similar factors when testing inputs of different sizes. Therefore we exclude obdurate integration from the following figures.

The next figure 5.4 shows the running times of the implementation when the input contains two parameters. The higher curve contains the times of the plain version.

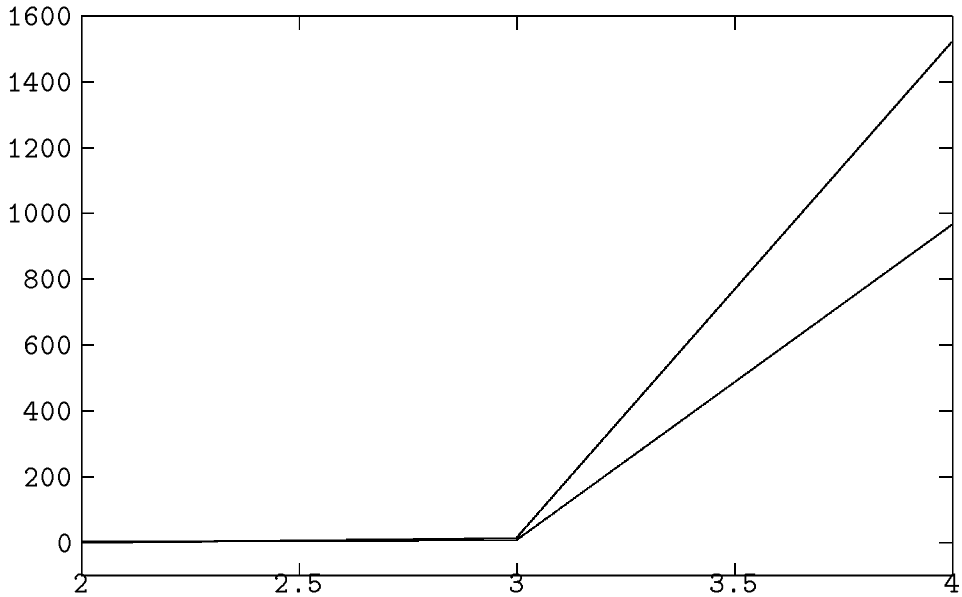

Figure 5.4: Running time: 2 parameters

We see that the shapes of the curves are similar to those ones of zero parameters. But the explosion of the running time already occurs with four variables. In other words: consistency check and redundancy detection aren't that efficient with parameters. Nevertheless there are certain speed ups. With 3 variables there is a speed up of 1.6. In the case of 4 variables it is 1.5.

The figure 5.5 contains the lengths of the corresponding outputs.

Again the curves look similar to the running times. Morover the improvements control the size of the output effectively. We get a factor of 5.8 for

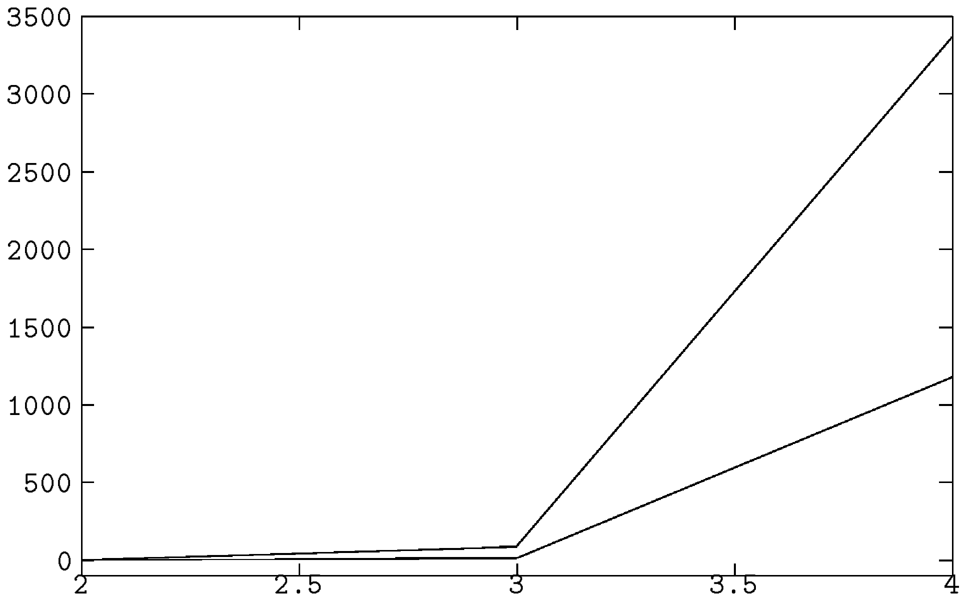

Figure 5.5: Length: 2 parameters

three variables and of 2.8 for four variables.

The results shown so far already indicate that the improvements of this chapter get less effective when increasing the number of parameters which is confirmed by the following results. The next figure 5.6 even shows cases in which the improvements don't have any effect at all.

This figure contains running times of the algorithm where the input has three integration variables but the number of parameters is increased.

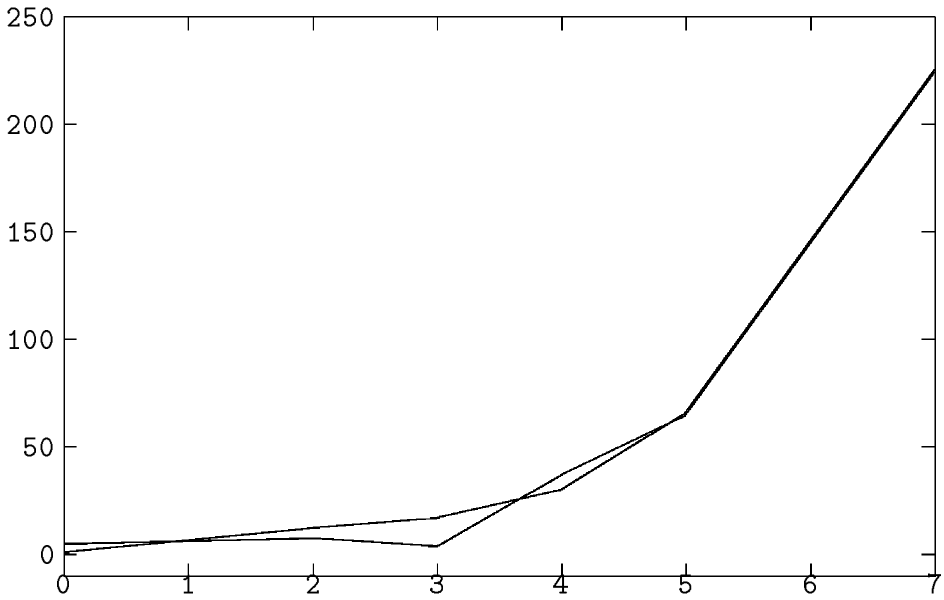

Figure 5.6: Running time: 3 variables

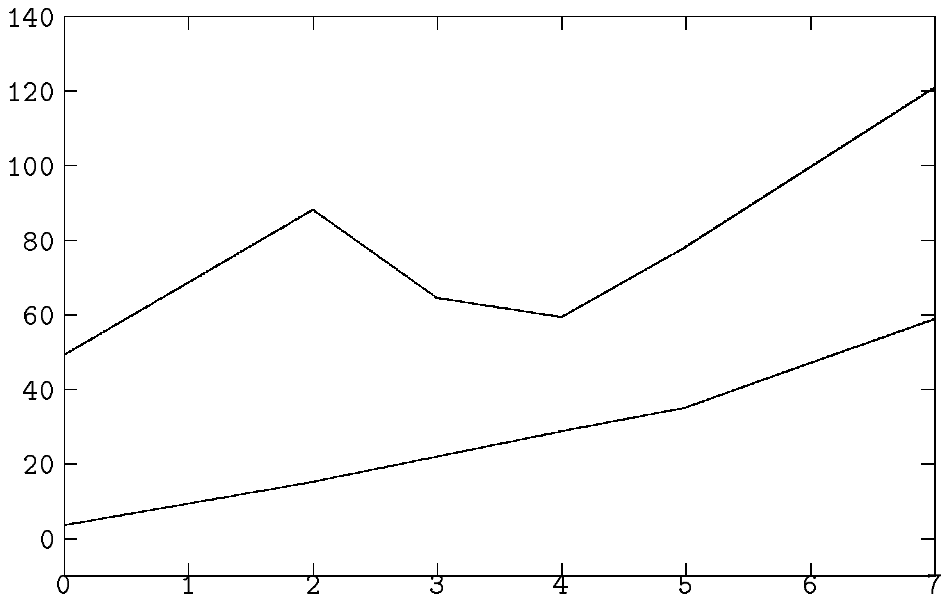

Figure 5.7: Length: 3 variables

The curve intersecting the y-axis a bit lower is the curve of the improved algorithm. The other one corresponds to the plain algorithm. Both curves grow slower when increasing the number of parameters in contrast to an increase in the number of integration variables. But at a certain point, here 7 parameters, there is also an explosion of the running times.

The figure 5.7 shows the corresponding lengths of the ouputs. The higher curve belongs to the plain algorithm. Here is a big difference in their shapes in contrast to the corresponding running time curves. The lengths of the outputs are increasing only slowly.

Initially we expected a similar behavior for the integer volume version of the algorithm as for the real volume version. But soon we had to accept that there was no systematic timing of the integer version possible. For special, very simple polyhedra as in example 1.4 we obtained correct results. The corresponding sizes and running times are similar to the real volume algorithm. But for random input it didn't terminate in most cases.

This effect can be explained very easily. The number of newly generated systems of linear inqualities in the integer volume algorithm depends on the sizes of the denominators in the intermediate results because of Pugh's splintering. But we already mentioned the enormous coefficient blow up happening in the real volume version. This blow up is even bigger in the integer version. Just imagine that a denominator of only the size of one line of the terminal screen means that at least about 10^{80} new linear systems are generated. But often the coefficients grow much longer than one screen line!

Summarizing we conclude that by the application of the improvements found in this work the blow up of intermediate and final results can be controlled in the volume algorithm significantly. Moreover remarkable speed ups could be achieved in some cases. Nevertheless the plain algorithm and the integer version in particular still must be improved until it can be used for arbitrary inputs.

Bibliography

[Ash72] Robert B. Ash. *Measure, Integration, and Functional Analysis.* Academic Press, New York, London, 1972.

[AW89] W. Arnold and H. Wussing. *Biographien bedeutender Mathematiker.* Aulis Verlag Deubner, 1989.

[Ber95] Fausto Bernardini. Integration of polynomials over n-dimensional polyhedra. Technical report, Dip. di Informatica e Sistemistica, Università La Sapienza, Via Buonarroti 12, 00185 Roma, 1995.

[Bun88] Peter Bundschuh. *Einführung in die Zahlentheorie.* Springer-Verlag, Berlin Heidelberg New York, 1988.

[Coh93] Henri Cohen. *A Course in Computational Algebraic Number Theory.* Number 138 in Graduate Texts in Mathematics. Springer-Verlag, Berlin Heidelberg New York, 1993.

[Ell91] T. M. R. Ellis. *Fortran 77 Programming.* Addison-Wesley Publishing Company, 1991.

[Fah93] T. Fahringer. *Automatic Performance Prediction for Parallel Programs on Massively Parallel Computers.* PhD thesis, Institute of Software Technology and Parallel Systems,University of Vienna, October 1993.

[Fah96] Thomas Fahringer. On estimating the useful work distribution of parallel programs under the P^3T: A static performance estimator. *Accepted for publication in journal: Concurrency Practice and Experience, Ed. Geoffrey Fox*, to appear in 1996.

[Fut89] Gerald Futschek. *Programmentwicklung und Verifikation.* Springers Angewandte Informatik. Springer-Verlag, Wien New York, 1989.

[GIS90] S. Gottwald, H.-J. Ilgauds, and K.-H. Schlote. *Lexikon bedeutender Mathematiker.* Verlag Harri Deutsch, Thun, Frankfurt, 1990.

[GK84] R. T. Gregory and E. V. Krishnamurthy. *Methods and applications of error-free computation.* Texts and monographs in computer science. Springer-Verlag, Berlin Heidelberg New York, 1984.

[Haz88] M. Hazewinkel. *Encyclopaedia of mathematics*, volume 1. Kluwer Academic Publishers, Dordrecht, Holland, 1988.

[Heu91] H. Heuser. *Lehrbuch der Analysis*, volume 1. B. G. Teubner, Stuttgard, 9th edition, 1991.

[Heu92] H. Heuser. *Funktionalanalysis.* B. G. Teubner, Stuttgart, 3rd edition, 1992.

[HKSF94] Hoon Hong, Franz Kriftner, Volker Stahl, and Thomas Fahringer. Computing the volume of a convex polytope using interval arithmetic. Technical report, RISC Linz, Johannes Kepler University Linz, and Institute for Software Technology and Parallel Systems, University Vienna, A-4040 Linz and Bruenner Str. 72, A-1210 Vienna, Austria, 1994.

[HP94] Mohammand R. Haghighat and Constantine D. Polychronopoulos. Symbolic analysis for parallelizing compilers. CSRD Report 1355, Center for Supercomputing Research and Development, University of Illinois at Urbana-Champaign, 415 CSRL - 1308 West Main Street, Urbana, IL 61801-2307, 1994.

[HU93] John E. Hopcroft and Jeffrey D. Ullman. *Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie.* Addison-Wesley Publishing Company, Bonn, Reading, Massachusetts, 2 edition, 1993.

[Kno64] Konrad Knopp. *Theorie und Anwendung der unendlichen Reihen.* Springer-Verlag, 4th edition, 1964.

[LM92] Jean-Louis Lassez and Ken McAloon. A canonical form for generalized linear constraints. *J. Symbolic Computation*, 13(1-24), 1992.

[Mar89] Dietrich Marsal. *Finite Differenzen und Elemente.* Springer-Verlag, Berlin Heidelberg New York, 1989.

[PS95] A. N. Parshin and I. R. Shafarevich, editors. *Number Theory I*, volume 49 of *Encyclopaedia of Mathematical Sciences*. Springer-Verlag, Berlin Heidelberg New York, 1995.

[Pug92] William Pugh. A practical algorithm for exact array dependency analysis. *Communications of the ACM*, 35(II):102–114, 1992.

[Pug94] William Pugh. Counting solutions to presburger formulas: How and why. UMIACS-TR-94-27 and CS-TR-3234, Institute for Advanced Computer Studies, Dept. of Computer Science, Univ. of Maryland, College Park, MD 20742, March 1994.

[Stu91] Peter J. Stuckey. A simplex-based algorithm for incremental linear arithmetic constraint solving and detection of implicit equalities. Technical report, Department of Computer Science, University of Melbourne, Parkville 3052, Australia, 1991.

[Taw95] Nadia Tawbi. Estimating the number of nested loops iterations by integer arithmetic in convex polyhedra. Technical report, BULL Corporate Research Center, Rue Jean Jaurès, F-78340 Les Clayes-sous-Bois, January 1995.

[vAHU83] A. von Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

[Val79] Leslie G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, August 1979.

[Zie94] Günter M. Ziegler. *Lectures on Polytopes*. Springer-Verlag, Berlin Heidelberg New York, 1994.