Universität Passau
Fakultät für Informatik und Mathematik

On Algorithmic and Heuristic Approaches to Integral Problems in the Polyhedron Model with Non-linear Parameters

Diplomarbeit

Autor:
Stefan Schuster

Aufgabensteller:
Priv.-Doz. Dr. Martin Griebl
Lehrstuhl für Programmierung
Universität Passau

Betreuer:
Dipl.-Inf. Armin Größlinger

26. Juli 2007

Abstract

The polyhedron model provides one possible approach to model-based program analysis and transformation. Over the years, it has undergone many improvements in order to handle a still growing class of programs. A recent approach enabled the use of non-linear parameters, which was in that generality not possible before. While this new framework uses real quantifier elimination, the present work examines possibilities to describe the integral solutions to certain problems occurring in the polyhedron model exactly. In particular, we study the solvability of systems of linear Diophantine equations in one and several non-linear integral parameters and demonstrate the applicability of the developed methods for Banerjee's data dependence analysis.

Acknowledgements

First of all, I want to thank my friend and tutor Armin Größlinger for giving me the opportunity and the initial impulse to dive into the problems that led to this work, for all the illuminating discussions and his clear comments and suggestions. I would like to thank Dr. Martin Griebl for providing and forming a sound platform to present those ideas: the LooPo-Team, to which I say thanks for listening and the feed-back. Michael Häusler helped me a lot with the proof-reading, thank you! I dedicate this work to my wife Anja, who carried me through some rough times.

Contents

Chapter 1

Introduction

Data dependence analysis lies at the core of many applications in computer science concerned with analysis and transformation of programs. Ranging from automatic parallelizers to sophisticated debuggers, whenever we have to take a close look at our programs and ask "which statement writes when and before or after which other statement to this memory cell?", data dependence analysis enters the scene.

Focusing on the construction of automatic parallel compilers, one approach which has been the subject of intensive study ([KMW67], [Lam74]) is a model-based approach which restricts the input program to nested for-loop programs having array access functions and bounds linear in the index variables and structural parameters. This constitutes the so-called polytope model ([Len93]), which was later extended to the polyhedron model ([LG95]).

When Armin Größlinger introduced real quantifier elimination in this model ([Grö03]), those restrictions were further eased as the coefficients of the index variables may now also contain structural parameters. In fact, he demonstrated that algorithms central to parallel-compiler construction can be generalized in such a way that those "non-linear parameters" can be handled adequately.

Real quantifier elimination, however, treats the parameters – as the name suggests – as variables from $\mathbb{R}$, while the applications actually require the parameters to be from $\mathbb{Z}$. The difference is that often real solutions to the application problems exist whereas integral solutions do not. To illustrate this point, just look at the simple equation

$$3x = 7.$$

It is clear that this equation has no solution in $\mathbb{Z}$ but it does in $\mathbb{R}$. Unfortunately, there is no integral quantifier elimination as implied by the unsolvability of Hilbert's Tenth Problem ([Mat70]).

Recent research introduced an interesting weak form of quantifier elimination "for the Full Linear Theory of the Integers" ([LS06]) and it would be one way to study in a top-down fashion how this general approach to integral problems can be adapted to the application's needs. The present work, however, takes a different, more bottom-up account and studies how existing methods can be extended such that they meet the needs of the integral, non-linear parametric case.

More precisely, we will concentrate on the classical approach to data dependence analysis developed by Utpal Banerjee in 1993 ([Ban93]) and try to extend the mathematical methods used by it.

To get a first impression of the mathematical tools involved let us consider the program

```
for $i = 0$ to $n$ do
  for $j = 0$ to $n$ do
    $S$: $A[p \cdot i + j] = A[p \cdot i + j] + 1$
  end for
end for
```

with one integral, non-linear parameter $p > 0$ (which could have occurred in rewriting a two-dimensional array) and one structural parameter n. Dependencies occur, for instance, if at iteration (i_0, j_0) some value is written to $A[pi_0 + j_0]$ which in turn is used at some later iteration (i_1, j_1). For example, let $n = 3$ and $p = 2$. Then iteration $(i, j) = (0, 3)$ increases the value of $A[3]$, which is read out again at iteration $(i, j) = (1, 1)$. In general, by Banerjee's approach, the possible dependencies are described by the equation system

$$pi + j = pi' + j'. \tag{1.1}$$

Let us see, how the set of integral solutions can be found. To this end, we rewrite (1.1) as

$$(i, j, i', j')\mathbf{A} = \mathbf{b}. \tag{1.2}$$

with $\mathbf{A} = (p, 1, -p, -1)^t$ and $\mathbf{b} = 0$. As described in Section 2.2, we first have to find some Echelon matrix $\mathbf{S}$ and some uni-modular matrix $\mathbf{U}$ such that

$$\mathbf{S} = \mathbf{UA}. \tag{1.3}$$

The set of solutions to (1.2) is now given by

$$\{\mathbf{tU} \mid \mathbf{t} \in \mathbb{Z}^4 \wedge \mathbf{tS} = \mathbf{b}\}. \tag{1.4}$$

For our example we find

$$\mathbf{S} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \mathbf{U} = \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & -p \\ 1 & 0 & 0 & p \end{pmatrix}. \tag{1.5}$$

Now, the set of vectors $\mathbf{t} = (t_1, t_2, t_3, t_4) \in \mathbb{Z}^4$ with $\mathbf{tS} = 0$ is given by

$$\{(0, t_2, t_3, t_4) \mid t_2, t_3, t_4 \in \mathbb{Z}\}$$

and therefore, the set of solutions to (1.1) is the set

$$\{(t_4, t_2, t_3, t_2 - pt_3 + pt_4) \mid t_2, t_3, t_4 \in \mathbb{Z}\}.$$

Note that we did not say how to find $\mathbf{S}$ and $\mathbf{U}$ in (1.5) – which is straightforward in the given example but will turn out to be harder in general. The systematic construction of solutions to systems of linear Diophantine equations of type (1.1) is the main theme of this thesis. That is, we consider equations in the form of

$$(x_1, \dots, x_m) \begin{pmatrix} f_{11}(\mathbf{p}) & \dots & f_{1n}(\mathbf{p}) \\ \vdots & & \vdots \\ f_{m1}(\mathbf{p}) & \dots & f_{mn}(\mathbf{p}) \end{pmatrix} = (b_1(\mathbf{p}), \dots, b_n(\mathbf{p})) \tag{1.6}$$

with $\mathbf{p} = (p_1, \dots, p_z)$, $f_{ij} \in \mathbb{Z}[X_1, \dots, X_z]$, $b_i \in \mathbb{Z}[X_1, \dots, X_z]$ and $x_i \in \mathbb{Z}$ and ask

- for which $\mathbf{p} \in \mathbb{Z}^z$ do solutions exist, and
- if there are solutions for certain $\mathbf{p}$, can we describe the set of *all* solutions to (1.6)?

We will give a positive answer for the case $z = 1$ and show that one can find a finite tree where each branch describes conditions φ in the single parameter p and each leaf describes (in a uniform way) the set of all solutions to (1.6) for the case that the condition $\varphi(p)$ of the branch leading to the respective leaf is true. For $z \geq 2$ our approach fails in its full generality and we will see why. Under certain circumstances it will, however, be possible to carry the results over to the multi-parametric case and we will describe some conditions where this happens.

This work is organized as follows. Chapter 2 discusses the mathematical prerequisites. Basic facts from elementary number theory such as integral division or the Chinese Remainder Theorem will be presented in Section 2.1. The reader may quickly skim through these pages and come back later, when the results are used. Section 2.3 studies a special kind of quasi-polynomials called aiq-polynomials. They essentially contain the information necessary to construct the decision trees mentioned above.

In the final analysis, solving linear Diophantine equations is nothing but the computation of gcd's with the Extended Euclidean Algorithm (hidden behind the term Echelon reduction). Therefore, Section 2.2 studies this well known algorithm once more in depth and shows how it is used to solve linear Diophantine equations.

Chapter 3 explains Banerjee's data dependence analysis. It is by no means an ersatz for studying Banerjee's original work ([Ban93]) but explains the main concepts and core ideas.

In Chapter 4, we study how the process of solving linear (non-parametric) Diophantine equations can be mimicked for one parameter (Section 4.1) and why it is problematic in the multi-parametric setting (Section 4.2).

Chapter 5 critically reflects the achievements and gives an outlook on future work.

Chapter 2

Mathematical Prerequisites

2.1 Integers

We repeat some well known facts about the ring of integers $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$, integer polynomials $f \in \mathbb{Z}[X]$ and their respective polynomial functions.

Remark 1 (Notation) Throughout this thesis, $\mathbb{N} = \{0, 1, \dots\}$ denotes the set of non-negative integers. We will use $\mathbb{N}_{\geq k}$ to denote the set $\{n \mid n \in \mathbb{N} \wedge n \geq k\}$. If $f : M \longrightarrow N$ is a mapping from M to N then $f(M) := \{n \mid n = f(m) \wedge m \in M\}$, as in $5\mathbb{Z} + 7 = \{n \mid n = 5z + 7 \wedge z \in \mathbb{Z}\}$.

We start with some facts about the functions $\lfloor \cdot \rfloor$ (*floor*) and $\lceil \cdot \rceil$ (*ceil*) which are defined by

$$\lfloor x \rfloor := \max\{z \in \mathbb{Z} \mid z \leq x\} \tag{2.1}$$

and

$$\lceil x \rceil := \min\{z \in \mathbb{Z} \mid z \geq x\}. \tag{2.2}$$

for all $x \in \mathbb{R}$. Because of (i) in the following lemma, the properties are formulated for $\lfloor \cdot \rfloor$ only.

Lemma 2 (Ceil and Floor) Let $x, y \in \mathbb{R}$. Then

(i) $\lceil x \rceil = -\lfloor -x \rfloor$

(ii) $\lfloor x \rfloor \leq x < \lfloor x \rfloor + 1$

(iii) $\lfloor x \rfloor \geq x \Rightarrow \lfloor x \rfloor = x$ (i.e., $x \in \mathbb{Z}$).

(iv) $x < y \Rightarrow \lfloor x \rfloor \leq \lfloor y \rfloor$

(v) $\lfloor x \rfloor < \lfloor y \rfloor \Rightarrow x < y$.

Proof. (i) follows from $\max D = -\min -D$ for every $D \subseteq \mathbb{R}$ (assuming that the maximum actually exists). (ii)-(v) follow directly from the definition. □

Division in $\mathbb{Z}$ Let a, $b \in \mathbb{Z}$. If $as = b$ for some $s \in \mathbb{Z}$ we say that a *divides* b, that a is a *divisor* of b, or that b *is divisible* by a and write $a \mid b$.

Lemma 3 (Properties of "|") Let a, b, $c \in \mathbb{Z}$. Then

(i) $a \mid a$.

(ii) if $a \mid b$ and $b \mid c$ then $a \mid c$.

(iii) if $a \mid b$ and $b \mid a$ then $a = \pm b$.

(iv) if $a \mid b$ and $a \mid c$ then $a \mid (b + c)$.

(v) if $a \mid b$ then $a \mid bc$.

(vi) $a \mid b$ iff $a \mid -b$ iff $-a \mid b$ iff $-a \mid -b$.

(vii) $a \mid 0$, but $0 \mid a$ iff $a = 0$.

(viii) $\pm 1 \mid a$.

Proof. Follows directly from the definition of $\mid$. □

By properties (i) and (ii) of the preceding lemma, $\mid$ is a quasi-order. It cannot be a partial order, since for all $a \in \mathbb{Z} - \{0\}$: $a \mid -a$ iff $-a \mid a$ but of course $a \neq -a$. We can alter this situation by defining an equivalence relation $\sim$ on $\mathbb{Z}$:

$$a \sim b \text{ :iff } a = (\pm 1)b;$$

in this situation we call a and b *associated*.

A replacement for division in $\mathbb{Z}$ is given by the notion of *quotient* and *remainder* of two integers a and b as supplied by the following, well known proposition.

Proposition 4 *Let a, $b \in \mathbb{Z}$. Then a and b uniquely determine a* quotient $q \in \mathbb{Z}$ *and a* remainder $r \in \{0, \dots, |b| - 1\}$ *such that*

$$a = qb + r. \tag{2.3}$$

Proof. See [Leu96], p. 13. □

A constructive approach to quotients and remainders is, for instance, given in [WBK93], Algorithm DIVINT on p. 12. A note to the reader looking up this source – what we strongly recommend: It is interesting to observe that the algorithm given there is in fact a little more complicated than one would expect at first. This is due to the fact that one requires the remainder r to be non-negative and one therefore has to treat the quotient "in the right way." The problem will become clear if we ask for a way to express the q and r of the last proposition in terms of a, b and common functions like $|\cdot|$ or $\lfloor \frac{\cdot}{\cdot} \rfloor$:

$$\begin{aligned} q &= \operatorname{sgn}(b) \left\lfloor \frac{a}{|b|} \right\rfloor \\ r &= a - b \operatorname{sgn}(b) \left\lfloor \frac{a}{|b|} \right\rfloor \\ &= a - |b| \left\lfloor \frac{a}{|b|} \right\rfloor \end{aligned}$$

While the last proposition is *remainder centric* (because it requires the remainder to be non-negative), the programing language Haskell provides two more *quotient centric* approaches to integral division that require the remainder only to be between $-|b|$ and $|b|$ (see [Jon02], sec. 6.4.2). In Haskell, the following equations hold for the integer division methods *quot* and *div*, assuming $b \neq 0$:

- $quot\ a\ b = \operatorname{sgn}(ab) \left\lfloor \frac{|a|}{|b|} \right\rfloor$, i.e., $\frac{a}{b}$ is truncated toward zero.
- $div\ a\ b = \lfloor \frac{a}{b} \rfloor$, i.e., $\frac{a}{b}$ is truncated toward minus infinity.

The respective remainder methods *rem* and *mod* are such that they satisfy the following laws ($b \neq 0$):

- $(x\ `quot`\ y) * y\ +\ (x\ `rem`\ y)\ ==\ x$
- $(x\ `div`\ y) * y\ +\ (x\ `mod`\ y)\ ==\ x$

For the purpose of this work, we only concentrate on the second versions, *div* and *mod*, and show the relation to their number theoretic siblings from Proposition 4.

Lemma 5 Let $a \in \mathbb{Z}$, $b \in \mathbb{Z} - \{0\}$. Define $q := \lfloor \frac{a}{b} \rfloor$ (i.e., $q = a\ `div`\ b$) and $r := a - qb$. Then

$$|r| < |b|. \tag{2.4}$$

In particular, if $b > 0$ then q and r coincide with the quotient and remainder given in Proposition 4.

Proof. Let $a \in \mathbb{Z}$, $b \in \mathbb{Z} - \{0\}$. Then by Lemma 2

$$\left\lfloor \frac{a}{b} \right\rfloor - 1 < \frac{a}{b} < \left\lfloor \frac{a}{b} \right\rfloor + 1.$$

If $b > 0$, then

$$\left\lfloor \frac{a}{b} \right\rfloor b - b < a < \left\lfloor \frac{a}{b} \right\rfloor b + b$$

and so

$$-b < a - \left\lfloor \frac{a}{b} \right\rfloor b < b.$$

I.e.,

$$-b < r < b$$

by definition of r. Similarly, if $b < 0$, then

$$-b > r > b.$$

In any case, inequality (2.4) follows. To prove the second statement, let q and r be as defined and observe that $b > 0$ implies $r \geq 0$ (again use $\lfloor x \rfloor \leq x$ for all $x \in \mathbb{Q}$):

$$q = \left\lfloor \frac{a}{b} \right\rfloor \leq \frac{a}{b} \text{ and } b > 0 \quad \Rightarrow \quad qb \leq a \quad \Rightarrow \quad 0 \leq a - qb = r.$$

Since $r \leq |b| - 1$ and of course $a = qb + r$, we know by Proposition 4 that q and r are uniquely determined and hence must be equal to the quotient and remainder given there. □

From now on, the functions div and mod are used in the same way they are used in Haskell

$$\mathrm{div} : \mathbb{Z} \times \mathbb{Z} - \{0\} \longrightarrow \mathbb{Z} : (a, b) \mapsto \left\lfloor \frac{a}{b} \right\rfloor$$

and

$$\mathrm{mod} : \mathbb{Z} \times \mathbb{Z} - \{0\} \longrightarrow \mathbb{Z} : (a, b) \mapsto a - b \left\lfloor \frac{a}{b} \right\rfloor$$

except that we will write them infix without back-ticks.

Congruences, gcd's and lcm's Let $l \in \mathbb{N}_{\geq 1}$ and let the equivalence relation $\equiv_l$ on $\mathbb{Z}$ be defined by

$$a \equiv_l b \quad \text{iff} \quad l \mid (a - b). \tag{2.5}$$

Note that in case $l = 1$ we have $a \equiv_1 b$ for all $a, b \in \mathbb{Z}$. For given $l \in \mathbb{Z}$,

$$[a]_l := \{x \in \mathbb{Z} \mid x \equiv_l a\}$$

denotes the equivalence class of *a modulo l*. It is well known that for $l \geq 1$, there are exactly l different equivalence classes that partition $\mathbb{Z}$:

$$\mathbb{Z} = [0]_l \cup [1]_l \cup \cdots \cup [l - 1]_l.$$

Therefore, each equivalence class modulo l has a representative within $\{0, \ldots, l - 1\}$ which we call the *normal representative (modulo l)*. Note that for any $p \in \mathbb{Z}$ its normal representative is given by $p \bmod l$. Note that

$$[a]_l = a + l\mathbb{Z}.$$

Sometimes, we prefer the notation on the right hand side, especially if $a = 0$. The set of equivalence classes modulo l itself is denoted by $\mathbb{Z}/l\mathbb{Z}$.

Proposition 6 *Let $a, a', b, b' \in \mathbb{Z}$ and $l \in \mathbb{N}_{\geq 1}$. Let $f \in \mathbb{Z}[X]$.*

(i) If $a \equiv_l a'$ and $b \equiv_l b'$ then $a + b \equiv_l a' + b'$ and $ab \equiv_l a'b'$.

(ii) If $a \equiv_l a'$ then $f(a) \equiv_l f(a')$.

Proof. (i) follows easily from the definitions. To prove (ii), observe that with $f(X) = a_0 + a_1X + \cdots + a_nX^n$ we get

$$\begin{aligned} f(a) - f(a') &= (a_0 + a_1a + \cdots + a_na^n) - (a_0 + a_1a' + \cdots + a_na'^n) \\ &= a_1(a - a') + \cdots + a_n(a^n - a'^n). \end{aligned}$$

By repeated application of (i) we derive $a^i \equiv_l a'^i$ for all $1 \leq i \leq n$, and therefore, $l \mid a^i - a'^i$. With Lemma 3 this implies $l \mid f(a) - f(a')$, i.e., $f(a) \equiv_l f(a')$. □

An integer d is called *greatest common divisor* (*gcd*) of $a_1, \ldots, a_n \in \mathbb{Z}$, if

(i) d is a common divisor of the a_i and

(ii) any further divisor d' of $a_1, \ldots, a_n$ also divides d.

Note that we do not require $d \geq 0$. Therefore, if d is a gcd of the a_i, then so is $-d$. On the other hand, if d and d' are any two gcd's of $a_1, \dots, a_n$, then $d \mid d'$ and $d' \mid d$ by definition. With Lemma 3(iii) we get $d = \pm d'$. This shows that the gcd of $a_1, \dots, a_n$ is determined *uniquely up to multiplication by* -1. In other words, the gcd is determined uniquely up to association. So we can write $d \sim \gcd(a_1, \dots, a_n)$ if d is any gcd of $a_1, \dots, a_n$. Two integers a, b are *coprime* if $\gcd(a, b) \sim 1$.

Let us repeat some elementary properties of gcd (cf. [Ban93], p. 55 and [Leu96], p. 15):

Lemma 7 (Properties of gcd) Let $a_1, \dots, a_n \in \mathbb{Z}$ $(n \geq 1)$. Then

(i) $\gcd(a_1, \dots, a_n) \sim \gcd(\pm a_1, \dots, \pm a_n)$ or in other words $\gcd(a_1, \dots, a_n) \sim \gcd(\pm b_1, \dots, \pm b_n)$ whenever $a_i \sim b_i$,

(ii) $\gcd(a_1, 0, \dots, 0) \sim a_1$,

(iii) $\gcd(1, a_1, \dots, a_n) \sim 1$,

(iv) $\gcd(a_1, \dots, a_n) \sim \gcd(a_{\pi(1)}, \dots, a_{\pi(n)})$
for any permutation $\pi : \{1, \dots, n\} \to \{1, \dots, n\}$,

(v) $\gcd(ka_1, \dots, ka_n) \sim k \cdot \gcd(a_1, \dots, a_n)$ for any $k \in \mathbb{Z}$,

(vi) $\gcd(a_1, a_2, \dots, a_n) \sim \gcd(a_1 - qa_2, a_2, \dots, a_n)$ for any $q \in \mathbb{Z}$,

(vii) $\gcd(a_1, a_2, \dots, a_n) \sim \gcd(a_1, \gcd(a_2, \dots, a_n))$ if $n \geq 3$,

(viii) $\gcd(a_1, \dots, a_n) \mid \gcd(ka_1, a_2, \dots, a_n)$ for all $k \in \mathbb{Z}$,

(ix) the a_i have a gcd d for which $d\mathbb{Z} = a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$ holds.

Further, let $a, b, c \in \mathbb{Z}$ with $\gcd(a, c) \sim 1$. Then

(x) $\gcd(ab, c) \sim \gcd(b, c)$.

Proof. For (i)-(vii) consult [Ban93], p. 55 and [Leu96], p.15. For (viii), note that $d \mid a_1 \implies d \mid ka_1$ (Lemma 3). Next, we prove (ix) in two steps: First we show that there is some $d \in \mathbb{N}_{\geq 1}$ such that

$$d\mathbb{Z} = a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$$

and second we reveal d as a gcd of the a_i. So let us define

$$d := \min \mathbb{N}_{\geq 1} \cap (a_1\mathbb{Z} + \cdots + a_n\mathbb{Z})$$

which must exist because each $a_i\mathbb{Z}$ and therefore $a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$, too, contain positive elements. Thus there are $s_1, \dots, s_n \in \mathbb{Z}$ such that $d = s_1a_1 + \cdots + s_na_n$. Now, if $p \in d\mathbb{Z}$, there is some p' such that

$$\begin{aligned} p &= p'd \\ &= p'(s_1a_1 + \cdots + s_na_n) \\ &= (p's_1)a_i + \cdots + (p's_n)a_n \\ &\in a_1\mathbb{Z} + \cdots + a_n\mathbb{Z} \end{aligned}$$

which means $d\mathbb{Z} \subseteq a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$. Next, let $p \in a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$ and set $q := \lfloor \frac{p}{d} \rfloor$. Observe that for any $x, y \in a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$ also $(x - ky) \in a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$ for any $k \in \mathbb{Z}$. Therefore, $r := p - qd$ must also be in $a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$. But by Lemma 5, we have $0 \leq r < d$. Because d was chosen as smallest positive integer in $a_1\mathbb{Z} + \cdots + a_n\mathbb{Z}$, r must be 0. It follows that $d \mid p$ and in particular $p \in d\mathbb{Z}$.

Hence, d also divides any a_i, i.e., d is a common divisor of all a_i. Let d' be any further common divisor such that $a_i = d'a_i'$ $(1 \leq i \leq n)$. Then

$$\begin{aligned} d &= s_1a_1 + \cdots + s_na_n \\ &= s_1(d'a_1') + \cdots + s_n(d'a_n') \\ &= d'(s_1a_1' + \cdots + s_na_n') \end{aligned}$$

which means $d' \mid d$. It follows that d must be a gcd.
(x) Finally, let $a, b, c \in \mathbb{Z}$ with $\gcd(a, c) \sim 1$. Let $d := |\gcd(ab, c)|$ and let $d' := |\gcd(b, c)|$. We have to show $d \mid d'$ and $d' \mid d$. The latter relation follows directly from (viii). To prove $d \mid d'$, we show that $d' \in d\mathbb{Z} = ab\mathbb{Z} + c\mathbb{Z}$. Since $\gcd(a, c) \sim 1$, by the proof of (ix) above there are $s, t \in \mathbb{Z}$ such that $as + ct = 1$. By the same reasoning there are $u, v \in \mathbb{Z}$ such that $bu + cv = d'$. Therefore,

$$\begin{aligned} d' &= bu + cv \\ &= bu(as + ct) + cv \\ &= ab(us) + c(but + v) \end{aligned}$$

which implies $d' \in \ d\mathbb{Z}$. □

Statement (ix) proves the existence of gcd's, unfortunately, in an unconstructive way. A constructive approach will be given later, when we study the Euclidean Algorithm.

An integer v is called *least common multiple* (*lcm*) of $a_1, \ldots, a_n \in \mathbb{Z}$ if

(i) v is a common multiple of all a_i

(ii) for any further common multiple v' of the a_i it holds $v \mid v'$.

As with the gcd, the lcm is only determined up to multiplication by -1. Since the lcm plays only a minor role within our work, we merely provide its most important properties.

Lemma 8 Let $a_1, \ldots, a_n \in \mathbb{Z}$ $(n \geq 1)$. Then the following statements hold:

(i) There is some $v \in \mathbb{Z}$ such that $v\mathbb{Z} = a_1\mathbb{Z} \cap \cdots \cap a_n\mathbb{Z}$ and $v \sim \operatorname{lcm}(a_1, \ldots, a_n)$,

(ii) $\operatorname{lcm}(a_1, a_2, \ldots, a_n) \sim \operatorname{lcm}(a_1, \operatorname{lcm}(a_2, \ldots, a_n))$ if $n \geq 3$,

(iii) If the a_i are pairwise coprime then $\operatorname{lcm}(a_1, \ldots, a_n) \sim a_1 \ldots a_n$.

Furthermore, let $a, b \in \mathbb{Z}$. Then

(iv) $\gcd(a, b) \operatorname{lcm}(a, b) \sim ab$.

Proof. (i) The proof is very similar to the proof of Lemma 7(ix) and we leave it to the reader.
(ii)

$$\begin{aligned}\operatorname{lcm}(a_1, a_2, \ldots, a_n)\mathbb{Z} &= a_1\mathbb{Z} \cap (a_2\mathbb{Z} \cdots \cap a_n\mathbb{Z})\\ &= a_1\mathbb{Z} \cap \operatorname{lcm}(a_2, \ldots, a_n)\mathbb{Z}\\ &= lcm(a_1, \operatorname{lcm}(a_2, \ldots, a_n))\mathbb{Z}\end{aligned}$$

(iv) See [Leu96], p.17.
(iii) We conduct a proof by induction on n. If $n = 2$ we get $\operatorname{lcm}(a_1, a_2) \sim a_1a_2$ by (iv) because $\gcd(a_1, a_2) \sim 1$. If $n + 1 \geq 3$ then

$$\begin{aligned}\operatorname{lcm}(a_1, \ldots, a_n, a_{n+1}) &\sim \operatorname{lcm}(\operatorname{lcm}(a_1, \ldots, a_n), a_{n+1})\\ &\sim \operatorname{lcm}(a_1 \ldots a_n, a_{n+1})\end{aligned}$$

by induction hypothesis. Again by (iv), $\gcd(a_1 \ldots a_n, a_{n+1}) \operatorname{lcm}(a_1 \ldots a_n, a_{n+1}) \sim a_1 \ldots a_na_{n+1}$ and by repeated application of Lemma 7(x), $\gcd(a_1 \ldots a_n, a_{n+1}) \sim 1$ which proves our claim. □

The Chinese Remainder Theorem Lemma 8(i) is a special case of the following problem: Let $a_1, \ldots, a_n \in \mathbb{Z}$ and $l_1, \ldots, l_n \in \mathbb{N}_{\geq 1}$ be given. Is it possible to find some $a \in \mathbb{Z}$ and $l \in \mathbb{N}_{\geq 1}$ such that

$$[a]_l = [a_1]_{l_1} \cap \cdots \cap [a_n]_{l_n}. \tag{2.6}$$

Lemma 9 Let $a_1, a_2 \in \mathbb{Z}$ and $l_1, l_2 \in \mathbb{N}_{\geq 2}$. Then there is some $a \in \mathbb{Z}$ such that $[a_1]_{l_1} \cap [a_2]_{l_2} = [a]_{\operatorname{lcm}(l_1,l_2)}$ iff $\gcd(l_1, l_2) \mid (a_2 - a_1)$.

Proof. There is some $a \in [a_1]_{l_1} \cap [a_2]_{l_2}$ iff there are $x_1, x_2 \in \mathbb{Z}$ such that

$$\begin{aligned}a &= a_1 + l_1x_1\\ a &= a_2 + l_2x_2\end{aligned}$$

iff there are $x_1, x_2 \in \mathbb{Z}$ such that

$$l_1x_1 - l_2x_2 = a_2 - a_1. \tag{2.7}$$

This equation in turn has a solution iff $\gcd(l_1, l_2) \mid (a_2 - a_1)$. So, let us assume that (2.7) has a solution $(x_1, x_2) = (s_1, s_2)$. Let $a := a_1 + l_1s_1$ such that $a \in [a_1]_{l_1} \cap [a_2]_{l_2}$ (and also $a = a_2 + l_2s_2$) and set $l := \operatorname{lcm}(l_1, l_2)$. It remains to show that

$$[a_1]_{l_1} \cap [a_2]_{l_2} = [a]_{\operatorname{lcm}(l_1,l_2)}.$$

"⊆": Let $a' \in [a_1]_{l_1} \cap [a_2]_{l_2}$. Then $a_1 \equiv_{l_1} a'$ and $a_2 \equiv_{l_2} a'$. Because

$$\begin{aligned}a_1 \equiv_{l_1} a' &\Leftrightarrow a - l_1s_1 \equiv_{l_1} a'\\ &\Leftrightarrow l_1 \mid (a - a') - l_1s_1\\ &\Leftrightarrow l_1 \mid (a - a')\end{aligned}$$

and also

$$a_2 \equiv_{l_2} a' \Leftrightarrow l_2 \mid (a - a')$$

$(a - a')$ is a common multiple of both l_1 and l_2 and therefore divisible by l, i.e., $a \equiv_l a'$ and therefore $a' \in [a]_l$.
"$\supseteq$": Let $a' \in [a]_l$. Then $l \mid (a - a')$. Because l is a multiple of l_1 and because $a \equiv_{l_1} a_1$ by our assumptions we get

$$\begin{aligned} l_1 \mid (a - a') &\Longrightarrow a \equiv_{l_1} a' \\ &\Longrightarrow a_1 \equiv_{l_1} a' \\ &\Longrightarrow a' \in [a_1]_{l_1} \end{aligned}$$

and in the same way since $l_2 \mid l$

$$a' \in [a_2]_{l_2}.$$

This shows $a' \in [a_1]_{l_1} \cap [a_2]_{l_2}$ and hence concludes the proof. □

Proposition 10 (Chinese Remainder Theorem) *Let* $[a_1]_{l_1}, \ldots, [a_n]_{l_n} \subseteq \mathbb{Z}$ $(l_i \geq 2)$. *Then*

$$\bigcap_{i=1}^{n} [a_i]_{l_1}$$

is either empty or equals $[a]_{\mathrm{lcm}(l_1,\ldots,l_n)}$ *for any* $a \in \bigcap_{i=1}^{n} [a_i]_{l_1}$

Proof. The proof is a simple induction on n. If $n = 2$, the statement reduces to Lemma 9. So assume the statement is true for $n > 2$. Then $M := \bigcap_{i=1}^{n} [a_i]_{l_1}$ is either empty in which case $M \cap [a_{n+1}]_{l_{n+1}}$ is empty, too. Or, $M = [a']_{\mathrm{lcm}(l_1,\ldots,l_n)}$ for some $a' \in \mathbb{Z}$. Then, again by Lemma 9, $M \cap [a_{n+1}]_{l_{n+1}}$ is either empty or there is some $a \in \mathbb{Z}$ such that

$$\begin{aligned} M \cap [a_{n+1}]_{l_{n+1}} &= [a']_{\mathrm{lcm}(l_1,\ldots,l_n)} \cap [a_{n+1}]_{l_{n+1}} \\ &= [a]_{\mathrm{lcm}(\mathrm{lcm}(l_1,\ldots,l_n),l_{n+1})} \\ &= [a]_{\mathrm{lcm}(l_1,\ldots,l_n,l_{n+1})} \end{aligned}$$ □

Often the following corollary is presented as Chinese Remainder Theorem, probably due to the fact that it guarantees non-emptiness of the intersection in question.

Corollary 11 If $l_1, \ldots, l_n$ are pairwise coprime then

$$\bigcap_{i=1}^{n} [a_i]_{l_i} \neq \emptyset.$$

Proof. by induction on n. If $n = 2$

$$\gcd(l_1, l_2) \sim 1 \Longrightarrow \gcd(l_1, l_2) \mid (a_1 - a_1)$$

and $[a_1]_{l_1} \cap [a_2]_{l_2} \neq \emptyset$ by Lemma 9. Assume the proposition true for $n > 2$, i.e., there exists some a such that

$$\bigcap_{i=1}^{n} [a_i]_{l_i} = [a]_{\mathrm{lcm}(l_1,\ldots,l_n)}.$$

Since $\gcd(l_i, l_j) \sim 1$ for all $i \neq j$ we obtain

$$\begin{aligned} \gcd(\mathrm{lcm}(l_1, \ldots, l_n), l_{n+1}) &\sim \gcd(l_1 \ldots l_n, l_{n+1}) \\ &\sim 1 \end{aligned}$$

by Lemma 8(iii) and repeated application of Lemma 7(x). Therefore, $\gcd(\mathrm{lcm}(l_1, \ldots, l_n), l_{n+1}) \mid (a_{n+1} - a)$ and there is some a' such that

$$\begin{aligned} [a]_{\mathrm{lcm}(l_1,\ldots,l_n)} \cap [a_{n+1}]_{l+1} &= [a']_{\mathrm{lcm}(\mathrm{lcm}(l_1,\ldots,l_n),l_{n+1})} \\ &= [a']_{\mathrm{lcm}(l_1,\ldots,l_n,l_{n+1})} \qquad \square \end{aligned}$$

The Polynomial Rings $\mathbb{Z}[X]$ and $\mathbb{Q}[X]$ We assume that the reader is familiar with the definitions of $\mathbb{Z}[X]$ and $\mathbb{Q}[X]$ (cf. [WBK93, Section 2.1]) as well as with polynomial division in $\mathbb{Q}[X]$ (cf. [WBK93, Section 2.2]). The *degree* of an polynomial f is denoted by $\deg(f)$ and we write $\mathrm{HC}\,(f)$ for the coefficient of the term with the highest exponent. By the algebraic structure of $\mathbb{Z}$ and $\mathbb{Q}$, it is possible to show that gcd's of polynomials f, g do always exist in both $\mathbb{Z}[X]$ and $\mathbb{Q}[X]$. While it is possible to determine $s, t \in \mathbb{Q}[X]$ for given $f, g \in \mathbb{Q}[X]$ such that

$$sf + tg \sim \gcd(f, g)$$

([WBK93, Theorem 2.32]), the same is not any more true for arbitrary $f, g \in \mathbb{Z}[X]$. Just consider, for instance, the polynomials $2, X \in \mathbb{Z}[X]$. We only note that gcd's in $\mathbb{Z}[X]$ can be computed with the aid of $\mathbb{Q}[X]$ as described in [WBK93, Section 2.5].

2.2 Linear Diophantine Equations

An equation of the form

$$f(a_1, \ldots, a_n) = 0 \tag{2.8}$$

where $f \in \mathbb{Z}[X_1,\ldots,X_n]$ and $a_i \in \mathbb{Z}$ is called *Diophantine equation.* In 1970, I. Matiyasevich showed that in the general case the existence of solutions for (2.8) is undecidable ([Mat70]). It is however possible to solve systems of *linear* Diophantine equations

$$\mathbf{Ax} = \mathbf{b} \tag{2.9}$$

where $\mathbf{A} \in \mathbb{Z}^{m \times n}$, $\mathbf{b} \in \mathbb{Z}^n$ and the variables $\mathbf{x} \in \mathbb{Z}^m$. Solving linear equations over the integers differs from solving such equations over the rationals or the reals, since in $\mathbb{Z}$ no division is available – while $\mathbb{Q}$ and $\mathbb{R}$ are fields, $\mathbb{Z}$ is a ring.

We will see in a moment that solving (2.9) is intimately connected to the computation of certain gcd's, which in turn is done by the well known Euclidean Algorithm.

The Euclidean Algorithm Revised By property (vii) of Lemma 7, the computation of $\gcd(a_1, \ldots, a_n)$ can be reduced to computing the gcd of only two of the numbers. This is done by Algorithms 1 and 2. Even though well known, they lie at the heart of our extension to Banerjee's data dependence analysis (Section 4.1.4) and therefore, we state them explicitly.

Remark 12 The algorithms presented in this work are written in a Haskell-like pseudocode that uses common mathematical notation. When vectors or matrices are involved, the signature indicates this, using $Vec_n(R)$ for vectors in R^n and $Mat_{m\times n}(R)$ for matrices in $R^{m\times n}$. If $\mathbf{x} = (x_1, \ldots, x_n)$ is a vector we write $\mathbf{x}_{\geq k}$ for $(x_k, \ldots, x_n)$. Similarly, if $\mathbf{A}$ is a $m \times n$-matrix, $\mathbf{A}_{\geq(2,2)}$ denotes the matrix one obtains by deleting the upper row and the left-most column. Sometimes we will use list-notation to denote vectors. That way, we can take advantage of some convenient, well-known Haskell functions as, for example, *map*.

Algorithm 1 Euclidean Algorithm for Two Integers

```
$gcd_2$       :: $Vec_2(\mathbb{Z}) \to Vec_2(\mathbb{Z})$
$gcd_2 \binom{a}{0} = \binom{a}{0}$
$gcd_2 \binom{a}{b} = gcd_2 \binom{b}{a-\lfloor\frac{a}{b}\rfloor b}$
```

Theorem 13 *For any given $a, b \in \mathbb{Z}$, Algorithm 1 terminates after a finite number of steps. Its output is of the form $\binom{d}{0}$, where $d \sim \gcd(a,b)$.*

Proof. We first show termination. Let $\{\binom{a_i}{b_i}\}_{i\in\mathbb{N}}$ be the sequence of consecutive instances in the calls of gcd_2 in Algorithm 1, i.e., let

$$\begin{pmatrix} a_0 \\ b_0 \end{pmatrix} := \begin{pmatrix} a \\ b \end{pmatrix} \tag{2.10}$$

$$\begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} := \begin{pmatrix} b_i \\ a_i - \left\lfloor \frac{a_i}{b_i} \right\rfloor b_i \end{pmatrix}. \tag{2.11}$$

By Lemma 5, $|b_i| > |b_{i+1}|$ for all $i \in \mathbb{N}$. Since each $|b_i| \in \mathbb{N}$, and there are no infinitely decreasing sequences in $\mathbb{N}$, we are done. In particular, there is some $n_0 \in \mathbb{N}$, such that $b_{n_0} = 0$. Therefore, the output is of the required form and with Lemma 7 correctness follows. □

Algorithm 2 Euclidean Algorithm for $n \geq 2$ Integers

```
$gcd$            :: $[Integer] \to [Integer]$
$gcd\ []$          $= []$
$gcd\ [a_1]$        $= [a_1]$
$gcd\ (a_1 : as) = (gcd_2\ [a_1,\ a_2']) : as'$
  where
    $a_2'$ $= head\ (gcd\ as)$
    $as'$ $= tail\ (gcd\ as)$
```

Corollary 14 Let $a_1, \ldots, a_n \in \mathbb{Z}$. Then Algorithm 2 terminates and $gcd\ [a_1, \ldots, a_n] = [d, 0, \ldots, 0]$ where $d \sim \gcd(a_1, \ldots, a_n)$.

Proof. The proof follows by an easy induction on the length n of the input list $[a_1, \dots, a_n]$. □

More important than mere computations of gcd's in $\mathbb{Z}$ is the *extended* version of the Euclidean Algorithm that will (for given $a_1, \dots, a_n \in \mathbb{Z}$) compute integers $u_1, \dots, u_n$, such that

$$u_1 a_1 + \cdots + u_n a_n \sim \gcd(a_1, \dots, a_n). \tag{2.12}$$

For this task, we need the notion of an unimodular matrix, as given in the following definition.

Definition 15 (Unimodular Matrix) Let $\mathbf{A} \in \mathbb{Z}^{n \times n}$. Then $\mathbf{A}$ is called *unimodular* if

$$\det \mathbf{A} \sim 1.$$

Lemma 16 (Properties of Unimodular Matrices) Let $\mathbf{A}, \mathbf{B} \in \mathbb{Z}^{n \times n}$ be unimodular matrices. Then the following propositions hold.

(i) $\mathbf{A}^{-1} \in \mathbb{Z}^{n \times n}$ exists and is unimodular.

(ii) $\mathbf{AB}$ is unimodular.

(iii) $\mathbf{A}^t$ is unimodular.

Proof. (i) follows from [Sch87, Theorem 4.3]. (ii) and (iii) follow from $\det \mathbf{AB} = \det \mathbf{A} \det \mathbf{B}$ and $\det \mathbf{A}^t = \det \mathbf{A}$ which can be found in any textbook on linear algebra. □

Example 17 Elementary row and column operations can be carried out by pre- and post-multiplication by corresponding elementary matrices. Moreover, it can be shown that every unimodular matrix is the product of (is generated by) elementary matrices. See [Ban93], pp.28-31 and Lemma 2.3, p. 45.

Recall that there are three *elementary row (column) operations* operating on integral matrices ([Ban93], p.28):

(1) reversal: Multiplying a row (column) by -1;

(2) interchange: Interchange two rows (columns);

(3) skewing: Add an integer multiple of one row (column) to another row (column).

These operations do not change the absolute value of the determinant of a the manipulated matrix.

The class of *elementary matrices* is obtained by application of *one* elementary row operation to the identity matrix I_n (note that applying column operations would yield exactly the same class of matrices). For instance, let us consider some elementary 2×2-matrices:

Multiply row (column) 1 by -1: $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$

Interchange rows (columns) 1 and 2: $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Add row 2 z-times to row 1 (of column 1 to column 2): $\begin{pmatrix} 1 & z \\ 0 & 1 \end{pmatrix}$

In general, to each elementary row (column) operation, there exists a corresponding elementary matrix; and to each elementary matrix there is one corresponding elementary row operation and one corresponding column operation.

Lemma 18 Let $\mathbf{U} \in \mathbb{Z}^{n \times n}$ be unimodular, $(a_1, \dots, a_n)^t \in \mathbb{Z}^n$ and

$$(a'_1, \dots, a'_n)^t := \mathbf{U}(a_1, \dots, a_n)^t.$$

Then

$$\gcd(a_1, \dots, a_n) \sim \gcd(a'_1, \dots, a'_n).$$

Proof. If $\mathbf{U}$ is an elementary matrix the proposition is immediately clear from Lemma 7(i), (iv) and (vi). The general case follows from the fact that any unimodular matrix can be decomposed into elementary matrices ([Sch87, Theorem 4.3]). □

Now we can turn back to the problem of finding integers $u_1, \dots, u_n$, such that (2.12) holds. Therefore, we consider the $n \times 1$ matrix $\mathbf{a} := (a_1, \dots, a_n)^t$. Our goal is to construct a unimodular matrix $\mathbf{U}$ that reflects the steps carried out by the Euclidean Algorithm given above:

$$\mathbf{Ua} = (d, 0, \dots, 0)^t \tag{2.13}$$

where $d \sim \gcd(a_1, \dots, a_n)$. The first row $(u_{11}, \dots, u_{1n})$ of $\mathbf{U}$ then obviously gives a solution to (2.12) (let $u_i := u_{1i}$).

To understand how $\mathbf{U}$ is constructed, we look again at the 2×2-case. Recall the finite sequence $\{\binom{a_i}{b_i}\}_{i \in \mathbb{N}}$ defined by (2.10) above, where

$$\begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -\lfloor \frac{a_i}{b_i} \rfloor \end{pmatrix} \begin{pmatrix} a_i \\ b_i \end{pmatrix}. \tag{2.14}$$

Let

$$\mathbf{U}_i := \begin{pmatrix} 0 & 1 \\ 1 & -\lfloor \frac{a_i}{b_i} \rfloor \end{pmatrix}. \tag{2.15}$$

Then

$$\begin{aligned} \begin{pmatrix} a_{i+1} \\ b_{i+1} \end{pmatrix} &= \mathbf{U}_i \cdot \dots \cdot \mathbf{U}_0 \begin{pmatrix} a_0 \\ b_0 \end{pmatrix} \\ &= \mathbf{U}_i \cdot \dots \cdot \mathbf{U}_0 \begin{pmatrix} a \\ b \end{pmatrix} \end{aligned}$$

and in particular, if $n_0 \in \mathbb{N}$ is again such that $b_{n_0} = 0$:

$$\begin{pmatrix} d \\ 0 \end{pmatrix} = \mathbf{U}_{n_0 - 1} \cdot \dots \cdot \mathbf{U}_0 \begin{pmatrix} a \\ b \end{pmatrix} \tag{2.16}$$

Note that each $\mathbf{U}_i$ is unimodular since it is a product of two elementary row operations. Algorithm 3 extends Algorithm 1, now taking care of the corresponding matrices.

Algorithm 3 Extended Euclidean Algorithm for Two Integers

```
$extGcd_2$          $:: [Integer] \to ([Integer],\ Mat_{2\times 2}\ (Integer))$
$extGcd_2\ [a, 0] = ([a, 0], \left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right))$
$extGcd_2\ [a, b] = (xs,\ \mathbf{U}_c\mathbf{U})$
  where
    $(xs,\ \mathbf{U}_c) = extGcd_2\ [b,\ a - \lfloor \frac{a}{b} \rfloor\ b]$
    $\mathbf{U}$        $= \left(\begin{smallmatrix}0&1\\1&-\lfloor \frac{a}{b}\rfloor\end{smallmatrix}\right)$
```

Corollary 19 For any given $a,\ b \in \mathbb{Z}$, Algorithm 3 terminates. Its output is of the form $([d, 0], \mathbf{U})$ where $d \sim \gcd(a, b)$ and $\mathbf{U} \in \mathbb{Z}^{2\times 2}$ such that

$$\mathbf{U}\begin{pmatrix}a\\b\end{pmatrix} = \begin{pmatrix}d\\0\end{pmatrix}$$

Example 20 Even though elementary, we depict the sequence of arguments, $extGcd_2[12, 18]$ is called with and the corresponding matrices occurring from call to call. Keep this picture in mind when we study Example 63.

$$\begin{tikzcd}
{} & & & & {} \arrow[llll, "\mathbf{U} = \mathbf{U}_3\mathbf{U}_2\mathbf{U}_1\mathbf{U}_0 = \left(\begin{smallmatrix}-1&1\\3&-2\end{smallmatrix}\right)"'] \\
\left(\begin{smallmatrix}12\\18\end{smallmatrix}\right) \arrow[r, "\mathbf{U}_0 = \left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right)"] & \left(\begin{smallmatrix}18\\12\end{smallmatrix}\right) \arrow[r, "\mathbf{U}_1 = \left(\begin{smallmatrix}0&1\\1&-1\end{smallmatrix}\right)"] & \left(\begin{smallmatrix}12\\6\end{smallmatrix}\right) \arrow[r, "\mathbf{U}_2 = \left(\begin{smallmatrix}0&1\\1&-2\end{smallmatrix}\right)"] & \left(\begin{smallmatrix}6\\0\end{smallmatrix}\right) \arrow[r, "\mathbf{U}_3 = \left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)"] & {} \arrow[u, bend right, no head]
\end{tikzcd}$$

□

To extend the Euclidean Algorithm for input lists of length greater than 2, note that $gcd\ [a_1, a_2, \ldots, a_n]$ first computes $gcd\ [a_2, \ldots, a_n] = [a_2', 0, \ldots, 0]$ and then $gcd_2\ [a_1, a_2'] = [d, 0]$. Therefore, in the extended version, $extGcd\ [a_2, \ldots, a_n]$ will provide us with a $(n-1) \times (n-1)$-matrix $\mathbf{U}_c$, such that

$$\mathbf{U}_c(a_2, \ldots, a_n)^t = (a_2', 0, \ldots, 0)^t \tag{2.17}$$

and $extGcd_2\ [a_1, a_2']$ will return the respective 2×2-matrix $\mathbf{U}$ with

$$\mathbf{U}(a_1, a_2')^t = (d, 0)^t \tag{2.18}$$

To see how these two matrices can be glued together, observe that

$$\begin{pmatrix}1 & \mathbf{0}\\ \mathbf{0} & \mathbf{U}_c\end{pmatrix}(a_1, a_2, \ldots, a_n)^t = (a_1, a_2', 0, \ldots, 0)^t \tag{2.19}$$

and

$$\begin{pmatrix}\mathbf{U} & \mathbf{0}\\ \mathbf{0} & I_{n-2}\end{pmatrix}(a_1, a_2', 0, \ldots, 0)^t = (d, 0, 0, \ldots, 0)^t. \tag{2.20}$$

We need to fill up the matrix bearing $\mathbf{U}$ with I_{n-2}, since otherwise unimodularity would be destroyed.

Algorithm 4 Extended Euclidean Algorithm for $n \geq 2$ Integers

```
$extGcd$            $:: [Integer] \rightarrow ([Integer],\ Mat_{n \times n}(Integer))$
$extGcd\ []$        $= ([],\ ())$
$extGcd\ [a_1]$     $= ([a_1],\ (1))$
$extGcd\ (a_1 : as) = (ds : as',\ \begin{pmatrix} \mathbf{U} & \mathbf{0} \\ \mathbf{0} & I_{n-2} \end{pmatrix} \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_c \end{pmatrix})$
  where
    $(xs,\ \mathbf{U}_c)$ $= extGcd\ as$
    $(ds, \mathbf{U})$  $= extGcd_2\ [a_1,\ a_2']$
    $a_2'$          $= head\ xs$
    $as'$           $= tail\ xs$
    $n$             $= length\ (a1 : as)$
```

Corollary 21 For any given $a_1, \ldots, a_n \in \mathbb{Z}$ Algorithm 4 terminates and $gcd\,[a_1, \ldots, a_n] = ([d, 0, \ldots, 0], \mathbf{U})$, where $d \sim \gcd(a_1, \ldots, a_n)$ and $\mathbf{U} \in \mathbb{Z}^{n \times n}$, such that

$$\mathbf{U}(a_1, \ldots, a_n)^t = (d, 0, \ldots, 0)^t.$$

Echelon Reduction A matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}^{m \times n}$ is an *echelon matrix* if:

(1) There is some $r \in \{0, \ldots, m\}$ such that

$$i > r \Rightarrow a_{ij} = 0$$

for all $j \in \{1, \ldots, n\}$;

(2) For all $1 \leq i \leq r$ the set $M_i := \{j \mid a_{ij} \neq 0\}$ is not empty;

(3) $\rho_1 < \rho_2 < \ldots < \rho_r$, where $\rho_i := \min M_i$.

An $1 \times n$ matrix (a row vector) already is an echelon matrix. An $m \times 1$ matrix $\mathbf{A}$ can be reduced to an echelon matrix $\mathbf{S}$ by the Extended Euclidean Algorithm, which will additionally find a unimodular matrix $\mathbf{U}$ such that $\mathbf{UA} = S$. The case of a general $m \times n$ matrix is covered by Algorithm 5.

Proposition 22 *Let* $\mathbf{A} \in \mathbb{Z}^{m \times n}$*. Algorithm 5 terminates and* *echelonReduction* $\mathbf{A} = (\mathbf{S}, \mathbf{U})$ *where* $\mathbf{U} \in \mathbb{Z}^{m \times m}$ *is a unimodular matrix and* $\mathbf{S} \in \mathbb{Z}^{m \times n}$ *is echelon such that* $\mathbf{UA} = \mathbf{S}$.

Proof. Despite the technical notation, Algorithm 5 is essentially a recursive algorithm. Each recursive call is instantiated with the matrix $\mathbf{A}'$ whose number of rows is reduced by 1 compared to the input matrix $\mathbf{A}$. Since any other operation done within the where clause terminates, the whole procedure will terminate.

Correctness is established by induction on the number m of rows of the input matrix $\mathbf{A}$. If $m = 1$, then $\mathbf{A} = \mathbf{S}$ is echelon and of course $\mathbf{A} = \mathbf{UA}$ with $\mathbf{U} = (1)$, as given by the base case of Algorithm 5.

The case $n = 1$, easily follows from Algorithm 4.

Algorithm 5 Echelon Reduction

```
$echelonReduction \quad :: Mat_{m\times n}(Integer) \to (Mat_{m\times n}(Integer),\ Mat_{m\times m}(Integer))$
$echelonReduction\ A$
  $\mid\ m == 1 \qquad = (A, (1))$
  $\mid\ n == 1 \qquad = extGcd\ [a_{11}, \ldots, a_{m1}]$
  $\mid\ otherwise \quad = (S, U)$
    where
      $(\_, V) \quad = extGcd\ [a_{11}, \ldots, a_{m1}]$
      $(a'_{ij}) \quad = VA$
      $(S', U') = echelonReduction\ ((a'_{i,j\geq 2}))$
      $S \quad = \begin{pmatrix} a'_{11} & a'_{12} \cdots a'_{1n} \\ \mathbf{0} & S' \end{pmatrix}$
      $U \quad = \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & S' \end{pmatrix} \cdot V$
```

Now, let $m > 1$ and $n > 1$. The call of *extGcd* $[a_{11}, \ldots, a_{m1}]$ returns $\mathbf{V} \in \mathbb{Z}^{m\times m}$, such that

$$\mathbf{VA} = \begin{pmatrix} a'_{11} & a'_{12} \cdots a'_{1n} \\ \mathbf{0} & \mathbf{A}' \end{pmatrix} \in \mathbb{Z}^{m\times n}$$

where $a_{11} \sim \gcd(a_{11}, \ldots, a_{m1})$. By the induction hypothesis, *echelonReduction* $\mathbf{A}'$ returns $\mathbf{S}' \in \mathbb{Z}^{(m-1)\times(n-1)}$ and $\mathbf{U}' \in \mathbb{Z}^{(m-1)\times(m-1)}$ with $\mathbf{U}'\mathbf{A}' = \mathbf{S}'$. Finally, let $\mathbf{U}$ and $\mathbf{S}$ be defined as in the algorithm, then

$$\begin{aligned}
\mathbf{UA} &= \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{S}' \end{pmatrix} \mathbf{VA} \\
&= \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & S' \end{pmatrix} \begin{pmatrix} a'_{11} & a'_{12} \cdots a'_{1n} \\ \mathbf{0} & \mathbf{A}' \end{pmatrix} \\
&= \begin{pmatrix} a'_{11} & a'_{12} \cdots a'_{1n} \\ \mathbf{0} & \mathbf{U}'\mathbf{A}' \end{pmatrix} \\
&= \begin{pmatrix} a'_{11} & a'_{12} \cdots a'_{1n} \\ \mathbf{0} & \mathbf{S}' \end{pmatrix} \\
&= \mathbf{S}.
\end{aligned}$$

□

Solving Linear Diophantine Equations Now to our main result (see [Ban93], Theorem 3.6).

Theorem 23 *Let* $\mathbf{A} \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{b} \in \mathbb{Z}^n$*. Let* $\mathbf{U} \in \mathbb{Z}^{m\times m}$ *denote a unimodular matrix and* $\mathbf{S} \in \mathbb{Z}^{m\times n}$ *an echelon matrix, such that* $\mathbf{UA} = \mathbf{S}$*. The system of equations*

$$\mathbf{xA} = \mathbf{b} \tag{2.21}$$

has a solution iff there exists a vector $\mathbf{t} \in \mathbb{Z}^m$ *such that*

$$\mathbf{tS} = \mathbf{b}. \tag{2.22}$$

When a solution exists, the set of all solutions is given by the formula (the general solution)

$$\mathbf{x} = \mathbf{tU} \tag{2.23}$$

where $\mathbf{t}$ *is any integer vector satisfying* $\mathbf{tS} = \mathbf{b}$.

Proof. See [Ban93] p. 63. □

Remark 24 In order to derive the description of the set of solutions (2.23), two steps are involved

(1) find an echelon matrix $\mathbf{S}$, and

(2) describe all vectors $\mathbf{t}$ with $\mathbf{tS} = \mathbf{b}$.

The first step is carried out by Algorithm 5, the second step comes down to simple divisibility tests as $\mathbf{S}$ is in echelon form.

Example 25 Consider

$$(t_1, t_2, t_3) \begin{pmatrix} 3 & 1 \\ 0 & 2 \\ 0 & 0 \end{pmatrix} = (6, 8). \tag{2.24}$$

Then $t_1 = 2$ as $3 \mid 6$ and and because $2 \mid 8 - t_1$ we get $t_2 = 3$. t_3 can be chosen freely from $\mathbb{Z}$ and, therefore, the set of vectors $\mathbf{t}$ satisfying (2.24) is given by

$$\{(2, 3, t_3) \mid t_3 \in \mathbb{Z}\}.$$

2.3 Quasi-Polynomials

This section introduces quasi-polynomials (sometimes also called pseudo-polynomials), a special extension of polynomials which are, in the form of Ehrhart-(Quasi-)polynomials, already known to the parallel computing community. As it turns out, quasi-polynomials constitute an interesting framework for solving parametric linear equation systems (cf. Chapter 4 where we will see that, given $f, g \in \mathbb{Z}[X]$, the mapping $p \mapsto \gcd(f(p), g(p))$ is a quasi-polynomial). To this end, however, we have to consider quasi-polynomials in a syntactical representation (Definition 30) different from the usual one (Definition 27). Both are linked together by Theorem 38.

Definition 26 (Periodic Number) Let $l \in \mathbb{N}_{\geq 1}$ and let $c_0, \ldots, c_{l-1} \in \mathbb{Q}$. A *periodic number* c is a mapping of the form

$$c : \mathbb{Z} \longrightarrow \mathbb{Q} : p \mapsto \begin{cases} c_0 & \text{if } p \equiv_l 0 \\ \ldots & \\ c_{l-1} & \text{if } p \equiv_l (l-1). \end{cases}$$

We call l the *period* of c. The set of all periodic numbers is denoted by $\mathcal{P}$.

More conveniently, c is often written as

$$c(p) = [c_0, \dots, c_{l-1}]_p.$$

Note that periodic numbers do not have a unique period. With l, for every $k \in \mathbb{N}_{\geq 1}$ also kl is a period.

$$[c_0, \dots, c_{l-1}]_p = [c_0, \dots, c_{l-1}, c_0, \dots, c_{l-1}]_p = \dots$$

Therefore, any two periodic numbers can be rewritten with a common period given by (a multiple of) the lcm of the two periods. With the usual pointwise operations, $\mathcal{P}$ is a ring. Indeed, if $[c_0, \dots, c_l]$ and $[d_0, \dots, d_l]$ are periodic numbers with a common period l then

$$[c_0, \dots, c_{l-1}] + [d_0, \dots, d_{l-1}] = [c_0 + d_0, \dots, c_{l-1} + d_{l-1}]$$

and

$$[c_0, \dots, c_{l-1}] \cdot [d_0, \dots, d_{l-1}] = [c_0 d_0, \dots, c_{l-1} d_{l-1}]$$

are periodic numbers, too.

Definition 27 (Quasi-Polynomials) Let $f \in \mathcal{P}[X]$. Then f is called a *(univariate) quasi-polynomial.*

Remark 28 Let P be a polytope in $\mathbb{R}^n$ and let $t \in \mathbb{N}$. In 1962, Ehrhart proved that the function

$$i_P : \mathbb{N}_{\geq 1} \longrightarrow \mathbb{N} : t \mapsto |(tP \cap \mathbb{Z}^n)|$$

is a Quasi-polynomial ([Ehr62], [Ehr77]), called Ehrhart-polynomial.

Note that each quasi-polynomial f can be written as

$$\begin{aligned} f(p) &= \sum_{i=0}^{n} c_i(p) p^i \\ &= \begin{cases} f_0(p) & \text{if } p \equiv_l 0 \\ f_1(p) & \text{if } p \equiv_l 1 \\ \dots & \\ f_{l-1}(p) & \text{if } p \equiv_l (l-1) \end{cases} \end{aligned}$$

where $f_i \in \mathbb{Q}[X]$ and $l \in \mathbb{Z}_{\geq 1}$ is a common period of the c_i. The f_i are called the *constituents* of f.

Example 29 Let

$$f = \frac{1}{4}X^2 + [0, -\frac{1}{2}]X + [0, \frac{1}{4}].$$

On closer inspection we find that

$$f(p) = \left\lfloor \frac{p}{2} \right\rfloor^2.$$

Indeed, remembering that $p \bmod 2 = i$ iff $p = 2p' + i \wedge p' = \lfloor \frac{p}{2} \rfloor$ for $i \in \{0, 1\}$ we get

$$\begin{aligned} f(p) &= \frac{1}{4}p^2 + [0, -\frac{1}{2}]p + [0, \frac{1}{4}] \\ &= \begin{cases} \frac{1}{4}p^2 & \text{if } p \equiv_2 0 \\ \frac{1}{4}p^2 - \frac{1}{2}p + \frac{1}{4} & \text{if } p \equiv_2 1 \end{cases} \\ &= \begin{cases} \frac{1}{4}4p'^2 & \text{if } p = 2p' \wedge p' = \lfloor \frac{p}{2} \rfloor \\ \frac{1}{4}(2p'+1)^2 - \frac{1}{2}(2p'+1) + \frac{1}{4} & \text{if } p = 2p' + 1 \wedge p' = \lfloor \frac{p}{2} \rfloor \end{cases} \\ &= \begin{cases} p'^2 & \text{if } p = 2p' \wedge p' = \lfloor \frac{p}{2} \rfloor \\ p'^2 & \text{if } p = 2p' + 1 \wedge p' = \lfloor \frac{p}{2} \rfloor \end{cases} \\ &= \left\lfloor \frac{p}{2} \right\rfloor^2 \end{aligned}$$

The next definition presents a variant of quasi-polynomials which will, as already mentioned, play an important role in Chapter 4.

Definition 30 (Aiq-polynomials) A map $f : \mathbb{Z} \longrightarrow \mathbb{Z}$ is called a *(univariate) almost integer quasi polynomial* (aiq-plynomial) if there is some $l \in \mathbb{N}_{\geq 1}$ and polynomial maps $f_0, \ldots, f_{l-1} \in \mathbb{Z}[X]$ such that

$$f(p) = \begin{cases} f_0(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 0 \\ \ldots & \\ f_{l-1}(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l (l-1). \end{cases}$$

We call l a *period* or a *modulus* of f and f_i the *i-th constituent* for each $0 \leq i < l$. The set of all aiq-polynomials is denoted by $\mathcal{AIQ}$.

An important feature of aiq-polynomials is that we find their period l within the arguments $\lfloor \frac{p}{l} \rfloor$ of the f_i. This seems to prohibit that, with l, every kl ($k \in \mathbb{N}_{\geq 1}$) is a period of f. However, the contrary is the case as we see in Lemma 35. In fact, we will see that the set $\mathcal{AIQ}$ is *exactly* the set of quasi-polynomials f with $f(\mathbb{Z}) \subseteq \mathbb{Z}$. But before we proceed, let us have a look at some examples of aiq-polynomials.

Example 31 (i) $\mathbb{Z}[X] \subseteq \mathcal{AIQ}$, as $f(p) = f(\lfloor \frac{p}{1} \rfloor)$ for every $f \in \mathbb{Z}[X]$.

(ii) Let $f(p) = p^2 - 3p + 1$. Then $\left\lfloor \frac{f(p)}{2} \right\rfloor \in \mathcal{AIQ}$. This example will be generalized in Proposition 33. Its proof will be rather technical, so we give the following detailed calculation which already contains the main ideas of the proof.

$$\left\lfloor \frac{p^2 - 3p + 1}{2} \right\rfloor = \begin{cases} \left\lfloor \frac{(2p')^2 - 3(2p') + 1}{2} \right\rfloor & \text{if } p = 2p' \\ \\ \left\lfloor \frac{(2p'+1)^2 - 3(2p'+1) + 1}{2} \right\rfloor & \text{if } p = 2p' + 1 \end{cases}$$

$$= \begin{cases} \left\lfloor \frac{4p'^2 - 6p' + 1}{2} \right\rfloor & \text{if } p = 2p' \\ \\ \left\lfloor \frac{(4p'^2 + 4p' + 1^2) + (-6p' - 3 \cdot 1) + 1}{2} \right\rfloor & \text{if } p = 2p' + 1 \end{cases}$$

$$= \begin{cases} \left\lfloor \frac{4p'^2-6p'+f(0)}{2} \right\rfloor & \text{if } p = 2p' \\[2ex] \left\lfloor \frac{4p'^2-2p'+f(1)}{2} \right\rfloor & \text{if } p = 2p'+1 \end{cases}$$

$$= \begin{cases} \left\lfloor 2p'^2 - 3p' + \frac{f(0)}{2} \right\rfloor & \text{if } p = 2p' \\[2ex] \left\lfloor 2p'^2 - p' + \frac{f(1)}{2} \right\rfloor & \text{if } p = 2p'+1 \end{cases}$$

$$= \begin{cases} 2p'^2 - 3p' + \left\lfloor \frac{1}{2} \right\rfloor & \text{if } p = 2p' \\[2ex] 2p'^2 - p' + \left\lfloor \frac{-1}{2} \right\rfloor & \text{if } p = 2p'+1 \end{cases}$$

$$= \begin{cases} 2p'^2 - 3p' & \text{if } p = 2p', \text{ i.e., } p' = \lfloor \frac{p}{2} \rfloor \wedge p \equiv_2 0 \\[2ex] 2p'^2 - p' - 1 & \text{if } p = 2p'+1, \text{ i.e., } p' = \lfloor \frac{p}{2} \rfloor \wedge p \equiv_2 1 \end{cases}$$

$$= \begin{cases} 2\lfloor \frac{p}{2} \rfloor^2 - 3\lfloor \frac{p}{2} \rfloor & \text{if } p \equiv_2 0 \\[2ex] 2\lfloor \frac{p}{2} \rfloor^2 - \lfloor \frac{p}{2} \rfloor - 1 & \text{if } p \equiv_2 1 \end{cases}$$

(iii) $f(p) := \lfloor \frac{p}{2} \rfloor + \lfloor \frac{p}{3} \rfloor \in \mathcal{AIQ}$. To see this, observe that

$$\left\lfloor \frac{p}{2} \right\rfloor = \begin{cases} p' & \text{if } p = 2p' \\ p' & \text{if } p = 2p'+1 \end{cases}$$

$$= \left\{ \begin{array}{rlll} 3p'' & \text{if } p = 2p' & \wedge\ p' = 3p'' & \text{, i.e., } p = 6p'' \\ 3p''+1 & \text{if } p = 2p' & \wedge\ p' = 3p''+1 & \text{, i.e., } p = 6p''+2 \\ 3p''+2 & \text{if } p = 2p' & \wedge\ p' = 3p''+2 & \text{, i.e., } p = 6p''+4 \\ 3p'' & \text{if } p = 2p'+1 & \wedge\ p' = 3p'' & \text{, i.e., } p = 6p''+1 \\ 3p''+1 & \text{if } p = 2p'+1 & \wedge\ p' = 3p''+1 & \text{, i.e., } p = 6p''+3 \\ 3p''+2 & \text{if } p = 2p'+1 & \wedge\ p' = 3p''+2 & \text{, i.e., } p = 6p''+5 \end{array} \right.$$

and

$$\left\lfloor \frac{p}{3} \right\rfloor = \begin{cases} p' & \text{if } p = 3p' \\ p' & \text{if } p = 3p'+1 \\ p' & \text{if } p = 3p'+1 \end{cases}$$

$$= \begin{cases} 2p'' & \text{if } p = 3p' \quad \wedge\ p' = 2p'' \quad , \text{i.e., } p = 6p'' \\ 2p''+1 & \text{if } p = 3p' \quad \wedge\ p' = 2p''+1 \quad , \text{i.e., } p = 6p''+3 \\ 2p'' & \text{if } p = 3p'+1 \quad \wedge\ p' = 2p'' \quad , \text{i.e., } p = 6p''+1 \\ 2p''+1 & \text{if } p = 3p'+1 \quad \wedge\ p' = 2p''+1 \quad , \text{i.e., } p = 6p''+4 \\ 2p'' & \text{if } p = 3p'+2 \quad \wedge\ p' = 2p'' \quad , \text{i.e., } p = 6p''+2 \\ 2p''+1 & \text{if } p = 3p'+2 \quad \wedge\ p' = 2p''+1 \quad , \text{i.e., } p = 6p''+5. \end{cases}$$

Therefore,

$$\left\lfloor \frac{p}{2} \right\rfloor + \left\lfloor \frac{p}{3} \right\rfloor = \begin{cases} 5p'' & \text{if } p = 6p'' \\ 5p'' & \text{if } p = 6p''+1 \\ 5p''+1 & \text{if } p = 6p''+2 \\ 5p''+2 & \text{if } p = 6p''+3 \\ 5p''+3 & \text{if } p = 6p''+4 \\ 5p''+3 & \text{if } p = 6p''+5 \end{cases}$$

$$= \begin{cases} 5\lfloor \frac{p}{6} \rfloor & \text{if } p \equiv_6 0 \\ 5\lfloor \frac{p}{6} \rfloor & \text{if } p \equiv_6 1 \\ 5\lfloor \frac{p}{6} \rfloor + 1 & \text{if } p \equiv_6 2 \\ 5\lfloor \frac{p}{6} \rfloor + 2 & \text{if } p \equiv_6 3 \\ 5\lfloor \frac{p}{6} \rfloor + 3 & \text{if } p \equiv_6 4 \\ 5\lfloor \frac{p}{6} \rfloor + 3 & \text{if } p \equiv_6 5 \end{cases}$$

The next lemma prepares the generalization of Example 31(ii).

Lemma 32 Let $f \in \mathbb{Z}[X]$ and $l, r \in \mathbb{Z}$. Then there exists some $g \in \mathbb{Z}[X]$ such that

$$f(lX + r) = lg(X) + f(r).$$

Proof. Let $f \in \mathbb{Z}[X]$ with degree n and let $l, r \in \mathbb{Z}$. Let $\mu_i(X) = a_i X^i$ denote the terms of f $(0 \leq i \leq n)$. Then

$$\begin{aligned} \mu_i(lX + r) &= a_i(lX + r)^i \\ &= a_i \sum_{j=0}^{i} \binom{i}{j} (lX)^j r^{i-j} \\ &= a_i r^i + a_i \sum_{j=1}^{i} \binom{i}{j} l^j r^{i-j} X^j \\ &= a_i r^i + l \sum_{j=1}^{i} \binom{i}{j} a_i l^{(j-1)} r^{i-j} X^j \\ &= a_i r^i + l\nu_i(X) \end{aligned}$$

with $\nu_i(X) := \sum_{j=1}^{i} \binom{i}{j} a_i l^{(j-1)} r^{i-j} X^j$. Note that $\nu_i(X) \in \mathbb{Z}[X]$. Now let $g(X) :=$

$\sum_{i=0}^{n} \nu_i(X)$ (again $\in \mathbb{Z}[X]$) such that

$$\begin{aligned} f(lX+r) &= \sum_{i=0}^{n} \mu_i(lX+r) \\ &= \sum_{i=0}^{n} (a_i r^i + l\nu_i(X)) \\ &= \sum_{i=0}^{n} a_i r^i + l \sum_{i=0}^{n} \nu_i(X) \\ &= f(r) + lg(X) \end{aligned}$$

which proves the proposed equality. □

Proposition 33 *Let $f \in \mathbb{Z}[X]$, $0 \neq l \in \mathbb{Z}$. Then $\left\lfloor \frac{f(p)}{l} \right\rfloor \in \mathcal{AIQ}$.*

Proof. Let f and l be as stated. Let $p \in \mathbb{Z}$. By Lemma 5 one can write $p = |l|p' + r$ where $0 \leq r < |l|$ and $p' = \lfloor \frac{p}{|l|} \rfloor$. Moreover, by Lemma 32 there is some $g_r \in \mathbb{Z}[X]$ dependent only on r (since l is fixed) such that

$$\begin{aligned} f(p) &= f(|l|p' + r) \\ &= |l| g_r(p') + f(r). \end{aligned}$$

Therefore,

$$\begin{aligned} \left\lfloor \frac{f(p)}{l} \right\rfloor &= \left\lfloor \frac{|l| g_r(p') + f(r)}{l} \right\rfloor \\ &= \left\lfloor \operatorname{sgn}(l) g_r(p') + \frac{f(r)}{l} \right\rfloor \\ &= \operatorname{sgn}(l) g_r(p') + \left\lfloor \frac{f(r)}{l} \right\rfloor \\ &= \operatorname{sgn}(l) g_r(\lfloor \frac{p}{|l|} \rfloor) + \left\lfloor \frac{f(r)}{l} \right\rfloor \end{aligned}$$

Finally, set

$$f_r(X) := \operatorname{sgn}(l) g_r(X) + \left\lfloor \frac{f(r)}{l} \right\rfloor.$$

Altogether, by case distinction on the residue class modulo l that p belongs to we get:

$$\left\lfloor \frac{f(p)}{l} \right\rfloor = \begin{cases} f_0(\lfloor \frac{p}{|l|} \rfloor) & \text{if } p \equiv_{|l|} 0 \\ \dots & \\ f_{(|l|-1)}(\lfloor \frac{p}{|l|} \rfloor) & \text{if } p \equiv_{|l|} (|l|-1). \end{cases}$$

This proves $\left\lfloor \frac{f(p)}{l} \right\rfloor \in \mathcal{AIQ}$. □

The proof of Proposition 33 demonstrates the simple but important concept of rewriting the argument of a given polynomial modulo some positive integer. It is of such importance that we give its own name to it within the following definition.

Definition 34 Let $f \in \mathbb{Z}[X]$ and $l \in \mathbb{N}_{\geq 1}$. Then we can write

$$f(p) = \begin{cases} f(lp') & \text{if } p = lp' \text{ for some } p' \in \mathbb{Z} \\ f(lp'+1) & \text{if } p = lp'+1 \text{ for some } p' \in \mathbb{Z} \\ \dots & \\ f(lp'+(l-1)) & \text{if } p = lp'+(l-1) \text{ for some } p' \in \mathbb{Z} \end{cases}$$

or equivalently

$$f(p) = \begin{cases} f(lp') & \text{if } p' = \lfloor \frac{p}{l} \rfloor \wedge p \equiv_l 0 \\ f(lp'+1) & \text{if } p' = \lfloor \frac{p}{l} \rfloor \wedge p \equiv_l 1 \\ \dots & \\ f(lp'+(l-1)) & \text{if } p' = \lfloor \frac{p}{l} \rfloor \wedge p \equiv_l (l-1) \end{cases}$$

or equivalently

$$f(p) = \begin{cases} f(l\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 0 \\ f(l\lfloor \frac{p}{l} \rfloor + 1) & \text{if } p \equiv_l 1 \\ \dots & \\ f(l\lfloor \frac{p}{l} \rfloor + (l-1)) & \text{if } p \equiv_l (l-1). \end{cases}$$

We call the transition to (equivalent variants of) the respective right-hand sides *l-extending* f and the respective right-hand side an *l-extension of* f.

By Lemma 32, for each $0 \leq i < l$ there is some $g_i \in \mathbb{Z}[X]$ such that $f(lX+i) = lg_i(X) + f(i)$. Set $f_i(X) := lg_i(X) + f(i)$. Then

$$f(p) = \begin{cases} f_0(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 0 \\ f_1(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 1 \\ \dots & \\ f_{l-1}(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l (l-1) \end{cases}$$

and we call the f_i the constituents *determined* by l-extending f.

The following lemma is a further example of l-extension in action and shows that periods of aiq-polynomials do behave as periods do.

Lemma 35 Let $f \in \mathcal{AIQ}$ and let $l \in \mathbb{N}_{\geq 1}$ be a modulus of f. Then kl is also a modulus of f for any $k \in \mathbb{N}_{\geq 1}$.

Proof. Let $f \in \mathcal{AIQ}$ and let l be its modulus. Let $k \in \mathbb{N}_{\geq 1}$. Denote the l constituents of f by $f_0, \dots, f_{l-1}$. Let $p \in \mathbb{Z}$ and let $0 \leq i_0 < l$ be such that $p \equiv_l i_0$. Then

$$\begin{aligned} f(p) &= f_{i_0}(\lfloor \frac{p}{l} \rfloor) \\ &= f_{i_0}(p') \end{aligned}$$

where p' is determined by $p = lp' + i_0$, or equivalently $\big(p' = \lfloor \frac{p}{l} \rfloor \wedge p \equiv_l i_0\big)$. Now, we k-extend $f_{i_0}(p')$:

$$
\begin{aligned}
f(p) &= f_{i_0}(p') && (2.25)\\
&= \begin{cases} f_{i_0}(kp'') & \text{if } p' = kp'' \wedge p = lp' + i_0 \\ f_{i_0}(kp''+1) & \text{if } p' = kp''+1 \wedge p = lp' + i_0 \\ \dots & \\ f_{i_0}(kp''+(k-1)) & \text{if } p' = kp''+(k-1) \wedge p = lp' + i_0 \end{cases} && (2.26)\\
&= \begin{cases} kg_{i_0 0}(p'') + f_{i_0}(0) & \text{if } p = lkp'' + i_0 \\ kg_{i_0 1}(p'') + f_{i_0}(1) & \text{if } p = l(kp''+1) + i_0 \\ \dots & \\ kg_{i_0 (k-1)}(p'') + f_{i_0}(k-1) & \text{if } p = l(kp''+(k-1)) + i_0. \end{cases} && (2.27)
\end{aligned}
$$

Let us define

$$f_{i_0 j}(X) := kg_{i_0 j}(X) + f_{i_0}(j) \qquad (0 \le j < k),$$

consider that for all $j \in \{0, \dots, k-1\}$

$$p = l(kp''+j) + i_0 \Longleftrightarrow \left(p'' = \left\lfloor \frac{p}{lk} \right\rfloor \wedge p \equiv_{lk} kj + i_0\right)$$

and hence get

$$
\begin{aligned}
f(p) &= \begin{cases} f_{i_0 0}(p'') & \text{if } p = lkp'' + i_0 \\ f_{i_0 1}(p'') & \text{if } p = l(kp''+1) + i_0 \\ \dots & \\ f_{i_0 (k-1)}(p'') & \text{if } p = l(kp''+(k-1)) + i_0 \end{cases}\\
&= \begin{cases} f_{i_0 0}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} i_0 \\ f_{i_0 1}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} k + i_0 \\ \dots & \\ f_{i_0 (k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} k(k-1) + i_0 \end{cases}
\end{aligned}
$$

The described k-extension can be carried out for every $0 \le i < l$ which leads to one big case distinction on the residue classes of p modulo lk, see Figure 2.1. Note that the nature of the given construction leads to a complete case distinction on the lk possible residue classes of p modulo lk. Therefore, f is expressed as an aiq-polynomial with modulus lk, which concludes the proof. □

Remark 36 Given $p \in \mathbb{Z}$ with $p \equiv_{lk} r$ $(0 \le r < lk-1)$, one can find the appropriate constituent f_{ij} of Figure 2.1 via the bijection

$$
\begin{aligned}
\varphi : \{0, \dots, lk-1\} &\longrightarrow \{0, \dots, l-1\} \times \{0, \dots, k-1\}\\
r &\longmapsto (i, j) = \left(r - \left\lfloor \frac{r}{l} \right\rfloor l, \left\lfloor \frac{r}{l} \right\rfloor\right),
\end{aligned}
$$

$$
\begin{aligned}
f(p) &= \begin{cases}
f_{00}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} 0 \\
f_{01}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l \\
\ldots & \\
f_{0(k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l(k-1) \\
f_{10}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} 1 \\
f_{11}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l+1 \\
\ldots & \\
f_{1(k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} k(k-1)+1 \\
 & \\
\ldots & \\
f_{(l-1)0}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} (l-1) \\
f_{(l-1)1}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l+(l-1) \\
\ldots & \\
f_{(l-1)(k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} k(k-1)+(l-1)
\end{cases} \\
&= \begin{cases}
f_{00}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} 0 \\
f_{10}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} 1 \\
\ldots & \\
f_{(l-1)0}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} (l-1) \\
f_{01}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l \\
f_{11}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l+1 \\
\ldots & \\
f_{(l-1)1}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} k+(l-1) \\
 & \\
\ldots & \\
f_{0(k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l(k-1) \\
f_{1(k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} l(k-1)+1 \\
\ldots & \\
f_{(l-1)(k-1)}(\lfloor \frac{p}{lk} \rfloor) & \text{if } p \equiv_{lk} lk-1
\end{cases}
\end{aligned}
$$

Figure 2.1: Complete case distinction on the residues p modulo lk.

such that

$$f_{ij} = f_{\varphi(r)}.$$

Moreover, note that for all r, $r' \in \{0, \ldots, lk-1\}$:

$$r \leq r' \Longleftrightarrow \varphi(r) \prec_{revlex} \varphi(r').$$

Theorem 37 *$\mathcal{AIQ}$ is a ring.*

Proof. By the subring test ([WBK93, Section 1.4]), we only have to show that with $f, g \in \mathcal{AIQ}$ also $f - g, fg \in \mathcal{AIQ}$. So let $f, g \in \mathcal{AIQ}$ and let $m \in \mathbb{N}_{\geq 1}$ be a common modulus of f and g (apply Lemma 35 if necessary). Let $f_i, g_i \in \mathbb{Z}[X]$ $(0 \leq i < m)$ denote the m constituents of f and g. Then one can immediately see that $f_i - g_i$ and $f_i g_i$, respectively, give the m constituents of $f - g$ and fg by pointwise addition and multiplication, which shows that $f - g$ and fg are actually aiq-polynomials with modulus m. □

The following theorem links aiq-polynomials and quasi-polynomials with $f(\mathbb{Z}) \subseteq \mathbb{Z}$ together. The reader may note that Theorem 37 actually is an easy consequence of it but we leave it to her to provide this alternative proof.

Theorem 38 *The set $\mathcal{AIQ}$ of aiq-polynomials is exactly the set of quasi-polynomials f with $f(\mathbb{Z}) \subseteq \mathbb{Z}$.*

Proof. Let $f \in \mathcal{AIQ}$ of period l and let $f_0, \ldots, f_{l-1}$ be the constituents of f. Further, let $p \in \mathbb{Z}$. Then with $i := p \bmod l$

$$\begin{aligned} f(p) &= f_i(\left\lfloor \frac{p}{l} \right\rfloor) \\ &= f_i(\frac{p-i}{l}). \end{aligned}$$

Thus, f is a quasi-polynomial because each $f_i(\frac{X-i}{l}) \in \mathbb{Q}[X]$ and clearly $f(\mathbb{Z}) \subseteq \mathbb{Z}$. Conversely, let f be a quasi-polynomial such that $f(\mathbb{Z}) \subseteq \mathbb{Z}$. Let f be of period l and let $f_0, \ldots, f_{l-1} \in \mathbb{Q}[X]$ be its constituents. For each constituent we have

$$p \bmod l = i \Longrightarrow f_i(p) \in \mathbb{Z}.$$

Let n denote the absolute value of the least common multiple of the denominators of all coefficients of all f_i. Then there are integral polynomials $g_i \in \mathbb{Z}[X]$ with $f_i = \frac{g_i}{n}$. Finally, for each $0 \leq i < l$ we set $h_i(X) := g_i(lX + i)$ such that for all $p \in \mathbb{Z}$

$$\begin{aligned} f(p) = f_i(p) &= \frac{g_i(p)}{n} \\ &= \frac{h_i(\lfloor \frac{p}{l} \rfloor)}{n} = \frac{h_i(p')}{n} \end{aligned}$$

whenever $i = p \bmod l$ and $p' = \lfloor \frac{p}{l} \rfloor$. Note in particular that

$$\frac{h_i(j)}{n} \in \mathbb{Z} \quad (j \in \{0, \ldots, n-1\}). \tag{2.28}$$

Then for all $i \in \{0, \ldots, l-1\}$ and all p with $p \equiv_l i$ and $p' := \lfloor \frac{p}{n} \rfloor$

$$\frac{h_i(p')}{n} = \begin{cases} \frac{h_i(np'')}{n} & \text{if } p' = np'' \\ \frac{h_i(np''+1)}{n} & \text{if } p' = np'' + 1 \\ \ldots & \\ \frac{h_i(np''+n-1)}{n} & \text{if } p' = np'' + (n-1) \end{cases}$$
$$= \begin{cases} h_{i0}(p'') + \frac{h_i(0)}{n} & \text{if } p' = np'' \\ h_{i1}(p'') + \frac{h_i(1)}{n} & \text{if } p' = np'' + 1 \\ \ldots & \\ h_{i(n-1)}(p'') + \frac{h_i(n-1)}{n} & \text{if } p' = np'' + (n-1) \end{cases}$$

where the $h_{ij} \in \mathbb{Z}[X]$ are defined such that $h_i(nX + j) = nh_{ij}(X) + h_i(j)$, cf. Lemma 32. With (2.28) we have $h_{ij}(X) + \frac{h_i(j)}{n} \in \mathbb{Z}[X]$. And so, by a line of reasoning similar to that of Lemma 35 and its proof, we conclude that f is indeed an aiq-polynomial. □

Generalizing Aiq-polynomials Further In [Wei90], a certain extension of Presburger arithmetic called almost linear arithmetic is studied. In simple terms, Weispfenning considers formulas of type $\exists X t(X) \, \rho \, 0$ with

$$t(X) = \left(f(X) + c_1 \left\lfloor \frac{f_1(X)}{g_1(X)} \right\rfloor + \cdots + c_m \left\lfloor \frac{f_m(X)}{g_m(X)} \right\rfloor \right) \tag{2.29}$$

and $\rho \in \{<, >, =, \equiv_l\}$, $c_i \in \mathbb{Z}$ and $p, p_i, q_i \in \mathbb{Z}[X]$. He shows that one can compute some $s \in \mathbb{N}_{\geq 1}$ such that

$$\exists X t(X) \, \rho \, 0 \text{ iff } \exists X \, (-s < X < s \wedge t(X) \, \rho \, 0)$$

and takes this result to generalize Presburger arithmetic accordingly. We take up the ideas given in his proof of Theorem 2.1 in [Wei90] and show how terms of type $\lfloor \frac{f}{g} \rfloor$ with $f, g \in \mathbb{Z}[X]$ are related to aiq-polynomials.

Let

$$h : \mathbb{Z} - P_u \longrightarrow \mathbb{Z} : p \mapsto \left\lfloor \frac{f(p)}{g(p)} \right\rfloor$$

with $f, g \in \mathbb{Q}[X]$ and $P_u := \{p \in \mathbb{Z} \mid g(p) = 0\}$. By usual polynomial division in $\mathbb{Q}[X]$ we can find polynomials $q, r \in \mathbb{Q}[X]$ such that $\deg(r) < \deg(g)$ and

$$\frac{f}{g} = q + \frac{r}{g}.$$

It is well known from calculus that $\frac{r(p)}{g(p)} \to 0$ when $p \to \pm\infty$. In particular, with $r(X) = \sum_{i=0}^{d} a_i X^i$, $g(X) = \sum_{i=0}^{d'} b_i X^i$ and $k \in \mathbb{N}_{\geq 2}$ for any $p \in \mathbb{Z}$ with $|p| > (k+1) \frac{a_d}{b_{d'}}$

$$\left| \frac{r(p)}{g(p)} \right| < \frac{1}{k-1}$$

(cf. Corollary 2.4, [Wei90]) and

$$\frac{r(p)}{g(p)} > 0 \text{ iff } \frac{a_d}{b_{d'}} p^{d-d'} > 0. \tag{2.30}$$

The question is now, how small do we want $\lfloor \frac{r(p)}{g(p)} \rfloor$ to be? The answer depends entirely on $q(p)$. Let l be the absolute value of a lcm of the denominators of the coefficients of q and choose $q' \in \mathbb{Z}[X]$ such that

$$q(X) = \frac{q'(X)}{l}.$$

For each $i \in \{0, \dots, l-1\}$, determine $q_i \in \mathbb{Z}[Y]$ such that $q'(lY + i) = lq_i(Y) + q'(i)$ (cf. Lemma 32). Then for all p, with $i = p \bmod l$

$$\begin{aligned}\Big\lfloor \frac{f(p)}{g(p)} \Big\rfloor &= \Big\lfloor q(p) + \frac{r(p)}{g(p)} \Big\rfloor \\ &= \Big\lfloor \frac{q'(p)}{l} + \frac{r(p)}{g(p)} \Big\rfloor \\ &= \Big\lfloor q_i(\lfloor \frac{p}{l} \rfloor) + \frac{q'(i)}{l} + \frac{r(p)}{g(p)} \Big\rfloor \\ &= q_i(\lfloor \frac{p}{l} \rfloor) + \Big\lfloor \frac{q'(i)}{l} + \frac{r(p)}{g(p)} \Big\rfloor\end{aligned}$$

The remaining floor expression can be computed if $|\frac{r(p)}{g(p)}|$ is sufficiently small which certainly is the case if

$$\left| \frac{r(p)}{g(p)} \right| < \min(M - \{0\}) \tag{2.31}$$

with

$$M = \{1\} \cup \bigcup_{i=0}^{l-1} \left\{ \frac{q'(i)}{l} - \Big\lfloor \frac{q'(i)}{l} \Big\rfloor, \Big\lceil \frac{q'(i)}{l} \Big\rceil - \frac{q'(i)}{l} \right\}.$$

With (2.30), we can find some $s \in \mathbb{N}_{\geq 1}$ such that $P_u \subseteq (\mathbb{Z} \cap [-s, s])$ and each $\Big\lfloor \frac{q'(i)}{l} + \frac{r(p)}{g(p)} \Big\rfloor$ is constant on $\mathbb{Z} - [-s, s]$. Therefore, choose $|p_0| > s$ and define

$$z_i := \Big\lfloor \frac{q'(i)}{l} + \frac{r(p_0)}{g(p_0)} \Big\rfloor.$$

Then

$$\Big\lfloor \frac{f(p)}{g(p)} \Big\rfloor = \begin{cases} q_0(\lfloor \frac{p}{l} \rfloor) + z_0 & \text{if } p \equiv_l 0 \\ \dots & \\ q_{l-1}(\lfloor \frac{p}{l} \rfloor) + z_{l-1} & \text{if } p \equiv_l l-1 \end{cases}$$

for all $|p| > s$. In other words, $\Big\lfloor \frac{f(p)}{g(p)} \Big\rfloor$ is an aiq-polynomials outside the interval $\mathbb{Z} \cap [-s, s]$.

What makes these considerations notably interesting is that we are now in the position to consider functions containing *nested* $\lfloor \div \rfloor$-fractions as the following example demonstrates. We do not investigate this topic any further within this thesis, but we claim that all the methods studied in Chapter 4 apply to those types of functions as well, allowing data dependency analysis for a wider class of programs.

Example 39 Consider the mapping

$$f : \mathbb{Z} \longrightarrow \mathbb{Z} : p \mapsto \left\lfloor \frac{\lfloor \frac{3p^2}{2p+1} \rfloor + 3p}{2} \right\rfloor \tag{2.32}$$

which is defined for all $p \in \mathbb{Z}$. We start by simplifying the inner $\lfloor \frac{\cdot}{\cdot} \rfloor$-expression in the way described above, i.e., we divide $3X^2$ by $2X+1$ and get

$$\frac{3X^2}{2X+1} = \frac{6X-3}{4} + \frac{3}{8X+4}. \tag{2.33}$$

Therefore,

$$\begin{aligned}
\left\lfloor \frac{3p^2}{2p+1} \right\rfloor &= \left\lfloor \frac{6p-3}{4} + \frac{3}{8p+4} \right\rfloor \\
&= \begin{cases} \lfloor 6\lfloor \frac{p}{4} \rfloor - \frac{3}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 0 \\ \lfloor 6\lfloor \frac{p}{4} \rfloor + \frac{3}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 1 \\ \lfloor 6\lfloor \frac{p}{4} \rfloor + \frac{9}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 2 \\ \lfloor 6\lfloor \frac{p}{4} \rfloor + \frac{15}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 3 \end{cases} \\
&= \begin{cases} 6\lfloor \frac{p}{4} \rfloor + \lfloor -\frac{3}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 0 \\ 6\lfloor \frac{p}{4} \rfloor + \lfloor \frac{3}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 1 \\ 6\lfloor \frac{p}{4} \rfloor + \lfloor \frac{9}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 2 \\ 6\lfloor \frac{p}{4} \rfloor + \lfloor \frac{15}{4} + \frac{3}{8p+4} \rfloor & \text{if } p \equiv_4 3 \end{cases}
\end{aligned}$$

We get $\frac{3}{8p+4} < \frac{1}{4}$ for all $|p| \geq 3$, i.e., we get

$$\left\lfloor \frac{3p^2}{2p+1} \right\rfloor = \begin{cases} 6\lfloor \frac{p}{4} \rfloor - 1 & \text{if } p \equiv_4 0 \\ 6\lfloor \frac{p}{4} \rfloor & \text{if } p \equiv_4 1 \\ 6\lfloor \frac{p}{4} \rfloor + 2 & \text{if } p \equiv_4 2 \\ 6\lfloor \frac{p}{4} \rfloor + 3 & \text{if } p \equiv_4 3 \end{cases}$$

for all $|p| \geq 3$. Altogether, we get

$$f(p) = \begin{cases} \lfloor \frac{18p'-1}{2} \rfloor & \text{if } p = 4p' \\ \lfloor \frac{18p'+3}{2} \rfloor & \text{if } p = 4p'+1 \\ \lfloor \frac{18p'+8}{2} \rfloor & \text{if } p = 4p'+2 \\ \lfloor \frac{18p'+12}{2} \rfloor & \text{if } p = 4p'+3 \end{cases}$$

for all $|p| \geq 3$ which is an aiq-polynomial.

Chapter 3

Banerjee's Data Dependence Analysis

While the Introduction (Chapter 1) was mainly concentrated on one particular aspect of data dependence analysis – the solution to certain equation systems –, this chapter describes the different steps of Banerjee's approach in more detail. By and large, our presentation follows [Ban93, Chapter 4] and [Kei97, Chapter 6].

The basic building blocks of programs are for-loops

```
for $i = p$ to $q$ do
  $S : S(i)$
  $T : T(i)$
  $\ldots$
end for
```

and assignment statements as indicated by $S, T, \ldots$. These statements constitute the *body* of the given loop. i is called *index variable*. The body statements are dependent on i since they can contain array accesses $A[f(i)], B[g(i)], \ldots$ where the *access functions* $f, g, \ldots$ take i as arguments. Note that the general signature of the access functions is $\mathbb{Z}^m \longrightarrow \mathbb{Z}^n$, since the arrays can of course be multi-dimensional. p is the *lower limit* and q the *upper limit* of the given for-loop. We assume that the strides of all for-loops are always 1. Loops can be nested as in

```
for $i_1 = p_1$ to $q_1$ do
  for $i_2 = p_2(i_1)$ to $q_2(i_1)$ do
    $S : S(i_1, i_2)$
  end for
  $T : T(i_1)$
end for
```

Note that the loops do not have to be nested perfectly. Observe further that the loop bounds can depend on the array indices of the surrounding loops. Similarly, the access functions of the arrays contained within the different statements can be functions in the index variables of the respective surrounding loops. An important question regards the way in which the bounds and access functions can be formed from the surrounding index variables. One usual condition is that the loop bounds and access functions are affine functions in the index variables and additional *structural parameters*. Hereby, structural parameters are constants whose value will be known at run time. For instance, in our example program from Chapter 1 n is a structural parameter. So, assume that $i_1, \ldots, i_k$ are surrounding index variables of some statement or loop and assume that $p_1, \ldots, p_z$ are structural parameters. Then

$$c_1 i_1 + \cdots + c_k i_k + c_{k+1} p_1 + \cdots + d_{k+z} p_z + c \tag{3.1}$$

with $c_j, c \in \mathbb{Z}$ is a legitimate dimension of an access function or a legitimate loop bound. We will come back to the problem of the concrete form of (3.1) at the end of this chapter after we discussed Banerjee's data dependence analysis.

A data dependency between statements $S(\mathbf{i})$ and $T(\mathbf{j})$, for given concrete instances of $\mathbf{i}$ and $\mathbf{j}$, occurs if

- S and T read from or write to the same memory location;
- S is executed before T.

As for the first condition, let $A[f(\mathbf{i})]$ be an array access in statement S and $A[g(\mathbf{j})]$ an array access in statement T. Then a dependency between S and T (with respect to the given common array A) can only occur if

$$f(\mathbf{i}) = g(\mathbf{j}). \tag{3.2}$$

If the target dimensions of f and g are in the form of (3.1), we deal with a system of linear Diophantine equations in the variables $\mathbf{i}$ and $\mathbf{j}$ and so we can derive a complete description of the possible dependencies by Theorem 23. Note in particular that S and T are not dependent on each other (with respect to the given array accesses) if (3.2) has no solution. This demonstrates why it is not enough to consider rational or real solutions to (3.2) as their existence does not imply integral feasibility.

Even if (3.2) has a solution, it is still possible that S and T are not dependent. For, on the one hand, we still have to consider the second condition given above, which requires that $\mathbf{i} \prec_{lex} \mathbf{j}$. On the other hand, there are the constraints imposed on $\mathbf{i}$ and $\mathbf{j}$ by the loop bounds. They describe a system of linear inequalities provided that the loop bounds are linear expressions as in (3.1). One possibility to handle these inequalities is given by the Fourier-Motzkin elimination method which decides if the inequalities are satisfiable over the reals. Again, this does not imply satisfiabilty over the integers, therefore, another possibility lies in the application of methods known from integer programming. In any case, the description of the integral solutions to (3.2) precedes any further action.

Let us return to the question of the actual form of the access functions and the loop bounds. In [Grö03], the restriction given above was eased such that in (3.1) now also $c_1, \ldots, c_k \in \mathbb{R}[p_1, \ldots, p_z]$ are admitted. Among other things, A. Größlinger shows that Fourier-Motzkin elimination is possible within the new framework as well as (real) linear programming. Since it is also possible to treat systems of linear equations with coefficients from $\mathbb{R}[p_1, \ldots, p_z]$, the algorithmic basis for an extended data dependence analysis is prepared. However, as argued above, we would prefer solutions to systems of linear equations within $\mathbb{Z}[p_1, \ldots, p_z]$. The next chapter attends this task and it relies on the assumption that the methods presented there will give results compatible with, for instance, the extended Fourier-Motzkin elimination. We support this assumption by the following example which continues the example from Chapter 1 and leads over to Chapter 4.

Example 40 Let us repeat the example program from Chapter 1:

```
for $i = 0$ to $n$ do
  for $j = 0$ to $n$ do
    $S$: $A[p \cdot i + j] = A[p \cdot i + j] + 1$
  end for
end for
```

S depends on itself if

$$pi + j = pi' + j'. \tag{3.3}$$

for $(i, j) \prec_{lex} (i', j')$, that is

$$\begin{aligned} & i < i' && (3.4)\\ \vee\quad & (i = i' \wedge j < j'). && (3.5)\end{aligned}$$

The constraints of the loop bounds are given by

$$\begin{array}{ccccc} 0 & \leq & i & \leq & n \\ 0 & \leq & j & \leq & n \\ 0 & \leq & i' & \leq & n \\ 0 & \leq & j' & \leq & n \end{array} \tag{3.6}$$

As we already know from Chapter 1, the set of solutions to (3.3) is given by

$$(i, j, i', j') = (t_4, t_2, t_3, t_2 - t_3 p + t_4 p) \tag{3.7}$$

for all $t_1, t_2, t_3 \in \mathbb{Z}$. The next step combines (3.7) with (3.6), (3.4) and (3.5), respectively. Thus, we get

$$\begin{aligned} & t_4 < t_3 && (3.8)\\ \vee\quad & (t_4 = t_3 \wedge t_2 < t_2 - t_3 p + t_4 p) && (3.9)\end{aligned}$$

and

$$\begin{array}{ccccc} 0 & \leq & t_4 & \leq & n \\ 0 & \leq & t_2 & \leq & n \\ 0 & \leq & t_3 & \leq & n \\ 0 & \leq & t_2 - t_3 p + t_4 p & \leq & n \end{array} \tag{3.10}$$

The extended Fourier-Motzkin eliminations applied to these inequalities returns the condition $p \leq n$ in case (3.10) $\wedge$ (3.8) and that no solutions exist in case (3.10) $\wedge$ (3.9).

Chapter 4

Equation Systems with Non-linear Parameters

In the following, we will describe an extension of Theorem 23 for systems of parametrized linear Diophantine equations

$$\mathbf{xA} = \mathbf{b}, \tag{4.1}$$

where $A \in \mathbb{Z}[p_1, \ldots, p_k]^{m\times n}$, $b \in \mathbb{Z}[p_1,\ldots, p_k]^m$ which we will call *modulo reduction.* In the case of a single non-linear parameter ($k = 1$) this reduction will lead to a complete description of the respective solutions, which in turn are descriptions of the data dependencies from where (4.1) arose. In the multi-parameter case ($k > 1$) this generality will not apply. However, our method can give partial information about the solutions that is still useful in later phases of dependence analysis.

4.1 Modulo Reduction with a Single Parameter

4.1.1 An Introductory Example

As in Section 2.2, we start with the computation of $\gcd(a, b)$, $a, b \in \mathbb{Z}[p]$ and then extend it to the full echelon reduction of the matrix A from (4.1). To introduce the idea of modulo reduction we start with a simple example and try to *compute*

$$d(p) := \gcd\,(\,2p^2 + 1,\, 3\,).$$

First, note that we are not interested in the *polynomial* gcd of $2X^2 + 1$ and 3, but in the *function*

$$d : \mathbb{Z} \to \mathbb{Z},\ p \mapsto \gcd\,(\,2p^2 + 1,\, 3\,).$$

Second, by *computation* we mean giving some explicit and finite representation of $d(p)$ avoiding any occurrence of "gcd". For instance, Table 4.1 suggests that

$$d(p) = \begin{cases} 1 & \text{if } p \equiv_3 0, \\ 3 & \text{if } p \equiv_3 1 \text{ or } p \equiv_3 2. \end{cases} \tag{4.2}$$

Of course, it is clear that $d(p)$ must be either 1 or 3, since these are the only possible positive divisors of 3 in $\gcd\,(\,2p^2\!+\!1,\, 3\,)$. It is less clear that $d(p)$ contains the

Table 4.1 $d(p) = \gcd(2p^2 + 1,\ 3)$

p	-5	-4	-3	-2	-1	0	1	2	3	4	5
$2p^2+1$	51	33	19	9	3	1	3	9	19	33	51
$d(p)$	3	3	1	3	3	1	3	3	1	3	3

claimed amount of internal structure, which, however, is verified by case distinction on the residue classes of p modulo 3:

Case $p \equiv_3 0$: Since $p = 3p'$ for some $p' \in \mathbb{Z}$, we get

$$\begin{aligned}
\gcd(2p^2+1,\ 3) &\sim \gcd(18p'^2+1, 3) & & \\
&\overset{(*)}{\sim} \gcd(1,3) & &\rightsquigarrow \begin{pmatrix} 1 & -6p'^2 \\ 0 & 1 \end{pmatrix} \\
&\sim \gcd(3,1) & &\rightsquigarrow \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
&\sim \gcd(0,1) & &\rightsquigarrow \begin{pmatrix} 1 & -3 \\ 0 & 1 \end{pmatrix} \\
&\sim \gcd(1,0) & &\rightsquigarrow \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
&\sim 1
\end{aligned}$$

In $(*)$ we apply Lemma 7(vi), $\gcd(a,b) = \gcd(a - qb, b)$ with $q = 6p'^2$. The matrices on the right hand side "collect" the actions along the transformations, similarly as is done in Algorithm 3. Check for instance that in $(*)$

$$\begin{pmatrix} 1 & -6p'^2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 18p'^2+1 \\ 3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix}.$$

Each matrix is obviously unimodular, and their product

$$\begin{aligned}
U_0 &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & -3 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & -6p'^2 \\ 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} 1 & -6p'^2 \\ -3 & 1+18p'^2 \end{pmatrix}
\end{aligned} \tag{4.3}$$

is again unimodular for every $p' \in \mathbb{Z}$.

Case $p \equiv_3 1$: Now $p = 3p' + 1$ for some $p' \in \mathbb{Z}$, and

$$\begin{aligned}
\gcd(2p^2+1,\ 3) &\sim \gcd(2(3p'+1)^2+1, 3) & & \\
&\sim \gcd(18p'^2+12p'+3, 3) & & \\
&\overset{(*)}{\sim} \gcd(0,3) & &\rightsquigarrow \begin{pmatrix} 1 & -6p'^2-4p'-1 \\ 0 & 1 \end{pmatrix} \\
&\sim \gcd(3,0) & &\rightsquigarrow \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
&\sim 3
\end{aligned}$$

with $q = 6p'^2 + 4p' + 1$ in $(*)$. Again,

$$\begin{aligned} U_1 &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & -6p'^2 - 4p' - 1 \\ 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 \\ 1 & -6p'^2 - 4p' - 1 \end{pmatrix} \end{aligned} \tag{4.4}$$

is unimodular.

Case $p \equiv_3 2$**:** Here, $p = 3p' + 2$ for some $p' \in \mathbb{Z}$, and

$$\begin{aligned} \gcd(2p^2 + 1,\, 3) &\sim \gcd(2(3p' + 2)^2 + 1, 3) \\ &\sim \gcd(18p'^2 + 24p' + 9, 3) \\ &\overset{(*)}{\sim} \gcd(0, 3) && \rightsquigarrow \begin{pmatrix} 1 & -6p'^2 - 8p' - 3 \\ 0 & 1 \end{pmatrix} \\ &\sim \gcd(3, 0) && \rightsquigarrow \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ &\sim 3 \end{aligned}$$

with $q = 6p'^2 + 8p' + 3$ in $(*)$ and

$$\begin{aligned} U_2 &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & -6p'^2 - 8p' - 3 \\ 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 \\ 1 & -6p'^2 - 8p' - 3 \end{pmatrix} \end{aligned} \tag{4.5}$$

unimodular and our guess from Equation (4.2) is finally justified. The gcd's of $2p^2+1$ and 3 can be described by a finite case distinction on the residue classes of p modulo 3 or, in other words, the gcd is an aiq-polynomial.

Let us go a step further and ask for the solutions of

$$(x, y)A = p. \tag{4.6}$$

with $A = \begin{pmatrix} 2p^2 + 1 \\ 3 \end{pmatrix}$. By Theorem 23 and the above calculation, (4.6) has a solution iff there is some $(t_1, t_2) \in \mathbb{Z}^2$ such that

$$\begin{cases} (t_1, t_2) \begin{pmatrix} 1 \\ 0 \end{pmatrix} = t_1 = 3p' & \text{if } p \equiv_3 0 \wedge p' = \lfloor \frac{p}{3} \rfloor \\ (t_1, t_2) \begin{pmatrix} 3 \\ 0 \end{pmatrix} = 3t_1 = 3p' + 1 & \text{if } p \equiv_3 1 \wedge p' = \lfloor \frac{p}{3} \rfloor \\ (t_1, t_2) \begin{pmatrix} 3 \\ 0 \end{pmatrix} = 3t_1 = 3p' + 2 & \text{if } p \equiv_3 2 \wedge p' = \lfloor \frac{p}{3} \rfloor \end{cases} \tag{4.7}$$

Therefore, (4.6) has a solution iff $p \equiv_3 0$ and $\{t = (p, t_2) \mid t_2 \in \mathbb{Z}\}$ then gives the set of integer vectors satisfying $t\binom{1}{0} = p$. So the set of all solutions in case $p \equiv_3 0$ is given by

$$\begin{aligned}(x, y) &= (p, t_2)U_0 \\ &= (p, t_2)\begin{pmatrix} 1 & -6\lfloor \frac{p}{3} \rfloor^2 \\ -3 & 1 + 18\lfloor \frac{p}{3} \rfloor^2 \end{pmatrix} \\ &= (p - 3t_2, -6p\lfloor \frac{p}{3} \rfloor^2 + t_2 + t_2\lfloor \frac{p}{3} \rfloor^2).\end{aligned} \tag{4.8}$$

If $p \equiv_3 1$ or $p \equiv_3 2$ Equation (4.6) has no solution.

In the following we will generalize this example step by step. Our agenda reads as follows:

1. In Section 4.1.2 we will demonstrate that for $f, g \in \mathbb{Z}[X]$ the function

   $$d : \mathbb{Z} \longrightarrow \mathbb{Z} : p \mapsto \gcd(f(p), g(p))$$

   is an aiq-polynomial (called the aiq-gcd of f and g) and will get a clear picture how the constituents look like (Proposition 45 and Theorem 48). In preparation for Section 4.1.3, we will prove a result about the divisibility of certain constituents involved (Corollary 49). Finally, we extend the previous results to the case of more than two polynomials (Corollary 52) and will see that the gcd of two aiq-polynomials is again an aiq-polynomial (Corollary 53).

2. Section 4.1.3 shows how we can describe the set of integers p such that $f(p) \mid g(p)$ for given $f, g \in \mathbb{Z}[X]$. Interestingly, aiq-gcd's are involved. This section leads to the extension of step (2) from Remark 24.

3. The unimodular matrices that we get from the Extended Euclidean Algorithm play a prominent role for the solution of linear Diophantine equations. While Section 4.1.2 introduces the ideas behind aiq-gcd's, no algorithms are developed. It is not even clear yet, if we can actually obtain some kind of "unimodular aiq-matrices" in the sense of Algorithms 4 and 5. Section 4.1.4 will show that this is possible and the respective algorithms are developed. A larger example illustrates the ideas behind the algorithms.

4. Resting on previous algorithms, Section 4.1.5 describes aiq-echelon reduction. This section is rather technical and focuses mainly on the principal feasibility of a Haskell implementation.

5. Section 4.1.6 finally describes how the different pieces can be put together in order to construct a tree that represents the different exact solutions of a system of linear Diophantine equations in one non-linear parameter.

4.1.2 The Aiq-gcd of Integer Polynomial Functions

Aiq-gcd's of two integer polynomials We initially treat only the case that two polynomials are relatively prime. This simplifies our considerations and leads to Proposition 45 which is prepared by following lemmas and corollaries.

Lemma 41 Let $f, g \in \mathbb{Z}[X]$ and let $\gcd(f(X), g(X)) = 1$, where the polynomial gcd of f and g is considered in $\mathbb{Q}[X]$. Then there is some $0 \neq l \in \mathbb{Z}$ such that for all $p \in \mathbb{Z}$

$$\gcd(f(p), g(p)) \mid l.$$

In other words, $\gcd(f(p), g(p))$ is bounded.

Proof. Let f and g be as stated. Then by [WBK93, Theorem 2.32] there are $s, t \in \mathbb{Q}[X]$ such that

$$sf + tg = 1.$$

Choose $0 \neq l \in \mathbb{Z}$ as a common multiple of the denominators of the coefficients of s and t and let $p \in \mathbb{Z}$. Then $ls, lt \in \mathbb{Z}[X]$, i.e., $ls(p), lt(p) \in \mathbb{Z}$ and therefore,

$$ls(p)f(p) + lt(p)g(p) = l.$$

But with Lemma 7(ix) this means just

$$l \in f(p)\mathbb{Z} + g(p)\mathbb{Z} = \gcd(f(p), g(p))\mathbb{Z}$$

which in turn implies

$$\gcd(f(p), g(p)) \mid l.$$

Since

$$a \mid l \wedge l \neq 0 \Longrightarrow |a| \leq |l|,$$

$\gcd(f(p), g(p))$ is indeed bounded. □

The bound l given by the last lemma will now be used to confirm the cyclic behaviour observed by the introductory example, in case f and g are relatively prime.

Lemma 42 Let $a, b \in \mathbb{Z}$ and let $l \in \mathbb{N}_{\geq 1}$. Then

$$a \equiv_l b \Longrightarrow \gcd(a, l) \sim \gcd(b, l).$$

Proof. $a \equiv_l b$ implies $a - b = ql$ for some $q \in \mathbb{Z}$. Therefore, by Lemma 7(vi)

$$\begin{aligned} \gcd(a, l) &\sim \gcd(a - ql, l) \\ &= \gcd(b, l). \quad \square \end{aligned}$$

Corollary 43 Let $f \in \mathbb{Z}[X]$ and $l \in \mathbb{N}_{\geq 1}$. Then

$$p_1 \equiv_l p_2 \Longrightarrow \gcd(f(p_1), l) \sim \gcd(f(p_2), l)$$

for all $p_1, p_2 \in \mathbb{Z}$.

Proof. Let f and l be as supposed. Let $p_1, p_2 \in \mathbb{Z}$. Then

$$p_1 \equiv_l p_2 \Longrightarrow f(p_1) \equiv_l f(p_2)$$

by Lemma 6 and

$$f(p_1) \equiv_l f(p_2) \Longrightarrow \gcd(f(p_1), l) \sim \gcd(f(p_2), l)$$

by the preceding lemma. □

Lemma 44 Let $a, b \in \mathbb{Z}$ and $l \in \mathbb{N}_{\geq 1}$ such that

$$\gcd(a, b) \mid l.$$

Then

$$\gcd(a, b) \sim \gcd(a, b, l).$$

Proof. Let a, b and l be as stated and denote by d a gcd of a and b, i.e., $d \sim \gcd(a, b)$. Since $d \mid l$ there is some $q \in \mathbb{Z}$ such that $l = dq$. Then follows

$$\begin{aligned} \gcd(a, b) &\sim d \cdot 1 \\ &\sim d \gcd(1, q) \\ &\sim \gcd(d, dq) \\ &\sim \gcd(\gcd(a, b), l) \\ &\sim \gcd(a, b, l). \quad \square \end{aligned}$$

Proposition 45 *Let $f, g \in \mathbb{Z}[X]$ be relatively prime in $\mathbb{Q}[X]$. Then there is some $l \in \mathbb{N}_{\geq 1}$ such that*

$$p_1 \equiv_l p_2 \Longrightarrow \gcd(f(p_1), g(p_1)) \sim \gcd(f(p_2), g(p_2))$$

for all $p_1, p_2 \in \mathbb{Z}$. In particular, set $c_i := |\gcd(f(i), g(i))|$ for $i \in \{0, \dots, l-1\}$. Then

$$\gcd(f(p), g(p)) \sim \begin{cases} c_0 & \text{if } p \equiv_l 0 \\ \dots & \\ c_{l-1} & \text{if } p \equiv_l (l-1) \end{cases}$$

Proof. Let f and g be as required. By Lemma 41 there is some $0 \neq l' \in \mathbb{Z}$ such that

$$\gcd(f(p), g(p)) \mid l'$$

and therefore, with $l := |l'| \geq 1$

$$\gcd(f(p), g(p)) \mid l$$

for all $p \in \mathbb{Z}$. Now let $p_1, p_2 \in \mathbb{Z}$ such that $p_1 \equiv_l p_2$. By repeated application of Lemma 7 and Corollary 43 we get

$$\gcd(f(p_1), g(p_1), l) \sim \gcd(f(p_2), g(p_2), l). \tag{4.9}$$

By the above Lemma 44 we additionally get

$$\gcd(f(p_1), g(p_1)) \sim \gcd(f(p_1), g(p_1), l) \tag{4.10}$$

and

$$\gcd(f(p_2), g(p_2)) \sim \gcd(f(p_2), g(p_2), l), \tag{4.11}$$

and finally, by putting (4.9), (4.10) and (4.11) together, the proposed relation. Now, set $c_i := |\gcd(f(i), g(i))|$ for $i \in \{0, \dots, l-1\}$ and define

$$\iota : \mathbb{Z} \to \mathbb{Z} : p \mapsto p \bmod l$$

Table 4.2 $d(p) \sim \gcd(\, p^2,\ 3p+2\,)$

p	-5	-4	-3	-2	-1	0	1	2	3	4	5
p^2	25	16	9	4	1	0	1	4	9	16	25
$3p+2$	-13	-10	-7	-4	-1	2	5	8	11	14	17
$d(p)$	1	2	1	4	1	2	1	4	1	2	1

such that $p \equiv_l \iota(p)$ and $0 \le \iota(p) < l$ for all $p \in \mathbb{Z}$. Then, with what we have already proven,

$$\begin{aligned} \gcd(f(p), g(p)) &\sim \gcd(f(\iota(p)), g(\iota(p))) \\ &\sim c_{\iota(p)} \\ &= \begin{cases} c_0 & \text{if } \iota(p) = 0, \text{ i.e., } p \equiv_l 0 \\ \dots & \\ c_{l-1} & \text{if } \iota(p) = (l-1), \text{ i.e., } p \equiv_l (l-1) \end{cases} . \end{aligned}$$

This concludes our proof. □

Remark 46 The preceding corollary contains a way to compute the gcd of two polynomial functions $f(p)$ and $g(p)$ where $f, g \in \mathbb{Z}[X]$ are relatively prime (in $\mathbb{Q}[X]$): Let $f, g \in \mathbb{Z}[X]$ be two such polynomials. As done in the proof of Lemma 41, find (with the extended Euclidean Algorithm for polynomials in $\mathbb{Q}[X]$) some $0 \neq l \in \mathbb{Z}$, such that for all $p \in \mathbb{Z}$

$$\gcd(f(p), g(p)) \mid l.$$

Then compute a gcd d_i of $f(i)$ and $g(i))$ for every $0 \le i < l$. Given any $p \in \mathbb{Z}$, to determine $d(p) \sim \gcd(f(p), g(p))$ it is now enough to identify the residue class (modulo l) p belongs to: if $p \equiv_l i_0$ (where $0 \le i_0 < l$) then

$$d(p) \sim d_{i_0}.$$

Example 47 Let $f := X^2$ and $g := 3X + 2$. By the Extended Euclidean Algorithm (which requires only one polynomial division) we find $s = \frac{9}{4}$ and $t = -\frac{3}{4}p + \frac{1}{2}$ such that

$$sf + tg = 1$$

which implies

$$9f + (-3p + 2)g = 4.$$

Therefore, by Lemma 41 and its proof for all $p \in \mathbb{Z}$

$$gcd(p^2, 3p + 2) \mid 4.$$

Table 4.2 shows–among others–the gcd's for $p \in \{0, 1, 2, 3\}$. By Proposition 45 we therefore know that

$$\gcd(p^2, 3p+2) \sim \begin{cases} 2 & \text{if } p \equiv_4 0 \\ 1 & \text{if } p \equiv_4 1 \\ 4 & \text{if } p \equiv_4 2 \\ 1 & \text{if } p \equiv_4 3 \end{cases}$$

Now we are prepared to see that the gcd function of two arbitrary integer polynomials is indeed an aiq-polynomial.

Theorem 48 *Let $f, g \in \mathbb{Z}[X]$. Then there is some $\delta \in \mathcal{AIQ}$ such that for all $p \in \mathbb{Z}$*

$$\delta(p) \sim \gcd(f(p), g(p)). \tag{4.12}$$

In particular, let l be a modulus of δ and let $h \sim \gcd(f, g)$ be a polynomial gcd of f and g in $\mathbb{Z}[X]$. Then each constituent d_i of δ has the form

$$d_i(X) = c_i h_i(X),\ (0 \leq i < l) \tag{4.13}$$

where $c_i \in \mathbb{Z}$ and h_i is the i-th constituent determined by l-extending h.

Proof. Let $f, g \in \mathbb{Z}[X]$, let $h \in \mathbb{Z}[X]$ be a gcd of f and g in $\mathbb{Z}[X]$ and let $f', g' \in \mathbb{Z}[X]$ such that $f = hf'$ and $g = hg'$, i.e., $\gcd(f', g') \sim 1$ in $\mathbb{Z}[X]$. Then, by Proposition 45, there is some $l \in \mathbb{N}_{\geq 1}$ and there are $c_0, \ldots, c_{l-1} \in \mathbb{Z}$, such that

$$\gcd(f'(p), g'(p)) \sim \begin{cases} c_0 & \text{if } p \equiv_l 0 \\ \ldots & \\ c_{l-1} & \text{if } p \equiv_l (l-1) \end{cases}.$$

Denote by $h_0(X), \ldots, h_{l-1}(X)$ the l-extensions of h such that for all $p \in \mathbb{Z}$

$$h(p) = \begin{cases} h_0(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 0 \\ \ldots & \\ h_{l-1}(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l (l-1) \end{cases}$$

Then for all $p \in \mathbb{Z}$

$$\begin{aligned} \gcd(f(p), g(p)) &\sim \gcd(h(p)f'(p), h(p)g'(p)) \\ &\overset{(*)}{\sim} h(p) \gcd(f'(p), g'(p)) \\ &\sim \begin{cases} h_0(\lfloor \frac{p}{l} \rfloor)c_0 & \text{if } p \equiv_l 0 \\ \ldots & \\ h_{l-1}(\lfloor \frac{p}{l} \rfloor)c_{l-1} & \text{if } p \equiv_l (l-1) \end{cases} \end{aligned} \tag{4.14}$$

where we used Lemma 7(v) for $(*)$. Clearly, the last line of (4.14) uncovers $\gcd(f(p), g(p)$ as an aiq-polynomial in the form required by our statement which was to be proven. □

So far, we considered pointwise properties of two integral polynomial functions f and g because we realized that a *mere polynomial view* was in general too coarse for the intended applications. This pointwise view led us to the insight that instead of observing f and g, we better turn our attention to certain related aiq-polynomials – and in particular we just proved that the pointwise gcd of f and g can be expressed as an aiq-polynomial. The following lemma considers properties of the involved constituents, now again from a more *polynomial* perspective.

Corollary 49 Let $f, g \in \mathbb{Z}[X]$, let $\delta \in \mathcal{AIQ}$ be their aiq-gcd and let l be a modulus of δ. Denote the constituents of δ by $d_0, \ldots, d_{l-1} \in \mathbb{Z}[X]$ and denote by $f_0, \ldots, f_{l-1} \in \mathbb{Z}[X]$ and $g_0, \ldots, g_{l-1} \in \mathbb{Z}[X]$ the constituents of f and g after l-extending them. Then

$$d_i \mid f_i \text{ and } d_i \mid g_i$$

for all $i \in \{0, \ldots, l-1\}$.

Proof. First, assume $\gcd(f, g) \sim 1$ such that each $d_i \in \mathbb{Z}$. Let $i \in \{0, \ldots, l-1\}$. Then we have to show that d_i divides every coefficient of f_i. Remember that by Definition 34

$$\begin{aligned} f_i(X) &= f(lX + i) \\ &= l\widehat{f_i}(X) + f(i). \end{aligned}$$

Since $d_i|l$ (cf. Proposition 45 and its proof) and of course $d_i|f(i)$, every coefficient of f_i is divisible by d_i which implies $d_i|f_i$. In the same way we can prove $d_i|g_i$.
Now, let f and g be arbitrary polynomials in $\mathbb{Z}[X]$ and let $h \in \mathbb{Z}[X]$ be their gcd. Let $f', g' \in \mathbb{Z}[X]$ such that $f = hf'$ and $g = hg'$. By Theorem 48, the aiq-gcd δ of f and g is determined by the aiq-gcd of f' and g' (let l again denote one of its modulus), and the l-extension of h, i.e., with the notation from the proof of Theorem 48, we get

$$\delta(p) \sim \begin{cases} c_0 h_0(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 0 \\ \ldots \\ c_{l-1} h_{l-1}(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l (l-1) \end{cases}$$

i.e., $d_i = c_i h_i$. As in the former case, $c_i|f'_i$, where $f'_0, \ldots, f'_{l-1}$ are the constituents of the l-extension of f'. Furthermore, for each $i \in \{0, \ldots, l-1\}$

$$\begin{aligned} f_i(X) &= f(lX + i) \\ &= h(lX + i) f'(lX + i) \\ &= h_i(X) f'(lX + i) \\ &= h_i(X) f'_i(X) \end{aligned}$$

such that obviously

$$d_i = c_i h_j | f'_i h_i = f_i$$

as stated. □

Aiq-gcd's of more than two polynomials So far we gained some insight into the structure of the gcd mapping of *two* integral polynomials. What, if more than two polynomials in $\mathbb{Z}[X]$ are given? We prepare the answer with the following lemma and corollary.

Lemma 50 Let $k \in \mathbb{N}_{\geq 1}$ and $f_0, \ldots, f_{(k-1)} \in \mathcal{AIQ}$. Then the mapping

$$f : \mathbb{Z} \longrightarrow \mathbb{Z} : p \mapsto \begin{cases} f_0(\lfloor \frac{p}{k} \rfloor) & \text{if } p \equiv_k 0 \\ \ldots \\ f_{(k-1)}(\lfloor \frac{p}{k} \rfloor) & \text{if } p \equiv_k (k-1) \end{cases}$$

is an aiq-polynomial.

Proof. Let $k \in \mathbb{N}_{\geq 1}$ and $f_0, \dots, f_{(k-1)} \in \mathcal{AIQ}$ be expressed with a common modulus l, cf. Lemma 35, such that for $0 \leq i < k$, each f_i consists of l constituents $f_{i0}, \dots, f_{i(l-1)}$. Now, we claim that for all $p \in \mathbb{Z}$

$$f(p) = f_{ij}(\lfloor \frac{p}{kl} \rfloor),$$

where $(i,j) = \varphi(r)$ with $r \equiv_{kl} p$ $(0 \leq r < kl)$ is determined by the bijection φ from Remark 36. If this is true, the f_{ij} will directly constitute kl constituents of f and $f \in \mathcal{AIQ}$ as proposed.

So, let $p \in \mathbb{Z}$ and choose $0 \leq r < kl$ such that $p \equiv_{lk} r$, i.e.,

$$p = klq + r$$

for some $q \in \mathbb{Z}$. Set

$$(i,j) := \varphi(r) = (r - \lfloor \frac{r}{k} \rfloor, \lfloor \frac{r}{k} \rfloor)$$

and note that $0 \leq i < k$ and $0 \leq j < l$ by definition of φ. Since $p \equiv_l r \equiv_k i$ we find that

$$f(p) = f_i(\lfloor \frac{p}{k} \rfloor)$$

must hold. Because of

$$\left\lfloor \frac{p}{k} \right\rfloor = \left\lfloor \frac{klq + r}{k} \right\rfloor = lq + \left\lfloor \frac{r}{k} \right\rfloor$$

we get

$$\left\lfloor \frac{p}{k} \right\rfloor \equiv_l \left\lfloor \frac{r}{k} \right\rfloor = j$$

which means that we have to choose the j-th constituent of f_i, such that indeed

$$\begin{aligned} f_i(\left\lfloor \frac{p}{k} \right\rfloor) &= f_{ij}(\left\lfloor \frac{\lfloor \frac{p}{k} \rfloor}{l} \right\rfloor) \\ &= f_{ij}(\left\lfloor \frac{p}{kl} \right\rfloor) \end{aligned}$$

as claimed. □

Corollary 51 Let $\delta \in \mathcal{AIQ}$ and $f \in \mathbb{Z}[X]$. Then there is some aiq-polynomial ϵ such that for all $p \in \mathbb{Z}$

$$\epsilon(p) \sim \gcd(\delta(p), f(p)).$$

Proof. We first define the desired ϵ. Let l denote a modulus of δ and let $\delta_0, \dots, \delta_{(l-1)} \in \mathbb{Z}[X]$ be the constituents of δ. Moreover, l-extend f such that $f_0, \dots, f_{(l-1)} \in \mathbb{Z}[X]$ are the constituents of f. By Theorem 48, for each $0 \leq i < l$ there is an aiq-polynomial ϵ_i such that for all $p \in \mathbb{Z}$

$$\gcd(\delta_i(p), f_i(p)) \sim \epsilon_i(p).$$

By Lemma 50,

$$\epsilon : \mathbb{Z} \longrightarrow \mathbb{Z} : p \mapsto \begin{cases} \epsilon_0(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l 0 \\ \dots & \\ \epsilon_{(l-1)}(\lfloor \frac{p}{l} \rfloor) & \text{if } p \equiv_l (l-1) \end{cases}$$

is again an aiq-polynomial. Let us next put everything together and verify that actually

$$\epsilon(p) \sim \gcd(\delta(p), f(p))$$

for all $p \in \mathbb{Z}$, as desired. To do so, let $p \in \mathbb{Z}$. Let $0 \leq i < l$ be such that $p \equiv_l i$. Then

$$\begin{aligned} \gcd(\delta(p), f(p)) &= \gcd(\delta_i(\lfloor \frac{p}{l} \rfloor), f_i(\lfloor \frac{p}{l} \rfloor)) \\ &\sim \epsilon_i(\lfloor \frac{p}{l} \rfloor) \\ &= \epsilon(p) \end{aligned}$$

which proves our claim. □

Given $f_1, \ldots, f_m \in \mathbb{Z}[X]$ $(m \geq 2)$, this corollary offers an inductive argument on the number m of given polynomials that there is indeed an aiq-polynomial δ such that

$$\delta(p) \sim \gcd(f_1(p), \ldots, f_m(p))$$

for all $p \in \mathbb{Z}$. If $m = 2$, apply Theorem 48. If $m > 2$, by induction hypothesis there is a $\delta \in \mathcal{AIQ}$ such that

$$\delta(p) \sim \gcd(f_1(p), \ldots, f_{(m-1)}(p))$$

for all $p \in \mathbb{Z}$ and therefore

$$\gcd(f_1(p), \ldots, f_{(m-1)}(p) f_m(p)) \sim \gcd(\delta(p), f_m(p)).$$

Now apply Corollary 51. The conclusion is drawn in the following corollary.

Corollary 52 Let $m \geq 2$ and $f_1, \ldots, f_m \in \mathbb{Z}[X]$. Then there is an aiq-polynomial ϵ such that for all $p \in \mathbb{Z}$

$$\epsilon(p) \sim \gcd(f_1(p), \ldots, f_m(p)).$$

Before we proceed with the more algorithmic aspects of the "aiq-gcd-mappings", let us note that it takes only a slight modification in the proof of Corollary 51 to state the following corollary.

Corollary 53 Let $\delta_1, \delta_2 \in \mathcal{AIQ}$. Then there is some $\epsilon \in \mathcal{AIQ}$ such that for all $p \in \mathbb{Z}$

$$\epsilon(p) \sim \gcd(\delta_1(p), \delta_2(p)).$$

Proof. Let $\delta_1, \delta_2 \in \mathcal{AIQ}$, let l be a common period of δ_1 and δ_2 and denote their constituents by $\delta_{10}, \ldots, \delta_{1(l-1)}$ and $\delta_{20}, \ldots, \delta_{2(l-1)}$, respectively. Then we can simply replace each occurrence of "δ" in the proof of Corollary 51 by δ_1 and each occurrence of "f" by δ_2, and so forth. □

4.1.3 Intermezzo: Divisibility of Integral Polynomial Functions

As a first application of aiq-gcd's, we consider the following problem: Let $f, g \in \mathbb{Z}[X]$. For which $p \in \mathbb{Z}$ does $f(p)$ divide $g(p)$? Or, stated in a different way, can we explicitly describe the set

$$D(f|g) := \{p \in \mathbb{Z} \mid f(p)|g(p)\} \ ?$$

The following lemma is the key to the answer which in turn is given by the subsequent corollary.

Lemma 54 Let $a, b \in \mathbb{Z}$. Then

$$a \mid b \text{ iff } a \sim \gcd(a, b).$$

Proof. The proof follows directly from the respective definitions. □

Lemma 55 Let $f, g \in \mathbb{Z}[X]$. Then

$$\forall\, p \in \mathbb{Z}\ (f(p) \sim g(p)) \quad \Longleftrightarrow \quad (f = g \vee f = -g).$$

Proof. "$\Rightarrow$": Let $n = \max\{\deg(f), \deg(g)\}$. Since there are infinitly many points $p \in \mathbb{Z}$ with either $f(p) = g(p)$ or $f(p) = -g(p)$ we can find in particular $n+1$ points such that either $f(p) = g(p)$ or $f(p) = -g(p)$. But any polynomials of degree n is completly detetermined by $n+1$ points. Therefore, either $f = g$ or $f = -g$.
"$\Leftarrow$": Clear. □

Corollary 56 Let $f, g \in \mathbb{Z}[X]$, let $\delta \in \mathcal{AIQ}$ be their aiq-gcd and let l be some modulus of δ. Then there is some finite (possibly empty) set $M \subset \mathbb{Z}$ and there exist $0 \leq k \leq l$ different integers $0 \leq n_1 < \cdots < n_k < l$ such that

$$D(f|g) = M \cup [n_1]_l \cup \cdots \cup [n_k]_l.$$

Proof. Let f, g, δ and l be as stated and let $d_0 \ldots, d_{(l-1)} \in \mathbb{Z}[X]$ be the constituents of δ. By l-extending f, let $f_0, \ldots, f_{(l-1)} \in \mathbb{Z}[X]$ denote the constituents of f. Now, consider the set

$$S_i := \{s \in \mathbb{Z} \mid f_i(s) \sim d_i(s)\}$$

for $0 \leq i < l$. Because $a \sim b$ iff $a = b \vee a = -b$, we get

$$\begin{aligned} S_i &= \{s \in \mathbb{Z} \mid f_i(s) = d_i(s)\} \cup \{s \in \mathbb{Z} \mid f_i(s) = -d_i(s)\} \\ &= \{s \in \mathbb{Z} \mid f_i(s) - d_i(s) = 0\} \cup \{s \in \mathbb{Z} \mid f_i(s) + d_i(s) = 0\}. \end{aligned}$$

Each S_i is either finite (if $f_i \neq g_i \wedge f_1 \neq -g_i$) and can be computed by the methods described in [WBK93] or equals $\mathbb{Z}$, in which case $f_i = g_i \vee f_i = -g_i$ (cf. Lemma 55). Therefore, each set $lS_i + i$ is either finite or equals $[i]_l$. Next, we set

$$S := \bigcup_{i=1}^{l-1} (lS_i + i)$$

(note that S is in the stated form) and show that

$$D(f|g) = S.$$

So, let $p \in D(f|g)$ and set $i := p \mod l$ such that $p = l\lfloor \frac{p}{l} \rfloor + i$. Then, with Lemma 54 (for $(*)$) and Theorem 48 (for $(**)$), we get

$$\begin{aligned} p \in D(f|g) &\Longleftrightarrow f(p)|g(p) \\ &\overset{(*)}{\Longleftrightarrow} f(p) \sim \gcd(f(p), g(p)) \\ &\overset{(**)}{\Longleftrightarrow} f(p) \sim \delta(p) \\ &\Longleftrightarrow i = p \bmod l \wedge f_i(\left\lfloor \frac{p}{l} \right\rfloor) \sim d_i(\left\lfloor \frac{p}{l} \right\rfloor) \\ &\Longleftrightarrow i = p \bmod l \wedge \left\lfloor \frac{p}{l} \right\rfloor \in S_i \\ &\Longleftrightarrow i = p \bmod l \wedge l\left\lfloor \frac{p}{l} \right\rfloor + i \in lS_i + i \\ &\Longrightarrow p \in S. \end{aligned}$$

Conversely, if $p \in S$, then there is some $i \in \{0, \ldots, l-1\}$ such that $p \in lS_i + i$, i.e., there is some $p' \in S_i$ such that $p = lp' + i$ which means

$$i = p \mod l \wedge l\left\lfloor \frac{p}{l} \right\rfloor + i \in lS_i + i$$

and the above equivalences go all the way up and lead to $p \in D(f|g)$. □

Example 57 Let $f = 2(X+1)$ and $g = X(X+1)$. The polynomial gcd of f and g is $X+1$ such that it is easy to see that

$$\begin{aligned} \gcd(f(p), g(p)) &\sim \begin{cases} 2(2p'+1) & \text{if } p = 2p' \\ 1(2p'+2) & \text{if } p = 2p'+1 \end{cases} \\ &= \begin{cases} 4\lfloor \frac{p}{2} \rfloor + 2 & \text{if } p \equiv_2 0 \\ 2\lfloor \frac{p}{2} \rfloor + 2 & \text{if } p \equiv_2 1 \end{cases} \end{aligned}$$

which gives us $d_0 = 4X + 2$ and $d_1 = 2X + 2$. The 2-extension of f is given by

$$\begin{aligned} f(p) &= \begin{cases} 2(2p'+1) & \text{if } p = 2p' \\ 2((2p'+1)+1) & \text{if } p = 2p'+1 \end{cases} \\ &= \begin{cases} 4\lfloor \frac{p}{2} \rfloor + 2 & \text{if } p \equiv_2 0 \\ 4\lfloor \frac{p}{2} \rfloor + 4 & \text{if } p \equiv_2 1 \end{cases} \end{aligned}$$

such that $f_0 = 4X + 2$ and $f_1 = 4X + 4$. This implies

$$\begin{aligned} S_0 &= \{s \in \mathbb{Z} \mid (f_0 \pm d_0)(s) = 0\} \\ &= \mathbb{Z} \end{aligned}$$

since $f_0 = d_0$, and

$$\begin{aligned} S_1 &= \{s \in \mathbb{Z} \mid (f_1 \pm d_1)(s) = 0\} \\ &= \{s \in \mathbb{Z} \mid 6s + 6 = 0 \vee 2s + 2 = 0\} \\ &= \{-1\}. \end{aligned}$$

Table 4.3

p	-5	-4	-3	-2	-1	0	1	2	3	4	5
$f(p)$	-8	-6	-4	2	0	2	4	6	8	10	12
$g(p)$	20	12	6	2	0	0	2	6	12	20	30
$\gcd(f(p), g(p))$	-4	-6	-2	-2	0	2	2	6	4	10	6

Therefore,

$$\begin{aligned} D(f|g) &= (2\mathbb{Z} + 0) \cup (2\{-1\} + 1) \\ &= 2\mathbb{Z} \cup \{-1\} \end{aligned}$$

c.f. Table 4.3.

Corollary 58 Let $f, g \in \mathbb{Z}[X]$ and let l be some period of the aiq-gcd d of f and g. Let $[i]_l \subset D(f|g)$ denote one of the equivalence classes as described in Corollary 56 ($i \in \{0, \ldots, l-1\}$). Then $f_i | g_i$, where $f_0, \ldots, f_{l-1}$ and $g_0, \ldots, g_{l-1}$ denote the l-extensions of f and g.

Proof. The statement follows from the fact that either $f_i = d_i$ or $f_i = -d_i$, similarly as in the proof of Corollary 56. □

4.1.4 The Extended Euclidean Algorithm for Aiq-gcd's

Recall the Introductory Example given above, where we not only provided an aiq-polynomial representing the gcd of given polynomials in $\mathbb{Z}[X]$ but also derived matrices $U_i \in \mathbb{Z}[X]^{m \times n}$ such that the $\mathbf{U}_i(p)$ were unimodular for every $p \in \mathbb{Z}$ and therefore led to solutions of a corresponding parametric equation system. In the following we describe how these matrices can be systematically constructed while also finding the aiq-gcd for given $f_1, \ldots, f_n \in \mathcal{AIQ}$. Thereby, we let us guide by the following observation.

Observation 59 Let $f = \sum_{i=0}^{m} a_i X^i$ and $g = \sum_{i=0}^{n} b_i X^i$ be integer polynomials with $a_m, b_n > 0$. Without loss of generality, we may assume $m \geq n$ and consider the following two cases:

- $\mathbf{m = n}$ Define a sequence $(f_i, g_i)_{i \in \mathbb{N}}$ of pairs of integer polynomials:

$$(f_0, g_0) := (f, g)$$

$$(f_{i+1}, g_{i+1}) := \begin{cases} (g_i, f_i - \lfloor \frac{\mathrm{HC}\,(f_i)}{\mathrm{HC}\,(g_i)} \rfloor g_i) & \text{if } \deg(g_i) = n \\ (f_i, g_i) & \text{otherwise.} \end{cases}$$

Comparing this sequence to Theorem 13 and its proof, we could say that the computation of the gcd of a_m and b_n is "lifted" to f and g. As the computation of $\gcd(a_m, b_n)$ in Algorithm 1 terminates, there is some $i_0 \in \mathbb{N}$ such that $\deg(g_{i_0}) < n$ and $\deg(g_{i_0 - 1}) = n$. Let

$$\mathbf{U} = \begin{pmatrix} s_1 & t_1 \\ s_2 & t_2 \end{pmatrix}$$

be the matrix from Algorithm 3. Then

$$\begin{aligned} f_{i_0} &= s_1 f + t_1 g \\ g_{i_0} &= s_2 f + t_2 g. \end{aligned}$$

Moreover, note that

$$\gcd(f(p), g(p)) \sim \gcd(f_i(p), g_i(p))$$

for all $i \in \mathbb{N}$ and all $p \in \mathbb{Z}$ (cf. Lemma 18). Altogether, we can say that by "lifting" Algorithm 1 to the leading coefficients of f and g, we can reduce the degree of one polynomial without loosing information about $\gcd(f(p), g(p))$.

- $\mathbf{m > n}$ For $0 \leq k < |b_n|$, consider the polynomials

$$\begin{aligned} f(b_n Y + k) &= \sum_{i=0}^{m} a_i (b_n Y + k)^i \\ &= \sum_{i=0}^{m} a_i \sum_{j=0}^{i} \binom{i}{j} (b_n Y)^j k^{(i-j)} \\ &= a_m b_n^m Y^m + \underbrace{a_m \sum_{j=0}^{m-1} \binom{m}{j} (b_n Y)^j k^{(m-j)} + \sum_{i=0}^{m-1} \sum_{j=0}^{i} a_i \binom{i}{j} (b_n Y)^j k^{(i-j)}}_{:=\hat{f}_k(Y)} \end{aligned}$$

and

$$\begin{aligned} g(b_n Y + k) &= \sum_{i=0}^{n} b_i (b_n Y + k)^i \\ &= \sum_{i=0}^{n} b_i \sum_{j=0}^{i} \binom{i}{j} (b_n Y)^j k^{(i-j)} \\ &= b_n^{n+1} Y^n + \underbrace{b_n \sum_{j=0}^{n-1} \binom{n}{j} (b_n Y)^j k^{(n-j)} + \sum_{i=0}^{n-1} b_i \sum_{j=0}^{i} \binom{i}{j} (b_n Y)^j k^{(i-j)}}_{:=\hat{g}_k(Y)} \end{aligned}$$

where $a_m b_n^m Y^m$ and $b_n^{n+1} Y^n$ are the respective highest terms, i.e., $\deg(\hat{f}_k) < m$ and $\deg(\hat{g}_k) < m$. Because $m > n$, we get

$$b_n^{n+1} Y^n \mid a_m b_n^m Y^m. \tag{4.15}$$

and therefore, the polynomial

$$\begin{aligned} h_k(Y) &:= f(b_n Y + k) - a_m b_n^{(m-n-1)} Y^{m-n} g(b_n Y + k) \\ &= (a_m b_n^m Y^m + \hat{f}_k(Y)) - a_m b_n^m Y^m - a_m b_n^{(m-n-1)} \hat{g}_k(Y) \\ &= \hat{f}_k(Y) - a_m b_n^{(m-n-1)} \hat{g}_k(Y) \end{aligned}$$

has degree $< m$. The idea behind the substitution of X by $b_nY + k$ is lent from b_n-extending $f(p)$ and $g(p)$: Any $p \in \mathbb{Z}$ can be uniquely written as

$$p = b_n p' + k$$

with $p' \in \mathbb{Z}$ and $0 \leq k < |b_n|$ and so

$$\begin{aligned} \gcd(f(p), g(p)) &\sim \gcd(f(b_np'+k), g(b_np'+k)) \\ &\sim \gcd(f(b_np'+k) - a_m b_n^{m-n-1} p^{m-n} g(b_np'+k), g(b_np'+k)) \\ &\sim \gcd(h_k(p'), g(b_np'+k)) \\ &\sim \gcd(g(b_np'+k), h_k(p')). \end{aligned}$$

Again, we reduced the degree of one polynomial and preserved information about $\gcd(f(p), g(p))$. Moreover, we can find a unimodular matrix $\mathbf{U}(p)$ such that

$$\mathbf{U}(p') \begin{pmatrix} f(b_np'+k) \\ g(b_np'+k) \end{pmatrix} = \begin{pmatrix} g(b_np'+k) \\ h_k(p') \end{pmatrix},$$

namely

$$\mathbf{U}(p') = \begin{pmatrix} 0 & 1 \\ 1 & -a_m b_n^{(m-n-1)} p'^{(m-n)} \end{pmatrix}$$

However, this comes at the cost of having to treat b_n different cases, one for each residue class modulo b_n that p could belong to. Note that the case distinction on b_n is essentially a b_n-reduction. Note moreover that $\mathbf{U}(p')$ does not depend on k! □

Observation 59 contains the two main concepts (cases $\mathbf{m} = \mathbf{n}$ and $\mathbf{m} > \mathbf{n}$) needed to develop the desired algorithm. The core idea of *both* concepts is to reduce the degrees of the polynomials involved to get more and more "information" about $\gcd(f(p), g(p))$. To see the idea of this kind of reducibility in the right light, we provide the following definition and lemma.

Definition 60 Let $f, g \in \mathbb{Z}[X]$, $f, g \neq 0$. Then we say that f is *d-reducible* (degree-reducible) by g and write $f \downarrow_d g$ if either $\deg(f) = \deg(g)$ or $\deg(f) > \deg(g)$ and $\mathrm{HC}(g) \mid \mathrm{HC}(f)$.

Lemma 61 *Let $f, g \in \mathbb{Z}[X]$ and $f \downarrow_d g$. Then one can find a unimodular matrix $\mathbf{U} \in \mathbb{Z}[X]^{2\times 2}$ and polynomials $f', g' \in \mathbb{Z}[X]$ such that*

(i) $\mathbf{U}\binom{f}{g} = \binom{f'}{g}$,

(ii) $\deg(f) > \deg(f')$

(iii) for all $p \in \mathbb{Z}$

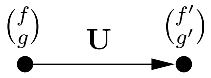
$$\gcd(f(p), g(p)) \sim \gcd(f'(p), g'(p)).$$

In particular, if $\deg(f) = \deg(g)$ then the matrix $\mathbf{U}$ is given by *extGcd* $[\mathrm{HC}(f), \mathrm{HC}(g)]$. If $\deg(f) > \deg(g)$ then $\mathbf{U}$ is given by

$$\begin{pmatrix} 1 & -\frac{\mathrm{HC}(f)}{\mathrm{HC}(g)} X^{\deg(f)-\deg(g)} \\ 0 & 1 \end{pmatrix}$$

Proof. The propositions follow immediately from Definition 60, Observation 59 and Lemma 18. □

In a very informal way, we could visualize the effect of the last lemma as follows:

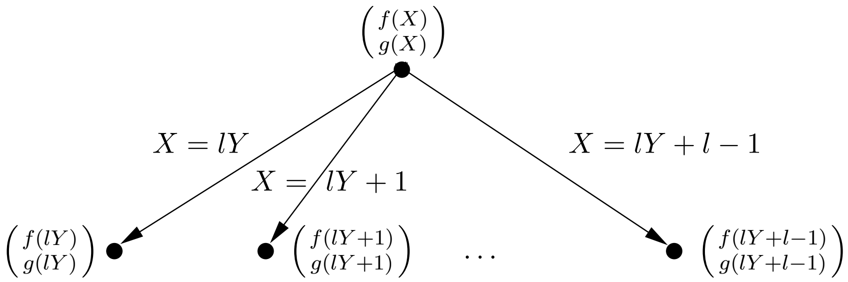


This shall mean that f is d-reducible by g and f', g' and $\mathbf{U}$ are such that they meet the three properties of Lemma 61. It is important to note that, by definition, if $f = 0$ or $g = 0$ then neither $f \downarrow_d g$ nor $g \downarrow_d f$. As well it is possible that even though $f, g \neq 0$ we find $f \not\downarrow_d g$ and $g \not\downarrow_d f$, too, take for instance $f = X^2$ and $g = 3X + 2$. In this situation the next lemma is helpful. It essentially restates Observation 59 with the terminology from Definition 60, therefore, we omit its proof.

Lemma 62 Let $f, g \in \mathbb{Z}[X]$ with $\deg(f) > \deg(g) \geq 0$ and $f \not\downarrow_d g$. Define $l := |\mathrm{HC}\,(g)|$. Then for $0 \leq i < l$ and $Y \neq X$

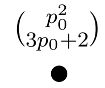
$$f(lY + i) \downarrow_d g(lY + i).$$

This specialization can be drawn as:

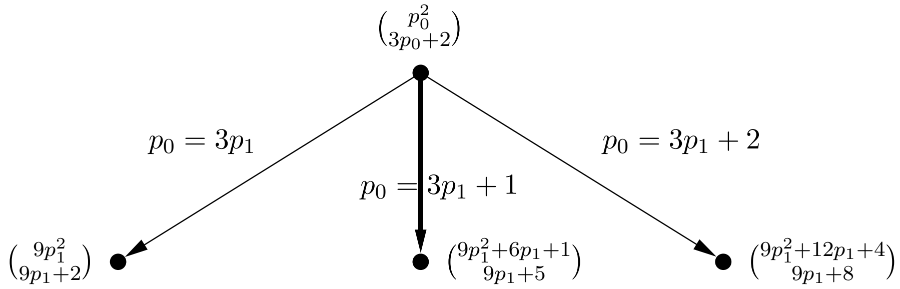


We are now ready to introduce the algorithm by an example illustrating the application of Observation 59 and the two preceding lemmas. It does so by construction of a tree that keeps track of the respective l-extensions and degree-reductions. It uses the informal visualisations given above to become familiar with the core idea.

Example 63 As in Example 47, let $f = X^2$ and $g = 3X + 2$. Let us assume $p_0 \in \mathbb{Z}$ and start the aiq-tree with a single node and labeled as follows:



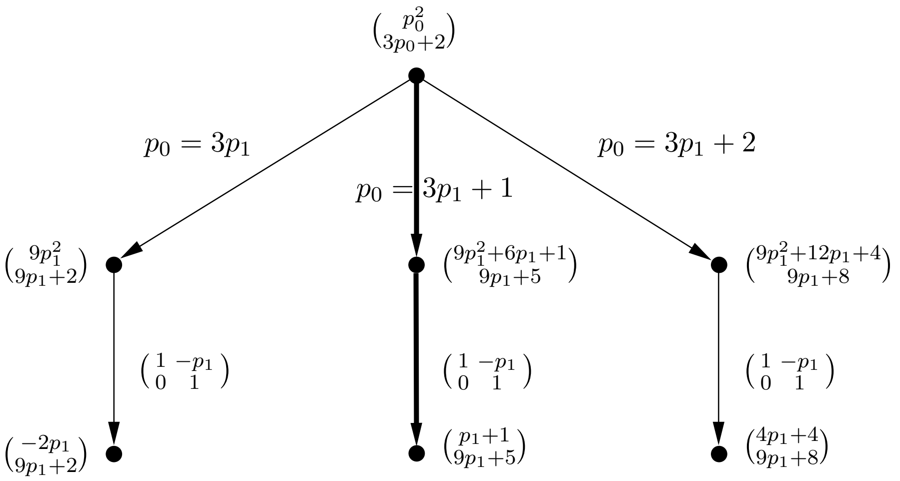
Since $\deg(f) > \deg(g)$, case $\mathbf{m} > \mathbf{n}$ of Observation 59 applies, and because $\mathrm{HC}\,(g) = 3$, we add three new cases for each $p_0 = 3p_1 + k$ where $k \in \{0, 1, 2\}$:



and with $h_k(p_1) = f(3p_1 + k) - p_1 g(3p_1 + k)$ and

$$\mathbf{U}(p_1) = \begin{pmatrix} 1 & -p_1 \\ 0 & 1 \end{pmatrix}$$

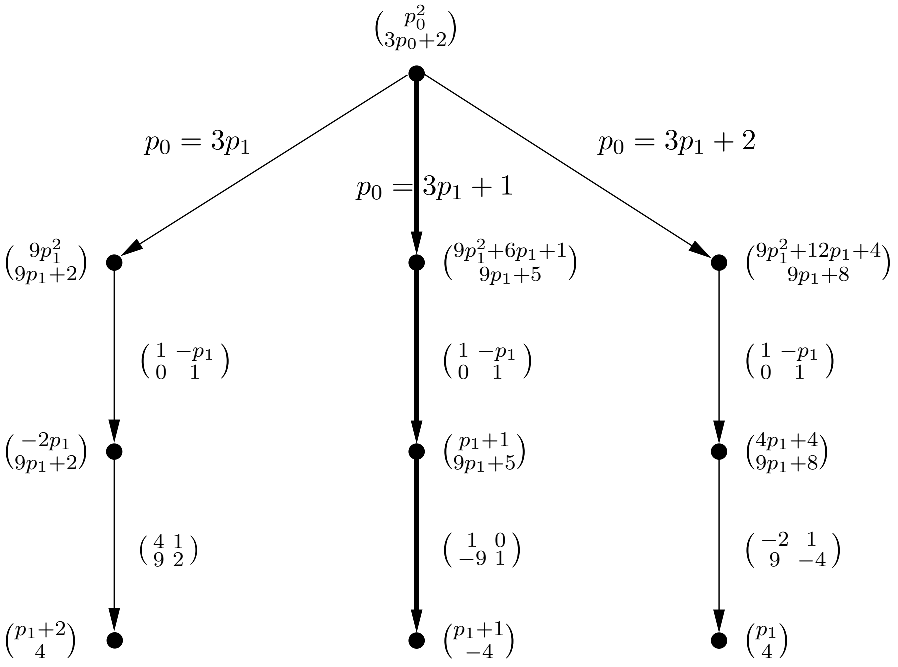
we get


$\binom{p_0^2}{3p_0+2}$
$p_0 = 3p_1$
$p_0 = 3p_1 + 1$
$p_0 = 3p_1 + 2$
$\binom{9p_1^2}{9p_1+2}$
$\binom{9p_1^2+6p_1+1}{9p_1+5}$
$\binom{9p_1^2+12p_1+4}{9p_1+8}$
$\begin{pmatrix} 1 & -p_1 \\ 0 & 1 \end{pmatrix}$
$\begin{pmatrix} 1 & -p_1 \\ 0 & 1 \end{pmatrix}$
$\begin{pmatrix} 1 & -p_1 \\ 0 & 1 \end{pmatrix}$
$\binom{-2p_1}{9p_1+2}$
$\binom{p_1+1}{9p_1+5}$
$\binom{4p_1+4}{9p_1+8}$

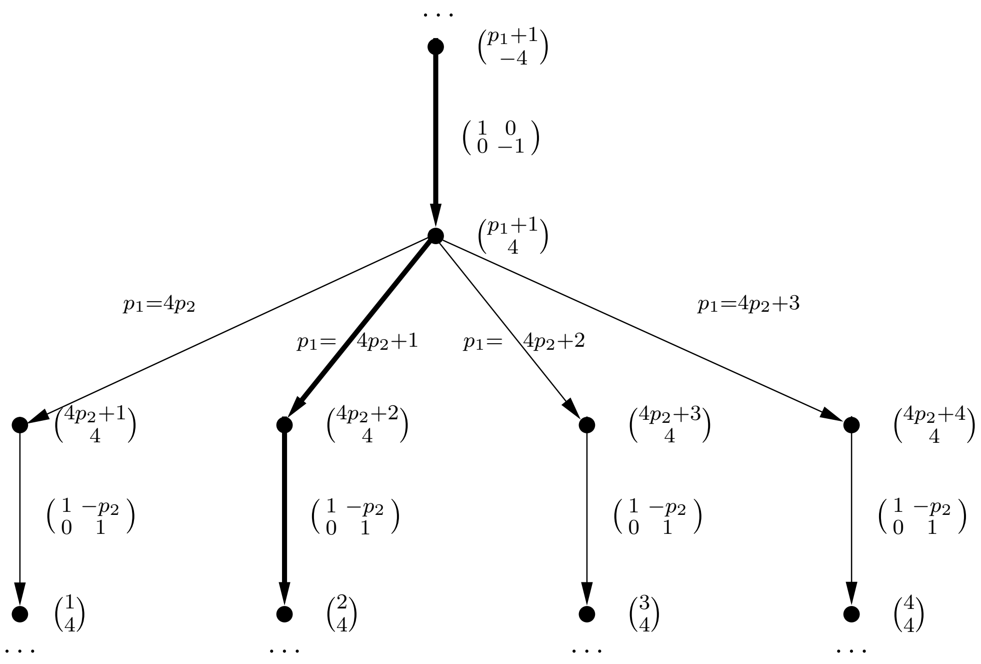
So, we were able to reduce the degree of f in each branch and can now apply case $\mathbf{m} = \mathbf{n}$ of Observation 59. Only considering head coefficients in the leaf of the tree constructed so far we compute

$$\begin{aligned}
&\mathit{extGcd}_2 \binom{-2}{9} = (\binom{1}{0}, \begin{pmatrix} 4 & 1 \\ 9 & 2 \end{pmatrix}) \\
&\mathit{extGcd}_2 \binom{1}{9} \;\;= (\binom{1}{0}, \begin{pmatrix} 1 & 0 \\ -9 & 1 \end{pmatrix}) \\
&\mathit{extGcd}_2 \binom{4}{9} \;\;= (\binom{1}{0}, \begin{pmatrix} -2 & 1 \\ 9 & -4 \end{pmatrix})
\end{aligned}$$

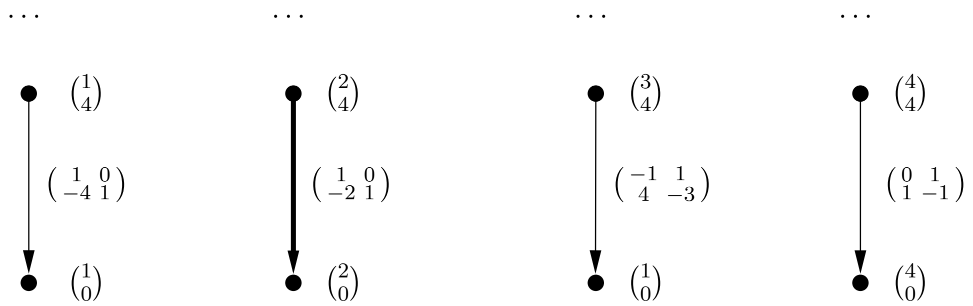
and extend the aiq-tree accordingly:


$\binom{p_0^2}{3p_0+2}$
$p_0 = 3p_1$
$p_0 = 3p_1 + 1$
$p_0 = 3p_1 + 2$
$\binom{9p_1^2}{9p_1+2}$
$\binom{9p_1^2+6p_1+1}{9p_1+5}$
$\binom{9p_1^2+12p_1+4}{9p_1+8}$
$\begin{pmatrix}1 & -p_1\\ 0 & 1\end{pmatrix}$
$\begin{pmatrix}1 & -p_1\\ 0 & 1\end{pmatrix}$
$\begin{pmatrix}1 & -p_1\\ 0 & 1\end{pmatrix}$
$\binom{-2p_1}{9p_1+2}$
$\binom{p_1+1}{9p_1+5}$
$\binom{4p_1+4}{9p_1+8}$
$\begin{pmatrix}4 & 1\\ 9 & 2\end{pmatrix}$
$\begin{pmatrix}1 & 0\\ -9 & 1\end{pmatrix}$
$\begin{pmatrix}-2 & 1\\ 9 & -4\end{pmatrix}$
$\binom{p_1+2}{4}$
$\binom{p_1+1}{-4}$
$\binom{p_1}{4}$

Observe again, how the degrees of the corresponding polynomials decrease. The two polynomials of each leaf have different degrees, so again case $\mathbf{m} < \mathbf{n}$ applies. However, the center branch needs a small adjustment such that all head coefficients are positive, an adjustment which is justified by Lemma 7(i). Here, due to the limited space, we only show the center branch:

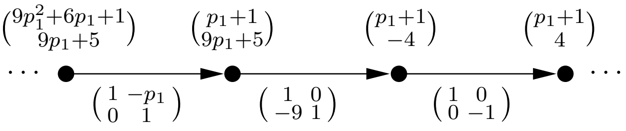


Now all polynomials are reduced to degree 0 and a final application of the Extended Euclidean Algorithm completes the subtree:



. . .
$\binom{1}{4}$
$\begin{pmatrix}1&0\\-4&1\end{pmatrix}$
$\binom{1}{0}$
. . .
$\binom{2}{4}$
$\begin{pmatrix}1&0\\-2&1\end{pmatrix}$
$\binom{2}{0}$
. . .
$\binom{3}{4}$
$\begin{pmatrix}-1&1\\4&-3\end{pmatrix}$
$\binom{1}{0}$
. . .
$\binom{4}{4}$
$\begin{pmatrix}0&1\\1&-1\end{pmatrix}$
$\binom{4}{0}$

We do not draw the two missing continuations of the whole graph (and leave this task as an exercise to the reader). Instead, we want to present a condensed version of the complete graph (see Figure 4.1) that we get as follows. Let us concentrate on the path marked by the bold edges in the preceding figures. Consider only the portion of the graph containing the first three consecutive transformations

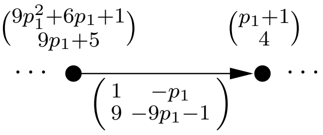


$\binom{9p_1^2+6p_1+1}{9p_1+5}$
$\binom{p_1+1}{9p_1+5}$
$\binom{p_1+1}{-4}$
$\binom{p_1+1}{4}$
. . .
. . .
$\begin{pmatrix}1&-p_1\\0&1\end{pmatrix}$
$\begin{pmatrix}1&0\\-9&1\end{pmatrix}$
$\begin{pmatrix}1&0\\0&-1\end{pmatrix}$

Since

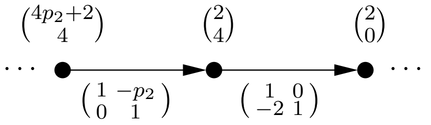
$$\begin{pmatrix}1&0\\0&-1\end{pmatrix}\begin{pmatrix}1&0\\-9&1\end{pmatrix}\begin{pmatrix}1&-p_1\\0&1\end{pmatrix}=\begin{pmatrix}1&-p_1\\9&-9p_1-1\end{pmatrix}$$

we can simply draw



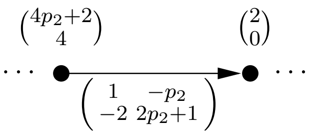
$\binom{9p_1^2+6p_1+1}{9p_1+5}$
$\binom{p_1+1}{4}$
. . .
. . .
$\begin{pmatrix}1&-p_1\\9&-9p_1-1\end{pmatrix}$

In the same way we can replace



$\binom{4p_2+2}{4}$
$\binom{2}{4}$
$\binom{2}{0}$
. . .
. . .
$\begin{pmatrix}1&-p_2\\0&1\end{pmatrix}$
$\begin{pmatrix}1&0\\-2&1\end{pmatrix}$

by



such that the whole bold path now reads as

$$\begin{tikzcd} \binom{p_0^2}{3p_0+2} \arrow[r, "p_0=3p_1+1"'] & \binom{9p_1^2+6p_1+1}{9p_1+5} \arrow[r, "\begin{pmatrix}1 & -p_1\\ 9 & -9p_1-1\end{pmatrix}"'] & \binom{p_1+1}{4} \arrow[r, "p_1=4p_2+1"'] & \binom{4p_2+2}{4} \arrow[r, "\begin{pmatrix}1 & -p_2\\ -2 & 2p_2+1\end{pmatrix}"'] & \binom{2}{0} \end{tikzcd}$$

Since any valid equation $\mathbf{U}(X)\mathbf{v}(X) = \mathbf{w}(X)$ remains valid if we substitute X by $lY+k$, we can substitute any occurrence of p_1 by $4p_2 + 1$ and delete the now unnecessary arrow labeled by $p_1 = 4p_2 + 1$. Thus, we get

$$\begin{tikzcd} \binom{p_0^2}{3p_0+2} \arrow[r, "p_0=12p_2+4"'] & \binom{144p_2^2+96p_2+16}{36p_2+14} \arrow[r, "\begin{pmatrix}1 & -4p_2-1\\ 9 & -36p_2-10\end{pmatrix}"'] & \binom{4p_2+2}{4} \arrow[r, "\begin{pmatrix}1 & -p_2\\ -2 & 2p_2+1\end{pmatrix}"'] & \binom{2}{0} \end{tikzcd}$$

or, simplified as above,

$$\begin{tikzcd} \binom{p_0^2}{3p_0+2} \arrow[r, "p_0=12p_2+4"'] & \binom{144p_2^2+96p_2+16}{36p_2+14} \arrow[r, "\begin{pmatrix}-9p_2+1 & 36p_2^2+6p_2-1\\ 18p_2+7 & -72p_2^2-48p_2-8\end{pmatrix}"'] & \binom{2}{0} \end{tikzcd}$$

This can be read as follows. Whenever $p_0 \in \mathbb{Z}$ and $p_0 = 12p_2 + 4$ (i.e., $p_0 \equiv_{12} 4$ and $p_2 = \lfloor \frac{p_0}{12} \rfloor$) then

$$\begin{pmatrix} -9\left\lfloor\frac{p_0}{12}\right\rfloor + 1 & 36\left\lfloor\frac{p_0}{12}\right\rfloor^2 + 6\left\lfloor\frac{p_0}{12}\right\rfloor - 1 \\ 18\left\lfloor\frac{p_0}{12}\right\rfloor + 7 & -72\left\lfloor\frac{p_0}{12}\right\rfloor^2 - 48\left\lfloor\frac{p_0}{12}\right\rfloor - 8 \end{pmatrix} \begin{pmatrix} p_0^2 \\ 3p_0 + 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix}.$$

Thus, if $p_0 \equiv_{12} 4$ the gcd of p_0^2 and $3p_0 + 2$ is always 2. Moreover, we have found a unimodular matrix that encapsulates (in dependence of p_0) all the steps which lead to this gcd.

The Algorithm Let us turn our attention now to the general data structure which will be used to represent aiq-polynomials as well as the result of algorithm *aiqGcd*$_2$ we are about to develop.

$$\begin{aligned} \textbf{data}\ \mathit{LTree}\ \alpha = {} & \mathit{LNode}\ \alpha\ [\mathit{LTree}\ \alpha] \\ \mid\ & \mathit{TNode}\ \alpha\ (\mathit{LTree}\ \alpha) \\ \mid\ & \mathit{EmptyLNode}\ [\mathit{LTree}\ \alpha] \\ \mid\ & \mathit{Leaf}\ \alpha \end{aligned}$$

As in the example above, an *LNode* will represent a specialisation (or l-extension) while a *TNode* will stand for a transformation. In both cases, these inner nodes can carry intermediate results, in contrast to *EmptyLNode*. For a given *LNode x ys*, *TNode x y* or *Leaf x* we will call x the *content* of the respective node.

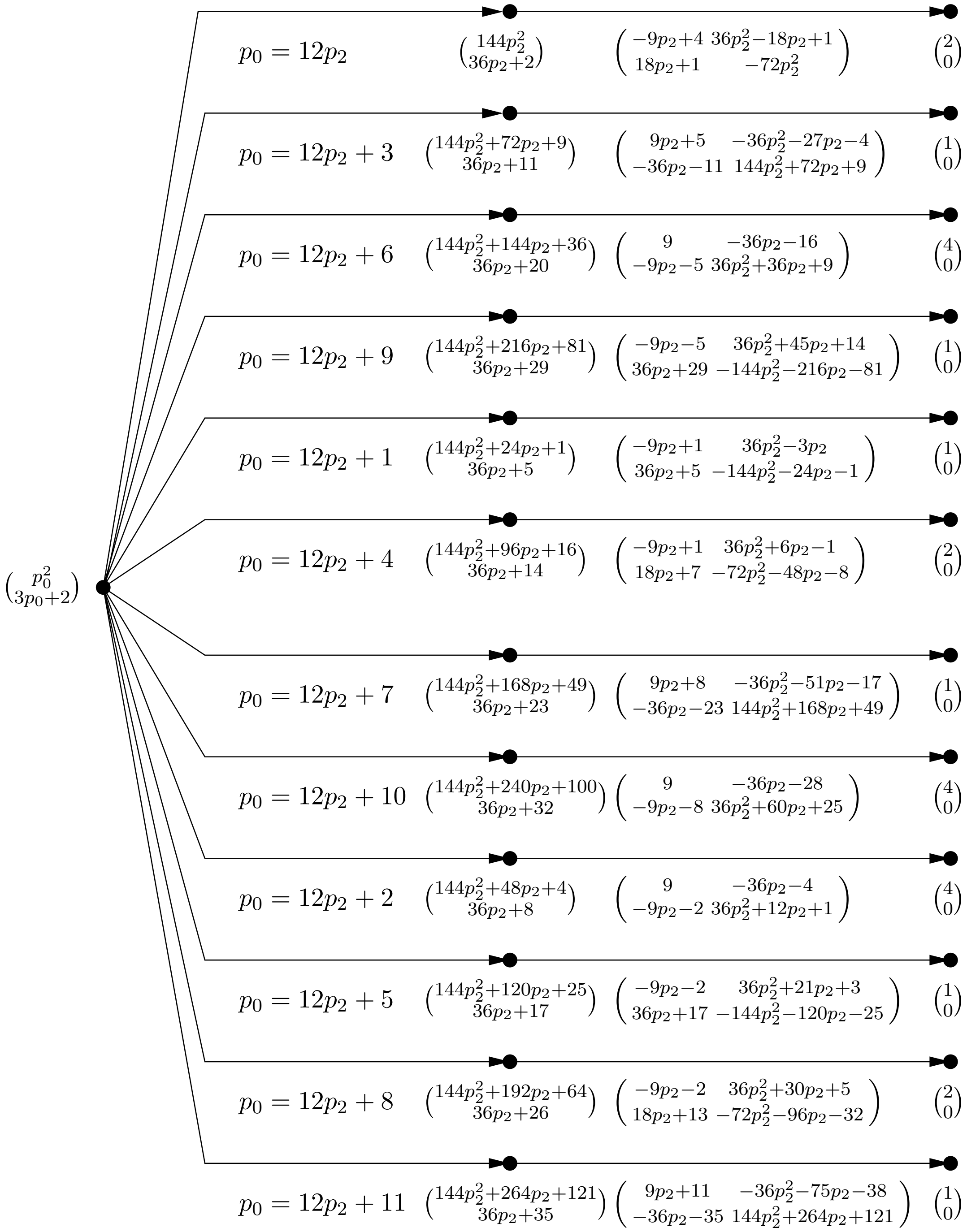

$\binom{p_0^2}{3p_0+2}$
$p_0 = 12p_2$ $\binom{144p_2^2}{36p_2+2}$ $\begin{pmatrix} -9p_2+4 & 36p_2^2-18p_2+1 \\ 18p_2+1 & -72p_2^2 \end{pmatrix}$ $\binom{2}{0}$
$p_0 = 12p_2 + 3$ $\binom{144p_2^2+72p_2+9}{36p_2+11}$ $\begin{pmatrix} 9p_2+5 & -36p_2^2-27p_2-4 \\ -36p_2-11 & 144p_2^2+72p_2+9 \end{pmatrix}$ $\binom{1}{0}$
$p_0 = 12p_2 + 6$ $\binom{144p_2^2+144p_2+36}{36p_2+20}$ $\begin{pmatrix} 9 & -36p_2-16 \\ -9p_2-5 & 36p_2^2+36p_2+9 \end{pmatrix}$ $\binom{4}{0}$
$p_0 = 12p_2 + 9$ $\binom{144p_2^2+216p_2+81}{36p_2+29}$ $\begin{pmatrix} -9p_2-5 & 36p_2^2+45p_2+14 \\ 36p_2+29 & -144p_2^2-216p_2-81 \end{pmatrix}$ $\binom{1}{0}$
$p_0 = 12p_2 + 1$ $\binom{144p_2^2+24p_2+1}{36p_2+5}$ $\begin{pmatrix} -9p_2+1 & 36p_2^2-3p_2 \\ 36p_2+5 & -144p_2^2-24p_2-1 \end{pmatrix}$ $\binom{1}{0}$
$p_0 = 12p_2 + 4$ $\binom{144p_2^2+96p_2+16}{36p_2+14}$ $\begin{pmatrix} -9p_2+1 & 36p_2^2+6p_2-1 \\ 18p_2+7 & -72p_2^2-48p_2-8 \end{pmatrix}$ $\binom{2}{0}$
$p_0 = 12p_2 + 7$ $\binom{144p_2^2+168p_2+49}{36p_2+23}$ $\begin{pmatrix} 9p_2+8 & -36p_2^2-51p_2-17 \\ -36p_2-23 & 144p_2^2+168p_2+49 \end{pmatrix}$ $\binom{1}{0}$
$p_0 = 12p_2 + 10$ $\binom{144p_2^2+240p_2+100}{36p_2+32}$ $\begin{pmatrix} 9 & -36p_2-28 \\ -9p_2-8 & 36p_2^2+60p_2+25 \end{pmatrix}$ $\binom{4}{0}$
$p_0 = 12p_2 + 2$ $\binom{144p_2^2+48p_2+4}{36p_2+8}$ $\begin{pmatrix} 9 & -36p_2-4 \\ -9p_2-2 & 36p_2^2+12p_2+1 \end{pmatrix}$ $\binom{4}{0}$
$p_0 = 12p_2 + 5$ $\binom{144p_2^2+120p_2+25}{36p_2+17}$ $\begin{pmatrix} -9p_2-2 & 36p_2^2+21p_2+3 \\ 36p_2+17 & -144p_2^2-120p_2-25 \end{pmatrix}$ $\binom{1}{0}$
$p_0 = 12p_2 + 8$ $\binom{144p_2^2+192p_2+64}{36p_2+26}$ $\begin{pmatrix} -9p_2-2 & 36p_2^2+30p_2+5 \\ 18p_2+13 & -72p_2^2-96p_2-32 \end{pmatrix}$ $\binom{2}{0}$
$p_0 = 12p_2 + 11$ $\binom{144p_2^2+264p_2+121}{36p_2+35}$ $\begin{pmatrix} 9p_2+11 & -36p_2^2-75p_2-38 \\ -36p_2-35 & 144p_2^2+264p_2+121 \end{pmatrix}$ $\binom{1}{0}$

Figure 4.1: All branches of the condensed aiq-graph from Example 63

Algorithm 6

```
nmap                             :: $(\alpha \to \beta) \to$ LTree $\alpha$ $\to$ LTree $\beta$
nmap fn (Leaf x)                 = Leaf (fn x)
nmap fn (LNode x xs)             = LNode (fn x) (map (nmap fn) xs)
nmap fn (TNode x y)              = TNode (fn x) (nmap fn y)
nmap fn (EmptyLNode xs)          = EmptyLNode (map (nmap fn) xs)
```

Algorithm 7

```
lmap                             :: $(\alpha \to$ LTree $\alpha) \to$ LTree $\alpha$ $\to$ LTree $\alpha$
lmap fn (Leaf x)                 = fn x
lmap fn (LNode x xs)             = LNode x (map (lmap fn) xs)
lmap fn (TNode x y)              = TNode x (lmap fn y)
lmap fn (EmptyLNode xs)          = EmptyLNode (map (lmap fn) xs)
```

Algorithms 6 and 7 introduce two useful functions that operate on *LTree*s: *nmap* which applies a given function to all nodes within some *LTree* and *lmap* that applies a given function to leaves only, thereby extending them.

We use the set

$$P = \{p, p_0, p_1, \dots\}$$

as a pool of parameters used for specialisation. To represent the modulo case distinctions we use linear equations of the form

$$p \approx lp_i + k$$

where $l \in \mathbb{N}_{\geq 1}$, $0 \leq k < l$ and $i \in \mathbb{N}$. To distinguish the "="-relation within equations from other occurrences within the algorithms, we use "$\approx$". Note that we always use p on the left hand side of the equations, i.e., we will always use the condensed information on the modulo cases. To use these equations as syntactical objects in our Haskell pseudocode, we provide the type *ModEqs*. It will be convenient to use $\varphi(x)$ as an abbreviation for the equation $p \approx lx + k$. The argument of φ refers to the right-hand parameter and hence allows to denote specialization in an easy manner.

Another special type is

$$\textbf{type}\,\mathbb{Z}_u[P] \ = \ \mathbb{Z}[p] \cup \textstyle\bigcup_{i\in\mathbb{N}} \mathbb{Z}[p_i]$$

the set of all univariate polynomials with a parameter from P. Algorithm *aiqGcd*$_2$ uses the two auxiliary functions provided by Algorithm 8 and Algorithm 9.

Lemma 64 Let $f, g \in \mathbb{Z}[X]$. Then Algorithm 8 terminates. Its result is of the form $([f', g'], \mathbf{U})$ with $f', g' \in \mathbb{Z}[X]$ such that neither $f' \downarrow_d g'$ nor $g' \downarrow_d f'$. Moreover, $\mathbf{U} \in \mathbb{Z}[X]^{2\times 2}$ is unimodular and for all $p \in \mathbb{Z}$

$$\begin{pmatrix} f'(p) \\ g'(p) \end{pmatrix} = \mathbf{U}(p) \begin{pmatrix} f(p) \\ g(p) \end{pmatrix}.$$

Algorithm 8

```
$\mathit{dreduce} \quad :: [\mathbb{Z}[X]] \to ([\mathbb{Z}[X]],\ \mathit{Mat}_{2\times 2}(\mathbb{Z}[X]))$
$\mathit{dreduce}\ [f,0] = ([f,0],\ \mathbf{I}_2)$
$\mathit{dreduce}\ [0,f] = ([f,0],\ \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right))$

$\mathit{dreduce}\ [f,g] =$ if $(f \downarrow_d g \vee g \downarrow_d f)$ then
        $([f',g'],\ \mathbf{WV})$
    else
        $([f,g],\ I_2)$
  where
    $\mathbf{V} = \begin{cases} \begin{pmatrix} 0 & 1 \\ 1 & -\frac{\mathrm{HC}\,(f)}{\mathrm{HC}\,(g)} X^{\deg(f)\,-\,\deg(g)} \end{pmatrix} & \textbf{if } \deg(f) > \deg(g) \wedge \mathrm{HC}\,(g) \mid \mathrm{HC}\,(f) \\ \mathit{snd}\,\$\,\mathit{extGcd}_2\ [\mathrm{HC}\,(f), \mathrm{HC}\,(g)] & \textbf{if } \deg(f) = \deg(g) \\ \begin{pmatrix} 1 & 0 \\ -\frac{\mathrm{HC}\,(g)}{\mathrm{HC}\,(f)} X^{\deg(g)\,-\,\deg(f)} & 1 \end{pmatrix} & \textbf{if } \deg\,(f) < \deg(g) \wedge \mathrm{HC}\,(f) \mid \mathrm{HC}\,(g) \end{cases}$
    $([f',g'],\ \mathbf{W}) = \mathit{dreduce}\ (\mathbf{V} \cdot [f,g])$
```

Proof. Follows by induction on $\deg(f) + \deg(g)$ together with Lemmas 61 and 62. □

Algorithm 9

```
$\mathit{specialize} :: (\mathit{ModEqs},\ \mathit{Vec}_2(\mathbb{Z}_u[P]),\ \mathit{Mat}_{2\times\,2}(\mathbb{Z}_u[P]))$
        $\to [(\mathit{ModEqs},\ \mathit{Vec}_2(\mathbb{Z}_u[P]),\ \mathit{Mat}_{2\times\,2}(\mathbb{Z}_u[P]))]$
$\mathit{specialize}\ (\varphi(p_i),\ \binom{f(p_i)}{g(p_i)},\ \mathbf{U}(p_i))$
  $= \mathit{map}$
    $(\lambda\, x \to (\varphi(l' p_{i+1} + x),\ \binom{f(l' p_{i+1}+x)}{g(l' p_{i+1}+x)},\ \mathbf{U}(l' p_{i+1} + x)))$
    $[0..l' - 1]$
  where
    $l' =$ if $\deg(f) < \deg(g)$ then
            $|\mathrm{HC}\,(f)|$
          else
            $|\mathrm{HC}\,(g)|$
```

Theorem 65 *Let $f,g \in \mathbb{Z}[X]$. Then $\mathit{aiqGcd}_2\ (p \approx p_0,\ \binom{f(p_0)}{g(p_0)},\ \mathbf{I}_2)$ terminates, i.e., its result, say T, is a finite LTree. Let $p \in \mathbb{Z}$. Then there is exactly one leaf of the form Leaf $(p \approx lp_i + k,\ \binom{f'(p_i)}{0},\ \mathbf{U}(p_i))$ in T such that $p = lp_i + k$ for some $p_i \in \mathbb{Z}$. Moreover,*

$$\gcd(f(p), g(p)) \sim f'(p_i)$$

and

$$\begin{pmatrix} f'(p_i) \\ 0 \end{pmatrix} = \mathbf{U}(p_i) \begin{pmatrix} f(p) \\ g(p) \end{pmatrix}$$

where p_i is determined by the equation $p = lp_i + k$, i.e., $p_i = \lfloor \frac{p}{l} \rfloor$. The matrix $\mathbf{U}$ *is unimodular.*

Proof. Again, the proof is an induction on $\deg(f) + \deg(g)$ and follows from Lemma 64. □

Algorithm 10

```
$aiqGcd_2 :: (ModEqs,\ Vec_2(\mathbb{Z}_u[P]),\ Mat_{2\times 2}(\mathbb{Z}_u[P]))$
        $\rightarrow\ LTree\ (ModEqs,\ Vec_2([\mathbb{Z}_u[P]),\ Mat_{2\times 2}(\mathbb{Z}_u[P]))$
$aiqGcd_2\ (\varphi,\ [f, g],\ \mathbf{U})$
  $|\ g\ =\ 0$               $= Leaf\ (\varphi,\ [f, g],\ \mathbf{U})$
  $|\ f\ =\ 0$               $= Leaf\ (\varphi,\ [g, f],\ (\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix})\ \mathbf{U})$
  $|\ (f \downarrow_d\ g \vee\ g \downarrow_d\ f)$ $= TNode\ (\varphi,\ [f, g],\ \mathbf{U})\ (aiqGcd_2\ (\varphi,\ [f', g'],\ \mathbf{VU}))$
        where
          $([f', g'],\ \mathbf{V})\ =\ dreduce\ [f, g]$
  $|\ otherwise$             $= LNode\ (\varphi,\ [f, g],\ \mathbf{U})\ (map\ aiqGcd_2\ (specialize\ (\varphi,\ \binom{f}{g},\ \mathbf{U})))$
```

As in Section 2.2, we are going to extend Algorithm 10 for an arbitrary finite number of polynomials $f_1, \ldots, f_n \in \mathbb{Z}[X]$, $n > 2$. Assume, for instance, that we want to compute the aiq-gcd of (f_1, f_2, f_3) First, the aiq-gcd is computed only for f_2 and f_3, resulting in leaves of the form $(p \approx lp_i + k,\ [f'(p_i), 0],\ \mathbf{U}')$. Then each leaf is "extended" by f_1 to look like $(p \approx lp_i + k,\ [f_1(lp_i + k),\ f'(p_i),\ 0],\ \mathbf{U}_{ext})$. Finally, the aiq-computation is continued with the *two* polynomials f_1 and f'. The algorithm which does all this is given by Algorithm 12. It uses Algorithm 11, a version of Algorithm 10 working on vectors of length ≥ 2.

There is one last step of generalization we can undertake, namely, to compute the aiq-gcd not just of polynomials from $\mathbb{Z}[X]$ but directly from $\mathcal{AIQ}$. So, let $f_1, \ldots, f_n \in \mathcal{AIQ}$ and let k be a common modulus, such that

$$f_i(p) = \begin{cases} f_{i0}(p') & \text{if } p = kp' \\ f_{i1}(p') & \text{if } p = kp' + 1 \\ \ldots & \\ f_{i(k-1)}(p') & \text{if } p = kp' + (k-1). \end{cases}$$

Therefore

$$(f_1(p), \ldots, f_n(p)) = \begin{cases} (f_{10}(p'), \ldots, f_{n0}(p')) & \text{if } p = kp' \\ (f_{11}(p'), \ldots, f_{n1}(p')) & \text{if } p = kp' + 1 \\ \ldots & \\ (f_{1(k-1)}(p'), \ldots, f_{n(k-1)}(p')) & \text{if } p = kp' + (k-1) \end{cases}$$

and we can *lmap* Algorithm 12 to the *initialDataStructure* given below.

```
$initialDataStructure\ ::\ LTree\ (ModEqs,\ Vec_n(\mathbb{Z}_u[P]),\ Mat_{n\times n}(\mathbb{Z}_u[P]))$
$initialDataStructure$
```

Algorithm 11

```
$aiqGcd_n$ ::($ModEqs$, $Vec_n(\mathbb{Z}_u[P])$, $Mat_{n\times n}(\mathbb{Z}_u[P])$)
        $\rightarrow$ $LTree$ ($ModEqs$, $Vec_n(\mathbb{Z}_u[P])$, $Mat_{n\times n}(\mathbb{Z}_u[P])$)
$aiqGcd_n$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$)
| $n$ == 1              = $Leaf$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$)
| $f_2$ == 0             = $Leaf$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$)
| $f_1$ == 0             = $Leaf$ ($\varphi$, $\mathbf{f}'$, $\mathbf{U}'$)

      where
        $\mathbf{f}' = \begin{pmatrix} 0 & 1 & \dots \\ 1 & 0 & \\ \vdots & & I_{(n-2)} \end{pmatrix} \mathbf{f}$
        $\mathbf{U}' = \begin{pmatrix} 0 & 1 & \\ 1 & 0 & \\ & & I_{(n-2)} \end{pmatrix} \mathbf{U}$
 | ($f_1 \downarrow_d f_2 \vee f_2 \downarrow_d f_1$) = $TNode$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$) ($aiqGcd_n$ ($\varphi$, $\mathbf{f}'$, $\mathbf{VU}$))
      where
        $\mathbf{f}'$          =$(f_1', f_2', f_3, \dots, f_n)^t$
        $\mathbf{V}$          $= \begin{pmatrix} \mathbf{V}' & 0 \\ 0 & I_{(n-2)} \end{pmatrix}$
        $\left(\binom{f_1'}{f_2'}, \mathbf{V}'\right)$ =$dreduce$ $\binom{f_1}{f_2}$
| $otherwise$             = $LNode$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$) ($map$ $aiqGcd_n$ ($specialize$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$)))
```

Algorithm 12

```
$aiqGcd$ ::($ModEqs$, $Vec_n(\mathbb{Z}_u[P])$, $Mat_{n\times n}(\mathbb{Z}_u[P])$))
       $\rightarrow$ $LTree$ ($ModEqs$, $Vec_n(\mathbb{Z}_u[P])$, $Mat_{n\times n}(\mathbb{Z}_u[P])$))
$aiqGcd$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$)
| $n$ == 1                                        = $Leaf$ ($\varphi$, $\mathbf{f}$, $\mathbf{U}$)
| $n \geq 2$                                       = $lmap$ $aiqGcd_n$ $T_1$
  where
    $T_1$      = $nmap$ ($extendBy$ $f_1$) $T_2$
    $T_2$      = $aiqGcd$ ($\varphi$, $\mathbf{f}_{\geq 2}$, $\mathbf{U}_{\geq(2,2)}$)
    $extendBy$ $v(p)$ ($p \approx kp' + l$, $\mathbf{v}(p')$, $\mathbf{M}(p')$)
             = ($p \approx kp' + l$, $\binom{v(kp'+l)}{\mathbf{v}(p')}$, $\begin{pmatrix} 1 & 0 \\ 0 & \mathbf{M}(p') \end{pmatrix}$ ))
```

$$\begin{aligned}
= \mathit{EmptyLNode}\,[\, & \mathit{Leaf}\ (p \approx kp',\ (f_{10}(p'),\dots,f_{n0}(p')),\ \mathbf{I}_n),\\
& \mathit{Leaf}\ (p \approx kp'+1,\ (f_{11}(p'),\dots,f_{n1}(p')),\ \mathbf{I}_n),\\
& \dots,\\
& \mathit{Leaf}\ (p \approx kp'+(k-1),\ (f_{1(k-1)}(p'),\dots,f_{n(k-1)}(p')),\ \mathbf{I}_n)\,]
\end{aligned}$$

4.1.5 Aiq-Echelon Reduction

By now, it may be already clear to the reader that aiq-echelon reduction for parametric matrices $\mathbf{U} \in \mathbb{Z}[p]^{m\times n}$ might be possible in a similar way as it was for pure integral matrices in Section 2.2, i.e., by repeated application of Algorithm 12 to smaller and smaller matrices. There are, however, some technical details we have to be aware of and the purpose of the current section is to provide a Haskell oriented approach to them.

Definition 66 (Weak Echelon Form) Let $\mathbf{A} = (a_{ij}) \in \mathbb{Z}[X]^{m\times n}$. Then $\mathbf{A}$ is in *weak echelon form* if

(1) There is some $r \in \{0,\dots,m\}$ such that

$$i > r \Rightarrow a_{ij} = 0$$

for all $j \in \{1,\dots,n\}$;

(2) For all $1 \le i \le r$ the set $M_i := \{j \mid a_{ij} \neq 0\}$ is not empty;

(3) $\rho_1 < \rho_2 < \ldots < \rho_r$, where $\rho_i := \min M_i$.

The reason for the definition of a *weak* echelon form is illustrated by the following example.

Example 67 Consider the matrix

$$\begin{pmatrix} X-2 & X-1 & 1 \\ 0 & X^2-1 & 2 \\ 0 & 0 & 3 \end{pmatrix}.$$

By the above definition, the matrix is echelon. However, for $X = 2$ the matrix becomes

$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & -1 & 2 \\ 0 & 0 & 3 \end{pmatrix}$$

which is not echelon because $l_1 = l_2$. For $X = -1$ we get

$$\begin{pmatrix} -3 & -2 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 3 \end{pmatrix}$$

again not echelon since $l_2 = l_3$. The same would be true in case $X = 1$.

Definition 68 (Strong Echelon Form) Let $\mathbf{A} = (a_{ij}) \in \mathbb{Z}[X]^{m\times n}$ be weak echelon. Then we say that $\mathbf{A}$ is in *strong* echelon form if $\mathbf{A}(p) \in \mathbb{Z}^{m\times n}$ is echelon for all $p \in \mathbb{Z}$.

The “repeated application of Algorithm 12” which we are going to describe will guarantee that all matrices contained within the leaves of the resulting *LTree* are in weak echelon form. The implications of a matrix not being strong echelon will be discussed later, when we describe the solutions of parametric equation systems (Section 4.1.6).

Algorithm 13

```
$aiqEchelon :: (ModEqs,\ Mat_{m\times n}(\mathbb{Z}_u[P]),\ Mat_{m\times m}(\mathbb{Z}_u[P]))$
                 $\rightarrow\ LTree\ (ModEqs,\ Mat_{m\times n}(\mathbb{Z}_u[P]),\ Mat_{m\times m}(\mathbb{Z}_u[P]))$
$aiqEchelon\quad (p \approx\ kp' + l,\ \mathbf{A}(p'),\ \mathbf{U}(p'))$
               $= T_3$
   where
      $T_1\ = aiqGcd\ (p' \approx\ p',\ \mathbf{a}_1(p'),\ \mathbf{I}_m)$
      $T_2\ = lift\ \mathbf{A}(p')\ T_1$
      $T_3\ = lmap\ f\ T_2$
      $f\ x\ = nmap\ (extend\ x)\ ((aiqEchelon \circ\ extract)\ x)$
```

Theorem 69 *Let* $\mathbf{A} \in \mathbb{Z}[p]^{m\times n}$ *and let* $T = aiqEchelon\,(p_0 \approx p_0,\ \mathbf{A}(p_0),\ \mathbf{I}_m)$. *Then* T *is a finite LTree such that for any* $p \in \mathbb{Z}$ *there is exactly one leaf of the form* $L = Leaf\,(p_0 \approx kp_i + l,\ \mathbf{S}(p_i),\ \mathbf{U}(p_i))$ *with* $\mathbf{S} \in \mathbb{Z}_u[P]^{m\times n}$, $\mathbf{U} \in \mathbb{Z}_u[P]^{m\times m}$ *and* $0 \le l < k$ *such that the following statements hold.*

(1) $p \equiv_k l$,

(2) $\mathbf{S}$ *is in weak echelon form,*

(3) $\mathbf{U}$ *is unimodular,*

(4) $\mathbf{S}(p_i) = \mathbf{U}(p_i)\mathbf{A}(p_i)$ *for all* $p_i \in \mathbb{Z}$.

Proof. Let $\mathbf{A} \in \mathbb{Z}[X]^{m\times n}$ and let $\mathbf{a}_1$ denote the first column of $\mathbf{A}$. Let $T_1 = aiqGcd\,(p \approx p,\ \mathbf{a}_1(p),\ \mathbf{I}_m)$ denote the *LTree* obtained by applying Algorithm 12 to $\mathbf{a}_1$. From there, we know that for any content $(p \approx kp' + l,\ \mathbf{a}_1'(p'),\ \mathbf{U}'(p'))$ of T_1 the equation

$$\mathbf{a}_1'(p') = \mathbf{U}'(p')\mathbf{a}_1(kp' + l)$$

applies. Therefore, we can lift the actions of $\mathbf{U}'$ on $\mathbf{a}_1$ to the whole matrix $\mathbf{A}$ by the function *lift*.

```
$lift\qquad :: Matrix_{m\times\ n}(\mathbb{Z}_u[P])$
              $\rightarrow\ LTree\ (ModEqs,\ Vec_m(\mathbb{Z}_u[P]),\ Mat_{m\times\ m}(\mathbb{Z}_u[P])\ )$
              $\rightarrow\ LTree\ (ModEqs,\ Mat_{m\times\ n}(\mathbb{Z}_u[P]),\ Mat_{m\times\ m}(\mathbb{Z}_u[P])\ )$
$lift\ \mathbf{A}(p) = lmap\ (\lambda\ (p\ \approx\ kp_i + l,\ \mathbf{x}(p_i),\ \mathbf{U}(p_i)) \rightarrow (p\ \approx\ kp_i + l,\ \mathbf{A}(kp_i + l),\ \mathbf{U}(p_i))$
```

Thus, for any column $\mathbf{a}_i$ of $\mathbf{A}$ and the corresponding column $\mathbf{a}'_i$ of $\mathbf{A}'$ we have

$$\mathbf{a}'_i(p') = \mathbf{U}'(p')\mathbf{a}_i(kp' + l)$$

or, viewing all columns simultaneously,

$$\mathbf{A}'(p') = \mathbf{U}'(p')\mathbf{A}(kp' + l).$$

Now let $T_2 \;=\; \mathit{lift}\,\mathbf{A}(p)\;T_1$, let

$$L := \mathit{Leaf}(p \approx kp' + l, \mathbf{A}'(p'), \mathbf{U}'(p'))$$

be a leaf of T_2 and let

$$C := (p \approx kp' + l, \mathbf{A}'(p'), \mathbf{U}'(p'))$$

be its content. The first row of $\mathbf{A}'$ is already in the desired form $\binom{a'_{11}}{\mathbf{0}}$, so we want to reduce $\mathbf{A}_s := \mathbf{A}'_{(i,j)\geq(2,2)}$. This step is accomplished by the function *extract*.

$$\begin{aligned}
&\mathit{extract} :: \;(\mathit{ModEqs},\; \mathit{Mat}_{m\times\, n}(\mathbb{Z}_u[P]),\; \mathit{Mat}_{m\times\, m}(\mathbb{Z}_u[P]))\\
&\qquad\qquad \rightarrow\; (\mathit{ModEqs},\; \mathit{Mat}_{(m-1)\times\,(n-1)}(\mathbb{Z}_u[P]),\; \mathit{Mat}_{(m-1)\times\,(m-1)}(\mathbb{Z}_u[P]))\\
&\mathit{extract}(p_i \approx\; kp_j + l,\; \mathbf{A}(p_j),\; \mathbf{U}(p_j)) \;=\; (p_j \approx\; p_j,\; \mathbf{A}(p_j)_{\geq\,(2,2)},\; \mathbf{I}_{(m-1)\times(m-1)}).
\end{aligned}$$

By applying the induction hypothesis, consider

$$T_s =\; \mathit{aiqEchelon}\;(\mathit{extract}\;C) =\; \mathit{aiqEchelon}\;(p' \approx\; p',\; \mathbf{A}_s(p'),\; \mathbf{I}_{(m-1)}).$$

If $(p' \approx k'p'' + l', \mathbf{A}''(p''), \mathbf{U}''(p''))$ is the content of any node N of T_s then

$$\mathbf{A}''(p'') = \mathbf{U}''(p'')\mathbf{A}_s(k'p'' + l').$$

If N is a leaf then $\mathbf{A}''$ is weak echelon. Now we have to combine T_s and L. First, we extend each matrix $\mathbf{A}''(p'')$ in L_s by the upper row and the zeroes we deleted in order to get $\mathbf{A}_s$. I.e., we replace each $\mathbf{A}''(p'')$ in T_s by

$$\hat{\mathbf{A}}''(p'') := \begin{pmatrix} a'_{11}(k'p'' + l') & a'_{12}(k'p'' + l') \;\;\ldots\;\; a'_{1n}(k'p'' + l') \\ \mathbf{0} & \mathbf{A}''(p'') \end{pmatrix}.$$

Moreover, we replace each $\mathbf{U}''(p'')$ by

$$\hat{\mathbf{U}}''(p'') := \begin{pmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{U}''(p'') \end{pmatrix},$$

such that

$$\hat{\mathbf{A}}''(p'') = \hat{\mathbf{U}}''(p'')\mathbf{A}'(k'p'' + l').$$

This is done by *nmap*ping the function *extend* to T_s.

Second, we replace each equation

$$p' \approx k'p'' + l'$$

$$\begin{aligned}
&\mathit{extend} :: (\mathit{ModEqs},\ \mathit{Mat}_{m\times n}(\mathbb{Z}_u[P]),\ \mathit{Mat}_{m\times m}(\mathbb{Z}_u[P]))\\
&\qquad\qquad \to (\mathit{ModEqs},\ \mathit{Mat}_{(m-1)\times(n-1)}(\mathbb{Z}_u[P]),\ \mathit{Mat}_{(m-1)\times(m-1)}(\mathbb{Z}_u[P]))\\
&\qquad\qquad \to (\mathit{ModEqs},\ \mathit{Mat}_{m\times n}(\mathbb{Z}_u[P]),\ \mathit{Mat}_{m\times m}(\mathbb{Z}_u[P]))\\
&\mathit{extend}\quad (p \approx\ qp'+r,\ \mathbf{A}(p'),\ \mathbf{U}(p'))\ (p' \approx\ kp''+l,\ \mathbf{A}'(p''),\ \mathbf{U}')\\
&\qquad =(p \approx\ (qk)p''+(ql+r),\ \widehat{\mathbf{A}}'(p''),\ \widehat{\mathbf{U}}'(p'')\mathbf{U}(kp''+l))\\
&\quad \textbf{where}\\
&\quad \widehat{\mathbf{A}}'(p'') = \begin{pmatrix} a'_{11}\,(kp''+l) & a'_{12}\,(kp''+l)\ \ldots\ a'_{1n}\,(kp''+l)\\ 0 & \mathbf{A}'(p'') \end{pmatrix}\\
&\quad \widehat{\mathbf{U}}'(p'') = \begin{pmatrix} 1 & 0\\ 0 & \mathbf{U}'(p'') \end{pmatrix}
\end{aligned}$$

by

$$p \approx k(k'p''+l')+l$$

and each $\hat{\mathbf{U}}''(p'')$ by $\hat{\mathbf{U}}''(p'')\mathbf{U}'(k'p''+l')$. Altogether, if N denotes again any node within T_s, then its former content

$$(p' \approx k'p''+l', \mathbf{A}''(p''), \mathbf{U}''(p''))$$

is now replaced by

$$(p \approx k(k'p''+l')+l, \hat{\mathbf{A}}''(p''), \hat{\mathbf{U}}''(p'')\mathbf{U}'(k'p''+l')).$$

Now, N directly relates to the root node of T_2 because

$$\begin{aligned}\hat{\mathbf{A}}''(p'') &= \hat{\mathbf{U}}''(p'')\mathbf{A}'(k'p''+l')\\ &= \hat{\mathbf{U}}''(p'')\left(\mathbf{U}'(p')\mathbf{A}(k(k'p''+l')+l)\right).\end{aligned}$$

If N is a leaf of T_s, then $\hat{\mathbf{A}}''$ is by the very nature of our construction in weak echelon form. □

4.1.6 Solving Parametric Equation Systems

By now, we possess all the tools necessary to describe the solutions of integral equation systems with one parameter. To this end, we initially consider the following special situation. Let $\mathbf{A} \in \mathbb{Z}[p]^{m\times n}$ and $\mathbf{b} \in \mathbb{Z}^n$. Suppose further that there already is some strong echelon matrix $\mathbf{S} \in \mathbb{Z}[p]^{m\times n}$ and some unimodular matrix $\mathbf{U} \in \mathbb{Z}[p]^{m\times m}$ such that

$$\mathbf{UA} = \mathbf{S}$$

for all $p \in \mathbb{Z}$. We aim to describe all solutions of the equation system

$$\mathbf{xA}(p) = \mathbf{b}(p) \tag{4.16}$$

in dependence of p.

Lemma 70 Let $f_i, g_i \in \mathbb{Z}[X]$ for $(1 \leq i \leq n)$. Then

$$\bigcap_{1=1}^{n} D(f_i \mid g_i) = M \cup [a_1]_l \cup \cdots \cup [a_m]_l$$

for some finite set $M \subseteq \mathbb{Z}$ and some $m \in \mathbb{N}$.

Proof. By Corollary 56, for each i there is some finite set M_i and some k_i such that

$$D(f_i \mid g_i) = M_i \cup \bigcup_{j=1}^{k_i} [a_{ij}]_{l_i}.$$

Set $A_{i0} := M_i$ and $A_{ij} := [a_{ij}]_{l_i}$ for $i \geq 1$. Then

$$\begin{aligned}\bigcap_{i=1}^{n} D(f_i \mid g_i) &= (A_{10} \cup A_{11} \cup \cdots \cup A_{1k_1}) \\ &\quad \cap (A_{20} \cup A_{21} \cup \cdots \cup A_{2k_2}) \\ &\quad \cap \ldots \\ &\quad \cap (A_{n0} \cup A_{n1} \cup \cdots \cup A_{nk_n}) \\ &= \bigcup_{\substack{(j_1,\ldots,j_n) \\ \in \prod_{i=1}^{n}\{0,\ldots,k_i\}}} A_{1j_1} \cap \cdots \cap A_{nj_n}.\end{aligned}$$

Each $A_{1j_1} \cap \cdots \cap A_{nj_n}$ is either finite if some $j_i = 0$ or of the form $[a_{1j_1}]_{l_1} \cap \cdots \cap [a_{nj_n}]_{l_n}$. By the Chinese Remainder Theorem (Proposition 10), the latter set is either empty or of the form $[a']_{\mathrm{lcm}(l_1,\ldots,l_n)}$. Therefore, with

$$M := \bigcup_{\text{some } j_i = 0} A_{1j_1} \cap \cdots \cap A_{nj_n} \text{ and } l := \mathrm{lcm}(l_1, \ldots, l_n)$$

the proposed statement follows. □

Theorem 71 *Let $\mathbf{A} \in \mathbb{Z}[p]^{m\times n}$, $\mathbf{b} \in \mathbb{Z}[p]^n$. Let $\mathbf{S} \in \mathbb{Z}[p]^{m\times n}$ be echelon and $\mathbf{U} \in \mathbb{Z}[p]^{m\times m}$ be unimodular such that*

$$\mathbf{U} = \mathbf{AS}.$$

Consider the system of equations

$$\mathbf{xA}(p) = \mathbf{b}(p). \tag{4.17}$$

Then one can construct a finite LTree (ModEqs, $Vec_n(\mathbb{Z}_u[P \cup T])$, _) where each leaf is either of the form

Leaf $(p \approx\ c,\ (c_1, \ldots, c_i, t_{i+1} \ldots, t_m), \_)\ \ (*)$

with $c_j \in \mathbb{Z}$, or of the form

Leaf $(p \approx\ lp' + k\ ,\ (f_1(p'), \ldots, f_i(p'),\ t_{i+1}, \ldots, t_m), \_)\quad (**)$

with $f_j \in \mathbb{Z}[p']$ such that for any $p \in \mathbb{Z}$ system (4.17) has a solution iff

(1) either there is a corresponding leaf $(*)$ and $p = c$. Then the set of solutions is given by the set of vectors

$$\{\mathbf{x} \mid \mathbf{x} = (c_1, \ldots, c_i, t_{i+1} \ldots, t_n)\mathbf{U}(c),\ t_{i+1} \ldots, t_m \in \mathbb{Z}\}$$

(2) or there is a corresponding leaf $(**)$ *with* $p \equiv_l k$*. The set of all solutions is given by*

$$\{\mathbf{x} \mid \mathbf{x} = (f_1(p'), \dots, f_i(p'), t_{i+1}, \dots, t_n)\mathbf{U}(lp' + k),\ t_{i+1} \dots, t_m \in \mathbb{Z}\}$$

where $p' = \lfloor \frac{p}{l} \rfloor$.

Proof. Let $\mathbf{A}, \mathbf{b}, \mathbf{S}$ and $\mathbf{U}$ be as stated. The construction of the proposed *LTree* starts with

$$T_0 := LNode(p \approx p, (t_1, \dots, t_m), E) \tag{4.18}$$

where E is the equation system $(t_1, \dots, t_m)\mathbf{S} = \mathbf{b}$. To begin with, we compute the set

$$\Gamma := \bigcup_{s_{ij} \neq 0} \{p \in \mathbb{Z} \mid s_{ij}(p) = 0\} \cup \bigcup_{b_i \neq 0} \{p \in \mathbb{Z} \mid b_i(p) = 0\}$$

because we want to treat all the cases where one s_{ij} or b_j disappears occasionally separately. The set Γ is finite and for each $c \in \Gamma$ we determine the solutions of $E(c)$ and extend T_0 by a leaf provided that $E(c)$ is solvable.
Consider the numbers ρ_i from Definition 66. Set $\varphi_\Gamma(p) := \bigwedge_{c \in \Gamma} p \not\approx c$. Since $\mathbf{S}$ is echelon, the first $\rho_2 - 1$ equations have the form

$$\begin{aligned} t_1 s_{11}(X) &= b_1(X) \\ t_1 s_{12}(X) &= b_2(X) \\ &\dots \\ t_1 s_{1(\rho_2-1)}(X) &= b_{(\rho_2-1)}(X) \end{aligned} \tag{4.19}$$

(note that if there is no ρ_2 then these equations already represent the whole equation system.) The j-th equation is satisfiable if $D(s_{1j}|b_j) \neq \emptyset$. To satisfy all $\rho_2 - 1$ equations, it is necessary that

(1)
$$\bigcap_{j=1}^{n} D(s_{1j}|b_j) \neq \emptyset$$

and

(2) for any $p \in \cap_{j=1}^{n} D(s_{1j}|b_j)$

$$\frac{b_j(p)}{s_{1j}(p)} = \frac{b_{j'}(p)}{s_{1j'}(p)} \tag{4.20}$$

for all $1 \leq j, j' \leq \rho_2 - 1$.

If $\bigcap_{j=1}^{n} D(s_{1j}|b_j) = \emptyset$, there is no p for which we can determine t_1 besides the solutions found above and so we are done. If $\bigcap_{j=1}^{n} D(s_{1j}|b_j) \neq \emptyset$ then by Lemma 70 we can find some $l \in \mathbb{Z}$, a finite set $M \subseteq \mathbb{Z}$ and $k_1, \dots, k_{i_0} \in \mathbb{Z}$ such that

$$\bigcap_{j=1}^{n} D(s_{1j}|b_j) = M \cup [k_1]_l \cup \dots \cup [k_{i_0}]_l.$$

Now, we have to check for each case if condition (4.20) does hold as follows. For each $c \in M$ we treat the parameter-free equation system $E(c)$ in the same way as we did above for Γ. So, let us concentrate on the $[k_j]_l$. By Corollary 58, we know that $s_{1j}(lX' + k_j) \mid b_j(lX + k_j)$, i.e., $\frac{b_j(lX'+k_j)}{s_{1j}(lX'+k_j)} \in \mathbb{Z}[X]$. Set $f_j(X') := \frac{b_j(lX'+k_j)}{s_{1j}(lX'+k_j)}$ such that (4.19) reads as

$$\begin{aligned} t_1 &= f_1(X') \\ t_1 &= f_2(X') \\ \dots & \\ t_1 &= f_{\rho_2-1}(X'). \end{aligned} \tag{4.21}$$

If $\rho_2 = 2$ then t_1 is fully determined by f_1 and we construct the new node

$$\mathit{LNode}\ (p \approx\ lp' + k_j \wedge \varphi_\Gamma(p),\ (f_1(p'),\ t_2, \dots, t_m),\ E'(p'))$$

where E' results from E by deleting the first row and replacing each row

$$t_1 s_{1j}(X) + t_2 s_{2j}(X) + \cdots = b_j(X)$$

by

$$t_2 s_{2j}(X) + \cdots = b_j(X) - f_1(X) s_{1j}(X).$$

If $\rho_2 > 2$ then we have to determine $N := \bigcap_{j<j'} \{p \mid f_j(p) = f_{j'}(p)\}$ whereas three cases can occur.

1. $N = \mathbb{Z}$, i.e., $f_j = f_{j'}$ for all $j < j'$. Proceed similar as in case $\rho_2 = 2$ but delete the first $\rho_2 - 1$ rows of E in order to get E'.
2. N is finite. For each $c \in N$, construct a leaf

   $$\mathit{Leaf}\ (p \approx\ lp' + k \wedge\ p' \approx\ c \wedge \varphi_\Gamma(p),\ \mathbf{t},\ ())$$

   where $\mathbf{t}$ is the solution of the parameter-free system $E(c)$.
3. $N = \emptyset$. In this case, nothing has to be done.

In this manner we construct new nodes for each k_j which concludes the first step. Each leaf of the LTree constructed up to now is either a *Leaf* or an *LNode*. In the latter case, we can continue the computation of the remaining t_i $(i > 2)$ based on the E' in the same way as is described for t_1. □

4.2 Limitations and Possibilities in the Multi-variate Case

While we have concrete results in the case of a single parameter, the situation changes completely when it comes to more than one parameter. Consider for instance the following equation system with parameters p_1 and p_2:

$$(x, y)\begin{pmatrix} p_1 \\ p_1 \end{pmatrix} = 0. \tag{4.22}$$

Of course, (4.22) has always a solution. However, to describe the set of all solutions we must be able to express the gcd of p_1 and p_2. Let us assume for a moment that we were able to extend our previous results in a straight forward way to more than one parameter, that is, that there are polynomials $f_{00}, f_{01}, \ldots, f_{(k-1)(k-1)} : \mathbb{Z}^2 \longrightarrow \mathbb{Z}$ such that for all $p_1, p_2 \in \mathbb{Z}$

$$gcd(p_1, p_2) \sim \begin{cases} f_{00}(\lfloor \frac{p_1}{k} \rfloor, \lfloor \frac{p_2}{k} \rfloor) & \text{if } p_1, p_2 \equiv_k 0 \\ f_{01}(\lfloor \frac{p_1}{k} \rfloor, \lfloor \frac{p_2}{k} \rfloor) & \text{if } p_1 \equiv_k 0 \wedge p_2 \equiv_k 1 \\ \ldots & \\ f_{(k-1)(k-1)}(\lfloor \frac{p_1}{k} \rfloor, \lfloor \frac{p_2}{k} \rfloor) & \text{if } p_1, p_2 \equiv_k k-1. \end{cases} \tag{4.23}$$

The right hand side of $\sim$ in (4.23) involves

1. a fixed number of case distinctions that are determined by the
2. computations of the moduli of p_1 and p_2,
3. computations of $\lfloor \dot{-} \rfloor$-expressions and
4. a finite and fixed number of additions and multiplications determined by the polynomials f_{ij}.

Therefore, (4.23) describes a primitive recursive function, using the functions $+$, $\cdot$, $\lfloor \dot{-} \rfloor$ and mod (to which $\equiv_k$ can be reduced). Moreover, if we assume these functions *as given*, (4.23) states that the gcd of two integers can be computed within a fixed number of steps. This, however, contradicts a result by L. van den Dries, given in [vdD03, Theorem 6.1]. There, he proves, by model theoretic considerations, that the number of steps needed to compute the gcd of two integers – under the assumption that addition, multiplication, $\lfloor \dot{-} \rfloor$ and mod are given – is unbounded. More precisely, for infinitely many p_1, p_2, the number of steps needed to compute $\gcd(p_1, p_2)$ has a lower bound linear in $p_1 + p_2$.

But there are cases, when modulo reduction is possible, though. Consider, for instance, the matrix

$$\mathbf{A} = \begin{pmatrix} p_1 p_2^2 \\ p_1 + 1 \\ 2 \end{pmatrix} \tag{4.24}$$

which contains the constant term 2. Considerations similar to those given in Section 4.1.1 show that

$$\gcd(p_1 p_2^2, p_1 + 1, 2) \sim \begin{cases} 1 & \text{if } p_1 \equiv_2 0 \wedge p_2 \equiv_2 0 \\ 1 & \text{if } p_1 \equiv_2 0 \wedge p_2 \equiv_2 1 \\ 2 & \text{if } p_1 \equiv_2 1 \wedge p_2 \equiv_2 0 \\ 1 & \text{if } p_1 \equiv_2 1 \wedge p_2 \equiv_2 1. \end{cases}$$

We can as well find unimodular matrices $\mathbf{U}_{ij}$ for each case $p_1 \equiv_2 i \wedge p_2 \equiv_2 j$ ($i, j \in \{0, 1\}$) such that

$$\mathbf{U}_{ij}\mathbf{A} \sim (\gcd(p_1 p_2^2, p_1 + 1, 2), 0, 0)^t.$$

For example, if $p_1 = 2p'_1 + 1$ and $p_2 = 2p'_2$ then

$$\mathbf{U}_{10} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & -(p'_1+1) \\ 1 & 0 & -2p'^2_2(2p'_1-1) \end{pmatrix}.$$

In fact, we can generalize this example to the case that, whenever some $m \times 1$-matrix contains a constant c, complete modulo reduction to the form $(d, \mathbf{0})^t$ is possible for all occurring cases $p_1 \equiv_{|c|} j_1 \wedge \cdots \wedge p_z \equiv_{|c|} j_z$ with $(j_1, \ldots, j_z) \in \{0, \ldots, |c|-1\}^z$.

What about $m \times n$-matrices with $n > 1$? Let us extend the example from above and let now

$$\mathbf{B} = \begin{pmatrix} p_1p_2^2 & 2 \\ p_1+1 & 3 \\ 2 & 4 \end{pmatrix}. \tag{4.25}$$

If we simply reduce the left column as above, we may introduce parameters to the right column. Take again $p_1 = 2p'_1 + 1$ and $p_2 = 2p'_2$. Then

$$\mathbf{U}_{10}\mathbf{B} = \begin{pmatrix} 2 & 4 \\ 0 & 3-4(p'_1+1) \\ 0 & 2-8p'^2_2(2p'_1-1) \end{pmatrix}. \tag{4.26}$$

We can avoid this undesirable accumulation of parameters within a former parameter-free column if we simply switch the columns. And again, this observation can be easily generalized. Whenever it is possible to reorder the entries of a matrix by interchanging rows or columns such that the obtained matrix contains a lower triangular matrix with constant entries only, complete modulo reduction is possible for that lower part of the matrix.

Even in the case that only one column can be completely reduced, this information may be enough to conclude that the respective equation system has no solution. Consider

$$\mathbf{x}\mathbf{B} = (2p_1p_2 + 1, 0) \tag{4.27}$$

in case $p_1 = 2p'_1 + 1$ and $p_2 = 2p'_2$. With $\mathbf{B}$ being only partially reduced, as in (4.26), we can already see that the equations do not have any solutions since 2 does not divide $2(2p'_1+1)(2p'_2)+1$ for any $p'_1, p'_2 \in \mathbb{Z}$. Any echelon form of $\mathbf{B}$ will have $(2,0,0)^t$ as left column but $2p_1p_2+1$, the first entry of the right hand side of (4.27), will always be odd.

Chapter 5

Conclusion

This work described certain possibilities and limits of solving parametric linear Diophantine equation systems as they occur in Banerjee's data dependence analysis. A complete algorithmic approach that extends the well-known non-parametric techniques such as Echelon reduction and integral divisibility could be developed in the case of one non-linear parameter. The algorithms were presented in such a way that it should be straight forward to implement them in Haskell, provided the respective libraries for vector and matrix arithmetic are present.

To get these results, we had to adapt quasi-polynomials to our needs, which in turn allows us now to treat even programs where the access functions contain coefficients from the set $\mathcal{AIQ}$, which is more than we initially expected. By the remarks given at the end of Section 2.3, we are confident that these methods equally apply for coefficients with *nested* $\lfloor \div \rfloor$-expressions. Moreover, we believe that the presented method of l-extension can be adjusted to extend other all-integral methods known for the polytope model.

A severe problem that could impose limitations in practical applications could be the growth of the decision trees. We think that this growth is similar to the growth of coefficients as it occurs for the computation of gcd's in $\mathbb{Z}[X]$.

The multi-parametric case turned out to be more complicated. Model theoretic considerations imply that a simple continuation of the uni-parametric approach is in general not possible. We gave, however, heuristic examples which demonstrate that in special situations some uni-parametric results can be carried over. This may be a starting point for further research.

Bibliography

[Ban93] Utpal K. Banerjee. *Loop Transformations for Restructuring Compilers: The Foundations.* Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[Ehr62] Eugène Ehrhart. Sur les polyèdres rationnels homothétiques à n dimensions. *C. R. Acad. Sci. Paris*, 254:616–618, 1962.

[Ehr77] Eugène Ehrhart. *Polynômes arithmétiques et méthode des polyèdres en combinatoire*, volume 35. Birkaueser Verlag, 1977. 165 pages.

[Grö03] Armin Größlinger. Extending the Polyhedron Model to Inequality Systems with Non-linear Parameters using Quantifier Elimination. Diploma thesis, Universität Passau, September 2003. `http://www.infosun.fmi.uni-passau.de/cl/arbeiten/groesslinger.ps.gz`.

[Jon02] Simon Peyton Jones, editor. *Haskell 98 Language and Libraries: The Revised Report.* September 2002. `http://haskell.org/definition/haskell98-report.pdf`.

[Kei97] Harald Keimer. Datenabhängigkeitsanalyse zur Schleifenparallelisierung: Vergleich und Erweiterung zweier Ansätze. Diploma thesis, Universität Passau, January 1997. `http://www.infosun.fim.uni-passau.de/cl/loopo/doc/keimer-d.ps.gz`.

[KMW67] Richard M. Karp, Raymond E. Miller, and Shmuel Winograd. The organization of computations for uniform recurrence equations. *Journal of the ACM*, 14(3):563–590, July 1967.

[Lam74] Leslie Lamport. The parallel execution of DO loops. *Communications of the ACM*, 17(2):83–93, February 1974.

[Len93] Christian Lengauer. Loop Parallelization in the Polytope Model. In Eike Best, editor, *CONCUR'93*, Lecture Notes in Computer Science 715, pages 398–416. Springer-Verlag, 1993.

[Leu96] Armin Leutbecher. *Zahlentheorie.* Springer, 1996.

[LG95] Christian Lengauer and Martin Griebl. On the parallelization of loop nests containing while loops. In N. N. Mirenkov, Q.-P. Gu, S. Peng, and S. Sedukhin, editors, *Proc. 1st Aizu Int. Symp. on Parallel Algorithm/Architecture Synthesis (pAs'95)*, pages 10–18. IEEE Computer Society Press, 1995.

[LS06] Aless Lasaruk and Thomas Sturm. Weak quantifier elimination for the full linear theory of the integers - a uniform generalization of presburger arithmetic. Number MIP-0604. 2006.

[Mat70] Yuri Matiyasevich. Enumerable sets are diophantine. *Soviet Mathematics Doklady*, 11(2):354–358, 1970.

[Sch87] Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1987.

[vdD03] Lou van den Dries. Generating the greatest common divisor, and limitations of primitive recursive algorithms. *Foundations of Computational Mathematics*, 3(3):297–324, 2003.

[WBK93] Volker Weispfenning, Thomas Becker, and Heinz Kredel. *Gröbner Bases: A Computational Approach to Commutative Algebra*. Graduate Texts in Mathematics. Springer, New York, 1993.

[Wei90] Volker Weispfenning. The complexity of almost linear diophantine problems. *J. Symb. Comput.*, 10(5):395–404, 1990.

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Diplomarbeit selbständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe und alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, als solche gekennzeichnet habe, sowie dass diese Diplomarbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt wurde.

Passau, den 26. Juli 2007

(Stefan Schuster)