Masarykova univerzita v Brně
Přírodovědecká fakulta

Unification Modulo Associativity and Idempotency

disertační práce

Vypracoval: Mgr. Ondřej Klíma
Školitel: doc. RNDr. Libor Polák, CSc.

Brno, srpen 2003

Prohlašuji, že jsem tuto disertační práci vypracoval samostatně a že jsem čerpal pouze z uvedených pramenů.

V Brně 25. 8. 2003

Acknowledgments

I would like to thank Libor Polák, my supervisor, for his guidance and advice in my research. He provided me with continuing motivation and encouragement.

I have to thank also other members of the Department of Algebra and Geometry, for many interesting courses, friendly environment and their interest in my research.

I am also indebted to Jorge Almeida and Christian Herrmann who invited me to study abroad. My stay in Porto certainly broadened my mathematical horizon. In Darmstadt, I began to work on major parts of my thesis.

My special thanks belong to Michal Kunc, my colleague and friend. It was very inspiring to study with him at first, and to co-operate with him later. He was the first reader of my works, and his attentive eye always spotted and corrected any mistake in both mathematics and English. I owe him a lot, since I definitely profited more from our co-operation.

It would not be possible to finish my thesis without the help of Ivana Černá, Jiří Srba, Pascal Tesson and Denis Thérien, with whom I collaborated on my research, and with whom I co-wrote some of the papers that became the basis of my thesis.

And last but not least, I have to thank my family and friends for their continuous support and encouragement. They were always at my side to help me get through the difficult times.

Abstract

This thesis shows that the unification problem for the theory of a single associative and idempotent binary function symbol (AI-unification), i.e. the problem of solving systems of equations in free idempotent semigroups, is NP-complete. The same result is established for the AI-matching problem, where only equations with variable-free right hand sides are considered. The complexity of the unification problem in the theory of a single associative, idempotent and commutative function symbol (ACI-unification) is known to be polynomial. We describe the complexity of the unification problems for all equational theories of a binary function symbol that consist of associativity and idempotency together with any set of additional identities. We also describe the complexity of the matching problems for these theories. Unification and matching problems are solved for combinations (disjoint union) of these theories as well. Finally, we consider the problem of solving equations from another point of view; namely, a finite idempotent semigroup is fixed and the equations form the input. We establish almost identical dichotomy results as in the case of the unification and matching problems.

Contents

Introduction

Unification is a fundamental process upon which many methods for automated deduction are based. Simply speaking, unification theory deals with solving equations. We consider *terms* that are built from function symbols (say f, a and b, where f is binary and a, b are nullary) and variables (say x and y). An instance of a *unification problem* is a pair of terms (an *equation*). The question is whether it is possible to replace the variables by terms such that the two terms obtained in this way are (syntactically) equal (i.e. whether there exists a *unifier* of the terms). For example, if $f(x,b) =^? f(f(a,b),y)$ is an equation, then we can see that there is a unifier $\{x \mapsto f(a,b), y \mapsto b\}$. For the equation $f(a,x) =^? f(x,b)$ there is no unifier, and for the equation $f(x,y) =^? f(y,x)$ there are infinitely many unifiers. Fortunately, if there is a unifier then there is a *most general unifier*, which has the property that any other unifier can be obtained from it by substitution. In the last example, $\{x \mapsto y\}$ is such a most general unifier.

This kind of unification is called *syntactic* unification. A different concept is obtained if one considers function symbols satisfying certain semantic properties. The most natural notion is *equational* unification where the symbols satisfy a given equational theory E (a set of identities). We speak about the E-*unification problem.* Now the question is whether there exists a unifier which makes terms equivalent with respect to the fully invariant congruence of the algebra of all terms generated by E. For example, if $E = \{f(x,x) \doteq a\}$, then the equation $f(x,b) =^? f(f(a,b),y)$ has also an E-unifier $\{x \mapsto b, y \mapsto f(a,b)\}$. From the point of view of universal algebra, the syntactic unification problem is solving of equations in a algebra of all terms, and the E-unification problem is solving them in a relatively free algebra in the variety corresponding to the set of identities E. For equational unification, things are not as nice as for syntactic unification. In fact, depending on the theory E, the E-unification problem may be undecidable[1], and even if it is decidable, solvable equations need not have a most general E-unifier. Classification of equational theories according to their behavior in this

[1]Such a theory is, for example, the theory of commutative rings with a unit for which the unification problem is just the well known Hilbert's tenth problem.

respect is a rather interesting question of unification theory. In the case of syntactic unification any system is equivalent to a single equation. In the case of equational unification it is not true,and therefore consideration of systems of equations is more general.

The name "unification" and the first formal investigation of this notion is due to J. A. Robinson [42], who introduced unification as the basic operation of his resolution principle, showed that unifiable terms have a most general unifier and described an algorithm for computing this unifier. The resolution principle is the base of many automated provers.

The notions of "unification" and "most general unifier" were independently reinvented by Knuth and Bendix [34] as a tool for testing term rewriting systems for local confluence by computing critical pairs. Equational unification was introduced both in resolution-based theorem proving and in term rewriting as a means for treating certain troublesome equational axioms (like associativity and commutativity) in a special manner. For example, in term rewriting, the axiom $f(x, y) \doteq f(y, x)$ cannot be oriented into terminating rewrite rules. One way of solving this problem is to take such non-orientable identities completely out of the rewrite process, and perform rewriting with respect to the remaining (orientable) rules modulo the unoriented ones. Another area where we can find applications of unification theory is logic programming.

If the E-unification problem is decidable, then other questions are the complexity of the problem and a description of the set of all unifiers. There is a natural generalization of the concept of a most general unifier, namely a minimal complete set of unifiers, i.e. a set M having the property that any unifier can be obtained from some unifier from M by substitution and that the unifiers in M are independent in the sense that one can not be obtained from another one. If a minimal complete set of unifiers exists, it is determined uniquely up to equivalence of unifiers. It is empty if the system has no unifier.

In the literature, there are often discussed the theories of associativity (A), commutativity (C), idempotency (I) and their combinations. The decidability of the A-unification problem, i.e. solving of word equations, has been open for a long time. The original (deep) proof of its decidability by Makanin [35] (the algorithm using the Makanin's proof is in 3-NEXPTIME) was improved by Plandowski [39] (his algorithm is in PSPACE). On the other hand, it is not hard to prove that the problem is NP-hard. It was shown by Plotkin [40] that any system of word equations has a minimal complete set of unifiers, although it can be infinite. The AC-unification problem is decidable in polynomial time and a minimal complete set of unifiers always exists and it is finite. Another theories for which a numerous interesting results were established are the theories of groups, abelian groups, commutative and boolean rings (see [6, 8, 28, 45] for surveys on the area).

In the center of our interest in this thesis is the AI-unification problem. The problem is decidable because idempotent semigroups are locally finite (i.e. each finitely generated idempotent semigroup is finite, hence for any system we have to check only finitely many substitutions for being unifiers). AI-unification is important for unification theory, because it is a natural example of a theory for which a system without a minimal complete set of unifiers exists. It was proved independently by Baader [2] and Schmidt-Schauss [43]. Some results concerning the complexity of the problem were established in [26] and minimal complete sets of unifiers were studied in [3].

In the thesis we will concentrate on the complexity questions. The main result is the NP-completeness of the AI-unification problem. We also establish the complexities of the E-unification problems for several related theories. Typically our goal for a certain class of theories will be a dichotomy theorem cutting the class to theories with NP-complete unification problem and theories with tractable unification problem.

The thesis is based mainly on the material concerning the AI-unification problems that has been published previously. The original articles have been, of course, further elaborated. The structure of the thesis is the following. Each article comprises one separate chapter: paper [32] is included in Chapter 2, paper [30] in Chapter 3 and paper [31] in Chapter 4. Chapter 1 contains all technical notation used in these papers. In Chapter 5 we view the problem of solving equations from a different perspective; this material will be contained in article [33], currently in preparation. For readers who already saw the original papers, the differences between the papers and the thesis are commented in the introductory part of each chapter.

Chapter 1 summarizes the necessary notation and results from semigroup theory (Section 1.1) and unification theory (Section 1.2) which are used in the thesis. The two parts have a different spirit. The first part introduces a lot of definitions needed for our constructions and it is quite technical. The second part can be viewed as a short introduction to unification theory and is written for algebraists which are not familiar with this theory.

The second chapter contains the proof of the NP-completeness of the AI-matching problem, a special case of the AI-unification problem where only equations with variable-free right hand sides are considered. We should mention that the part of the proof showing the problem to be in NP was overridden by the results from Chapter 3. We put the original proof to the thesis too, for we think the method of the combinatorial proof is interesting and brings a good feeling of the used term rewriting system solving the identity problem for idempotent semigroups developed by Siekmann and Szabo [46]. The proof of the NP-hardness can be seen as a classical reduction from the 3-SAT problem, but it is quite tricky. The proofs of NP-hardness in Chapter 4 are modifications of this basic construction.

The main result of the thesis is contained in Chapter 3, where we prove that also the AI-unification problem is NP-complete. The idea of the nondeterministic polynomial algorithm is really easy and natural. For any variable we guess some additional assumptions (of polynomial length) and then we directly solve the resulting system of equations. So, we concentrate on the problem with these additional assumptions, which we call the *Simple AI-unification problem*. Our method of solving the Simple AI-unification problem has several consequences. This chapter also contains an observation that systems consisting of a single equation can be solved in polynomial time, although two equations are as hard as systems.

In Chapter 4 we consider the unification problem in the theory of a single associative and idempotent function symbol which satisfies some additional identities. In contrast to the AI-unification problem we are able to prove that the problems belong to NP in an easier way. On the other hand, for almost all theories an easy modification of the NP-hardness proof for the AI-unification problem is possible. For the other theories we find polynomial algorithms, which complete the dichotomy theorem for these theories. Finally, we consider combinations of such theories which can also contain uninterpreted function symbols. A dichotomy theorem for these theories is again established.

We drop the subject of unification theory in Chapter 5 which contains a quite independent research. Here, we consider equations not in a free semigroup but in a given concrete finite semigroup. We show some dichotomy theorems for idempotent semigroups which are similar to those for unification problems.

The last chapter simply summarizes the main results of the thesis.

We assume the reader has a basic knowledge of universal algebra (terms, identities, varieties, congruences) and complexity theory (NP-completeness, reductions). For indefinite terms we refer to the book [13] for universal algebra and the book [37] for complexity theory. Also knowledge of basic results from semigroup theory and unification theory can help, though we hope that the thesis contains all necessary concepts. If not, see [24] and [8] which are basic introductions in the areas. In fact, we take the survey paper [8] as a base of notation and terminology from unification theory with certain exceptions from which the following differences are the most important: substitutions or mappings in general are written in prefix notation (i.e. $\alpha(x)$ instead of $x\alpha$) and consequently a composition of substitutions is read from right to left, identities (equational axioms) are written $s \doteq t$ and by a problem we mean a problem from the point of view of the theory of computation, not a system of equations (an instance of a problem here).

Chapter 1

Background and Notations

In this chapter we fix notation and recall known results which are needed for formulations and proofs of our results.

The first part deals with semigroups. Paragraphs 1.1.1 – 1.1.4 use word invariants for complete description of varieties of idempotent semigroups and monoids, i.e. theories of one binary function symbol which is associative, idempotent and may satisfy some additional identities. Paragraph 1.1.5 contains short introduction to the structure of semigroups and notation of certain pseudovarieties of semigroups. The concepts defined in this paragraph are further used in Chapter 5.

The second part consists of two paragraphs. Paragraph 1.2.1 can be viewed as a quick overview of the unification theory. Paragraph 1.2.2 contains reformulation of unification problems in terms of word equations and known results concerning the unification problem modulo associativity and idempotency.

1.1 Semigroup Theory

1.1.1 Basic Definitions of Invariants

We denote the set of all positive integers and the set of all non-negative integers by $\mathbb{N}$ and $\mathbb{N}_0$ respectively, the set of all subsets of a given set X by $\mathcal{P}(X)$ and the cardinality of this set X by $\|X\|$.

Let Σ be an arbitrary finite alphabet. The *free semigroup* over Σ will be denoted Σ^+, the *free monoid* $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$, where ϵ is the *empty word*. For a word $u \in \Sigma^*$ we denote $\mathsf{c}(u)$ the set of all letters occurring in u (the *content* of u). Further $|u|$ stands for the *length* of the word u and $u_\sharp$ for the cardinality of $\mathsf{c}(u)$ (i.e. the number of letters occurring in the word u). We say that $u \in \Sigma^*$ is a *factor* of $v \in \Sigma^*$ if there are a *prefix* $v_1 \in \Sigma^*$ of v and a *suffix* $v_2 \in \Sigma^*$ of v

such that $v = v_1 u v_2$.

By an *invariant* we mean a mapping from the set Σ^* to some set Θ (in this thesis the set Θ is always either Σ^* or 2^Σ). We will use the following invariants. For $u \in \Sigma^*$ we define

$\mathsf{h}(u)$ – the first letter of u (*head*),
$\mathsf{t}(u)$ – the last letter of u (*tail*),
$\overrightarrow{u}$ – the sequence of the first occurrences of all letters from $\mathsf{c}(u)$ in u,
$\overleftarrow{u}$ – the sequence of the last occurrences of all letters from $\mathsf{c}(u)$ in u,
$0(u)$ – the longest prefix of u which contains at most $u_\sharp - 1$ different letters,
$1(u)$ – the longest suffix of u which contains at most $u_\sharp - 1$ different letters,

with conventions $\mathsf{h}(\epsilon) = \mathsf{t}(\epsilon) = \overrightarrow{\epsilon} = \overleftarrow{\epsilon} = 0(\epsilon) = 1(\epsilon) = \epsilon$.

Because the invariants 0 and 1 are mappings from the set Σ^* to the same set Σ^*, we can compose them. In this way each element from the set $\{0,1\}^*$ can be viewed as an invariant (so called *iterated invariant*) defined inductively: for an arbitrary $u \in \Sigma^+$ we put $\mathsf{e}(u) = u$ for the *empty invariant* e (e is a neutral element in $\{0,1\}^*$) and $(0\mu)(u) = 0(\mu(u))$, $(1\mu)(u) = 1(\mu(u))$ for all iterated invariants $\mu \in \{0,1\}^*$.

An arbitrary iterated invariant $\mu \in \{0,1\}^*$ can be written in a unique way as $\mu = 0^{k_0} 1^{l_1} 0^{k_1} \cdots 0^{k_n} 1^{l_{n+1}}$ where $n \geq 0$, $k_i, l_i \in \mathbb{N}$ for all $i \in \{1, \dots, n\}$ and $k_0, l_{n+1} \in \mathbb{N}_0$ (here $0^0 = \mathsf{e}$, $1^0 = \mathsf{e}$). Then we define the *number of blocks* in μ as $\mathrm{block}(\mu) = 2n + \mathrm{sgn}(k_0) + \mathrm{sgn}(l_{n+1})$, where $\mathrm{sgn}(m) = 1$ for all $m \in \mathbb{N}$ and $\mathrm{sgn}(0) = 0$. (The blocks are the iterated invariants 0^{k_0}, 1^{l_1}, …, $1^{l_{n+1}}$.) The last invariant 0 or 1 employed during the application of the iterated invariant $\mu \in \{0,1\}^*$, i.e. the first letter of μ if we consider μ as a word over the alphabet $\{0,1\}$, is denoted by $\mathrm{last}(\mu)$.

For $i \in \mathbb{N}$ and a word $u \in \Sigma^+$ we define

$\mathrm{pref}_i(u)$ – the longest prefix of u which contains at most i different letters,
$\mathrm{suff}_i(u)$ – the longest suffix of u which contains at most i different letters.

We have $\mathrm{pref}_i(u) = u$ and $\mathrm{suff}_i(u) = u$ for all $i \geq u_\sharp$. For $i < u_\sharp$, $\mathrm{pref}_i(u)$ is the $(u_\sharp - i)$-th iteration of the invariant 0 on u and dually $\mathrm{suff}_i(u)$ is the $(u_\sharp - i)$-th iteration of the invariant 1 on u, i.e. $\mathrm{pref}_i(u) = 0^{u_\sharp - i}(u)$, $\mathrm{suff}_i(u) = 1^{u_\sharp - i}(u)$. So, for two words $u, v \in \Sigma^+$ we have $\overrightarrow{u} = \overrightarrow{v}$ if and only if $u_\sharp = v_\sharp$ and $\mathsf{c}(\mathrm{pref}_i(u)) = \mathsf{c}(\mathrm{pref}_i(v))$ for all $i = 1, \dots, u_\sharp$.

For technical reasons we need an invariant n which is defined as $\mathsf{n}(u) = \epsilon$ for an arbitrary word $u \in \Sigma^*$.

For a word $u \in \Sigma^+$ and $n = u_\sharp$ we define $\overset{\Rightarrow}{u} \in \Sigma^+$ in this way: $\overset{\Rightarrow}{u} = a_n b_n \cdots a_1 b_1$ where $a_i, b_i \in \Sigma$, $a_n a_{n-1} \cdots a_1 = \overrightarrow{u}$ and $b_i = \mathsf{t}(0^{i-1}(u))$ for all $i \in \{1, \dots, n\}$. Dually we define $\overset{\Leftarrow}{u}$.

Example 1.1.1. We demonstrate these definitions on the word

$$u = bbdabdbcacbdb$$

over the alphabet $\{a, b, c, d, e\}$. We have $\mathsf{c}(u) = \{a, b, c, d\}$, $u_\sharp = 4$, $|u| = 13$, $\mathsf{h}(u) = b$, $\mathsf{t}(u) = b$, $0(u) = bbdabdb$, $1(u) = cbdb$. Further,

$$10(u) = 1(0(u)) = 1(bbdabdb) = bdb$$

and analogically $0^2(u) = \mathrm{pref}_2(u) = bbd$, $01(u) = cb$, $1^2(u) = bdb$, $10^2(u) = d$, $010(u) = b$ and $0^3(u) = \mathrm{pref}_1(u) = bb$. Finally, $\overrightarrow{u} = bdac$ and $\overleftarrow{u} = acdb$, $\overset{\Rightarrow}{u} = bbddabcb$ and $\overset{\Leftarrow}{u} = baccbdbb$.

1.1.2 Varieties of Idempotent Semigroups

An *idempotent semigroup* (in short called a *band*) is a semigroup where the identity $x^2 \doteq x$ is satisfied. The variety of all bands is denoted as $\mathcal{B}$. The lattice of all varieties of bands $\mathcal{L}(\mathcal{B})$ (see Figure 1.1) was described independently by Birjukov [10], Fennemore [17] and Gerhard [18] (see [19] for a revised version).

For a given variety of semigroups (particularly bands) $\mathcal{V}$ and a given set Σ we denote by $\sim_{\mathcal{V},\Sigma}$ the (fully invariant) congruence of Σ^+ such that for $p, q \in \Sigma^+$ we have $p \sim_{\mathcal{V},\Sigma} q$ if and only if $p \doteq q$ is an identity satisfied in the variety $\mathcal{V}$. (The indices $\mathcal{V}$ and Σ are omitted if they are obvious from context.)

It appears to be more useful to know a solution of the *identity problem* for the given variety $\mathcal{V}$, i.e. an effective description of the congruence $\sim_\mathcal{V}$, rather than a base of identities for $\mathcal{V}$. In this paper we will use a description of a solution of the identity problem for varieties of bands due to Polák [41] using the invariants defined above. We will present this description later, now we just explain the appearance of these invariants on Figure 1.1. We associate to the variety of bands $\mathcal{V}$ the set of invariants $\{\iota_1, \ldots, \iota_n\}$ (as on Figure 1.1) if the following condition is satisfied: for two words $p, q \in \Sigma^+$ we have $p \sim_\mathcal{V} q$ if and only if $\iota_i(p) = \iota_i(q)$ for all $i \in \{1, \ldots, n\}$.

Example 1.1.2. For the word $u = bbdabdbcacbdb$ from Example 1.1.1 we have

$$u \sim_{\mathcal{LSNB}} bdabcb = v$$

because

$$\overset{\Rightarrow}{u} = \overset{\Rightarrow}{v}.$$

Analogically,

$$u \sim_{\mathcal{RB}} bdacdb = w$$

because $\overrightarrow{u} = \overrightarrow{w}$ and $\overleftarrow{u} = \overleftarrow{w}$. We have also

$$u \sim_{\mathcal{LNB}} bacd$$

because $\mathsf{c}(u) = \mathsf{c}(bacd)$ and $\mathsf{h}(u) = \mathsf{h}(bacd)$.

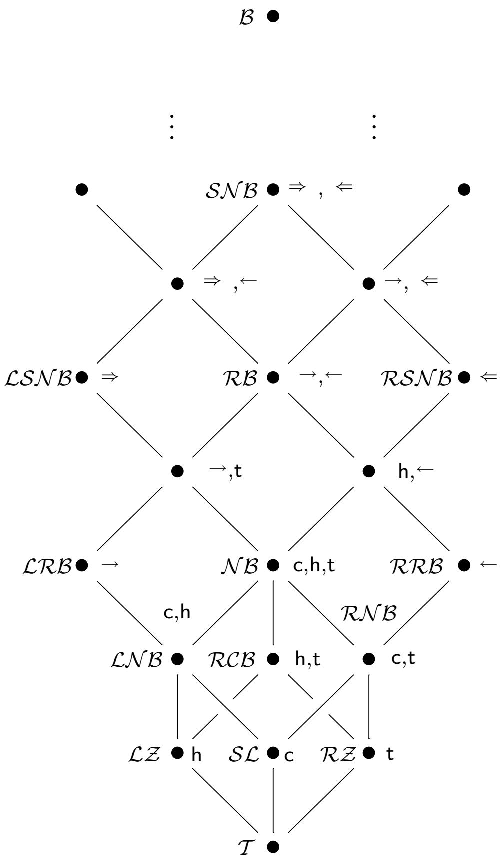

Figure 1.1: The lattice of varieties of bands $\mathcal{L}(\mathcal{B})$.

In this paper we will meet especially the varieties from the bottom of the lattice $\mathcal{L}(\mathcal{B})$ which were assigned certain names in literature. Here is a list of some of them with the corresponding bases of identities.

$\mathcal{SNB} = \mathrm{Mod}(x^2 \simeq x, xyzuzyx \simeq xyzxzuzxzyx)$ – all seminormal bands,
$\mathcal{LSNB} = \mathrm{Mod}(x^2 \simeq x, xyzxz \simeq xyz)$ – all left seminormal bands,
$\mathcal{RSNB} = \mathrm{Mod}(x^2 \simeq x, xzxyz \simeq xyz)$ – all right seminormal bands,
$\mathcal{RB} = \mathrm{Mod}(x^2 \simeq x, xyxzx \simeq xyzx)$ – all regular bands,
$\mathcal{LRB} = \mathrm{Mod}(x^2 \simeq x, xyx \simeq xy)$ – all left regular bands,
$\mathcal{RRB} = \mathrm{Mod}(x^2 \simeq x, xyx \simeq yx)$ – all right regular bands,
$\mathcal{NB} = \mathrm{Mod}(x^2 \simeq x, xyzx \simeq xzyx)$ – all normal bands,
$\mathcal{LNB} = \mathrm{Mod}(x^2 \simeq x, xyz \simeq xzy)$ – all left normal bands,
$\mathcal{RNB} = \mathrm{Mod}(x^2 \simeq x, xyz \simeq yxz)$ – all right normal bands,
$\mathcal{SL} = \mathrm{Mod}(x^2 \simeq x, xy \simeq yx)$ – all semilattices,
$\mathcal{RCB} = \mathrm{Mod}(x^2 \simeq x, xyz \simeq xz)$ – all rectangular bands,

$\mathcal{LZ} = \mathrm{Mod}(xy \doteq x)$ – all left zero semigroups,
$\mathcal{RZ} = \mathrm{Mod}(xy \doteq y)$ – all right zero semigroups,
$\mathcal{T} = \mathrm{Mod}(x \doteq y)$ – all trivial semigroups.

1.1.3 Identity Problems for Varieties of Bands

The identity problem for the variety of all bands $\mathcal{B}$ is solvable by the following well known proposition.

Proposition 1.1.3 ([21]). *Let* $p, q \in \Sigma^+$.

(i) *If* $p_\sharp = 1$ *then* $p \sim_{\mathcal{B}} q$ *iff* $\mathsf{c}(p) = \mathsf{c}(q)$

(ii) *If* $p_\sharp > 1$ *then* $p \sim_{\mathcal{B}} q$ *iff* $\mathsf{c}(p) = \mathsf{c}(q)$, $0(p) \sim_{\mathcal{B}} 0(q)$ *and* $1(p) \sim_{\mathcal{B}} 1(q)$.

The following useful lemma is a trivial consequence of Proposition 1.1.3.

Lemma 1.1.4. *Let* $u, v, w \in \Sigma^+$ *be such that* $\mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)$. *Then* $uvw \sim_{\mathcal{B}} uw$.

It follows from Proposition 1.1.3 that for $u, v \in \Sigma^+$ satisfying $u \sim v$ we have also $\mathrm{pref}_i(u) \sim \mathrm{pref}_i(v)$, $\mathrm{suff}_i(u) \sim \mathrm{suff}_i(v)$ for all $i \in \mathbb{N}$, $\overrightarrow{u} = \overrightarrow{v}$ and $\overleftarrow{u} = \overleftarrow{v}$.

We can reformulate the previous proposition in the following way. For each word $u \in \Sigma^+$ we define the mapping $\mathrm{inv}_u : \{0,1\}^* \to 2^\Sigma$ by the rule $\mathrm{inv}_u(\mu) = \mathsf{c}(\mu(u))$ for all $\mu \in \{0,1\}^*$. Proposition 1.1.3 says that for $p, q \in \Sigma^+$ we have $p \sim_{\mathcal{B}} q$ if and only if the mappings inv_p and inv_q are equal.

The mapping inv_u can be also used for the construction of a short word w which is $\sim_{\mathcal{B}}$-related to the word u (we use such a technique in Section 4.2). This is not usual in literature because there is a natural normal form in the class of words $\sim_{\mathcal{B}}$-related to the given word u based on the term rewriting system for free bands developed in [46]. We introduce it in Section 2.2 and we will see that this rewriting system gives also a polynomial algorithm solving the identity problem in the variety of all bands $\mathcal{B}$.

We turn our attention to the description of the identity problem for varieties of bands. First of all we define two ordered sets Λ and Ξ (see Figure 1.2). Ordering on the set $\Lambda = \{0,1\} \times \mathbb{N}$ is given in this way: $(\iota, n) < (\kappa, m)$ iff $n > m$ and order on the set of invariants $\Xi = \{\mathsf{n}, \mathsf{h}, \mathsf{t}, \mathsf{c}\}$ is given by $\mathsf{n} < \mathsf{h}, \mathsf{t}$ and $\mathsf{h}, \mathsf{t} < \mathsf{c}$.

Now we consider the ordered set

$$I = \{\varphi : \Lambda \to \Xi \mid \varphi \text{ is isotone}, \varphi(0,n) \neq \mathsf{t}, \varphi(1,n) \neq \mathsf{h} \text{ for } n \in \mathbb{N}\}.$$

Ordering on the set I is given in a usual way: $\varphi \leq \varphi'$ if and only if $\varphi(\iota, n) \leq \varphi'(\iota, n)$ for all $(\iota, n) \in \Lambda$.

The next proposition describes the interval $[\mathcal{SL}, \mathcal{B}]$ in the lattice $\mathcal{L}(\mathcal{B})$ between the varieties $\mathcal{SL}$ and $\mathcal{B}$.

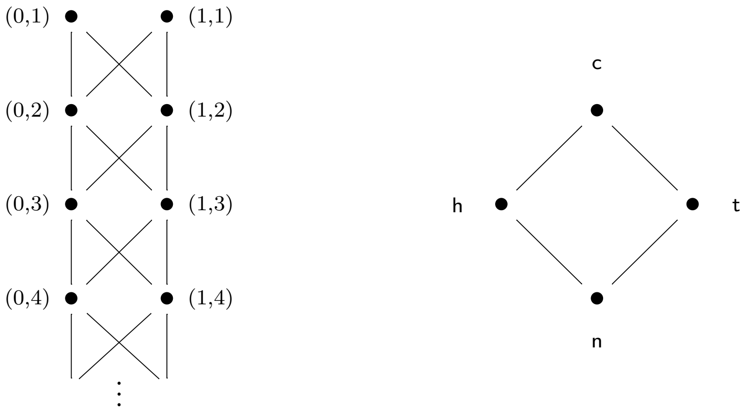

Figure 1.2: The ordered sets Λ and Ξ.

Proposition 1.1.5 ([41]). $[\mathcal{SL}, \mathcal{B}] \cong I$, *where every variety* $\mathcal{V}$ *corresponds to the mapping* $\varphi_{\mathcal{V}} \in I$ *such that for two words* $p, q \in \Sigma^+$: $p \sim_{\mathcal{V}} q$ *if and only if* $\mathsf{c}(p) = \mathsf{c}(q)$ *and for all iterated invariants* $\mu \in \{0,1\}^+$ *we have*

$$\overline{\mu}(\mu(p)) = \overline{\mu}(\mu(q)), \quad \textit{where} \quad \overline{\mu} = \varphi_{\mathcal{V}}(\mathrm{last}(\mu), \mathrm{block}(\mu)) \in \Xi.$$

Example 1.1.6. The variety $\mathcal{LRB}$ corresponds to $\varphi \in I$ satisfying $\varphi(0,1) = \mathsf{c}$ and $\varphi(\iota, n) = \mathsf{n}$ otherwise. This means that

$$p \sim_{\mathcal{LRB}} q \quad \text{iff} \quad \mathsf{c}(0^i(p)) = \mathsf{c}(0^i(q)) \quad \text{for all} \quad i \in \mathbb{N}_0,$$

because an iterated invariant $\mu \in \{0,1\}^+$ satisfies $\mathrm{last}(\mu) = 0$ and $\mathrm{block}(\mu) = 1$ if and only if $\mu = 0^i$ for some $i \in \mathbb{N}$. In other words

$$p \sim_{\mathcal{LRB}} q \quad \text{if and only if} \quad \overrightarrow{p} = \overrightarrow{q}.$$

The corresponding mapping φ for the variety $\mathcal{LSNB}$ is defined in the following way: $\varphi(0,1) = \mathsf{c}$, $\varphi(1,1) = \varphi(1,2) = \mathsf{t}$ and $\varphi(\iota, n) = \mathsf{n}$ otherwise. We have $p \sim_{\mathcal{LSNB}} q$ if and only if

i) $\mathsf{c}(p) = \mathsf{c}(q)$,

ii) $\overrightarrow{p} = \overrightarrow{q}$ (because $\varphi(0,1) = \mathsf{c}$) and

iii) $\mathsf{t}(1^i 0^j(p)) = \mathsf{t}(1^i 0^j(q))$ for all $i \in \mathbb{N}, j \in \mathbb{N}_0$ (because $\varphi(1,1) = \varphi(1,2) = \mathsf{t}$),

i.e.

$$p \sim_{\mathcal{LSNB}} q \quad \text{if and only if} \quad \overrightarrow{\overrightarrow{p}} = \overrightarrow{\overrightarrow{q}}.$$

Remark 1.1.7. By a *proper subvariety* $\mathcal{V}$ of the variety of all bands $\mathcal{B}$ we mean each subvariety of $\mathcal{B}$ except $\mathcal{B}$ itself. For each proper subvariety $\mathcal{V} \in [\mathcal{SL}, \mathcal{B}]$ there is an integer k such that $\varphi_{\mathcal{V}}(\iota, n) = \mathsf{n}$ for all $n \geq k$. For the variety of all bands $\mathcal{B}$ the corresponding mapping $\varphi_{\mathcal{B}}$ satisfies $\varphi_{\mathcal{B}}(\iota, n) = \mathsf{c}$ for all $(\iota, n) \in \Lambda$. In general, if we want to know whether $p \sim_{\mathcal{V}} q$ for two words $p, q \in \Sigma^+$, $\mathsf{c}(p) = \mathsf{c}(q)$, then it is enough to check the condition from the previous proposition only for the iterated invariants $\mu \in \{0,1\}^+$ such that $|\mu| < p_\sharp$, because $\mu(p) = \epsilon = \mu(q)$ for all μ, $|\mu| \geq p_\sharp$.

We will work in section 4.2 with some special varieties of bands. The variety $\mathcal{V}_k$ for $k \in \mathbb{N}$ is the variety corresponding to the mapping φ_k defined by the rule $\varphi_k(\iota, n) = \mathsf{c}$ if $n < k$ and $\varphi_k(\iota, n) = \mathsf{n}$ if $n \geq k$. This means that $p \sim_{\mathcal{V}_k} q$ if and only if $\mathsf{c}(\mu(p)) = \mathsf{c}(\mu(q))$ (i.e. $\mathrm{inv}_p(\mu) = \mathrm{inv}_q(\mu)$) for all iterated invariants $\mu \in \{0,1\}^*$ such that $\mathrm{block}(\mu) < k$. So, for example $\mathcal{V}_1 = \mathcal{SL}$ and $\mathcal{V}_2 = \mathcal{RB}$.

1.1.4 Varieties of Idempotent Monoids

For any idempotent monoid M with neutral element 1 the subset $M \setminus \{1\}$ forms a subsemigroup. Really, for $a, b \in M$ such that $a \cdot b = 1$ we have

$$a = a \cdot 1 = a(ab) = a^2 b = ab = 1.$$

This easy observation is behind the following well-known modifications of the previous results.

We denote $\mathcal{IM}$ the variety of all idempotent monoids. Other important varieties of idempotent monoids are the varieties of all regular, left regular, right regular, commutative and trivial idempotent monoids.

$\mathcal{RIM} = \mathrm{Mod}(x^2 \simeq x, 1 \cdot x \simeq x, x \cdot 1 \simeq x, xyxzx \simeq xyzx)$

$\mathcal{LRIM} = \mathrm{Mod}(x^2 \simeq x, 1 \cdot x \simeq x, xyx \simeq xy)$

$\mathcal{RRIM} = \mathrm{Mod}(x^2 \simeq x, x \cdot 1 \simeq x, xyx \simeq yx)$

$\mathcal{CIM} = \mathrm{Mod}(x^2 \simeq x, x \cdot 1 \simeq x, xy \simeq yx)$

$\mathcal{TM} = \mathrm{Mod}(x \simeq 1)$

Let $\mathcal{V}$ be a variety of idempotent monoids and let Σ be a set. We denote by $\sim_{\mathcal{V},\Sigma}$ the (fully invariant) congruence of Σ^* corresponding to the variety $\mathcal{V}$. One can prove that its restriction $\sim_{\mathcal{V},\Sigma} \cap (\Sigma^+ \times \Sigma^+)$ is a fully invariant congruence of the semigroup Σ^+. (Notice that $\sim_{\mathcal{V},\Sigma} \cap (\Sigma^+ \times \Sigma^+) = \sim_{\mathcal{V},\Sigma} \setminus \{(\epsilon, \epsilon)\}$ in the case $\mathcal{V} \neq \mathcal{TM}$.) This means that $\mathcal{V}$ determines a variety of idempotent semigroups, denote it $\mathbf{S}(\mathcal{V})$, corresponding to this congruence. For example

$$\begin{aligned}\mathbf{S}(\mathcal{TM}) &= \mathcal{T},\\ \mathbf{S}(\mathcal{CIM}) &= \mathcal{SL},\\ \mathbf{S}(\mathcal{RRIM}) &= \mathcal{RRB},\\ \mathbf{S}(\mathcal{LRIM}) &= \mathcal{LRB},\\ \mathbf{S}(\mathcal{RIM}) &= \mathcal{RB}.\end{aligned}$$

In fact, $\mathbf{S}$ is an embedding of the lattice of varieties of idempotent monoids $\mathcal{L}(\mathcal{IM})$ (see Figure 1.3) to the lattice $\mathcal{L}(\mathcal{B})$ and $\mathbf{S}(\mathcal{V})$ is the smallest variety containing $\mathcal{V}$.

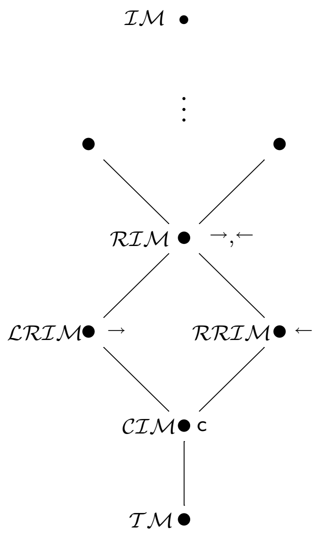

Figure 1.3: The lattice of varieties of idempotent monoids $\mathcal{L}(\mathcal{IM})$.

Moreover, the mappings φ corresponding to varieties of bands of the form $\mathbf{S}(\mathcal{V})$ from the interval $[\mathcal{SL}, \mathcal{B}]$ satisfy $\varphi(\iota, n) \in \{\mathsf{n}, \mathsf{c}\}$ for all $(\iota, n) \in \Lambda$. So, Proposition 1.1.5 can be modified to a description of the interval $[\mathcal{CIM}, \mathcal{IM}]$ in the lattice of all varieties of idempotent monoids if we consider the ordered set $\{\mathsf{n}, \mathsf{c}\}$ with order $\mathsf{n} < \mathsf{c}$ instead of the set Ξ. The solution of the identity problem for the variety of idempotent monoids $\mathcal{V}$ is the same as for the variety of idempotent semigroups $\mathbf{S}(\mathcal{V})$; only in addition the empty word ϵ (which forms a one-element $\sim_{\mathcal{V}}$-class in the case $\mathcal{V} \neq \mathcal{TM}$) is allowed.

1.1.5 Structure Theory of Semigroups

We recall some basic facts from semigroup theory concerning the structure of semigroups. These basic concepts can be found in any textbook on semigroup theory, e.g. [24].

Let S be a semigroup. We define the monoid S^1 in the following way. If S is a monoid itself, then $S^1 = S$, and if S is not a monoid, then $S^1 = S \cup \{e\}$, where $e \notin S$ is a new element and $es = se = e$ for any $s \in S^1$.

The *left ideal generated by an element* $a \in S$ is the set

$$S^1a = \{sa \mid s \in S\} \cup \{a\}.$$

An equivalence relation $\mathcal{L}$ on S is defined by the rule that $a\mathcal{L}b$ if and only if a and b generate the same left ideal, that is, if and only if $S^1a = S^1b$. Similarly, we define the equivalence relation $\mathcal{R}$ by the rule that $a\mathcal{R}b$ if and only if $aS^1 = bS^1$. The intersection of $\mathcal{L}$ and $\mathcal{R}$ is denoted by $\mathcal{H}$ and the join $\mathcal{L} \vee \mathcal{R}$, i.e. the least equivalence relation on S containing $\mathcal{L}$ and $\mathcal{R}$, is denoted by $\mathcal{D}$. Because the relations $\mathcal{L}$ and $\mathcal{R}$ commute, we can write

$$\mathcal{D} = \mathcal{L} \circ \mathcal{R} = \mathcal{R} \circ \mathcal{L} = \mathcal{L} \vee \mathcal{R}.$$

Finally, we define the two-sided analogue of $\mathcal{L}$ and $\mathcal{R}$. The *ideal generated by an element* $a \in S$ is the set S^1aS^1 and we define the equivalence $\mathcal{J}$ by the rule $a\mathcal{J}b$ if and only if $S^1aS^1 = S^1bS^1$. Since $\mathcal{D}$ is the smallest equivalence containing $\mathcal{L}$ and $\mathcal{R}$ and it is immediate that $\mathcal{L} \subseteq \mathcal{J}$ and $\mathcal{R} \subseteq \mathcal{J}$, we have also $\mathcal{D} \subseteq \mathcal{J}$. In certain important cases, for example in the case of finite semigroups, the equality $\mathcal{D} = \mathcal{J}$ holds.

The previous equivalences $\mathcal{L}$, $\mathcal{R}$, $\mathcal{H}$, $\mathcal{D}$ and $\mathcal{J}$ are so-called *Green relations*.

We denote the $\mathcal{L}$-class ($\mathcal{R}$-class, $\mathcal{H}$-class, $\mathcal{D}$-class, $\mathcal{J}$-class) of an element a by $\mathcal{L}_a$ (resp. $\mathcal{R}_a$, $\mathcal{H}_a$, $\mathcal{D}_a$, $\mathcal{J}_a$). Since $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{J}$ are defined in terms of ideals, the inclusion order among these ideals induces pre-orders $\leq_{\mathcal{L}}$, $\leq_{\mathcal{R}}$, $\leq_{\mathcal{J}}$ on S, e.g. $a \leq_{\mathcal{J}} b$ if and only if $S^1aS^1 \subseteq S^1bS^1$. These pre-orders give partial orders on the partitions $S/\mathcal{L}$, $S/\mathcal{R}$ and $S/\mathcal{J}$ respectively.

We say that an $\mathcal{L}$-class ($\mathcal{R}$-class, $\mathcal{H}$-class, $\mathcal{D}$-class, $\mathcal{J}$-class) is *regular* if it contains some idempotent. A semigroup is called *regular* if any $\mathcal{D}$-class is regular. One can prove that if any $\mathcal{D}$-class is regular then also any $\mathcal{L}$-class and $\mathcal{R}$-class is regular. Note that any regular $\mathcal{H}$-class is a group and in the case when all $\mathcal{H}$-classes are regular we call the semigroup S a *union of groups* or a *completely regular semigroup*.

We turn our attention to the structure of bands.

Lemma 1.1.8 ([24]). *Let S be a band and $a, b, c \in S$ arbitrary elements. Then*
(i) $\quad a \leq_{\mathcal{L}} b \iff ab = a,$

(ii) $a \leq_{\mathcal{R}} b \iff ba = a$,
(iii) $a \leq_{\mathcal{J}} b \iff aba = a$,
(iv) $a \leq_{\mathcal{L}} b,\ b \leq_{\mathcal{J}} a \implies a \mathrel{\mathcal{L}} b$,
(v) $a \leq_{\mathcal{J}} b \implies ac \leq_{\mathcal{J}} bc,\ ca \leq_{\mathcal{J}} cb$.

Proof. The implications from right to left are trivial. If $a \leq_{\mathcal{L}} b$ then there is $s \in S^1$ such that $sb = a$. Hence

$$ab = (sb)b = sb = a$$

and we have the first observation. The property (ii) is dual. If $a \leq_{\mathcal{J}} b$ then there is a pair of elements $s, t \in S^1$ satisfying $a = sbt$. Hence

$$aba = (sbt)b(sbt) = (sbt)(sbt) = aa = a$$

by Lemma 1.1.4. The property (iv) follows directly from (i) and (iii). If $a \leq_{\mathcal{J}} b$, i.e. $a = aba$, then

$$(ac)bc(ac) = (abac)bc(abac) = (abac)(abac) = ac$$

again by Lemma 1.1.4. Analogically, we can derive the second part of the property (v). □

These easy properties lead to the well known fact that any band is a semilattice of rectangular bands.

Lemma 1.1.9 ([24]). *Let S be a band.*
(i) $\mathcal{J} = \mathcal{D}$,
(ii) *$\mathcal{J}$ is a congruence relation and $S/\mathcal{J}$ is a semilattice,*
(iii) *any $\mathcal{J}$-class is a subsemigroup of S which forms a rectangular band.*

For finite semigroups, instead of varieties one has to consider another natural classes, so-called *pseudovarieties*, which are closed under taking non-empty subsemigroups, morphic images and finite products. In fact, not every pseudovariety of semigroups is the class of finite members of some variety; for example the class of all finite groups forms a pseudovariety of semigroups (we denote it $\mathbf{G}$), although the only variety of semigroups containing $\mathbf{G}$ is the variety of all semigroups. Some natural subclasses of $\mathbf{G}$ (the class of all finite nilpotent groups $\mathbf{G_{NIL}}$ or the class of all finite abelian groups $\mathbf{AB}$ for example) also form pseudovarieties of semigroups.

There are certain natural operators on the pseudovarieties of semigroups. For an arbitrary pseudovariety of groups[1] $\mathbf{H}$ we denote by $\overline{\mathbf{H}}$ the class of all finite

[1] Any pseudovariety of groups is a pseudovariety of semigroups, because any non-empty subsemigroup of a finite group forms a subgroup.

semigroups whose all regular $\mathcal{H}$-classes are groups from the class **H**. For any pseudovariety of semigroups **V** we denote by **DV** the pseudovariety of all semigroups whose regular $\mathcal{D}$-classes lie in **V**. For example, if **O** denotes the class of orthodox semigroups (a semigroup is *orthodox* iff any product of two idempotents is an idempotent), then a semigroup S belongs to the class **DO** if and only if for any idempotent $e \in S$ the class $\mathcal{D}_e$ is a subsemigroup of S and any product of two idempotents from $\mathcal{D}_e$ is an idempotent. It is not hard to prove that $\mathbf{DO} \subseteq \mathbf{O}$. Another important example is the class **DA** of all finite semigroups whose regular $\mathcal{D}$-classes are aperiodic subsemigroups (a semigroup is *aperiodic* if it contains only trivial subgroups). For more information on pseudovarieties see [1].

In the case of bands (and idempotent monoids respectively) every pseudovariety consists of finite members of some variety. Hence the lattice of pseudovarieties of bands is the same as in the case of varieties. We will denote the pseudovarieties of bands in the same way as varieties, for example **SL** denotes the class of all finite semilattices, **NB** and **RB** are the pseudovarieties of normal and regular bands respectively.

Now we introduce another semigroups for which $\mathcal{J}$ is a congruence.

Definition 1.1.10. Let Y be a semilattice, $\{S_e \mid e \in Y\}$ be a set of disjoint semigroups from a certain class $\mathcal{S}$ and $\{\varphi_{e,f} : S_e \to S_f \mid e, f \in Y,\ f \leq e\}$ be a set of homomorphisms such that $\varphi_{e,e} = \mathrm{id}_{S_e}$ and $\varphi_{f,g} \circ \varphi_{e,f} = \varphi_{e,g}$ for any $g \leq f \leq e$. We define the multiplication $\times$ on the set $S = \bigcup_{e \in Y} S_e$ by the rule

$$a \times b = \varphi_{e,ef}(a) \cdot \varphi_{f,ef}(b),$$

where $e, f \in Y$ are such that $a \in S_e$ and $b \in S_f$ and $\cdot$ is the multiplication in the semigroup S_{ef}. The resulting groupoid $(S, \times)$ is denoted $(Y, S_e, \varphi_{e,f})$ and called a *strong semilattice of semigroups*.

It is not hard to see that $S = (Y, S_e, \varphi_{e,f})$ is a semigroup. If we take the class of left zero semigroups $\mathcal{LZ}$ for $\mathcal{S}$, then the resulting strong semilattice of left zero semigroups is a left normal band. In fact, one can prove the converse, i.e. that any left normal band is a strong semilattice of left zero semigroups. The same is true for right zero semigroups and right normal bands and because any rectangular band is a product of a right zero semigroup and a left zero semigroup, the strong semilattices of rectangular bands are exactly the normal bands.

Note that the previous construction of a strong semilattice of semigroups can be done also for an arbitrary band Y and an arbitrary class of groups $\mathcal{G}$. In this case it leads to a description of the structure of certain interesting semigroups, namely orthodox unions of groups for which the relation $\mathcal{H}$ is a congruence. One can prove that for any pseudovariety of bands **V** the supremum $\mathbf{V} \vee \mathbf{AB}$

in the lattice of all pseudovarieties of semigroups is the class of all finite strong **V**-bands of abelian groups. We will meet pseudovarieties like the pseudovariety of all strong regular bands of abelian groups $\mathbf{RB} \vee \mathbf{AB}$ in Chapter 5.

1.2 Unification Theory

The goal of this section is to fix the notation from unification theory. We are not able nor we want to supply overview articles like [7, 25] which are recommended for real introduction to unification theory. In fact, we have taken the basic definitions from [8] and modified notation for our purpose.

1.2.1 General Introduction

A *signature* $\mathcal{F}$ is a (finite or countably infinite) set of function symbols together with their arities. For an arbitrary set of variables X we denote by $\mathcal{T}(\mathcal{F}, X)$ the term algebra (or absolutely free algebra) generated in a signature $\mathcal{F}$ by X. The elements of $\mathcal{T}(\mathcal{F}, X)$ are called $\mathcal{F}$*-terms* or simply *terms* when $\mathcal{F}$ is known from the context. For two sets $X_1 \subset X_2$ we have also $\mathcal{T}(\mathcal{F}, X_1) \subseteq \mathcal{T}(\mathcal{F}, X_2)$. A special role is played by the terms from the set $\mathcal{T}(\mathcal{F}, \emptyset)$, so-called *ground terms*, i.e. terms from $\mathcal{T}(\mathcal{F}, X)$ which contain no variable. From this moment we fix X as a countable infinite set of variables.

A *substitution* σ is a mapping from the set of variables X to $\mathcal{T}(\mathcal{F}, X)$ which is almost everywhere equal to the identity. Such a mapping can be extended in a unique way to a morphism $\sigma : \mathcal{T}(\mathcal{F}, X) \to \mathcal{T}(\mathcal{F}, X)$ which is denoted by the same symbol. This means that substitutions correspond to endomorphisms of the free algebra $\mathcal{T}(\mathcal{F}, X)$ with finite domain — for a morphism $\sigma : \mathcal{T}(\mathcal{F}, X) \to \mathcal{T}(\mathcal{F}, X)$, the *domain* of σ is the set $\mathrm{Dom}(\sigma) = \{x \in X \mid \sigma(x) \neq x\}$. We say that a substitution σ is a *unifier* of two terms $s, t \in \mathcal{T}(\mathcal{F}, X)$ when $\sigma(s) = \sigma(t)$; it is the *most general unifier* if for every unifier θ of s and t there is a substitution ρ such that $\theta = \rho \circ \sigma$.

Definition 1.2.1. An instance of the *syntactic unification problem* is a pair of terms $(s, t) \in \mathcal{T}(\mathcal{F}, X) \times \mathcal{T}(\mathcal{F}, X)$ and the question is whether there exists a unifier of terms s and t.

We denote the instance (s, t) of the syntactic unification problem by $s =^? t$ and call it also an *equation*. Unifiers are sometimes called *solutions*.

The basic result which was a starting point of unification theory is the fact that any instance $s =^? t$ which has a unifier has also the most general unifier [42]. Moreover, the syntactic unification problem can be decided and the most general unifier can be found in linear time [38].

Equational unification is a generalization of syntactic unification in which semantic properties of function symbols are taken into account.

An *equational theory* is a set of identities $\mathrm{E} \subseteq \mathcal{T}(\mathcal{F}, X) \times \mathcal{T}(\mathcal{F}, X)$. The identities from E are sometimes called *axioms* of the theory E. We denote by $=_{\mathrm{E}}$ the least congruence relation of the term algebra $\mathcal{T}(\mathcal{F}, X)$ that is closed under substitutions (so-called *fully invariant congruence*) and contains E. From another point of view, we can consider the class $\mathcal{V} = \mathrm{Mod}(\mathrm{E})$ consisting of all algebras of signature $\mathcal{F}$ which satisfy all identities from the set E. The class $\mathcal{V}$ is called a *variety*. Now, the fully invariant congruence $=_{\mathrm{E}} \subseteq \mathcal{T}(\mathcal{F}, X) \times \mathcal{T}(\mathcal{F}, X)$ is the set of all identities which are satisfied in all algebras in $\mathcal{V}$ and the factoralgebra $\mathcal{T}(\mathcal{F}, X)/{=_{\mathrm{E}}}$ is a free algebra over the set X in the variety $\mathcal{V}$. In the literature the fully invariant congruence $=_{\mathrm{E}}$ is also sometimes called equational theory.

For a given theory E, we denote by Sig(E) the set of all function symbols occurring in E. Symbols from the set $\mathcal{F} \setminus \mathrm{Sig}(\mathrm{E})$ are called *free*. An equational theory E is called *trivial* if $x =_{\mathrm{E}} y$ for two different variables x and y from X, i.e. if there are only one-element algebras in the corresponding variety. Two theories E and E′ of signature $\mathcal{F}$ are *equivalent* if and only if for any pair of terms $s, t \in \mathcal{T}(\mathcal{F}, X)$ we have $s =_{\mathrm{E}} t$ iff $s =_{\mathrm{E}'} t$.

By a *system of equations* S we mean a finite set $S = \{s_1 =^? t_1, \ldots, s_n =^? t_n\}$, where $s_1, t_1, \ldots, s_n, t_n \in \mathcal{T}(\mathcal{F}, X)$. An E-*unifier* of the system S is a substitution $\sigma : \mathcal{T}(\mathcal{F}, X) \to \mathcal{T}(\mathcal{F}, X)$ such that $\sigma(s_i) =_{\mathrm{E}} \sigma(t_i)$ for all $i = 1, \ldots, n$. The set of all E-unifiers of the system S is denoted by $\mathcal{U}_{\mathrm{E}}(S)$. We say that the system S is E-*satisfiable* whenever $\mathcal{U}_{\mathrm{E}}(S) \neq \emptyset$.

Remark 1.2.2. Let $\mathrm{E} \subseteq \mathrm{E}'$ be two theories and S be a system of equations. If a system S is E-satisfiable then it is also E′-satisfiable, because any E-unifier of the system S is also an E′-unifier of S.

Definition 1.2.3. Let E be an equational theory and $\mathcal{F}$ be a signature containing Sig(E). Given a system of equations $S = \{s_1 =^? t_1, \ldots, s_n =^? t_n\}$ with terms $s_1, \ldots, s_n, t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, X)$ as an instance of the E-*unification problem*, the task is to decide whether this system S is E-satisfiable.

From the point of view of the previous definition, the syntactic unification problem is just the $\emptyset$-unification problem.

In the literature various equational theories are studied. The following basic identities of a binary function symbol f are in the center of our interest.

Associativity (A)	$f(x, f(y, z)) \doteq f(f(x, y), z)$
Idempotency (I)	$f(x, x) \doteq x$
Existence of Unit (U)	$f(1, x) \doteq f(x, 1) \doteq x$

It is usual to distinguish the following variants of the E-unification problem in the literature.

Definition 1.2.4. An E-unification problem is an *elementary* E-unification problem if and only if $\mathcal{F} = \mathrm{Sig}(\mathrm{E})$; it is an E-unification problem *with constants* if and only if $\mathcal{F} \setminus \mathrm{Sig}(\mathrm{E})$ consists of infinitely many constant symbols; and it is a *general* E-unification problem if and only if $\mathcal{F} \setminus \mathrm{Sig}(\mathrm{E})$ contains infinitely many function symbols of every arity.

Remark 1.2.5. Our formulation of the variants of the E-unification problem is not standard in the literature, where it is usual to use a less restrictive definition. In the case of the E-unification problem with constants we can replace the condition of the existence of infinitely many constants with the condition that the free algebra $\mathcal{T}(\mathcal{F}, \emptyset)/{=_\mathrm{E}}$ is infinite. Then the ground terms from the set $\mathcal{T}(\mathcal{F}, \emptyset)$ (corresponding to different elements of $\mathcal{T}(\mathcal{F}, \emptyset)/{=_\mathrm{E}}$) can play the role of constants. Hence the E-unification problem with constants can be reduced to the elementary E-unification problem whenever $\mathcal{T}(\mathcal{F}, \emptyset)/{=_\mathrm{E}}$ is infinite.

Also in the case of the general E-unification problem one can construct arbitrary new operations from the original function symbols. For example, for a free binary function symbol f we can consider the term $f(f(x, y), z)$ which can be viewed as a new ternary operation $g(x, y, z)$. On the other hand, from a free ternary function symbol g and a free constant c we can construct a binary operation, for example $g(x, y, c)$.

For these reasons we will sometimes say that $\mathcal{F}$ contains *enough* free (constant) function symbols and speak about the E-unification problem with constants and the general E-unification problem although the conditions from the definition will not be satisfied.

Remark 1.2.6. For syntactic unification, solving a system of equations can be directly reduced to solving a single equation. For equational unification, the same holds if one considers general unification. In fact, if $f \in \mathcal{F}$ is a free n-ary function symbol, then the system $\{s_1 =^? t_1, \ldots, s_n =^? t_n\}$ has the same set of E-unifiers as $\{f(s_1, \ldots, s_n) =^? f(t_1, \ldots, t_n)\}$. For elementary unification and for unification with constants, there may be significant differences.

Remark 1.2.7. If we want to solve the E-unification problem, we have to know a solution of the *identity problem* of the theory E, i.e. an effective description of the congruence relation $=_\mathrm{E}$. In fact, the identity problem can be reduced to the E-unification problem with constants. If we consider a pair of terms s and t and a substitution σ which maps variables to different free constants, then $s =_\mathrm{E} t$ iff $\sigma(s) =_\mathrm{E} \sigma(t)$. Because $\sigma(s)$ and $\sigma(t)$ are ground terms, we have $\sigma(s) =_\mathrm{E} \sigma(t)$ iff there is an E-unifier of the equation $\sigma(s) =^? \sigma(t)$.

Remark 1.2.8. In general, if we ask for existence of a solution of a given system S and if $\mathcal{T}(\mathcal{F}, \emptyset) \neq \emptyset$, i.e. $\mathcal{F}$ contains some constant symbol, then we can consider only substitutions which map variables to ground terms (so called *ground substitutions*). It is because any solution of the system S can be prolonged to a ground solution.

The distinction between the three variants of unification problems can be substantial as the following example shows.

Example 1.2.9. Let Sig(E) consist of one binary operation symbol, which is idempotent, i.e. $\mathrm{E} = \mathrm{I} = \{f(x, x) \doteq x\}$ where $f \in \mathcal{F}$. Then the elementary I-unification problem is trivial because any system S has an I-unifier — we can choose any term $u \in \mathcal{T}(\mathcal{F}, X)$ and define the substitution σ by the rule $\sigma(x) = u$ for any variable $x \in X$ which occurs in S. We can not say this for the other problems. Note that Kapur and Narendran [27, 28] proved that both the I-unification problem with constants and the general I-unification problem are NP-complete.

It was shown [12] that there exists an equational theory for which the elementary unification problem is decidable, but the unification problem with constants is undecidable. It is still an open problem whether there is such a gap between the unification with constants and the general unification. We recall the result based on the so-called combination method (see [6]) which suggests that these two problems could be much closer.

Definition 1.2.10. Let $\mathcal{F} \setminus \mathrm{Sig(E)}$ contain only constant function symbols. An instance of the E-unification problem *with linear constant restriction* consists of a system S and a strict linear order $<$ on the set Γ_S of all variables and all free constant symbols occurring in S. The question is whether there exists an E-unifier σ of S that satisfies

$$x < c \quad \text{implies} \quad c \text{ does not occur in } \sigma(x) \tag{1.1}$$

for all variables x and free constants c in Γ_S.

Note that the assumption that $\mathcal{F} \setminus \mathrm{Sig(E)}$ contains only constant function symbols is not restrictive, because for any free n-ary function symbol f one can add the trivial identity $f(x_1, \ldots, x_n) \doteq f(x_1, \ldots, x_n)$ to E.

It is easy to see that the E-unification problem with linear constant restriction is a generalization of the E-unification problem with constants; for an instance of the E-unification problem with constants one can consider an arbitrary strict linear order $<$ such that $c < x$ for any free constant c and any variable x. Note also that the $\emptyset$-unification problem with linear constant restriction can be solved in linear time, because we have to only check whether the most general unifier satisfies condition (1.1).

On the other hand, the E-unification problem with linear constant restriction can be reduced to the general E-unification problem in the following way. For any free constant symbol c occurring in an instance $(S, \Gamma_S, <)$ of the E-unification problem with linear constant restriction, let $x_1, \ldots, x_k$ be all variables less than c in the order $<$ and consider an arbitrary free k-ary function symbol g_c which does not occur in the instance. Then we replace any occurrence of c in S with $g_c(x_1, \ldots, x_k)$. One can prove that the resulting system has an E-unifier if and only if the original instance has an E-unifier satisfying condition (1.1).

The proof of the following result is based on combination of the algorithms for the unification problems for disjoint theories.

Proposition 1.2.11 ([6]). *Let* $\mathrm{E}_1, \ldots, \mathrm{E}_n$ *be non-trivial theories over disjoint signatures. If the* E_i*-unification problem with linear constant restriction is decidable (in NP) for* $i = 1, \ldots, n$*, then the elementary* $(\mathrm{E}_1 \cup \cdots \cup \mathrm{E}_n)$*-unification problem is decidable (in NP).*

If we consider the general E-unification problem and in the previous proposition we let $n = 2$, $\mathrm{E}_1 = \mathrm{E}$, $\mathcal{F}_1 = \mathrm{Sig}(\mathrm{E})$, $\mathcal{F}_2 = \mathcal{F} \setminus \mathcal{F}_1$ and for any k-ary function symbol f put the identity $f(x_1, \ldots, x_k) \doteq f(x_1, \ldots, x_k)$ to E_2, we obtain the following interesting consequence.[2]

Proposition 1.2.12 ([6]). *Let* E *be a non-trivial equational theory. The general* E*-unification problem is decidable (in NP) if and only if the* E*-unification problem with linear constant restriction is decidable (in NP).*

Remark 1.2.13. If in Proposition 1.2.11 one of the theories E_i is trivial then the theory $(\mathrm{E}_1 \cup \cdots \cup \mathrm{E}_n)$ is trivial. For a trivial theory E any system is satisfiable because the equality $s =_{\mathrm{E}} t$ holds for any pair of terms s, t. For this reasons we consider only non-trivial theories in the whole thesis.

In Chapter 4 we will work with theories which are a disjoint union of certain basic theories. For this situation we introduce another notation.

Let E be a theory of signature $\mathcal{F}$. We say that a function symbol $f \in \mathcal{F}$ is *independent* (in the theory E) if and only if the theory E is equivalent to the theory $\mathrm{E}_f \cup \mathrm{E}'$, where the set E_f consists of all axioms from E which contain only the function symbol f and the set E' consists of all axioms from E which do not contain the function symbol f. We say that the theory E *consists of independent function symbols* if each symbol from $\mathcal{F}$ is independent in E.

[2]We put the trivial identities to E_2 because we transform the general E-unification problem to the elementary $(\mathrm{E}_1 \cup \mathrm{E}_2)$-unification problem.

We recall an important modification of unification, namely matching. In E-matching we ask for an E-matcher of a pair of terms s, t, i.e. a substitution σ such that $\sigma(s) =_{\mathrm{E}} t$. The E-matching problem can be viewed as a special case of the E-unification problem with constants in the same way as the identity problem does (Remark 1.2.7). For this reason we do not distinguish between elementary matching and matching with constants, which are equivalent, but only between matching and general matching. The precise definition follows.

Definition 1.2.14. Let E be an equational theory and $\mathcal{F}$ be a signature containing Sig(E) and enough free function symbols. Given a system of equations $S = \{s_1 =^? t_1, \ldots, s_n =^? t_n\}$ with terms $s_1, \ldots, s_n \in \mathcal{T}(\mathcal{F}, X)$ and ground terms $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, \emptyset)$ as an instance of the *general* E-*matching problem*, the task is to decide whether there is a substitution $\sigma : \mathcal{T}(\mathcal{F}, X) \to \mathcal{T}(\mathcal{F}, \emptyset)$ such that $\sigma(s_i) =_{\mathrm{E}} t_i$ for all $i = 1, \ldots, n$. We speak about the E-*matching problem* if $\mathcal{F} \setminus \mathrm{Sig}(\mathrm{E})$ contains only constant symbols.

If we are not only interested in existence of an E-unifier but we want to know all E-unifiers, then the following generalization of the notion of the most general unifier is useful. Let S be a system of equations with the set of all E-unifiers $\mathcal{U}_{\mathrm{E}}(S)$. We define a pre-order $\preceq_E^S$ on the set $\mathcal{U}_{\mathrm{E}}(S)$ in a natural way: for $\sigma, \theta \in \mathcal{U}_{\mathrm{E}}(S)$ we write $\sigma \preceq_E^S \theta$ iff there is a substitution $\rho : \mathcal{T}(\mathcal{F}, X) \to \mathcal{T}(\mathcal{F}, X)$ such that $\theta(x) =_{\mathrm{E}} \rho(\sigma(x))$ for any variable x which occurs in the system S. The corresponding equivalence relation is denoted $\simeq_E^S$. A *minimal complete set of* E-*unifiers* $\mathcal{MU}_{\mathrm{E}}(S)$ is a set of substitutions such that

- $\mathcal{MU}_{\mathrm{E}}(S) \subseteq \mathcal{U}_{\mathrm{E}}(S)$,
- for each $\theta \in \mathcal{U}_{\mathrm{E}}(S)$ there exists $\sigma \in \mathcal{MU}_{\mathrm{E}}(S)$ such that $\sigma \preceq_E^S \theta$,
- for all $\sigma, \sigma' \in \mathcal{MU}_{\mathrm{E}}(S)$, $\sigma \preceq_E^S \sigma'$ implies $\sigma = \sigma'$.

If the system S is not E-satisfiable, then the empty set is a minimal complete set of E-unifiers of S. It is easy to see that a minimal complete set of E-unifiers is unique up to the equivalence $\simeq_E^S$ (if it exists). This makes sure that the following definition of the unification type of the system S and the unification theory E is correct.

Definition 1.2.15. Let E be an equational theory and S be a E-satisfiable system of equations. The system S has *type unitary* (*finitary*, *infinitary*) iff it has a minimal complete set of E-unifiers of cardinality 1 (finite cardinality, infinite cardinality). If S does not have a minimal complete set of E-unifiers, then it is of *type zero*. We denote unification types by 1, ω, ∞ and 0 and order these types as follows: $1 < \omega < \infty < 0$. The *unification type* of E (with respect to $\mathcal{F}$) is the maximal type of E-satisfiable systems.

For example, unification type is unitary for the syntactic unification problem [42], because there is the most general unifier as we mentioned. The elementary AC-unification problem for the theory of a single associative and commutative function symbol is of finitary type [44]. An example of infinitary unification type is the elementary A-unification problem for the theory of a single associative function symbol; for example, the equation $f(x,y) =^? f(y,x)$ has minimal solutions $\sigma(x) = z^k, \sigma(y) = z^l$ where k and l are relatively prime (see [40]). The first unification problem of nullary unification type was described by Fages and Huet in [16]. Then Schmidt-Schauss [43] and Baader [2] gave a more natural E-unification problem of type zero, namely the AI-unification problem, which is in the center of our interest.

For a comprehensive overview not only on the unification type see [7, 8] and for complexity aspects see [28].

1.2.2 Reformulation of the AI-Unification Problem

In the thesis we will discuss unification problems for theories of associative and idempotent function symbols, so-called *ai-symbols*. Each theory of a single ai-symbol corresponds to a certain variety $\mathcal{V}$ of bands; we speak about a *$\mathcal{V}$-symbol*, e.g. an $\mathcal{RCB}$-symbol is also called a *rectangular symbol* and an $\mathcal{LZ}$-symbol is called a *left zero symbol*. As we saw in Example 1.2.9, the elementary unification problems for the theories of ai-symbols are trivial.

Lemma 1.2.16. *Let* E *consist of independent ai-symbols. Then the elementary* E*-unification problem is trivially decidable because any system of equations has a solution.*

Hence the crucial problem is the unification problem with constants because we are able to use Proposition 1.2.12 for the general unification problem as we will see in Section 3.4. From these reasons we slightly rewrite previous general definitions for the case of unification with constants. Assume that $\mathcal{F} = \mathcal{C} \cup \{\cdot\}$, where $\mathcal{C}$ is a set of free constants and $\cdot$ is an independent $\mathcal{V}$-symbol for a given variety of bands $\mathcal{V}$. Because the symbol $\cdot$ is associative, we prefer to work with the congruence relation $\sim_{\mathcal{V},\mathcal{C}}$ of the free semigroup $\mathcal{C}^+$ (and the free monoid $\mathcal{C}^*$ respectively) corresponding to the variety $\mathcal{V}$ instead of the congruence relation $=_{\mathrm{E}}$ of the free groupoid $\mathcal{T}(\mathcal{F}, X)$ (see Remark 1.2.8). We will also assume that the set of all constant symbols $\mathcal{C}$ and the set of variables X are finite and they are part of an instance of the problem.

Let $\mathcal{C}$ be a finite set of *constants* (constant symbols) and X be a finite set of *variables* such that $\mathcal{C} \cap \mathrm{X} = \emptyset$. A *word equation* $L =^? R$ is a pair $(L, R) \in (\mathcal{C} \cup \mathrm{X})^* \times (\mathcal{C} \cup \mathrm{X})^*$. We say that an equation $L =^? R$ is *trivial* if $L = R$. A *system of word equations* S is a finite set of equations, i.e. $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$

for $n \geq 0$. A *solution* of the system S in a variety of idempotent monoids $\mathcal{V}$ is a homomorphism $\alpha : (\mathcal{C} \cup \mathrm{X})^* \to \mathcal{C}^*$ which behaves as an identity on the set $\mathcal{C}$ and equates all the equations of S, i.e. $\alpha(L_i) \sim_{\mathcal{V},\mathcal{C}} \alpha(R_i)$ for all $i = 1, \ldots, n$. Such a homomorphism is fully established by a mapping $\alpha|_{\mathrm{X}} : \mathrm{X} \to \mathcal{C}^*$. We will use the same symbol α for this mapping and its unique extension to a homomorphism $\alpha : (\mathcal{C} \cup \mathrm{X})^* \to \mathcal{C}^*$. If we want to work with a variety of bands, we consider the free semigroups instead of the free monoids. More precisely, a word equation $L =^? R$ is a pair $(L, R) \in (\mathcal{C} \cup \mathrm{X})^+ \times (\mathcal{C} \cup \mathrm{X})^+$ and a solution of the system S in a variety of bands $\mathcal{V}$ is a homomorphism $\alpha : (\mathcal{C} \cup \mathrm{X})^+ \to \mathcal{C}^+$ which behaves as an identity on the set $\mathcal{C}$ etc.

We introduce the size of a solution α as $\mathrm{size}(\alpha) := \max_{x \in \mathrm{X}} |\alpha(x)|$ and say that α is the *shortest solution* if and only if for any solution β of the system we have $\mathrm{size}(\alpha) \leq \mathrm{size}(\beta)$.

Remark 1.2.17. Let $\mathcal{V} \subseteq \mathcal{U}$ be two varieties of bands. If the system of word equations S is satisfiable in the variety $\mathcal{U}$ then it is also satisfiable in the variety $\mathcal{V}$, because any solution of the system S in the variety $\mathcal{U}$ is also a solution in the variety $\mathcal{V}$ (see Remark 1.2.2).

A system of word equations $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ is called a *matching system* if $R_i \in \mathcal{C}^*$ and a *pattern equation system* if $L_i \in \mathrm{X}^*$ and $R_i \in \mathcal{C}^*$ for all $1 \leq i \leq n$.

Definition 1.2.18. Let $\mathcal{V}$ be a variety of idempotent monoids or bands. Given a system of word equations $\{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ with a set of variables X and a set of constants $\mathcal{C}$ as an instance of the $\mathcal{V}$-UNIFICATION problem, the task is to decide whether this system has a solution in the variety $\mathcal{V}$. The $\mathcal{V}$-MATCHING problem is the modification of the $\mathcal{V}$-UNIFICATION problem where only systems with $R_i \in \mathcal{C}^*$ are considered. Sometimes we speak about the UNIFICATION (resp. MATCHING) problem for the variety $\mathcal{V}$ instead of the $\mathcal{V}$-UNIFICATION (resp. $\mathcal{V}$-MATCHING) problem.

In the case of the $\mathcal{B}$-UNIFICATION and $\mathcal{IM}$-UNIFICATION problems we will use the following standard terminology.

Definition 1.2.19. The $\mathcal{IM}$-UNIFICATION problem and the $\mathcal{B}$-UNIFICATION problem are called the AIU-UNIFICATION problem and the AI-UNIFICATION problem respectively. Also the $\mathcal{IM}$-MATCHING problem and the $\mathcal{B}$-MATCHING problem are called the AIU-MATCHING problem and the AI-MATCHING problem respectively.

In the case of the $\mathcal{V}$-MATCHING problem we can restrict ourselves to pattern equation systems since for every instance (i.e. a matching system) it is easy to construct an equivalent pattern equation system: just notice that an equation of the form $L_1aL_2 =^? A$, where $a \in \mathcal{C}$, can be transformed into equations $L_1xL_2 =^? A$ and $x =^? a$, where x is a new variable.

We do not change the definition of general unification problems, we only introduce different notation. For a theory E of a single $\mathcal{V}$-symbol we will speak about the general $\mathcal{V}$-unification problem instead of the general E-unification problem. We also say a $\mathcal{V}$-unifier instead of an E-unifier.

Now we summarize the known results concerning AI-unification.

First of all, the AI-UNIFICATION problem is decidable because finitely generated bands are finite; it is a consequence of Proposition 1.1.3. As we will see in Chapter 3, it is possible to use Proposition 1.2.12 to obtain the following result.

Proposition 1.2.20 ([5]). *The general AIU-unification problem and the general AI-unification problem are decidable.*

The only already known result concerning complexity of the AI-unification problem is the following. (Note that this result is referred inaccurately in the literature.)

Proposition 1.2.21 ([26]). *The general AIU-unification problem and the general AI-unification problem are NP-hard.*

As we mentioned, the AI-UNIFICATION problem is of interest for unification theory because it was the first natural example of an equational theory known to be of unification type zero. It was proved independently by Schmidt-Schauss [43] and Baader [2]. Schmidt-Schauss's result is given for the AI-UNIFICATION problem, whereas Baader showed type zero for the elementary AI-unification problem.

Proposition 1.2.22 ([2]). *The elementary AI-unification problem is of unification type zero.*

If we look for known results about theories with a single associative and idempotent function symbol satisfying certain additional identities, the case of a commutative symbol, i.e. the $\mathcal{SL}$-unification problem, was discussed first. (The following result is contained in Chapter 4.)

Proposition 1.2.23 ([28]). *The $\mathcal{SL}$-*UNIFICATION *and $\mathcal{CIM}$-*UNIFICATION *problems are decidable in polynomial time.*

Note that it was proved in [22] that these problems are P-complete. In the case of the general unification we have the following result.

Proposition 1.2.24 ([26]). *The general $\mathcal{SL}$-unification problem and the general $\mathcal{CIM}$-unification problem are NP-complete.*

Actually, Kapur and Narendran proved in paper [26] also NP-completeness of the general $\mathcal{SL}$-matching problem.

As the unification type of the AI-unification problem is zero, the same is true for almost all varieties of bands (see Figure 1.1).

Proposition 1.2.25 ([3]). *Let $\mathcal{V}$ be a variety of bands. The $\mathcal{V}$-UNIFICATION problem has unitary type if $\mathcal{V} \in [\mathcal{T}, \mathcal{RCB}]$, finitary type if $\mathcal{V} \in [\mathcal{SL}, \mathcal{NB}]$ and nullary type otherwise.*

In fact, Baader [3] studied unification types of single equations, but the previous formulation can be easily obtained from his results. Moreover, his equations of unification type zero for $\mathcal{V} \nsubseteq \mathcal{NB}$ are variable-free, i.e. Baader proved the same statement for the elementary unification problems.

Chapter 2

AI-Matching Problem

In the first section, we prove NP-hardness of the AIU-Matching problem. The proof uses reduction from the NP-complete problem 3-SAT. In the rest of the chapter we prove that the AIU-Matching problem belongs to NP. In order to do this, we employ the term rewriting system of J. Siekmann and P. Szabo [46] and show that any solvable matching system has a solution of polynomial length. There are no new concepts in this chapter compared to paper [32], which did not contain all technical details. In the original proofs we have considered a pattern equation systems, but these proofs work as well for the AIU-Matching problem, where constants occur on both sides of equations.

In the whole chapter the symbol $\sim$ stands for $\sim_{\mathcal{IM},\mathcal{C}}$.

2.1 NP–hardness of the AI-Matching Problem

In this section we show that the AI-Matching problem is NP–hard. We will use a reduction from the NP-complete problem 3-SAT (see [37]) to prove NP-hardness of our problems. An instance of 3-SAT is a conjunction

$$\Phi \equiv \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$$

of clauses and each clause Φ_i, $1 \leq i \leq n$, is of the form

$$l_1 \vee l_2 \vee l_3$$

where l_j, $1 \leq j \leq 3$, is a *literal* (i.e. l_j is a *variable* from the set *Var*, possibly negated — we call it *positive* resp. *negative* literal). A *valuation* is a mapping $v : Var \rightarrow \{T, F\}$. Every valuation extends naturally to Φ and we say that Φ is *satisfiable* if and only if there exists a valuation v such that $v(\Phi) = T$.

Proposition 2.1.1. *The* AIU-Matching *problem is NP–hard.*

Proof. We use reduction from the 3-SAT problem. Let Φ be an instance of the 3-SAT problem. For a formula Φ we construct a pattern equation system such that the system is satisfiable if and only if Φ is satisfiable. The system will consist of the following sets of equations (2.1 – 2.6) and

$$\mathcal{C} = \{a, b, c\}, \quad \mathrm{X} = \{x, s_1^x, t_1^x, s_2^x, t_2^x \mid x \in \mathit{Var} \cup \overline{\mathit{Var}}\} \cup \{y_a, y_b, y_c\}$$

where $\overline{\mathit{Var}} = \{\overline{x} \mid x \in \mathit{Var}\}$ is a disjoint copy of Var. For the constants a, b and c there are three equations

$$y_a =^? a, \quad y_b =^? b, \quad y_c =^? c. \tag{2.1}$$

We define $\widetilde{x} = x$ if x is a positive literal, $\widetilde{\neg x} = \overline{x}$ if $\neg x$ is a negative literal and for all clauses $\Phi_i \equiv l_1 \vee l_2 \vee l_3$ we have the equation

$$y_a \widetilde{l_1} \widetilde{l_2} \widetilde{l_3} y_a =^? aba, \tag{2.2}$$

for each $x \in \mathit{Var}$ we add the equations

$$y_b x \overline{x} y_b =^? bab, \tag{2.3}$$

$$y_a x \overline{x} y_a =^? aba \tag{2.4}$$

and finally for each $x \in \mathit{Var} \cup \overline{\mathit{Var}}$ we have the following equations:

$$s_1^x x t_1^x =^? acb, \quad s_1^x y_c =^? ac, \tag{2.5}$$

$$s_2^x x t_2^x =^? bca, \quad s_2^x y_c =^? bc. \tag{2.6}$$

The intuition behind the construction is following. If a variable x is true then $x = b$ and if x is false then $x = a$. The second equation ensures that at least one literal in each clause is true and the other equations imply consistency, i.e. a literal and its negation cannot be both true (false). In particular, the equation (2.3) means that at least one of x and $\overline{x}$ contains a. Similarly for b and (2.4). The last two equations make sure that a variable $x \in \mathit{Var} \cup \overline{\mathit{Var}}$ cannot contain both a and b.

Suppose that Φ is satisfiable, i.e. there is a valuation v such that $v(\Phi) = T$. Then we show that α defined below is a solution of our system. Let us state

$$\alpha(y_a) = a, \quad \alpha(y_b) = b, \quad \alpha(y_c) = c$$

and for all $x \in \mathit{Var}$ such that $v(x) = T$ let

$$\alpha(x) = b, \quad \alpha(\overline{x}) = a$$

$$\alpha(s_1^x) = ac, \quad \alpha(t_1^x) = b, \quad \alpha(s_1^{\overline{x}}) = a, \quad \alpha(t_1^{\overline{x}}) = cb$$
$$\alpha(s_2^x) = b, \quad \alpha(t_2^x) = ca, \quad \alpha(s_2^{\overline{x}}) = bc, \quad \alpha(t_2^{\overline{x}}) = a$$

and if $v(x) = F$ then

$$\alpha(x) = a, \quad \alpha(\overline{x}) = b$$
$$\alpha(s_1^x) = a, \quad \alpha(t_1^x) = cb, \quad \alpha(s_1^{\overline{x}}) = ac, \quad \alpha(t_1^{\overline{x}}) = b$$
$$\alpha(s_2^x) = bc, \quad \alpha(t_2^x) = a, \quad \alpha(s_2^{\overline{x}}) = b, \quad \alpha(t_2^{\overline{x}}) = ca.$$

Checking that α is a solution is a routine. The only interesting equation is (2.2). This equation is also satisfied by α since we have the assumption that under the valuation v there is at least one true literal in each clause.

Let us suppose that α is an arbitrary solution of our system and we find a valuation that satisfies Φ. The equation (2.3) implies that $\mathsf{c}(\alpha(x)) \subseteq \{a, b\}$ for all $x \in \mathit{Var} \cup \overline{\mathit{Var}}$. We will conclude that it is not possible that $\mathsf{c}(\alpha(x)) = \{a, b\}$.

Suppose that it is the case and using the equations (2.5) we get that $\alpha(x)$ does not begin with the constant a. For the moment assume that $\alpha(x)$ begins with a. We have $ac = 0(acb) \sim 0(\alpha(s_1^x x t_1^x))$ and from (2.1) and (2.5) we get that $a \in \mathsf{c}(\alpha(s_1^x)) \subseteq \{a, c\}$. If $\mathsf{c}(\alpha(s_1^x)) = \{a\}$ then $\mathsf{c}(0(\alpha(s_1^x x t_1^x))) = \{a, b\}$ whereas $\mathsf{c}(0(acb)) = \{a, c\}$, which is a contradiction. Otherwise we have $\mathsf{c}(\alpha(s_1^x)) = \{a, c\}$ and we get

$$0(\alpha(s_1^x x t_1^x)) \sim aca \not\sim ac = 0(acb).$$

By the similar arguments and using the equations (2.6) we get that $\alpha(x)$ does not begin with the constant b. This yields that there are just three possibilities for $\alpha(x)$, namely $\alpha(x) \sim a$, $\alpha(x) \sim b$ or $\alpha(x) = \epsilon$.

By the equations (2.3) and (2.1) we know that for all $x \in \mathit{Var}$ at least $\alpha(x) \sim a$ or $\alpha(\overline{x}) \sim a$. The equation (2.4) implies that either $\alpha(x) \sim b$ or $\alpha(\overline{x}) \sim b$. Hence $\alpha(x) \neq \epsilon$ for all $x \in \mathit{Var} \cup \overline{\mathit{Var}}$. Similarly for each clause, the equation (2.2) with (2.1) gives that there is j, $1 \leq j \leq 3$, such that $\alpha(\widetilde{l_j}) \sim b$. Let us finally define the valuation v as $v(x) = T$ if $\alpha(x) \sim b$ and $v(x) = F$ if $\alpha(x) \sim a$ for each $x \in \mathit{Var}$. The valuation is consistent and it holds that $v(\Phi) = T$.

This is enough to demonstrate that the AIU-Matching problem is NP–hard since the reduction can be done effectively in polynomial time. □

It is easy to see that the same proof works in the case of the AI-Matching problem.

Corollary 2.1.2. *The* AI-Matching *problem is NP–hard.*

Because matching problem is a special case of unification problem we immediately obtain the following consequence.

Corollary 2.1.3. *The* AIU-Unification *problem and the* AI-Unification *problem are NP–hard.*

2.2 Rewriting System for Idempotent Monoids

In this section we summarize several properties of the rewriting system by Siekmann and Szabo in [46] and prove some technical lemmas. First of all we have to give some definitions and results concerning rewriting systems as it can be found e.g. in [4, 11].

A *rewriting system* R over $\mathcal{C}$ is a subset of $\mathcal{C}^* \times \mathcal{C}^*$. The elements of R will be called *rules*. Having such a system R we can define a *rewrite relation* $\rightarrow \subseteq \mathcal{C}^* \times \mathcal{C}^*$ in the following way:

$$\forall p, q \in \mathcal{C}^* : p \rightarrow q \text{ iff } \exists (u, v) \in R, s, t \in \mathcal{C}^* : p = sut, q = svt.$$

The elements (u, v) of R will be often written as $u \rightarrow v$. The reflexive transitive closure of $\rightarrow$ is denoted by $\rightarrow^*$. For a word $q \in \mathcal{C}^*$ we write $q \not\rightarrow$ iff there is no q' such that $q \rightarrow q'$ and we say that q is in a *normal form*. We define the set of normal forms of $p \in \mathcal{C}^*$ as $\langle p \rangle = \{q \mid p \rightarrow^* q \not\rightarrow\}$. We say that R (resp. the relation $\rightarrow$) is *terminating* if and only if there is no infinite sequence $p_1, p_2, p_3, \ldots \in \mathcal{C}^*$ such that $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow \ldots$. For a pair $p, q \in \mathcal{C}^*$ we write $p \downarrow q$ if and only if there exists $r \in \mathcal{C}^*$ such that $p \rightarrow^* r$, $q \rightarrow^* r$. The system R (resp. the relation $\rightarrow$) is *confluent* iff

$$\forall p, p_1, p_2 \in \mathcal{C}^* : p \rightarrow^* p_1, p \rightarrow^* p_2 \implies p_1 \downarrow p_2.$$

The system R (resp. the relation $\rightarrow$) is *locally confluent* iff

$$\forall p, p_1, p_2 \in \mathcal{C}^* : p \rightarrow p_1, p \rightarrow p_2 \implies p_1 \downarrow p_2.$$

The following well known lemma shows the relationship between confluence and local confluence.

Lemma 2.2.1 ([11]). *Let R be a terminating rewriting system. Then R is confluent if and only if R is locally confluent.*

It is easy to see that if R is a confluent and terminating rewriting system, then a word $p \in \mathcal{C}^*$ has exactly one normal form, i.e. $\langle p \rangle = \{q\}$ for some q, and in such a case we simply write $\langle p \rangle = q$.

Example 2.2.2. Let $\{xx \rightarrow x \mid x \in \mathcal{C}^*, \mathsf{c}(x) \neq \emptyset\}$ be a rewriting system over $\mathcal{C}$. Then this system is terminating but it is not confluent. For $p = ababcbabc$ we have

$$p = (ab)(ab)cbabc \rightarrow abcbabc$$

and

$$p = a(babc)(babc) \rightarrow (ab)(ab)c \rightarrow abc$$

where both $abcbabc$ and abc are in normal forms. It is easy to see that $\langle p \rangle = \{abc, abcbabc\}$.

In this chapter we will exploit the rewriting system by Siekmann and Szabo in [46].

Proposition 2.2.3 ([46]). *The rewriting system*

$$\{xx \to x \mid x \in \mathcal{C}^*, \mathsf{c}(x) \neq \emptyset\} \cup \{uvw \to uw \mid u,v,w \in \mathcal{C}^*, \emptyset \neq \mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)\}$$

is confluent and terminating. Moreover for $p, q \in \mathcal{C}^*$ we have $p \sim q$ if and only if p and q have the same normal form with respect to the system.

We will refer the rewriting system $\{xx \to x \mid \mathsf{c}(x) \neq \emptyset\} \cup \{uvw \to uw \mid \emptyset \neq \mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)\}$ as RS. Since RS contains two different types of rewriting rules we denote RS_1 the rewriting system $\{xx \to x \mid x \in \mathcal{C}^*, \mathsf{c}(x) \neq \emptyset\}$ and RS_2 the rewriting system $\{uvw \to uw \mid u,v,w \in \mathcal{C}^*, \emptyset \neq \mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)\}$. The corresponding rewrite relations are denoted $\to$, $\to_1$ resp. $\to_2$ and for a word $p \in \mathcal{C}^*$ the set of its normal forms is denoted by $\langle p \rangle$, $\langle p \rangle_1$ resp. $\langle p \rangle_2$.

Remark 2.2.4. If we want to investigate the complexity issues for equations, the first question we have to answer is the complexity of checking whether some identity holds in a free band. We can see that the identity problem for variety $\mathcal{IM}$ (i.e. the problem whether $p \sim q$) can be decided in polynomial time by using the rewriting system RS. If we note that a word of length k contains $\mathcal{O}(k^2)$ factors (each factor is identified by its beginning and its length) we get that each reduction of RS can be done in polynomial time. Since every reduction decreases the length of the word, we have a polynomial time decision algorithm for the identity problem.

We know that RS is confluent and terminating. Our goal in this section is to show that RS_2 is also a confluent and terminating rewriting system and that $\langle p \rangle = \langle \langle p \rangle_2 \rangle_1$.

We define a rewrite relation $\to_{2l} \subset \to_2$ such that $suvwt \to_{2l} suwt$ if and only if $|v| = 1$ and $\mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)$. It is easy to see that $\to_2 \subseteq \to_{2l}^*$ and hence $\to_{2l}^* = \to_2^*$. The last relation we will use is $\to_{2m} \subset \to_2$, consisting of all rules that leave out the maximal number of letters in the following sense. We write $suvwt \to_{2m} suwt$ if and only if $\emptyset \neq \mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)$ and the following conditions hold:

(i) $\mathsf{c}(s_0 u) \neq \mathsf{c}(w t_0)$, for any suffix s_0 of s and any prefix t_0 of t (including empty s_0 or t_0, but not both)

(ii) $u = 0(u)\mathsf{t}(u)$

(iii) $w = \mathsf{h}(w)1(w)$.

Note that if $suvwt \to_{2m} suwt$ then the last letter of s and the first letter of t (if they exist) are new and different letters[1]. Also note that $\mathsf{t}(u)$ is the only occurrence of this letter in u and we can write $u = 0(u)\mathsf{t}(u)$. Similarly $w = \mathsf{h}(w)1(w)$.

We remind that whenever $\to_2$ rewriting applies then so does $\to_{2m}$ and $\to_{2l}$. Moreover a word is in normal form with respect to $\to_2$ iff it is in normal form with respect to $\to_{2m}$ and iff it is in normal form with respect to $\to_{2l}$. In what follows, we will use these trivial observations without any explicit reference.

We show that $\langle p\rangle_{2m} = \langle p\rangle_2$. The inclusion $\langle p\rangle_{2m} \subseteq \langle p\rangle_2$ is obvious and the rest is the content of the following lemmas. For more transparent proofs we use the notation

$$s\underline{u}v\underline{w}t \to_2 suwt$$

in the sense that $suvwt \to_2 suwt$ where $\emptyset \neq \mathsf{c}(v) \subseteq \mathsf{c}(u) = \mathsf{c}(w)$ (and the same for $\to_{2l}$, $\to_{2m}$). In the following, whenever we say that u is a factor of sut, we always refer to the concrete (and obvious) occurrence of the factor u in sut.

Lemma 2.2.5. *The relation $\to_{2m}$ is confluent and terminating.*

Proof. The termination is obvious. Let p be a word and suppose that we can apply two different rules of $\to_{2m}$ on p, say

$$p = s_1\underline{u_1}v_1\underline{w_1}t_1 \to_{2m} s_1u_1w_1t_1$$

and

$$p = s_2\underline{u_2}v_2\underline{w_2}t_2 \to_{2m} s_2u_2w_2t_2.$$

Let us suppose that u_1 is a factor of $u_2v_2w_2$. Then the whole $u_1v_1w_1$ is a factor of $u_2v_2w_2$, because $u_2v_2w_2$ is followed by a new letter (if t_2 is non-empty), which is not contained in u_1. If $u_1v_1w_1$ is a factor of u_2 resp. w_2 then our two rules commute (i.e. they are independent of the order of their applications). If it is not the case, we will show that v_1 is a factor of v_2. Suppose that the occurrence of $\mathsf{t}(u_2)$ (the last letter of u_2) is in $u_1v_1w_1$, then it is surely in u_1. Similarly if the occurrence of $\mathsf{h}(w_2)$ is in $u_1v_1w_1$, then it is in w_1. This implies that v_1 is a factor of v_2. Let us define s_0 as a prefix of u_2 and t_0 as a suffix of w_2 s.t. $s_0u_1v_1w_1t_0 = u_2v_2w_2$. Observe that $\mathsf{c}(s_0u_1) = \mathsf{c}(u_2) = \mathsf{c}(w_2) = \mathsf{c}(w_1t_0)$ and using the condition (i) in the definition of $\to_{2m}$ we get that $s_0 = t_0 = \epsilon$. So we have that $u_1v_1w_1 = u_2v_2w_2$ and v_1 is a factor of v_2, which is a contradiction with the conditions (ii) and (iii).

If u_1 resp. w_1 is not a factor of $u_2v_2w_2$, and u_2 resp. w_2 is not a factor of $u_1v_1w_1$, then our two rules commute. □

[1]Observe that it doesn't hold in general that if $p \to_{2m} q$ then $spt \to_{2m} sqt$ for $s, t \in \mathcal{C}^*$. This means that $\to_{2m}$ is not a rewriting relation in the previously introduced sense.

Remark 2.2.6. From the previous proof we can see that two arbitrary applications of $\rightarrow_{2m}$, say

$$p = s_1\underline{u_1}v_1\underline{w_1}t_1 \rightarrow_{2m} s_1u_1w_1t_1$$

and

$$p = s_2\underline{u_2}v_2\underline{w_2}t_2 \rightarrow_{2m} s_2u_2w_2t_2,$$

commute and they can be nested exactly in one of the following ways (up to symmetry):

(i) $w_1 = w_1'q, u_2 = qu_2'$ and $p = s_1u_1v_1w_1'qu_2'v_2w_2t_2$

(ii) $u_1v_1w_1$ is a factor of u_2

Lemma 2.2.7. *RS_2 is a confluent and terminating rewriting system and $\langle p\rangle_{2m} = \langle p\rangle_2$ for any $p \in \mathcal{C}^*$.*

Proof. The termination is clear. If we have $u \rightarrow_2 v$, $u \rightarrow_2 w$ then there is v_1 and w_1 such that $v \rightarrow_2^* v_1$, $w \rightarrow_2^* w_1$, $u \rightarrow_{2m} v_1$ and $u \rightarrow_{2m} w_1$. Since $\rightarrow_{2m}$ is confluent, we have the confluence of RS_2. The equality $\langle p\rangle_{2m} = \langle p\rangle_2$ is a trivial consequence. $\square$

Lemma 2.2.8. *For any $p, q \in \mathcal{C}^*$ such that $p = \langle p\rangle_2$ and $p \rightarrow_1 q$ it holds that $\langle q\rangle_2 = q$. In particular for a word $p \in \mathcal{C}^*$ we have $\langle\langle p\rangle_2\rangle_1 = \langle p\rangle$.*

Proof. Assume for the moment that $\langle q\rangle_2 \neq q$, which means that $q = suawt$ where $s, u, w, t \in \mathcal{C}^*$, $a \in \mathcal{C}$, $a \in \mathsf{c}(u) = \mathsf{c}(w)$, i.e.

$$q = s\underline{u}a\underline{w}t \rightarrow_{2l} suwt.$$

Then (up to symmetry)

$$p = su_1xu_2awt$$

where $u_1, x, u_2 \in \mathcal{C}^*$, $u_1u_2 = u$ and su_1 has a suffix x or u_2awt has a prefix x. We discuss four different cases:

1) x is a suffix of u_1

2) x is a suffix of su_1 and $|x| > |u_1|$

3) x is a prefix of u_2aw

4) x is a prefix of u_2awt and $|x| > |u_2aw|$.

In the case 1) we get $\mathsf{c}(x) \subseteq \mathsf{c}(u) = \mathsf{c}(w)$ and we could also use the reduction

$$p = s\underline{u_1xu_2}a\underline{w}t \rightarrow_{2l} su_1xu_2wt$$

since $a \in \mathsf{c}(u_1xu_2) = \mathsf{c}(u) = \mathsf{c}(w)$. In the case 2) we may write $x = x_1u_1$ and then

$$p = su_1x_1u_1u_2awt = su_1x_1\underline{u}a\underline{w}t \rightarrow_{2l} su_1x_1uwt.$$

Cases 3) and 4) are similar and all the four cases lead to a contradiction. $\square$

2.3 Upper Bound for the Size of the Solution

This section aims to prove that the AIU-Matching problem is in NP by giving a polynomial upper bound on the size of the shortest solution. In the following we assume implicitly that $A, B \in \mathcal{C}^*$. Realize that each reduction of RS just leaves out some factor, the case $uvw \to uw$ is clear, and in the case $xx \to x$ we leave out the right occurrence of x in the square. If we have a word uAv, we can speak about the *residual* of A in the sense that the residual consists of all letter occurrences of A that were not left out during the sequence of reductions. Moreover, if we use two different sequences of reductions by $\to_{2m}$, which give normal form with respect to $\to_2$, then the residuals are the same after the both reduction sequences, since any two applications of $\to_{2m}$ commute by Remark 2.2.6.

Lemma 2.3.1. *Let A and B be in normal form and $AB \to_{2m} AB' \not\to_{2m}$ where B' is the residual of B. Then the word B' contains at most one square x^2. If B' contains such a square, then x^2 arises from xvx where v is the word left out by the reduction rule $\underline{u}v\underline{w} \to_{2m} uw$, and x is both a suffix of u and a prefix of w. Moreover, in the case when B' contains a square we have $B' \to_1 \langle B' \rangle$.*

Proof. Assume that we have used

$$AB = s\underline{u}v\underline{w}t \to_{2m} suwt = AB'$$

and B' contains a square x^2. Since B is in normal form, x^2 contains ‘space” of the canceled v, i.e. $xx = u_1w_1$ where u_1 is a suffix of u (u starts in A) and w_1 is a prefix of wt.

We show that w_1 is a prefix of w. In the case when $|w_1| > |w|$ we can deduce that occurrences of $\mathsf{t}(u_1)$ and $\mathsf{h}(t)$ must lie in the left x since they are the first occurrences of the constants $\mathsf{t}(u_1)$ and $\mathsf{h}(t)$ in B (from the maximality of $\to_{2m}$). It means that $x = u_1wz$ where z is a prefix of t. Since $\mathsf{c}(u_1) \subseteq \mathsf{c}(u) = \mathsf{c}(w) \subseteq \mathsf{c}(wz)$, we can reduce

$$xx = u_1\underline{wz}u_1\underline{wz} \to_2 u_1wzwz$$

and this is a contradiction with $B' \not\to_2$.

So, w_1 is a prefix of w. The last letter of u_1 is in the left x and the first letter of w_1 is in the right x and we see that x^2 arises from xvx and x is a suffix of u and a prefix of w (i.e. $u = u_0x$, $w = xw_0$).

We have

$$uwt = u_0xxw_0t \to_1 u_0xw_0t.$$

Let us denote B'' as the residual of B'; in fact $B'' = u_0''xw_0t$ where u_0'' is a suffix of u_0.

It is enough to show that the word B'' does not contain a square and in such a case we get $B' \to_1 \langle B' \rangle_1 = B''$. Assume that $u_0''xw_0t$ contains a square y^2. Recall $xw_0t = wt$ is a suffix of B. Thus y^2 is a factor of $u_0''xw_0$ since $\mathsf{h}(t) \notin \mathsf{c}(u_0''xw_0)$ and because of the similar arguments as in the second paragraph. Since $u_0''x$ and $xw_0 = w$ are factors of B then y^2 contains both $\mathsf{t}(u_0'')$ and $\mathsf{h}(w_0)$. However, $\mathsf{t}(x) = \mathsf{t}(u)$ is the first occurrence of this letter in $u_0''xw_0$ and it must be in the left y and from the same reason $\mathsf{h}(x) = \mathsf{h}(w)$ is in the right y. This is impossible. So $u_0''xw_0t$ contains no square.

If B' contains two (or more) squares then one is a factor of the other and we get two different residuals of B'. This is a contradiction with $B' \to_1 \langle B' \rangle_1 = \langle B' \rangle$. So B' contains at most one square. $\square$

Remark 2.3.2. The same arguments as in the proof of Lemma 2.3.1 give the following analogue. If A_1, B, A_2 are in normal form and

$$A_1BA_2 \to_{2m} A_1B'A_2 \not\to_{2m}$$

and if the residual B' of B contains a square x^2, then $B' \to_1 \langle B' \rangle_1$ and x has the same properties as in Lemma 2.3.1.

Proposition 2.3.3. *Let A and B be in normal form such that $\langle AB \rangle_2 = AB'$ where B' is the residual of B, then $|B'| \leq |\langle B' \rangle|^2$.*

Proof. By Lemma 2.2.7 we have $\langle AB \rangle_2 = \langle AB \rangle_{2m}$ and we can use the maximal reductions. Without loss of generality assume that the reductions $\to_{2m}$ did not leave out some prefix B_1 of the word B, otherwise we can start with the words A and B_2 where $B = B_1B_2$. Remark 2.2.6 shows how two applications of $\to_{2m}$ can be nested. Since A and B are in normal form, we can see that any reduction $\to_{2m}$ uses some letters from both A and B. Since A is left untouched, we can write

$$A = s_{n+1}s_n \dots s_1, \quad B = u_1v_1w_1 \dots u_nv_nw_nu_{n+1}$$

where $s_i, u_i, v_i, w_i \in \mathcal{C}^*$ for all possible i and we have n reductions of the form

$$s_{n+1} \dots \underline{s_i \dots s_1u_1w_1 \dots u_i}v_i\underline{w_i} \dots u_{n+1} \to_{2m}$$

$$s_{n+1} \dots s_1u_1w_1 \dots u_iw_i \dots u_{n+1}$$

where $\mathsf{c}(v_i) \subseteq \mathsf{c}(s_i \dots s_1u_1w_1 \dots u_i) = \mathsf{c}(w_i)$ and $B' = u_1w_1 \dots u_nw_nu_{n+1}$.

Since each step of the maximal reduction needs a new letter (the letter that immediately succeeds w_i), we get an upper bound for n (the number of steps in $\to_{2m}^*$), $n + 1 \leq \|\mathsf{c}(B)\|$. Let us denote $B'' = \langle B' \rangle_1 = \langle B' \rangle$ and $w_0 = \epsilon$.

By induction (where $i = 1, \dots, n$) and by Lemma 2.3.1 applied on A and $\langle w_0 u_1 \dots w_{i-1} u_i \rangle v_i w_i u_{i+1}$ we can see that

$$|B''| \geq \max_{i=1}^{n+1} \{|w_{i-1} u_i|\}$$

since after every application $xx \to x$ we can find each $w_{i-1}u_i$ as a factor in the residual of B. Hence we get

$$|B''| \geq \max_{i=1}^{n+1} \{|w_{i-1} u_i|\} \geq \frac{1}{n+1} \sum_{i=1}^{n+1} |w_{i-1} u_i| = \frac{|B'|}{n+1}$$

and from the fact $n + 1 \leq \|\mathsf{c}(B)\| = \|\mathsf{c}(B'')\|$ we can deduce that

$$|\langle B' \rangle|^2 = |B''|^2 \geq \|\mathsf{c}(B'')\| \cdot |B''| \geq (n+1) \frac{|B'|}{n+1} = |B'|.$$

□

The previous proposition can be generalized in the following way.

Proposition 2.3.4. *Let A_1, B and A_2 be in normal form such that $\langle A_1 B A_2 \rangle_2 = A'_1 B' A'_2$ where A'_1, B', A'_2 are the residuals of A_1, B, A_2. Then*

$$|B'| \leq 2 \cdot |\langle B' \rangle|^2.$$

Proof. We prove the assertion for $A'_1 = A_1$ and $A'_2 = A_2$, because in the case when we leave out some occurrences of letters in A_1 and A_2, we can reduce $A'_i \to_1^* \langle A'_i \rangle$ and start with these new surroundings $\langle A'_1 \rangle$ and $\langle A'_2 \rangle$ of the word B, since $\langle \langle A'_1 \rangle B \langle A'_2 \rangle \rangle_2 = \langle A'_1 \rangle B' \langle A'_2 \rangle$. So, $\langle A_1 B A_2 \rangle_2 = A_1 B' A_2$ where B' is the residual of B.

We will use the maximal reduction again and for an arbitrary word $B_i \in \mathcal{C}^*$ we denote B'_i its residual (after the applications of $\to_{2m}$). Three different cases must be discussed.

1) There is a reduction $\to_{2m}$ using letters from both A_1 and A_2.

2) There is a letter in B which is not involved in any reduction $\to_{2m}$.

3) Otherwise.

In the case 1) we can write $B = B_1 v B_2$, $A_1 = s u_1$ and $A_2 = w_2 t$ where

$$s \underline{u_1 B_1} v \underline{B_2 w_2} t \to_{2m} A_1 B_1 B_2 A_2.$$

We apply twice Proposition 2.3.3 on the words $A_1B'_1$ and B'_2A_2. We can deduce

$$|B'| = |B'_1B'_2| = |B'_1|+|B'_2| \leq |\langle B'_1\rangle|^2+|\langle B'_2\rangle|^2 \leq 2\cdot|\langle B'_1B'_2\rangle|^2 = 2\cdot|\langle B'\rangle|^2$$

where the last inequality holds, because by Remark 2.3.2 we have $\langle B'_1\rangle\langle B'_2\rangle \to_1 \langle B'\rangle$ (in the case when $\langle B'_1\rangle\langle B'_2\rangle$ contain a square) and so

$$|\langle B'_1B'_2\rangle| \geq \max\{|\langle B'_1\rangle|, |\langle B'_2\rangle|\},$$

which implies that

$$2\cdot|\langle B'_1B'_2\rangle|^2 \geq |\langle B'_1\rangle|^2+|\langle B'_2\rangle|^2.$$

In the case 2) we can write $B = B_1B_2B_3$ where B_2 is not involved in any reduction by $\to_{2m}$. Then we have $\langle B'\rangle = \langle B'_1\rangle B_2\langle B'_3\rangle$ and so by Proposition 2.3.3 we get

$$\begin{aligned} |B'| &= |B'_1B_2B'_3| \\ &= |B'_1|+|B_2|+|B'_3| \\ &\leq |\langle B'_1\rangle|^2+|B_2|+|\langle B'_3\rangle|^2 \\ &\leq |\langle B'_1\rangle B_2\langle B'_3\rangle|^2 \\ &= |\langle B'\rangle|^2. \end{aligned}$$

In the case 3) we have $B = B_1v_1B_2B_3B_4v_2B_5$, $A_1 = su_1$, $A_2 = w_2t$ where

$$A_1BA_2 = s\underline{u_1B_1}v_1\underline{B_2B_3}B_4v_2B_5w_2t \to_{2m} A_1B_1B_2B_3B_4v_2B_5A_2,$$

$$A_1BA_2 = su_1B_1v_1B_2\underline{B_3B_4}v_2\underline{B_5w_2}t \to_{2m} A_1B_1v_1B_2B_3B_4B_5A_2$$

are the unique overlapping reductions (see Remark 2.2.6). Now we can write $B' = B'_1B_2B_3B_4B'_5$ and $B_0 = \langle B'_1B_2B_3\rangle$ is a prefix of $\langle B'\rangle = \langle B_0B_4B'_5\rangle$, and so $|B_0| \leq |\langle B'\rangle|$. Let us observe that

$$\begin{aligned} |B'| &= |B'_1B_2B_3B_4B'_5| \\ &= |B'_1B_2B_3|+|B_4B'_5| \\ &\leq |\langle B'_1B_2B_3\rangle|^2+|B_0B_4B'_5| \\ &\leq |B_0|^2+|\langle B_0B_4B'_5\rangle|^2 \\ &\leq 2\cdot|\langle B'\rangle|^2. \end{aligned}$$

□

Lemma 2.3.5. *Let $sxxt$ be a word such that $\langle sxxt\rangle_2 = sxxt$ and $sxxt$ contains another square y^2 ($|y| \leq |x|$) such that one of these occurrences of y lies inside the x^2. Then one of the following conditions holds:*

(i) y *is a suffix of* s *and a prefix of* x

(ii) y *is a prefix of* t *and a suffix of* x

(iii) y^2 *is a factor of* x

Proof. Since y is a factor of x^2, we have $\mathsf{c}(y) \subseteq \mathsf{c}(x)$ and let us suppose that conditions (i) and (ii) do not hold. If $|x| = |y|$ then we get $\mathsf{c}(x) = \mathsf{c}(y)$ and we can apply $\rightarrow_2$ (without loss of generality $\underline{x}v\underline{y} \rightarrow_2 xy$ where v is both a prefix of x and a suffix of y), which is a contradiction. Assume that $|y| < |x|$. Notice that the first and the last letter of x are unique occurrences of these constants in the word x because in another case we can apply $\rightarrow_{2l}$ on x^2. This implies that y^2 does not contain the first letter of the right x and y^2 also does not contain the last letter of the left x, requiring that y^2 is a factor of x. □

Remark 2.3.6. The previous lemma shows that for two applications of the rules $xx \rightarrow_1 x$ and $yy \rightarrow_1 y$ on a word p in normal form with respect to $\rightarrow_2$, one of the following conditions holds (up to symmetry):

(i) xx and yy do not overlap

(ii) yy is a factor of x

(iii) $x = x'z$, $y = zy'$ and $xx'zy'y$ is a factor of p

Lemma 2.3.7. *If* $\langle sxxt \rangle_2 = sxxt$ *and* sxt *contains a square* y^2 *which is not in* $sxxt$ *then* $y = s_1xt_1$ *where* $|s_1|$, $|t_1| \geq 1$*.*

Proof. Since y^2 is not in $sxxt$, we have $y^2 = s_0xt_0$ where s_0 is a suffix of s and t_0 is a prefix of t, $|s_0|, |t_0| \geq 1$. If x is not a factor of y then $y = s_0x_1$, $y = x_2t_0$ where $x = x_1x_2$. Hence

$$s_0xxt_0 = \underline{s_0x_1}x_2x_1\underline{x_2t_0} \rightarrow_2 s_0x_1x_2t_0$$

since $\mathsf{c}(x_2x_1) \subseteq \mathsf{c}(y) = \mathsf{c}(s_0x_1) = \mathsf{c}(x_2t_0)$. The case when x_1 or x_2 is an empty word (i.e. x is a prefix or a suffix of y) is also included and we can conclude that $y = s_1xt_1$ and $|s_1|, |t_1| \geq 1$. □

Proposition 2.3.8. *Let* A *and* B *be in normal form,* $\|\mathsf{c}(AB)\| \geq 2$ *and* $\langle AB \rangle_2 = AB$*. Then* $|AB| \leq |\langle AB \rangle|^2$*.*

Proof. We denote k the length of $\langle AB \rangle_1 = \langle AB \rangle$. The case $k = 2$ is trivial: there is no word (in normal form with respect to $\rightarrow_2$) over a 2–letter alphabet of length greater than 4. Now we assume that $k \geq 3$.

At first we have a look at the squares in AB. Since A and B are in normal form, each square has got some letters from A and some from B. By Remark 2.3.6 we have that AB contains at most two squares because the cases 1. and 2. of Remark 2.3.6 are impossible.

By Lemma 2.2.8, anytime during any reduction sequence by $\rightarrow_1$ the residuals of A and B remain in normal form and their concatenations are in normal form with respect to $\rightarrow_2$. So by the previous arguments there are at most two squares and by Lemma 2.3.7, if the reduction $\rightarrow_1$ introduces a new square then it is larger at least by two letters than the previous one. We use the sequence of reductions which in each step reduces the smallest square. Then the last reduction in this sequence reduces a square x^2 with the property $|x| \leq k$. The previous reduction reduces a square y^2 with the property $|y| \leq k-1$ (the case $|y| = |x| = k = |\langle AB\rangle|$ is a contradiction with AB is in normal form with respect to $\rightarrow_2$). Thus we can see that for the length of AB we have an upper bound $k + k + (k-1) + (k-2) + \dots \ \leq \ k^2$. ☐

Proposition 2.3.9. *There is a polynomial* $p : \mathbb{N} \rightarrow \mathbb{N}$*, such that for an arbitrary* A_1*,* B*,* $A_2 \in \mathcal{C}^*$ *in normal form and* $\langle A_1 B A_2\rangle_2 = A'_1 B' A'_2$ *where* A'_i*,* $1 \leq i \leq 2$*, is the residual of* A_i *and* B' *is the residual of* B*, we have* $|B'| \leq p(|\langle A_1 B A_2\rangle|)$*.*

Proof. We may assume that $\|\mathsf{c}(B)\| \geq 2$ and by Proposition 2.3.4 we know that $|B'| \leq 2 \cdot |\langle B'\rangle|^2$, which is of course less or equal to $2 \cdot |\langle A'_1\rangle\langle B'\rangle|^2$. Since $\langle\langle A'_1\rangle\langle B'\rangle\rangle_2 = \langle A'_1\rangle\langle B'\rangle$ by Lemma 2.2.8, we can use Proposition 2.3.8 and we get that $2\cdot|\langle A'_1\rangle\langle B'\rangle|^2 \leq 2\cdot|\langle A'_1 B'\rangle|^4$, which is again less or equal to $2\cdot|\langle A'_1 B'\rangle\langle A'_2\rangle|^4$. Analogously we have that $2 \cdot |\langle A'_1 B'\rangle\langle A'_2\rangle|^4 \leq 2 \cdot |\langle A'_1 B' A'_2\rangle|^8$. Thus we have $|B'| \leq p(|\langle A_1 B A_2\rangle|)$ for the polynomial $p(n) = 2 \cdot n^8$. ☐

Theorem 2.3.10. *Let* p *be a polynomial that satisfies the condition from Proposition 2.3.9. If a pattern equation system* $\{L_1 =^? R_1, \dots, L_n =^? R_n\}$ *is satisfiable then there exists a solution* α *with*

$$\mathrm{size}(\alpha) \leq \sum_{i=1}^{n} |L_i| \cdot p(|R_i|).$$

Proof. Of course, we can assume that all R_i's are in normal form. Let α be a solution of the pattern equation system $\{L_1 =^? R_1, \dots, L_n =^? R_n\}$ which minimizes both $\mathrm{size}(\alpha)$ and the number of variables x such that $|\alpha(x)| = \mathrm{size}(\alpha)$. Denote

$$K = \sum_{i=1}^{n} |L_i| \cdot p(|R_i|)$$

and assume for the moment that there is some x such that $\mathrm{size}(\alpha) = |\alpha(x)| > K$. We may assume that $\alpha(x)$ is in normal form, otherwise we have a smaller solution.

We now reduce $\alpha(L_i) \rightarrow_{2m}^{*} \langle\alpha(L_i)\rangle_2$. If we look at an arbitrary residual B' of an occurrence of $\alpha(x)$ in $\langle\alpha(L_i)\rangle_2$, we see that $|B'| \leq p(|R_i|)$ by Proposition 2.3.9. This means that there are at most K letter's occurrences in the residuals of all occurrences of $\alpha(x)$ in all $\langle\alpha(L_i)\rangle_2$.

By the assumption $|\alpha(x)| > K$ we get that there is an occurrence of a letter a in $\alpha(x)$, i.e. $\alpha(x) = u_1 a u_2$, that has been left out from all the occurrences of $\alpha(x)$ by the rule $\rightarrow_{2m}$. We can erase this occurrence of the letter a from $\alpha(x)$ and we get a smaller solution β such that $\beta(y) = \alpha(y)$ for $y \neq x$ and $\beta(x) = u_1 u_2$. The homomorphism β is indeed a solution since $\alpha(L_i) \rightarrow_{2l}^{*} \beta(L_i)$. This is a contradiction because we have found a smaller solution. □

The previous considerations lead to the following corollary.

Corollary 2.3.11. *The* AIU-MATCHING *problem and* AI-MATCHING *problem are in NP.*

Proof. We can guess a solution α and by Theorem 2.3.10, if the system is satisfiable then there is a solution of a polynomial length. Checking whether α solves all the equations takes also polynomial time and so the problems are in NP. □

The following example shows that in order to prove that the AI-UNIFICATION problem belongs to NP we cannot use the same method as for the AI-MATCHING problem, because there are satisfiable systems which have no solution of polynomial size.

Example 2.3.12. Let S be a system over the set of constants $\mathcal{C} = \{a_0, \ldots, a_n\}$ and the set of variables $\mathrm{X} = \{x_1, \ldots, x_n\}$ which contains for each $i \in \mathbb{N}$, $1 < i \leq n$, the equation $x_i =^{?} x_{i-1} a_i x_{i-1}$ and in addition $x_1 =^{?} a_0 a_1 a_0$. It is easy to see that the system S has a unique solution in the free monoid $\mathcal{C}^*$, which is the shortest solution in the free idempotent monoid over $\mathcal{C}$. For this solution α we have $|\alpha(x_i)| = 2^{i+1} - 1$. (These words $\alpha(x_i)$ are usually called the Zimin words in literature.)

Chapter 3

AI-Unification Problem

This chapter contains the main result of the thesis — the NP-completeness of the AI-UNIFICATION problem. The chapter is an extension of the original paper [30] in two ways: Sections 3.2 – 3.4 contain complete proofs and all technical details and some parts (Section 3.1 for example) consist of new observations and comments.

The first section solves the special case of the AI-UNIFICATION problem for one equation and discusses the case when the number of equations in an instance of the problem is bounded. Section 3.2 introduces the basic concept of the SIMPLE AIU-UNIFICATION problem which is, in fact, a special case of the AIU-UNIFICATION problem, but on the other hand can help to solve it. The complexity of the SIMPLE AIU-UNIFICATION problem is described in Section 3.3. The NP-completeness of the (general) AI-UNIFICATION problem and the AI-MATCHING problem respectively, is a consequence of this description and the proofs are contained in Section 3.4.

3.1 One Equation

We explain how one can solve one equation in the free band or in a free idempotent monoid in polynomial time. The proof is based on the idea that any solvable equation has a constant solution. We use Lemma 1 from our paper [29]. In order to formulate the lemma we need a new definition.

For any word $u \in (\mathcal{C} \cup \mathrm{X})^*$ we define u_0 the longest prefix of u such that $\mathsf{c}(u_0) \subseteq \mathcal{C}$; and dually u_1 the longest suffix of u such that $\mathsf{c}(u_1) \subseteq \mathcal{C}$.

Lemma 3.1.1. *The equation $L =^? R$ is solvable in the free idempotent monoid if and only if the following conditions are satisfied.*

(i) $L_0R_0 \sim R_0L_0R_0$,

(ii) $R_1L_1 \sim R_1L_1R_1$,

(iii) *if* $\mathsf{c}(L) \subseteq \mathcal{C}$ *then* $\mathsf{c}(R) \cap \mathcal{C} \subseteq \mathsf{c}(L)$,
(iv) *if* $\mathsf{c}(R) \subseteq \mathcal{C}$ *then* $\mathsf{c}(L) \cap \mathcal{C} \subseteq \mathsf{c}(R)$.

Proof. Let α be a solution of the equation $L =^? R$. After substituting α into the equation, we get $\alpha(L) = L_0 L' L_1 \sim R_0 R' R_1 = \alpha(R)$ for some $L', R' \in \mathcal{C}^*$. It is easy to see

$$L_0 R_0 R' R_1 L_1 \sim R_0 L_0 R_0 R' R_1 L_1 R_1,$$

which implies the first two conditions by Proposition 1.1.3. The next two conditions are clear.

Now assume that the conditions are satisfied. Let us denote $w = L_0 R_0 e R_1 L_1$, where e is a product of all letters from the set $\mathsf{c}(LR) \cap \mathcal{C}$. If we define $\alpha : \mathrm{X} \to \mathcal{C}^*$ by the relations $\alpha(x) = w$ for all $x \in \mathrm{X}$, then one can see that α is a solution of the equation $L =^? R$. Indeed, conditions (iii) and (iv) ensure that $\mathsf{c}(\alpha(L)) = \mathsf{c}(\alpha(R)) = \mathsf{c}(w) = \mathsf{c}(e)$ and conditions (i) and (ii) give $0(\alpha(L)) = 0(L_0 L_0 R_0 e) \sim 0(R_0 L_0 R_0 e) = 0(\alpha(R))$ and dually $1(\alpha(L)) \sim 1(\alpha(R))$. □

It is easy to see that all conditions from the lemma can be checked in polynomial time using Remark 2.2.4 for example. We did not use the neutral element in the lemma (except for the trivial case $\mathsf{c}(LR) \subseteq \mathrm{X}$), hence the same holds for a free band.

We show that solving of a pair of equations is as hard as solving of a system of equations.

Lemma 3.1.2. *For any system of equations $S = \{L_1 =^? R_1, \dots, L_n =^? R_n\}$ there exists a system consisting of a pair of equations $S' = \{L =^? R, L' =^? R'\}$ such that the system S' is solvable if and only if the system S is solvable and the size of the system S' is polynomial in the size of the system S.*

Proof. Let $S = \{L_1 =^? R_1, \dots, L_n =^? R_n\}$ be a system. Let $\{x_1, \dots, x_l\}$ be the set of all variables and $\{a_1, \dots, a_m\}$ be the set of all constants occurring in the system S. Denote $\{e_1, \dots, e_{n-1}\}$ a set of new different constant symbols which is disjoint from the set $\{a_1, \dots, a_m\}$.

Now, we consider a pair of equations:

$$L_1 e_1 L_2 e_2 \dots e_{n-1} L_n =^? R_1 e_1 R_2 e_2 \dots e_{n-1} R_n, \tag{3.1}$$

$$a_1 a_2 \dots a_m x_1 x_2 \dots x_l a_1 a_2 \dots a_m =^? a_1 a_2 \dots a_m. \tag{3.2}$$

We show that the system S' consisting of the equations (3.1 – 3.2) has the property from the statement. If S is solvable then there is a solution α of S such that $\alpha(x_i) \subseteq \{a_1, \dots, a_m\}$ for any $i = 1, \dots, l$. It is easy to see that such α is a solution of the system S' too. Conversely, if α is a solution of the system S' then we can see that $\alpha(x_i) \subseteq \{a_1, \dots, a_m\}$ for any $i = 1, \dots, l$ by the equation (3.2). Hence each $e \in \{e_1, \ldots, e_{n-1}\}$ does not occur in $\alpha(L_k)$ and $\alpha(R_k)$ for all $k = 1, \ldots, n$. We can deduce that $\alpha(L_k) \sim \alpha(R_k)$ for any $k = 1, \ldots, n$, because

$$\alpha(L_k) = 1^p 0^r(\alpha(L_1 e_1 L_2 e_2 \ldots e_{n-1} L_n)), \quad \alpha(R_k) = 1^p 0^r(\alpha(R_1 e_1 R_2 e_2 \ldots e_{n-1} R_n))$$

for certain $p, r \in \mathbb{N}$. □

Of course, we will use an opposite method for solving of systems, i.e. we will try to decompose one equation to certain equivalent equations which will be simpler in some sense. This is the content of the following sections.

3.2 Simple AIU-Unification Problem

Let $L =^? R$ be a word equation and $\alpha : \mathrm{X} \to \mathcal{C}^*$ be a mapping. Which information about α do we need to know if we want to decide whether α is a solution of the equation $L =^? R$? We would like to apply Proposition 1.1.3. We have to compute $\mathsf{c}(\alpha(L))$ and $\mathsf{c}(\alpha(R))$, hence we need to know $\mathsf{c}(\alpha(x))$ for all variables $x \in \mathrm{X}$. If we want to compute $\mathsf{c}(\mathrm{pref}_i(\alpha(L)))$ then we need to know $\overrightarrow{\alpha(x)}$ for some variables. Therefore, we will assume that $\overrightarrow{\alpha(x)}$ and $\overleftarrow{\alpha(x)}$ are given for all variables $x \in \mathrm{X}$. This assumption is natural; the AIU-Unification problem with this assumption is almost the same as without it as we show in this section.

For two mappings $\alpha, \beta : \mathrm{X} \to \mathcal{C}^*$ we write

$$\alpha \approx_{\mathrm{X}} \beta \quad \text{if and only if} \quad \overrightarrow{\alpha(x)} = \overrightarrow{\beta(x)},\ \overleftarrow{\alpha(x)} = \overleftarrow{\beta(x)} \quad \text{for all} \quad x \in \mathrm{X}\,.$$

For mappings $\alpha, \beta : \mathrm{X} \to \mathcal{C}^*$ and $I : \mathrm{X} \to \mathcal{P}(\{\to, \leftarrow\})$ we write

$$\alpha \approx^I_{\mathrm{X}} \beta$$

if and only if for all $x \in \mathrm{X}$ we have

$$\overrightarrow{\alpha(x)} = \overrightarrow{\beta(x)} \quad \text{when} \quad \to \in I(x) \quad \text{and} \quad \overleftarrow{\alpha(x)} = \overleftarrow{\beta(x)} \quad \text{when} \quad \leftarrow \in I(x).$$

A triple (S, ν, I) is called a *system with partial information about invariants* if $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ is a system of word equations and $\nu : \mathrm{X} \to \mathcal{C}^*$ and $I : \mathrm{X} \to \mathcal{P}(\{\to, \leftarrow\})$ are mappings. A solution of the system with partial information about invariants (S, ν, I) is a solution $\alpha : \mathrm{X} \to \mathcal{C}^*$ of the system S in the free idempotent monoid over $\mathcal{C}$ such that $\alpha \approx^I_{\mathrm{X}} \nu$. (We can assume that $|\nu(x)| \le 2 \cdot \|\mathcal{C}\|$ because we can always exchange $\nu(x)$ for the word $\overrightarrow{\nu(x)}\,\overleftarrow{\nu(x)}$.)

We speak about a *system with complete information about invariants* if

$$I(x) = \{\to, \leftarrow\} \quad \text{for all} \quad x \in \mathrm{X}\,.$$

Such a system is denoted simply (S, ν) and a solution α of this system is a solution of the system S satisfying $\alpha \approx_{\mathrm{X}} \nu$.

Definition 3.2.1. The SIMPLE AIU-UNIFICATION problem asks to provide an algorithm deciding whether a given system with complete information about invariants has a solution.

Lemma 3.2.2. *Let (S, ν, I) be a system with partial information about invariants. Then one can construct (in polynomial time) a system S' such that the set of all solutions of (S, ν, I) and the set of all solutions of $S \cup S'$ coincide.*

Proof. We put the system S to the system S' and for each $x \in \mathrm{X}$ we add to the system S' a subsystem T_x whose solutions α are characterized by the conditions which ν and I determine for the variable x, i.e. $\overrightarrow{\alpha(x)} = \overrightarrow{\nu(x)}$ if $\rightarrow \in I(x)$ and $\overleftarrow{\alpha(x)} = \overleftarrow{\nu(x)}$ if $\leftarrow \in I(x)$.

Let $n \in \mathbb{N}$, $n > 1$ and let $a_1, \ldots, a_n \in \mathcal{C}$ be pairwise distinct letters. We show that for an arbitrary mapping $\alpha : \mathrm{X} \to \mathcal{C}^*$ the condition $\overrightarrow{\alpha(x)} = a_1 \cdots a_n$ holds if and only if α is a solution of the system T consisting of the equations

$$a_1 \cdots a_n x a_1 \cdots a_n =^? a_1 \cdots a_n \tag{3.3}$$

$$a_1 a_2 x =^? x \tag{3.4}$$

and the equations

$$a_1 \cdots a_k a_1 x =^? a_1 \cdots a_k a_2 x \tag{3.5}$$

for $k = 3, \ldots, n$.

Let α be a solution of the system T and let $u = \alpha(x)$. Then $\mathsf{c}(u) \subseteq \{a_1, \ldots, a_n\}$ follows from (3.3). We show inductively

$$\overrightarrow{\mathrm{pref}_i(u)} = a_1 \cdots a_i \tag{3.6}$$

for $i = 1, \ldots, n$. For $i = 1, 2$ it follows directly from (3.4). Assume that (3.6) holds for certain i, $1 < i < n$, and does not hold for $i + 1$. Then

$$w := \mathrm{pref}_{i+1}(a_1 \cdots a_{i+1} a_1 u) = a_1 \cdots a_{i+1} a_1 \,\mathrm{pref}_i(u),$$

hence $1(w) = a_1 \,\mathrm{pref}_i(u)$. In the same way

$$w' := \mathrm{pref}_{i+1}(a_1 \cdots a_{i+1} a_2 u) = a_1 \cdots a_{i+1} a_2 \,\mathrm{pref}_i(u),$$

hence $1(w') = a_2 \,\mathrm{pref}_i(u)$. So, $1(w) \not\sim 1(w')$ and this implies $w \not\sim w'$. We see that α is not a solution of the equation (3.5) for $k = i + 1$ and this is a contradiction. For $i = n$ the equality (3.6) means

$$\overrightarrow{\alpha(x)} = \overrightarrow{\mathrm{pref}_n(u)} = a_1 \cdots a_n.$$

Conversely, if $\alpha(x) = u$ is a word such that $\overrightarrow{u} = a_1 \cdots a_n$, then

$$a_1 \cdots a_k a_1 \operatorname{pref}_k(u) \sim a_1 \cdots a_k \operatorname{pref}_k(u) \sim a_1 \cdots a_k a_2 \operatorname{pref}_k(u)$$

for $k \in \{3, \ldots, n\}$ by Lemma 1.1.4. Hence α is a solution of the subsystem (3.5). It is easy to see that α is also a solution of the equations (3.3) and (3.4).

For a given variable $x \in \mathrm{X}$ and a word $a_1 \cdots a_n \in \mathcal{C}^*$ consisting of distinct letters we denote the system constructed in the previous paragraphs by $T_x^{\rightarrow}(a_1 \cdots a_n)$. From left-right duality we obtain a system $T_x^{\leftarrow}(a_1 \cdots a_n)$ whose solutions α are characterized by the condition $\overleftarrow{\alpha(x)} = a_1 \cdots a_n$. Now for $x \in \mathrm{X}$ we put

$$T_x = \begin{cases} T_x^{\rightarrow}\left(\overrightarrow{\nu(x)}\right) \cup \; T_x^{\leftarrow}\left(\overleftarrow{\nu(x)}\right) & \text{if} \quad I(x) = \{\rightarrow, \leftarrow\} \\ T_x^{\rightarrow}\left(\overrightarrow{\nu(x)}\right) & \text{if} \quad I(x) = \{\rightarrow\} \\ T_x^{\leftarrow}\left(\overleftarrow{\nu(x)}\right) & \text{if} \quad I(x) = \{\leftarrow\} \\ \emptyset & \text{if} \quad I(x) = \emptyset \end{cases}$$

It is easy to see that the system $S' = \bigcup_{x \in \mathrm{X}} T_x$ satisfies the statement. $\square$

As a consequence of Lemma 3.2.2 we obtain a polynomial reduction from the Simple AIU-Unification problem to the AIU-Unification problem. On the other hand, it is easy to see that there exist non-deterministic polynomial reductions from the AIU-Unification and AI-Unification problems to the Simple AIU-Unification problem (they guess a mapping $\nu : \mathrm{X} \rightarrow \mathcal{C}^*$ satisfying $|\nu(x)| \leq 2 \cdot \|\mathcal{C}\|$). So, if we want to prove that the AIU-Unification and AI-Unification problems are in NP, it is enough to show that the Simple AIU-Unification problem is in NP. We show much more; we describe a polynomial algorithm for the Simple AIU-Unification problem in the next section.

3.3 Solution of Simple AIU-Unification

In this section we prove the crucial result.

Proposition 3.3.1. *The* Simple AIU-Unification *problem is decidable in polynomial time.*

Proof. Let (S, ν) be a system with complete information about invariants over a finite set of constants $\mathcal{C}$ and a finite set of variables X.

We may assume that $\nu(x)_\sharp \geq 2$ for any $x \in \mathrm{X}$ because if $\nu(x)_\sharp \leq 1$ for some $x \in \mathrm{X}$ then $\alpha(x) \sim \nu(x)$ for any solution α of the system (S, ν) and we can replace all occurrences of the variable x in S with $\nu(x)$. We also assume that all constants from $\mathcal{C}$ and all variables from X occur in S. We denote m the cardinality of $\mathcal{C}$, i.e. $\|\mathcal{C}\| = m$. So, we have $2 \leq \nu(x)_\sharp \leq m$ for any $x \in \mathrm{X}$.

We extend the set of variables X. For each $x \in \mathrm{X}$ and $i = 1, \ldots, \nu(x)_\sharp - 1$ we introduce new variables x_i^{pref} and x_i^{suff}. Moreover, for each such new variable y we also introduce the variable $\overline{y}$. We denote the sets of these variables in the following way:

$$\mathrm{X}^{\mathrm{pref}} = \{x_i^{\mathrm{pref}} \mid x \in \mathrm{X},\ i = 1, \ldots, \nu(x)_\sharp - 1\},$$

$$\mathrm{X}^{\mathrm{suff}} = \{x_i^{\mathrm{suff}} \mid x \in \mathrm{X},\ i = 1, \ldots, \nu(x)_\sharp - 1\},$$

$$\overline{\mathrm{X}}^{\mathrm{pref}} = \{\overline{y} \mid y \in \mathrm{X}^{\mathrm{pref}}\}, \quad \overline{\mathrm{X}}^{\mathrm{suff}} = \{\overline{y} \mid y \in \mathrm{X}^{\mathrm{suff}}\}.$$

Note that these sets are pairwise disjoint and each of them has $\sum_{x \in \mathrm{X}}(\nu(x)_\sharp - 1)$ elements. We put

$$\mathrm{X}^{\mathrm{ext}} = \mathrm{X} \cup \mathrm{X}^{\mathrm{pref}} \cup \mathrm{X}^{\mathrm{suff}} \qquad \overline{\mathrm{X}} = \overline{\mathrm{X}}^{\mathrm{pref}} \cup \overline{\mathrm{X}}^{\mathrm{suff}} \quad \text{and} \quad \mathcal{X} = \mathrm{X}^{\mathrm{ext}} \cup \overline{\mathrm{X}}.$$

(By the way, we have $\|\mathcal{X}\| < 4 \cdot \|\,\mathrm{X}\,\| \cdot m$.) Moreover, for $k \in \mathbb{N}$ we define $\overline{\mathrm{X}}_k^{\mathrm{pref}}$ as the set of all elements from $\overline{\mathrm{X}}^{\mathrm{pref}}$ with index k, i.e.

$$\overline{\mathrm{X}}_k^{\mathrm{pref}} = \{\overline{x}_k^{\mathrm{pref}} \mid x \in \mathrm{X},\, \nu(x)_\sharp > k\}.$$

Analogically

$$\overline{\mathrm{X}}_k^{\mathrm{suff}} = \{\overline{x}_k^{\mathrm{suff}} \mid x \in \mathrm{X},\, \nu(x)_\sharp > k\} \quad \text{and} \quad \overline{\mathrm{X}}_k = \overline{\mathrm{X}}_k^{\mathrm{pref}} \cup \overline{\mathrm{X}}_k^{\mathrm{suff}}.$$

We say that a mapping $\alpha : \mathcal{X} \to \mathcal{C}^*$ is *compatible* on the set $\mathcal{X}$ if and only if the following conditions hold for all $x \in \mathrm{X}$ and $i = 1, \ldots, \nu(x)_\sharp - 1$:

$$\begin{aligned} \alpha(x_i^{\mathrm{pref}}) &= \mathrm{pref}_i(\alpha(x)), && (3.7)\\ \alpha(\overline{x}_i^{\mathrm{pref}}) &= 1(\alpha(x_i^{\mathrm{pref}})), && (3.8)\\ \alpha(x_i^{\mathrm{suff}}) &= \mathrm{suff}_i(\alpha(x)), && (3.9)\\ \alpha(\overline{x}_i^{\mathrm{suff}}) &= 0(\alpha(x_i^{\mathrm{suff}})). && (3.10) \end{aligned}$$

This definition explains the role of variables from the set $\mathcal{X}$. If α is compatible on $\mathcal{X}$ and $\alpha \approx_{\mathrm{X}} \nu$, then we have

$$\alpha(x_1^{\mathrm{pref}}) = \mathrm{pref}_1\left(\alpha(x)\right) \sim \mathrm{pref}_1\left(\overrightarrow{\alpha(x)}\right) = \mathrm{pref}_1(\overrightarrow{\nu(x)}),$$

so $\alpha(x_1^{\mathrm{pref}})$ is determined up to $\sim$. Moreover, it means $\alpha(\overline{x}_1^{\mathrm{pref}}) = \epsilon$.

We will define a certain system with partial information about invariants over the set of variables $\mathcal{X}$ such that the set of all compatible solutions on $\mathcal{X}$ of this system corresponds to the set of all solutions of the original system with complete information about invariants (S, ν) over the set of variables X. We define $I(x) = \{\rightarrow, \leftarrow\}$, $I(x_i^{\text{pref}}) = \{\rightarrow\}$ and $I(x_i^{\text{suff}}) = \{\leftarrow\}$ for all $x \in \mathrm{X}$ and $i = 1, \dots, \nu(x)_\sharp - 1$. We put $I(y) = \emptyset$ for all $y \in \overline{\mathrm{X}}$. Now we extend the domain of the mapping ν to the set $\mathcal{X}$. We put $\nu(x_i^{\text{pref}}) = \text{pref}_i(\nu(x))$ and $\nu(x_i^{\text{suff}}) = \text{suff}_i(\nu(x))$ for all $x \in \mathrm{X}$ and $i = 1, \dots, \nu(x)_\sharp - 1$. Finally, we define $\nu(\overline{x}_i^{\text{pref}}) = 1(\nu(x_i^{\text{pref}}))$ and $\nu(\overline{x}_i^{\text{suff}}) = 0(\nu(x_i^{\text{suff}}))$. So, ν is compatible on the set $\mathcal{X}$ under this definition.

Now we can see that any solution $\alpha : \mathrm{X} \rightarrow \mathcal{C}^*$ of the system (S, ν) with complete information about invariants can be uniquely (by the rules (3.7–3.10)) extended to a compatible solution of the system (S, ν, I) with partial information about invariants over the set of variables $\mathcal{X}$. (Let us point out that $\mathsf{c}(\alpha(\overline{x}_i^{\text{pref}}))$ and $\mathsf{c}(\nu(\overline{x}_i^{\text{pref}}))$ are not necessarily the same while $\mathsf{c}(\alpha(x_i^{\text{pref}})) = \mathsf{c}(\nu(x_i^{\text{pref}}))$.) Conversely, the restriction of any (not necessarily compatible) solution $\alpha : \mathcal{X} \rightarrow \mathcal{C}^*$ of the system (S, ν, I) with partial information about invariants to the set of variables X is a solution of the original system (S, ν). From this moment the mappings ν and I are fixed and we concentrate only on compatible solutions of systems (T, ν, I) with partial information about invariants, where T is some system of equations over the set of constants $\mathcal{C}$ and the set of variables $\mathcal{X}$. We say that two such systems T_1 and T_2 are *equivalent* if and only if the systems with partial information about invariants (T_1, ν, I) and (T_2, ν, I) have the same compatible solutions.

We describe an algorithm which simplifies the system S to some equivalent system step by step. The basic idea of the simplification is that the system

$$T \cup \{L =^? R\}$$

is exchanged for the equivalent system

$$T \cup \{\mathfrak{o}(L) =^? \mathfrak{o}(R), \mathfrak{j}(L) =^? \mathfrak{j}(R)\},$$

where $\mathfrak{o}$, $\mathfrak{j}$ are certain mappings with the properties

$$\alpha(\mathfrak{o}(U)) = 0(\alpha(U)), \quad \alpha(\mathfrak{j}(U)) = 1(\alpha(U)) \quad \text{for} \quad U \in \{L, R\}$$

and for any compatible mapping $\alpha : \mathcal{X} \rightarrow \mathcal{C}^*$.

We define the set $\mathcal{EF} \subseteq (\mathcal{C} \cup \mathrm{X}^{\text{ext}})^*$, which has the property that both sides of any equation appearing in systems within the algorithm belong to the set $\mathcal{EF} \cup \overline{\mathrm{X}}$. At first, for an arbitrary word $U \in (\mathcal{C} \cup \mathrm{X})^*$ we define the set

$$\mathcal{EF}(U) \subseteq (\mathcal{C} \cup \mathrm{X}^{\text{ext}})^*$$

of *extended factors* of U in the following way: for $V \in (\mathcal{C} \cup \mathrm{X})^*$, $x, y \in \mathrm{X}$, $i \in \{1, \ldots, \nu(x)_\sharp - 1\}$ and $k \in \{1, \ldots, \nu(y)_\sharp - 1\}$, the set $\mathcal{EF}(U)$ contains the words

$$x_i^{\mathrm{suff}} V y_k^{\mathrm{pref}}, \quad x_i^{\mathrm{suff}} V, \quad V y_k^{\mathrm{pref}} \quad \text{and} \quad V$$

if and only if

$$xVy, \quad xV, \quad Vy \quad \text{and} \quad V$$

respectively are factors of U. Finally, we denote by $\mathcal{EF}$ the union of all sets $\mathcal{EF}(U)$ such that U is a left-hand or right-hand side of some equation of the system S.

We define the mappings $\mathfrak{c}$, $\mathfrak{o}$, $\mathfrak{j}$ on the set $\mathcal{EF}$ which correspond to the invariants c, 0 and 1. We put $\mathfrak{c}(A) = \mathsf{c}(\nu(A))$ for a word $A \in \mathcal{EF}$. It is easy to see that for every mapping $\alpha : \mathrm{X} \to \mathcal{C}^*$ we have:

$$\forall A \in \mathcal{EF} : \alpha \approx^I_{\mathcal{X}} \nu \implies \mathsf{c}(\alpha(A)) = \mathfrak{c}(A). \tag{3.11}$$

The following definition of the mapping $\mathfrak{o}$ from the set $\mathcal{EF} \setminus \{\epsilon\}$ to the set $\mathcal{EF} \cup \overline{\mathrm{X}}^{\mathrm{suff}}$ is correct because ν is compatible on the set $\mathcal{X}$, which implies

$$\mathfrak{c}(x_i^{\mathrm{pref}}) \subseteq \mathfrak{c}(x_j^{\mathrm{pref}}) \subset \mathfrak{c}(x)$$

for all $x \in \mathrm{X}$ and $1 \le i < j < \nu(x)_\sharp = \|\mathfrak{c}(x)\|$.

Let $A \in \mathcal{EF}$ be a non-empty word and let $B \in \mathcal{EF}$ be the longest prefix of A such that $\mathfrak{c}(B) \neq \mathfrak{c}(A)$ and $b \in \mathcal{C} \cup \mathrm{X}^{\mathrm{ext}}$ be the next letter in A, i.e. Bb is also a prefix of A — the shortest prefix satisfying $\mathfrak{c}(Bb) = \mathfrak{c}(A)$.

- If $b \in \mathcal{C}$ then we put $\mathfrak{o}(A) = B$.
- If $b = x$ or $b = x_j^{\mathrm{pref}}$ for some $x \in \mathrm{X}$ and $j \in \mathbb{N}$, then we put $\mathfrak{o}(A) = Bx_i^{\mathrm{pref}}$, where $i \in \mathbb{N}$ is the largest number such that $\mathfrak{c}(Bx_i^{\mathrm{pref}}) \subset \mathfrak{c}(A)$. If such i does not exist, i.e. $\mathfrak{c}(Bx_1^{\mathrm{pref}}) = \mathfrak{c}(A)$, we put again $\mathfrak{o}(A) = B$.
- If $b = x_i^{\mathrm{suff}} \in \mathrm{X}^{\mathrm{suff}}$ (which means $B = \epsilon$, $\mathfrak{c}(A) = \mathfrak{c}(x_i^{\mathrm{suff}})$), we put $\mathfrak{o}(A) = \overline{x}_i^{\mathrm{suff}}$.

The mapping $\mathfrak{j} : \mathcal{EF} \setminus \{\epsilon\} \to \mathcal{EF} \cup \overline{\mathrm{X}}^{\mathrm{pref}}$ is defined dually. The following lemma is a trivial consequence of the definitions of $\mathfrak{o}$ and $\mathfrak{j}$.

Lemma 3.3.2. *Let $A \in \mathcal{EF} \setminus \{\epsilon\}$ and let $\alpha : \mathcal{X} \to \mathcal{C}^*$ be a compatible mapping on $\mathcal{X}$ such that $\alpha \approx^I_{\mathcal{X}} \nu$. Then $0(\alpha(A)) = \alpha(\mathfrak{o}(A))$ and $1(\alpha(A)) = \alpha(\mathfrak{j}(A))$.*

Now we can formulate the desired lemma.

Lemma 3.3.3. *Let $L, R \in \mathcal{EF} \setminus \{\epsilon\}$ and let α be a compatible mapping on $\mathcal{X}$ such that $\alpha \approx^I_{\mathcal{X}} \nu$. Then α is a solution of the equation $L =^? R$ if and only if $\mathfrak{c}(L) = \mathfrak{c}(R)$ and α is a solution of the system $\{\mathfrak{o}(L) =^? \mathfrak{o}(R), \mathfrak{j}(L) =^? \mathfrak{j}(R)\}$.*

Proof. It follows from Proposition 1.1.3, the property (3.11) and Lemma 3.3.2. □

We have introduced all required technical notation, so we are ready to describe the algorithm. The algorithm exchanges certain equations $L =^? R$ from the current system satisfying $\mathfrak{c}(L) = \mathfrak{c}(R)$ for $\mathfrak{o}(L) =^? \mathfrak{o}(R)$ and $\mathfrak{j}(L) =^? \mathfrak{j}(R)$ in each step. It does this in such a way that a variable from the set $\overline{\mathrm{X}}$ appears at most one time in the process. It is possible because $\overline{x}_i^{\mathrm{pref}} \in \overline{\mathrm{X}}$ and $\overline{x}_i^{\mathrm{suff}} \in \overline{\mathrm{X}}$ can appear in the equations $\mathfrak{o}(L) =^? \mathfrak{o}(R)$ or $\mathfrak{j}(L) =^? \mathfrak{j}(R)$ only if $\|\mathfrak{c}(L)\| = \|\mathfrak{c}(R)\| = i$. That is why we exchange in every step exactly all equations $L =^? R$ with maximal sets $\mathfrak{c}(L) = \mathfrak{c}(R)$. If such an equation containing a variable from the set $\overline{\mathrm{X}}$ occurs after the exchange, then we put it aside to a new system $S^E \cup S^X$; more precisely, we put the equations of the form $\overline{x}_i^{\mathrm{pref}} =^? \overline{y}_i^{\mathrm{pref}}$ to the system S^E and the equations of the form $\overline{x}_i^{\mathrm{pref}} =^? A$, where $A \in \mathcal{EF}$, to the system S^X. In this way we guarantee that the current system S is always a subset of the set $\mathcal{EF} \times \mathcal{EF}$.

The algorithm.
The algorithm uses systems of equations S, S^E, S^X, S', T and S_k^E, S_k^X for $k \in \{1, \ldots, m\}$ which are specified as follows. The system S is modified within the algorithm, but all the time

$$S \subseteq \mathcal{EF} \times \mathcal{EF}.$$

The systems S^E and S^X are constructed in the algorithm in such a way that

$$S^E = \bigcup_{k=1}^{m} S_k^E \quad \text{and} \quad S^X = \bigcup_{k=1}^{m} S_k^X,$$

where S_k^E and S_k^X are constructed when the main cycle of the algorithm is performed for the parameter k and

$$S_k^E \subseteq \overline{\mathrm{X}}_k^{\mathrm{pref}} \times \overline{\mathrm{X}}_k^{\mathrm{pref}} \cup \overline{\mathrm{X}}_k^{\mathrm{suff}} \times \overline{\mathrm{X}}_k^{\mathrm{suff}}, \quad S_k^X \subseteq \overline{\mathrm{X}}_k \times \mathcal{EF}.$$

We denote

$$S' = S \cup S^E \cup S^X.$$

Finally, the system T is auxiliary and is used locally during each iteration of the main cycle.

Initialization
Put $T = \emptyset$ and $S_k^E = S_k^X = \emptyset$ for any $k \in \{1, \ldots, m\}$. Delete trivial equations and check whether $\mathfrak{c}(L) = \mathfrak{c}(R)$ holds for all equations $L =^? R$ in S. If $\mathfrak{c}(L) \neq \mathfrak{c}(R)$ for some equation, then stop with the answer 'unsolvable".

Main cycle.
For any k from m down to 1 perform the following three steps.

1. Replacing of equations.

Put to the system T all non-trivial equations $\mathfrak{o}(L) =^? \mathfrak{o}(R)$ and $\mathfrak{j}(L) =^? \mathfrak{j}(R)$ where $L =^? R$ is an equation in S satisfying $\|\mathfrak{c}(L)\| = k$. Then remove all equations $L =^? R$ with $\|\mathfrak{c}(L)\| = k$ from S.

Remark 3.3.4. It follows directly from the definition of $\mathfrak{o}$ that the variable $\overline{x}_i^{\text{pref}}$ can not occur in any equation $\mathfrak{o}(L) =^? \mathfrak{o}(R)$. If $\overline{x}_i^{\text{pref}}$ occurs in some non-trivial equation $\mathfrak{j}(L) =^? \mathfrak{j}(R) \in T$ in this step, then $i = k$ and this equation is of the form $\mathfrak{j}(L) =^? \overline{x}_i^{\text{pref}}$ or $\overline{x}_i^{\text{pref}} =^? \mathfrak{j}(R)$ where $\overline{x}_i^{\text{pref}} \notin \mathfrak{c}(\mathfrak{j}(L))$, $\overline{x}_i^{\text{pref}} \notin \mathfrak{c}(\mathfrak{j}(R))$ respectively. Analogically $\overline{x}_i^{\text{suff}}$ can appear in T only as one side of an equation $\mathfrak{o}(L) =^? \mathfrak{o}(R)$ and only if $i = k$.

2. Construction of systems S_k^E and S_k^X.

Define S_k^E as $T \cap (\overline{\mathrm{X}}_k \times \overline{\mathrm{X}}_k)$ and remove these equations from T. Let E_k be the equivalence relation on $\overline{\mathrm{X}}_k$ generated by S_k^E and let Q_k be the partition on $\overline{\mathrm{X}}_k$ corresponding to E_k. For every $q \in Q_k$ choose some $A \in \mathcal{EF}$ such that there is an equation $A =^? \overline{y}$ or $\overline{y} =^? A$ in T for some $\overline{y} \in q$; if such A does not exist, do nothing for this q. Then put the equation $\overline{y} =^? A$ to S_k^X and replace all equations $B =^? \overline{z}$ and $\overline{z} =^? B$ in T, where $\overline{z} \in q$, $B \in \mathcal{EF}$, with $A =^? B$.

Remark 3.3.5. From Remark 3.3.4 we have $S_k^E \subseteq \overline{\mathrm{X}}_k^{\text{pref}} \times \overline{\mathrm{X}}_k^{\text{pref}} \cup \overline{\mathrm{X}}_k^{\text{suff}} \times \overline{\mathrm{X}}_k^{\text{suff}}$, $S_k^X \subseteq \overline{\mathrm{X}}_k \times \mathcal{EF}$ and $T \subseteq \mathcal{EF} \times \mathcal{EF}$.

3. Negative answer.

If there is an equation $L =^? R$ in T such that $\mathfrak{c}(L) \neq \mathfrak{c}(R)$, then stop with the answer 'unsolvable". If it is not the case, transfer all non-trivial equations from T to S.

Remark 3.3.6. If the algorithm transfers some equation $L =^? R$ from T into the system S, then $\mathfrak{c}(L) = \mathfrak{c}(R)$, $L, R \in \mathcal{EF}$ and $\|\mathfrak{c}(L)\| = k - 1$. This means that at the end of this step S contains only non-trivial equations $L =^? R$ such that $\mathfrak{c}(L) = \mathfrak{c}(R)$, $L, R \in \mathcal{EF}$ and $\|\mathfrak{c}(L)\| < k$. In particular, for $k = 1$ the resulting system S is empty.

Positive answer.
Stop with the answer 'solvable".

Lemma 3.3.7. *The algorithm stops with the correct answer. Moreover, if the original system S is solvable then it is equivalent to the resulting system S'.*

Proof. It is clear that the initialization of the algorithm is correct. The proof will be complete if we prove the following two claims:
Claim 1: For any k, either the algorithm stops with the answer 'unsolvable"

in the third step of the main cycle and there is no compatible solution of the current system S' or the three steps of the cycle modify the current system S' to an equivalent one.

Claim 2: If the algorithm stops with the answer 'solvable", then there is a compatible solution of the resulting system S'.

Proof of Claim 1: From Lemma 3.3.3 we know that the first step of the cycle exchanges the current system S' for the equivalent system $S' \cup T$. It is easy to see that the second step also modifies the system $S' \cup T$ to an equivalent one. If in the third step there is some equation $L =^? R$ in T such that $\mathfrak{c}(L) \neq \mathfrak{c}(R)$, then the system $S' \cup T$ is unsolvable and hence the previous system S' was also unsolvable. If there is no such equation in T, then the algorithm just moves all non-trivial equations from T to S', so the resulting system S' is equivalent to the previous system $S' \cup T$.

Proof of Claim 2: If the algorithm stops with the answer 'solvable", then the resulting system S is empty by Remark 3.3.6. Hence $S' = S^E \cup S^X$. Let $E = \bigcup_{k=1}^{m} E_k$ and $Q = \bigcup_{k=1}^{m} Q_k$. Then E is the equivalence relation on the set $\overline{\mathrm{X}}$ generated by S^E and Q is the partition on $\overline{\mathrm{X}}$ corresponding to E. The following properties also follow from the construction. If $\overline{x}_i^{\mathrm{pref}} =^? A \in S^X$ then $\|\mathfrak{c}(A)\| = i - 1$; in particular, $A = \epsilon$ when $i = 1$. If $\overline{y} =^? \overline{z} \in S^E$, where $y, z \in \mathrm{X}^{\mathrm{pref}} \cup \mathrm{X}^{\mathrm{suff}}$, then $\mathfrak{c}(y) = \mathfrak{c}(z)$ because $\mathfrak{c}(y) = \mathfrak{c}(L) = \mathfrak{c}(R) = \mathfrak{c}(z)$ where $L =^? R$ is the equation from which $\overline{y} =^? \overline{z}$ was created in the first step of the main cycle.

We construct a new system $\overline{S}$ which has the property that any compatible solution of the system $\overline{S}$ is a compatible solution of the system $S^E \cup S^X$. If for a class $q \in Q$ there is an equation $\overline{y} =^? A$ in S^X, where $\overline{y} \in q$, $A \in \mathcal{EF}$, then it is the unique such equation in S^X. In this case we put to the system $\overline{S}$ all equations from the set $q \times \{A\}$. If for a class $q \in Q_k \subseteq Q$ there is no equation $\overline{y} =^? A$ in S^X, where $\overline{y} \in q$ and $A \in \mathcal{EF}$, then we choose any word $u \in \mathcal{C}^*$ such that $u_\sharp = k - 1$ and $\mathfrak{c}(u) \subseteq \mathfrak{c}(y)$ for every $\overline{y} \in q$ (recall that all $\mathfrak{c}(y)$ are equal). In this case we put to the system $\overline{S}$ all equations from the set $q \times \{u\}$. So, the resulting system $\overline{S}$ is a subset of the set $\overline{\mathrm{X}} \times (\mathcal{EF} \cup \mathcal{C}^*)$ and for each $\overline{y} \in \overline{\mathrm{X}}$ the system $\overline{S}$ contains exactly one equation $\overline{y} =^? A$. Note that this equation is $\overline{y} =^? \epsilon$ for all $\overline{y} \in \overline{\mathrm{X}}_1$. Now, the system $S^E \cup S^X$ is clearly equivalent to a subsystem of the system $\overline{S}$, hence any compatible solution of $\overline{S}$ is a compatible solution of $S^E \cup S^X$.

We construct a solution α of the system $\overline{S}$ inductively with respect to $\|\mathfrak{c}(\omega)\|$ for $\omega \in \mathcal{X}$. At first we define $\alpha(\omega)$ for all $\omega \in \mathcal{X}$ such that $\|\mathfrak{c}(\omega)\| = 0$:

$$\alpha(\overline{x}_1^{\mathrm{pref}}) = \alpha(\overline{x}_1^{\mathrm{suff}}) = \epsilon.$$

Now assume that we have already determined $\alpha(\omega)$ for all $\omega \in \mathcal{X}$ satisfying $\|\mathfrak{c}(\omega)\| < i$ and let us determine $\alpha(\omega)$ for all $\omega \in \mathcal{X}$ satisfying $\|\mathfrak{c}(\omega)\| = i$. We do this in the following order: we define $\alpha(y)$ for $y \in \mathrm{X}^{\mathrm{ext}}$ so that α is compatible and then $\alpha(\overline{y})$ for $\overline{y} \in \overline{\mathrm{X}}$ so that α is a solution of the system $\overline{S}$.

For $x \in \mathrm{X}$ with $\|\mathfrak{c}(x)\| = i > 1$ we put

$$\alpha(x) = \alpha(x_{i-1}^{\mathrm{pref}})\, ab\, \alpha(x_{i-1}^{\mathrm{suff}})$$

where a and b are uniquely determined by the conditions $a \in \mathfrak{c}(x) \setminus \mathfrak{c}(x_{i-1}^{\mathrm{pref}})$ and $b \in \mathfrak{c}(x) \setminus \mathfrak{c}(x_{i-1}^{\mathrm{suff}})$. (Recall that there is no $x \in \mathrm{X}$ satisfying $\|\mathfrak{c}(x)\| = \nu(x)_\sharp = 1$.)

For $i = 1$ we put $\alpha(x_1^{\mathrm{pref}}) = \nu(x_1^{\mathrm{pref}})$ and $\alpha(x_1^{\mathrm{suff}}) = \nu(x_1^{\mathrm{suff}})$ for all $x \in \mathrm{X}$. For $i > 1$ we put

$$\alpha(x_i^{\mathrm{pref}}) = \alpha(x_{i-1}^{\mathrm{pref}})\, ab\, \alpha(\overline{x}_i^{\mathrm{pref}})$$

where a and b are uniquely determined by the conditions $a \in \mathfrak{c}(x_i^{\mathrm{pref}}) \setminus \mathfrak{c}(x_{i-1}^{\mathrm{pref}})$ and $b \in \mathfrak{c}(x_i^{\mathrm{pref}}) \setminus \mathfrak{c}(\overline{x}_i^{\mathrm{pref}})$. We define $\alpha(x_i^{\mathrm{suff}})$ analogically.

Now we have fixed $\alpha(y)$ for all $y \in \mathrm{X}^{\mathrm{ext}}$, $\|\mathfrak{c}(y)\| \leq i$. Let $\overline{x}_{i+1}^{\mathrm{pref}} =^? A \in \overline{S}$, where $A \in \mathcal{EF} \cup \mathcal{C}^*$. Then $\|\mathfrak{c}(A)\| = i$, so $\alpha(y)$ is already given for all $y \in \mathsf{c}(A) \subseteq \mathcal{C} \cup \mathrm{X}^{\mathrm{ext}}$. Hence we can correctly define $\alpha(\overline{x}_{i+1}^{\mathrm{pref}}) = \alpha(A)$ and analogically $\alpha(\overline{x}_{i+1}^{\mathrm{suff}}) = \alpha(B)$ for $\overline{x}_{i+1}^{\mathrm{suff}} =^? B \in \overline{S}$.

It is clear that α is a compatible solution of the system $\overline{S}$, so it is a compatible solution of the system $S^E \cup S^X$. This concludes the proof of Claim 2 and also the proof of the lemma. □

Remark 3.3.8. The solution α constructed in the previous proof is up to $\sim$ unique compatible solution of the system $\overline{S}$, i.e. if β is a compatible solution of the system $\overline{S}$, then $\alpha(\omega) \sim \beta(\omega)$ for all $\omega \in \mathcal{X}$, which is in fact equivalent to $\alpha(x) \sim \beta(x)$ for all $x \in \mathrm{X}$.

Let us turn our attention to the complexity aspects.

Lemma 3.3.9. *The algorithm decides the* Simple AIU-Unification *problem in polynomial time.*

Proof. The size of the set $\mathcal{EF}$ is polynomial in the size of the original system (S, ν). Really, for an arbitrary word $U \in (\mathcal{C} \cup \mathrm{X})^*$ we have $\|\mathcal{EF}(U)\| < m^2 \cdot (|U| + 1)^2$. Hence we obtain a polynomial bound for the number of equations which can occur within the algorithm. Finally, it is easy to see that for fixed k the three steps of the main cycle take only polynomial time in the size of all equations $L =^? R$ from S with the property $\|\mathfrak{c}(L)\| = k$. □

The previous lemma concludes the proof of the proposition. □

Example 3.3.10. We return to Example 2.3.12. Let $\nu(x_i) = a_0 a_1 \ldots a_i \ldots a_1 a_0$ for all $i \in \{1, \ldots, n\}$. This means that $\nu(x_i)_\sharp = i + 1$ and $|\nu(x_i)| = 2i + 1$. We demonstrate the computation of our algorithm on the system (S, ν) with complete information about invariants. At the beginning of the first step of the main cycle for $k \in \{3, \dots, n+1\}$ the current system S contains the original equations $x_i =^? x_{i-1}a_ix_{i-1}$ for $1 < i < k$, the original equation $x_1 =^? a_0a_1a_0$ and the following equations produced by the algorithm:

$$(x_i)_k^{\mathrm{pref}} =^? (x_{i-1})_k^{\mathrm{pref}} \tag{3.12}$$

for $i = n, n-1, \dots, k+1$ and

$$(x_k)_k^{\mathrm{pref}} =^? x_{k-1}, \tag{3.13}$$

and the analogical equations for suffixes.

If we apply $\mathfrak{o}$ to the equations (3.12) and (3.13), we obtain the equations

$$(x_i)_{k-1}^{\mathrm{pref}} =^? (x_{i-1})_{k-1}^{\mathrm{pref}} \tag{3.14}$$

for $i = n, n-1, \dots, k$. If we apply $\mathfrak{j}$ to the equations (3.12) and (3.13), we obtain the equations

$$\overline{(x_i)_k^{\mathrm{pref}}} =^? \overline{(x_{i-1})_k^{\mathrm{pref}}} \tag{3.15}$$

for $i = n, n-1, \dots, k+1$ and

$$\overline{(x_k)_k^{\mathrm{pref}}} =^? (x_{k-1})_{k-1}^{\mathrm{suff}}. \tag{3.16}$$

We also get the equation

$$(x_{k-1})_{k-1}^{\mathrm{pref}} =^? x_{k-2} \tag{3.17}$$

after applying $\mathfrak{o}$ to the equation $x_{k-1} =^? x_{k-2}a_{k-1}x_{k-2}$. This means that the system T computed in the first step of the cycle for k consists of the equations (3.14)–(3.17) and the analogical equations for suffixes.

In the second step, the partition Q_k consists of two classes. One of them contains $\overline{(x_i)_k^{\mathrm{pref}}}$ for all $i = n, \dots, k$ and the other contains $\overline{(x_i)_k^{\mathrm{suff}}}$ for all $i = n, \dots, k$. In this step we only put the equations (3.15) (and the analogical equations for suffixes) to the system S^E and the equation (3.16) (and the analogical equation for suffixes) to the set S^X. In the third step we transfer the equations (3.14) and (3.17) to the system S (they are the equations (3.12) and (3.13) for $k-1$).

For $k = 2$ the algorithm produces the equation

$$(x_1)_1^{\mathrm{pref}} =^? a_0 \tag{3.18}$$

instead of the equation (3.17). So, after this step S consists of the equations (3.12) where $k = 1$ and the equation (3.18) (and the analogical equations for suffixes). For $k = 1$ the applications of $\mathfrak{o}$ and $\mathfrak{j}$ to the equations (3.12) and (3.18) give the trivial equation $\epsilon =^? \epsilon$ and the equation

$$\overline{(x_1)_1}^{\text{pref}} =^? \epsilon. \tag{3.19}$$

This means that the original system is solvable and equivalent to the system $S^E \cup S^X$, where S^E consists of the equations (3.15) for all $k, i \in \mathbb{N}$ such that $1 \leq k < i \leq n$ and the analogical equations for suffixes, and S^X consists of the equations (3.16) for $2 \leq k \leq n$, the equation (3.19) and the analogical equations for suffixes. Note that S^E contains $n \cdot (n + 1)$ equations and S^X contains $2n$ equations.

The system $\overline{S}$ constructed in the proof of Lemma 3.3.7 contains for any pair $i, j \in \mathbb{N}$, such that $1 < j \leq i \leq n$, the equations

$$\overline{(x_i)_j}^{\text{pref}} =^? (x_{j-1})_{j-1}^{\text{suff}} \quad \text{and} \quad \overline{(x_i)_j}^{\text{suff}} =^? (x_{j-1})_{j-1}^{\text{pref}}$$

and the equations from the set $\overline{\mathrm{X}}_1 \times \{\epsilon\}$. Because $\overline{S}$ is constructed without any choice, the system (S, ν) with complete information about invariants has a unique solution up to $\sim$ (consisting of Zimin words).

Remark 3.3.11. The identity problem for the idempotent monoids can be viewed as a special case of the Simple AIU-Unification problem — as usual the system consists of one equation $u =^? v$, which does not contain variables. This means that Proposition 1.1.3 can be used for a decision of the word problem for the free idempotent monoid in polynomial time. It may be a little surprise, because on the first sight we need to compute exponentially many pairs of words like $0(1(u))$ and $0(1(v))$ and to compare their contents. But all these words are factors of u and v, so there are only polynomially many such pairs of words. This is a basic reason why our algorithm is polynomial.

3.4 Consequences

We present a few consequences of our new method of solving equations in free idempotent monoids in this section.

3.4.1 AIU-Unification Problem

The main result of this chapter is the following theorem.

Theorem 3.4.1. *The* AIU-Unification *problem and the* AI-Unification *problem are NP-complete.*

Proof. The NP-hardness is a consequence of Proposition 2.1.1.

Let S be a system. We can guess a mapping $\nu : \mathrm{X} \to \mathcal{C}^*$ such that $|\nu(x)| \leq 2 \cdot \|\mathcal{C}\|$ for all $x \in \mathrm{X}$ and then check in polynomial time whether the system (S, ν) with complete information about invariants has a solution by Proposition 3.3.1. The difference between the AIU-Unification and AI-Unification problems is only in the additional assumption $\nu(x) \neq \epsilon$ for all $x \in \mathrm{X}$. $\square$

3.4.2 General AIU-Unification Problem

We want to apply Proposition 1.2.12 to obtain that the general AIU-unification problem and the general AI-unification problem are also in NP. If we solve the AIU-unification problem with linear constant restriction, we can use the same trick as in the proof of Lemma 3.2.2.

Lemma 3.4.2. *There exists a polynomial reduction from the AI-unification problem with linear constant restriction to the* AIU-Unification *problem.*

Proof. If we consider an instance of the AI-unification problem with linear constant restriction, then for any variable x we have a list of constants which can not occur in the word $\alpha(x)$, where α is a unifier of the instance. This means that we have the condition $\mathsf{c}(\alpha(x)) \subseteq \{a_1, \ldots, a_n\}$, which can be ensured by the equation (3.3) as we have seen in the proof of Lemma 3.2.2. In this manner we can construct a system of equations which has the same set of unifiers as the given instance of the AI-unification problem with linear constant restriction. $\square$

It is easy to see that the same holds in the case of the AI-unification problem and we can formulate the desired result.

Theorem 3.4.3. *The general AIU-unification and AI-unification problems are NP-complete.*

Proof. It is an immediate consequence of Proposition 1.2.12, Lemma 3.4.2 and Theorem 3.4.1. $\square$

3.4.3 AIU-Matching Problem — Revised

Let S be a solvable matching system and r be the length of the longest right-hand (variable-free) side in the system. We can assume that $r \geq 2$ since otherwise a solution α exists such that $|\alpha(x)| \leq 1$ for any $x \in \mathrm{X}$. If we consider a solution ν of S, then the system (S, ν) with complete information about invariants is solvable. Let $S' = S^E \cup S^X$ be the system computed by the algorithm. It is easy to see that $S^E = \emptyset$. Any equation in S^X is of the form $\overline{y} = A$ where $\overline{y} \in \overline{\mathrm{X}}$ and A is a factor of some right-hand side such that $|A| \leq r - 2$. Hence, we can assume $|\alpha(\overline{y})| \leq r - 2$ for all $\overline{y} \in \overline{\mathrm{X}}$. Then we can inductively prove

$$|\alpha(x_i^{\mathrm{pref}})| \leq i \cdot r$$

for a solution defined by the formulas from the proof of Lemma 3.3.7 (the same holds for x_i^{suff}). Altogether we have

$$|\alpha(x)| \leq 2 \cdot (\nu(x)_\sharp - 1) \cdot r + 2 < 2 \cdot \nu(x)_\sharp \cdot r \leq 2 \cdot \|\mathcal{C}\| \cdot r.$$

This is a bound for the shortest solution of the solvable matching system. It is better than the one we have obtained in Theorem 2.3.10.

Proposition 3.4.4. *Let $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ be a matching system over the set of constants $\mathcal{C}$, i.e. $R_i \in \mathcal{C}^*$. If there exists a solution of the system S, then there exists a solution α satisfying $|\alpha(x)| < 2 \cdot \|\mathcal{C}\| \cdot \max_i\{|R_i|\}$ for all $x \in \mathrm{X}$.*

3.4.4 Number of Solutions

In the proof of Lemma 3.3.7 we choose a word u for certain classes of variables $q \in Q$. By Remark 3.3.8 for any collection of such choices we obtain (up to $\sim$) unique solution of the original system. Moreover, for different collections of choices we obtain different solutions modulo $\sim$. So, we can describe the number of solutions (up to $\sim$) of the given system with complete information about invariants.

Let $\overline{Q}$ be the set of all classes $q \in Q$ such that S^X does not contain any equation $\overline{y} =^? A$, where $\overline{y} \in q$ and $A \in \mathcal{EF}$. For each class $q \in Q$ we denote $\hat{q} = i$ if $\overline{x}_i^{\mathrm{pref}} \in q$ or $\overline{x}_i^{\mathrm{suff}} \in q$. This means that the word u chosen for the class $q \in \overline{Q}$ contains exactly $\hat{q} - 1$ letters. Let d_n be the number of all $\sim$-classes of words over an n-element alphabet which contain all n letters. By Proposition 1.1.3 we have $d_n = n^2 \cdot d_{n-1}^2$ for all $n \in \mathbb{N}$, so

$$d_n = \prod_{k=1}^{n-1} (n - k + 1)^{2^k}.$$

Then the number of all solutions (up to $\sim$) of the system with complete information about invariants is given by the formula

$$\prod_{q \in \overline{Q}} \hat{q} \cdot d_{\hat{q}-1}.$$

Chapter 4

Unification in Proper Varieties of Bands

In this chapter we deal with the unification problem modulo binary function symbol which is associative, idempotent and satisfies some additional identities. The dichotomy theorem for complexity of unification problems is formulated in Section 4.1 and proved in Sections 4.2 – 4.4. The content of these sections is identical with paper [31]. New results, that concern complexity of the unification problem in theories with more associative and idempotent function symbols, are contained in the last section.

4.1 Formulation of Results

The main results of the chapter are the following two theorems which describe (in the case P$\neq$NP) the borderline between varieties with UNIFICATION (resp. MATCHING) problems decidable in polynomial time and varieties with NP-complete UNIFICATION (resp. MATCHING) problems. This borderline is depicted on Figure 4.1.

Theorem 4.1.1. *Let $\mathcal{V}$ be a variety of bands. The $\mathcal{V}$-UNIFICATION problem is decidable in polynomial time if $\mathcal{V} \subseteq \mathcal{NB}$ and it is NP-complete otherwise.*

Proof. The statement is a consequence of Propositions 4.2.4, 4.3.2 and 4.4.2 below in the case of proper subvarieties of $\mathcal{B}$. For $\mathcal{B}$ it is Theorem 3.4.1. $\square$

Theorem 4.1.2. *Let $\mathcal{V}$ be a variety of bands. The $\mathcal{V}$-MATCHING problem is decidable in polynomial time if $\mathcal{V} \subseteq \mathcal{RB}$ and it is NP-complete otherwise.*

Proof. It is a consequence of Propositions 4.2.4, 4.3.1, 4.4.7 below and Corollary 2.3.11. $\square$

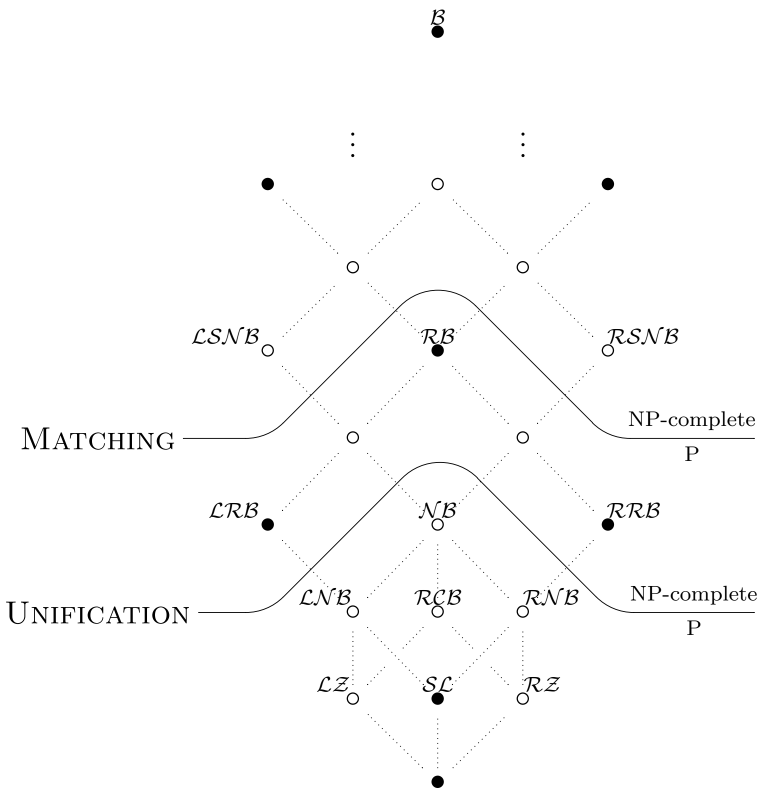

Figure 4.1: The complexity of the UNIFICATION and MATCHING problems in the subvarieties of the variety of all bands $\mathcal{B}$.

Both theorems can be formulated for the UNIFICATION and MATCHING problems for the varieties of idempotent monoids as well. It is easy to see that the following proofs work also if the empty word is allowed in the case of varieties of the form $\mathbf{S}(\mathcal{V})$ for some variety of idempotent monoids $\mathcal{V}$. The varieties of the form $\mathbf{S}(\mathcal{V})$ are noted by filled black circles on Figure 4.1.

Corollary 4.1.3. *Let $\mathcal{V}$ be an arbitrary variety of idempotent monoids. The $\mathcal{V}$-UNIFICATION problem is decidable in polynomial time if $\mathcal{V} \subseteq \mathcal{CIM}$ and it is NP-complete otherwise. The $\mathcal{V}$-MATCHING problem is decidable in polynomial time if $\mathcal{V} \subseteq \mathcal{RIM}$ and it is NP-complete otherwise.*

4.2 NP Algorithms

Lemma 4.2.1. *Let k and n be positive integers, $n > 1$. Then the number of all iterated invariants $\mu \in \{0,1\}^+$ such that $|\mu| \leq n$ and $\text{block}(\mu) = k$ is equal to*

$2\binom{n}{k}$ *and the number of all iterated invariants* $\mu \in \{0,1\}^*$ *such that* $|\mu| \leq n$ *and* $\mathrm{block}(\mu) < k$ *is less than* $2n^k$.

Proof. Each invariant $\mu \in \{0,1\}^+$ such that $|\mu| \leq n$ and $\mathrm{block}(\mu) = k$ is determined by $\mathrm{last}(\mu)$ and by the positions of ends of blocks. Hence there are $2\binom{n}{k}$ of them. The number of all invariants $\mu \in \{0,1\}^*$ such that $|\mu| \leq n$ and $\mathrm{block}(\mu) < k$ is

$$K = 1 + \sum_{l=1}^{k-1} 2\binom{n}{l}.$$

Because $\binom{n}{l} \leq n^l$, we get

$$K \leq 1 + 2\sum_{l=1}^{k-1} n^l < 2\sum_{l=0}^{k-1} n^l = 2\frac{n^k - 1}{n-1} < 2n^k$$

for $n > 1$. □

Remark 4.2.2. From Proposition 1.1.5 and Lemma 4.2.1 we have a polynomial-time algorithm for solving the identity problem for an arbitrary proper subvariety of the variety of all bands, because we must test only polynomially many invariants for given two words (see Remark 1.1.7).

Lemma 4.2.3. *Let* $\mathcal{C}$ *be a finite set and* $k \in \mathbb{N}$. *Let* K *denote the number of all iterated invariants* $\mu \in \{0,1\}^*$ *with the properties* $|\mu| < \|\mathcal{C}\|$ *and* $\mathrm{block}(\mu) < k$. *Then for an arbitrary word* $u \in \mathcal{C}^+$ *there exists a word* $w \in \mathcal{C}^+$ *such that*

$$w \sim_{\mathcal{V}_k} u \quad \textit{and} \quad |w| < 2K.$$

Proof. Let T be the set of all iterated invariants $\mu \in \{0,1\}^*$ which satisfy the conditions $|\mu| < u_\sharp$ and $\mathrm{block}(\mu) < k$. Then $w \sim_{\mathcal{V}_k} u$ if and only if $\mathrm{inv}_w(\mu) = \mathrm{inv}_u(\mu)$ for all $\mu \in T$, because $\mathrm{inv}_u(\mu) = \emptyset$ for $\mu \in \{0,1\}^*$ such that $|\mu| \geq u_\sharp$. We order the set T in this way: $\mu_1 \leq \mu_2$ if and only if μ_2 is a suffix of μ_1. We can view T as a binary rooted tree with the root e where each node μ has successors 0μ and 1μ if they are in T. Now we define inductively the mapping $\omega : T \to \mathcal{C}^+$. If μ is a leaf, i.e. $|\mu| = u_\sharp - 1$, then the word $\omega(\mu)$ is the unique letter from the set $\mathrm{inv}_u(\mu)$. If the node μ has only one successor 0μ (resp. 1μ) then $\omega(\mu) = \omega(0\mu)a$ (resp. $\omega(\mu) = a\omega(1\mu)$) where a is determined by the condition $a \in \mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(0\mu)$ (resp. $a \in \mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(1\mu)$). Finally, if the node μ has both successors 0μ and 1μ then $\omega(\mu) = \omega(0\mu)ab\omega(1\mu)$ where $a \in \mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(0\mu)$ and $b \in \mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(1\mu)$.

We prove by induction on the tree structure of T (starting from leaves) that for each $\mu \in T$

$$\mathsf{c}(\omega(\mu)) = \mathrm{inv}_u(\mu). \tag{4.1}$$

If $\mu \in T$ is a leaf then it is clear. Let $\mu \in T$ be an arbitrary node and assume that (4.1) holds for all successors of μ. We distinguish between two cases, namely the case when μ has only one successor $\iota\mu$ where $\iota \in \{0,1\}$ and the case when μ has both successors. In the first case $\mathsf{c}(\omega(\iota\mu)) = \mathrm{inv}_u(\iota\mu)$ by the assumption and from the definition of $\omega(\mu)$ we can deduce

$$\begin{aligned}\mathsf{c}(\omega(\mu)) &= \mathsf{c}(\omega(\iota\mu)) \cup (\mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(\iota\mu)) \\ &= \mathrm{inv}_u(\iota\mu) \cup (\mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(\iota\mu)) = \mathrm{inv}_u(\mu).\end{aligned}$$

In the second case we have

$$\begin{aligned}\mathsf{c}(\omega(\mu)) &= \mathsf{c}(\omega(0\mu)) \cup (\mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(0\mu)) \cup (\mathrm{inv}_u(\mu) \setminus \mathrm{inv}_u(1\mu)) \cup \mathsf{c}(\omega(1\mu)) \\ &= \mathrm{inv}_u(\mu).\end{aligned}$$

So, (4.1) holds for every $\mu \in T$. Now from (4.1) and the definition of $\omega(\mu)$ we can deduce $\iota(\omega(\mu)) = \omega(\iota\mu)$ for any $\mu \in T$ and $\iota \in \{0,1\}$ such that $\iota\mu \in T$. Hence $\lambda(\omega(\mu)) = \omega(\lambda\mu)$ holds for any $\mu \in T$ and $\lambda \in \{0,1\}^*$ such that $\lambda\mu \in T$. Now, for an arbitrary $\mu \in T$ we have $\mathrm{inv}_{\omega(\mathsf{e})}(\mu) = \mathsf{c}(\mu(\omega(\mathsf{e}))) = \mathsf{c}(\omega(\mu)) = \mathrm{inv}_u(\mu)$ by (4.1). This means

$$\omega(\mathsf{e}) \sim_{\mathcal{V}_k} u.$$

Moreover, one can prove by induction on the tree structure of T that for each $\mu \in T$

$$|\omega(\mu)| < 2 \cdot \|\{\ \lambda \in \{0,1\}^* \mid \lambda\mu \in T\}\|.$$

Hence, for $\mu = \mathsf{e}$ we get $|\omega(\mathsf{e})| < 2 \cdot \|T\| = 2K$. $\square$

Proposition 4.2.4. *The $\mathcal{V}$-Unification problem is in NP for an arbitrary proper subvariety $\mathcal{V}$ of the variety $\mathcal{B}$.*

Proof. Let $\mathcal{V}$ be a proper subvariety of the variety $\mathcal{B}$. We denote k the least integer such that $\mathcal{V} \subseteq \mathcal{V}_k$. Let S be a system over a set of variables X and an n-element set of constants $\mathcal{C}$. From Lemmas 4.2.1 and 4.2.3 we know that for an arbitrary word $u \in \mathcal{C}^+$ there is a word w satisfying $|w| < 4(n-1)^k$ and $w \sim_{\mathcal{V}} u$. So, if there is a solution α of the system S, then we can assume that $\mathrm{size}(\alpha(x)) < 4n^k$. Hence we can guess a solution α of this length and then we check whether α solves all equations of the system in polynomial time by Remark 4.2.2. So, the $\mathcal{V}$-Unification problem is in NP. $\square$

4.3 NP-hard Problems

Proposition 4.3.1. *Let $\mathcal{V}$ be a subvariety of the variety of all bands $\mathcal{B}$ which contains some non-regular band, i.e. $\mathcal{V} \not\subseteq \mathcal{RB}$. Then the $\mathcal{V}$-Matching problem is NP-hard.*

Proof. This proof is a modification of the original proof of the NP-hardness of the AI-Matching problem 2.1.1. It is enough to show the proposition for $\mathcal{V} \in [\mathcal{LSNB}, \mathcal{B}]$. The rest is a consequence of left-right duality.

We use a reduction from the 3-SAT problem. For the formula Φ we construct a system of word equations S such that the system S is satisfiable in the varieties from the interval $[\mathcal{LSNB}, \mathcal{B}]$ if and only if Φ is satisfiable. By Remark 1.2.17 we must prove the following two claims: (i) if the formula Φ is satisfiable then there is a solution of the system S in the variety $\mathcal{B}$ and (ii) if the system S has a solution in the variety $\mathcal{LSNB}$ then the formula Φ is satisfiable.

The system S will consist of the following equations (4.2) – (4.6) with constants $\mathcal{C} = \{a, f, t\}$ and variables $\mathrm{X} = \{x, r_1^x, s_1^x, r_2^x, s_2^x \mid x \in \mathit{Var} \cup \overline{\mathit{Var}}\}$ where $\overline{\mathit{Var}} = \{\overline{z} \mid z \in \mathit{Var}\}$ is a disjoint copy of Var. We define $\widetilde{z} = z$ and $\widetilde{\neg z} = \overline{z}$ for $z \in \mathit{Var}$ and for every clause $\Phi_i \equiv l_1 \vee l_2 \vee l_3$ we have the equation

$$f\widetilde{l_1}\widetilde{l_2}\widetilde{l_3}f =^? ftf. \tag{4.2}$$

For each $z \in \mathit{Var}$ we add the equations

$$tz\overline{z}t \;=^?\; tft, \tag{4.3}$$
$$fz\overline{z}f \;=^?\; ftf \tag{4.4}$$

and finally for each $x \in \mathit{Var} \cup \overline{\mathit{Var}}$ we have the following equations:

$$r_1^x x s_1^x \;=^?\; fat, \quad r_1^x a =^? fa, \tag{4.5}$$
$$r_2^x x s_2^x \;=^?\; taf, \quad r_2^x a =^? ta. \tag{4.6}$$

The intuition behind the construction is the following. If a variable $z \in \mathit{Var}$ is true then $z \sim_{\mathcal{V}} t$ and if z is false then $z \sim_{\mathcal{V}} f$. The first equation ensures that at least one literal in each clause is true and the other equations imply consistency, i.e. a literal and its negation cannot be both true (false). In particular, the equation (4.3) means that at least one of z and $\overline{z}$ contains f. Similarly for t and (4.4). The last equations make sure that variables z (and also $\overline{z}$) cannot contain both f and t.

Proof of the claim (i): Suppose that Φ is satisfiable, i.e. there is a valuation v such that $v(\Phi) = T$. Then we can easily define a solution of our system in the variety of all bands $\mathcal{B}$: for all $z \in \mathit{Var}$ such that $v(z) = T$ let $\alpha(z) = t, \alpha(\overline{z}) = f$ and

$$\alpha(r_1^z) = fa, \quad \alpha(s_1^z) = t, \quad \alpha(r_1^{\overline{z}}) = f, \quad \alpha(s_1^{\overline{z}}) = at,$$
$$\alpha(r_2^z) = t, \quad \alpha(s_2^z) = af, \quad \alpha(r_2^{\overline{z}}) = ta, \quad \alpha(s_2^{\overline{z}}) = f,$$

and if $v(z) = F$ then $\alpha(z) = f, \alpha(\overline{z}) = t$ and

$$\alpha(r_1^z) = f, \quad \alpha(s_1^z) = at, \quad \alpha(r_1^{\overline{z}}) = fa, \quad \alpha(s_1^{\overline{z}}) = t,$$

$$\alpha(r_2^z) = ta, \quad \alpha(s_2^z) = f, \quad \alpha(r_2^{\overline{z}}) = t, \quad \alpha(s_2^{\overline{z}}) = af.$$

Proof of the claim (ii): Let us suppose that α is an arbitrary solution of the system S in the variety $\mathcal{LSNB}$ (recall $p \sim_{\mathcal{LSNB}} q$ if and only if $\overset{\Rightarrow}{p} = \overset{\Rightarrow}{q}$) and we find a valuation which satisfies Φ. The equation (4.3) implies $\mathsf{c}(\alpha(x)) \subseteq \{f,t\}$ for all $x \in Var \cup \overline{Var}$. We will conclude that $\mathsf{c}(\alpha(x)) = \{f,t\}$ is not possible. If it is the case, using the equations (4.5) we get that $\alpha(x)$ does not begin with the constant f. For the moment assume that $\alpha(x)$ begins with f. From (4.5) we obtain $f \in \mathsf{c}(\alpha(r_1^x)) \subseteq \{f,a\}$. If $\mathsf{c}(\alpha(r_1^x)) = \{f\}$ then $\mathsf{c}(0(\alpha(r_1^x x s_1^x))) = \{f,t\}$ whereas $\mathsf{c}(0(fat)) = \{f,a\}$, which is a contradiction. Otherwise we have $\mathsf{c}(\alpha(r_1^x)) = \{f,a\}$ and we get $\mathsf{t}(0(\alpha(r_1^x x s_1^x))) = f$ but $\mathsf{t}(0(fat)) = a$. By similar arguments and using the equations (4.6) we obtain that $\alpha(x)$ does not begin with the constant t.

So, there are just two possibilities for $\alpha(x)$, namely $\alpha(x) \sim f$ or $\alpha(x) \sim t$ for any $x \in Var \cup \overline{Var}$. By the equation (4.3) we have $\alpha(z) \sim f$ or $\alpha(\overline{z}) \sim f$ for all $z \in Var$. The equation (4.4) implies either $\alpha(z) \sim t$ or $\alpha(\overline{z}) \sim t$. For each clause $\Phi_i \equiv l_1 \vee l_2 \vee l_3$ the equation (4.2) implies that $\alpha(\widetilde{l_j}) \sim t$ for some j, $1 \leq j \leq 3$. Let us finally define the valuation v as $v(z) = T$ if $\alpha(z) \sim t$ and $v(z) = F$ if $\alpha(z) \sim f$ for each $z \in Var$. The valuation is consistent and $v(\Phi) = T$. $\square$

Proposition 4.3.2. *Let $\mathcal{V}$ be a proper subvariety of the variety of all bands $\mathcal{B}$ which contains some non-normal band, i.e. $\mathcal{V} \nsubseteq \mathcal{NB}$. Then the $\mathcal{V}$-*Unification *problem is NP-hard.*

Proof. It is enough to prove the proposition only for the three varieties from the interval $[\mathcal{LRB}, \mathcal{RB}]$. The rest is a consequence of Proposition 4.3.1 and duality.

We use a reduction from 3-SAT again. For the formula Φ we construct a system S such that the formula Φ is satisfiable if and only if the system S is satisfiable in the varieties from the interval $[\mathcal{LRB}, \mathcal{RB}]$. By Remark 1.2.17 we must prove the following two claims: (i) if Φ is satisfiable then there is a solution of the system S in the variety $\mathcal{RB}$ and (ii) if the system S has a solution in the variety $\mathcal{LRB}$ then Φ is satisfiable.

The system S will consist of the following equations (4.7 – 4.13) with constants $\mathcal{C} = \{a,b,c,t\}$ and variables $\mathrm{X} = \{x, s_x \mid x \in Var \cup \overline{Var}\}$ where $\overline{Var} = \{\overline{z} \mid z \in Var\}$ is a disjoint copy of Var.

We define $\widetilde{z} = z$ and $\widetilde{\neg z} = \overline{z}$ for $z \in Var$ and for every clause $\Phi_i \equiv l_1 \vee l_2 \vee l_3$ we have the equation

$$ab\widetilde{l_1}\widetilde{l_2}\widetilde{l_3}ab =^? abtab. \tag{4.7}$$

For each $z \in \mathit{Var}$ we add the equations

$$zt =^? at, \tag{4.8}$$

$$\overline{z}t =^? bt, \tag{4.9}$$

$$abz\overline{z}ab =^? abtab, \tag{4.10}$$

$$bs_zc =^? bc, \tag{4.11}$$

$$as_{\overline{z}}c =^? ac, \tag{4.12}$$

$$s_zz\overline{z}s_zs_{\overline{z}} =^? s_{\overline{z}}\overline{z}zs_zs_{\overline{z}}. \tag{4.13}$$

Proof of the claim (i): Suppose that Φ is satisfiable, i.e. there is a valuation v such that $v(\Phi) = T$. For all $z \in \mathit{Var}$ such that $v(z) = T$, let $\alpha(z) = at$, $\alpha(\overline{z}) = b$, $\alpha(s_z) = cb$, $\alpha(s_{\overline{z}}) = c$ and if $v(z) = F$ then $\alpha(z) = a$, $\alpha(\overline{z}) = bt$, $\alpha(s_z) = c$, $\alpha(s_{\overline{z}}) = ca$. Checking that α is a solution of S in the variety $\mathcal{RB}$ is a routine.

Proof of the claim (ii): Let α be a solution of the system S in the variety $\mathcal{LRB}$ (recall $p \sim_{\mathcal{LRB}} q$ if and only if $\overrightarrow{p} = \overrightarrow{q}$). From (4.8) and (4.9) we have $a \in \mathsf{c}(\alpha(z)) \subseteq \{a, t\}$, $b \in \mathsf{c}(\alpha(\overline{z})) \subseteq \{b, t\}$ and from (4.10) $t \in \mathsf{c}(\alpha(z\overline{z}))$ for each $z \in \mathit{Var}$. The equations (4.11 – 4.13) serve for elimination of the case $t \in \mathsf{c}(\alpha(z)) \cap \mathsf{c}(\alpha(\overline{z}))$.

Assume for the moment that $t \in \mathsf{c}(\alpha(z)) \cap \mathsf{c}(\alpha(\overline{z}))$. Then $\mathsf{c}(\alpha(z)) = \{a, t\}$ and $\mathsf{c}(\alpha(\overline{z})) = \{b, t\}$. But (4.8) and (4.9) imply $\alpha(z) \sim at$, $\alpha(\overline{z}) \sim bt$, hence the equation (4.13) can be rephrased as

$$s_zatbs_zs_{\overline{z}} =^? s_{\overline{z}}btas_zs_{\overline{z}}.$$

From (4.11) and (4.12) we have $\mathsf{c}(\alpha(s_z)) \subseteq \{b, c\}$ and $\mathsf{c}(\alpha(s_{\overline{z}})) \subseteq \{a, c\}$, so $\alpha(s_z) \sim c$ or $\alpha(s_z) \sim cb$ and $\alpha(s_{\overline{z}}) \sim c$ or $\alpha(s_{\overline{z}}) \sim ca$ because $\mathsf{h}(\alpha(s_z)) = \mathsf{h}(\alpha(s_{\overline{z}}))$ by (4.13). Hence $\alpha(s_zatbs_zs_{\overline{z}}) \sim catb$ or $\alpha(s_zatbs_zs_{\overline{z}}) \sim cbat$ and $\alpha(s_{\overline{z}}btas_zs_{\overline{z}}) \sim cbta$ or $\alpha(s_{\overline{z}}btas_zs_{\overline{z}}) \sim cabt$ and this is a contradiction.

Now we see that there are two possibilities for each variable $z \in \mathit{Var}$, namely $\alpha(z) \sim at$, $\alpha(\overline{z}) \sim b$ or $\alpha(z) \sim a$, $\alpha(\overline{z}) \sim bt$. In the first case we define $v(z) = T$ and in the second case we define $v(z) = F$. From (4.7) we obtain $v(\Phi) = T$. □

4.4 Polynomial Algorithms

Let us now restate Proposition 1.2.23 and present the original proof from [28] because we use it in the proof of Proposition 4.4.2.

Proposition 4.4.1 ([28]). *The $\mathcal{SL}$-*Unification *problem is decidable in polynomial time.*

Proof. For the system $S = \{L_1 =^? R_1, \dots, L_n =^? R_n\}$ we construct a set $\mathcal{H}$ of propositional Horn clauses such that $\mathcal{H}$ is satisfiable if and only if S is satisfiable. (See [23] for an overview on Horn clauses.)

For every variable $x \in \mathrm{X}$ and every constant $a \in \mathcal{C}$ we form the propositional variable $P_{x,a}$. The intuition behind the construction is the following: the variable $P_{x,a}$ is true if and only if $a \notin \mathsf{c}(\alpha(x))$.

For each variable x we have the Horn clause

$$\bigwedge_{a \in \mathcal{C}} P_{x,a} \implies F. \tag{4.14}$$

For every equation $L_i = R_i \in S$ and every constant $a \in (\mathcal{C} \cap \mathsf{c}(R_i)) \setminus \mathsf{c}(L_i)$ we add the clause

$$\bigwedge_{x \in \mathrm{X} \cap \mathsf{c}(L_i)} P_{x,a} \implies F. \tag{4.15}$$

Analogically, we add the clause

$$\bigwedge_{x \in \mathrm{X} \cap \mathsf{c}(R_i)} P_{x,a} \implies F \tag{4.16}$$

for $a \in (\mathcal{C} \cap \mathsf{c}(L_i)) \setminus \mathsf{c}(R_i)$.

Finally, for $L_i = R_i \in S$ and $a \in \mathcal{C} \setminus (\mathsf{c}(L_i) \cup \mathsf{c}(R_i))$ we add the clauses

$$\bigwedge_{x \in \mathrm{X} \cap \mathsf{c}(L_i)} P_{x,a} \implies P_{y,a} \quad \text{for all} \quad y \in \mathrm{X} \cap \mathsf{c}(R_i), \tag{4.17}$$

$$\bigwedge_{x \in \mathrm{X} \cap \mathsf{c}(R_i)} P_{x,a} \implies P_{y,a} \quad \text{for all} \quad y \in \mathrm{X} \cap \mathsf{c}(L_i). \tag{4.18}$$

Note that by $\bigwedge_{x \in \emptyset} P_{x,a}$ we mean T.

It is easy to see that if there is a valuation v which satisfies $\mathcal{H}$ then we can define a solution α of the system S in the following way: $\alpha(x)$ is a product of all constants $a \in \mathcal{C}$ such that $v(P_{x,a}) = F$. The clause (4.14) ensures that $\alpha(x)$ is not the empty word and the clauses (4.15) – (4.18) ensure that α is a solution.

Since satisfiability of a set of propositional Horn clauses can be checked in linear time [15] and the size of the set $\mathcal{H}$ is polynomial in the size of the original system S, the proof is finished. $\square$

Proposition 4.4.2. *The $\mathcal{V}$-*Unification *problem is decidable in polynomial time for each variety $\mathcal{V} \subseteq \mathcal{NB}$.*

Proof. For $\mathcal{LZ}$ (and dually for $\mathcal{RZ}$) it is trivial: we can assume that the system of equations is of the form $\{L_1 = R_1, \ldots, L_n = R_n\}$ with $L_i, R_i \in \mathcal{C} \cup \mathrm{X}$, so it gives a partition of the set $\mathcal{C} \cup \mathrm{X}$. The system is satisfiable if and only if each class of the partition contains at most one constant — if a class M contains a constant a then $\alpha(x) \sim_{\mathcal{LZ}} a$ for all $x \in M \cap \mathrm{X}$ and any solution α. The system is satisfiable in $\mathcal{RCB}$ if and only if it is satisfiable in both $\mathcal{LZ}$ and $\mathcal{RZ}$.

For the $\mathcal{SL}$-Unification problem we use Proposition 4.4.1. Moreover, we can slightly modify the proof of Proposition 4.4.1 for the varieties $\mathcal{LNB}$ ($\mathcal{RNB}$, $\mathcal{NB}$ respectively). First we test whether the given system is satisfiable in $\mathcal{LZ}$ ($\mathcal{RZ}$, $\mathcal{RCB}$ respectively). If it is the case then for any $a \in \mathcal{C}$ and $x \in \mathrm{X}$ such that $a = \mathsf{h}(\alpha(x))$ or $a = \mathsf{t}(\alpha(x))$ holds for an arbitrary solution α of the system in $\mathcal{LZ}$ ($\mathcal{RZ}$, $\mathcal{RCB}$ respectively), we add the Horn clause $P_{x,a} \implies F$ to the set $\mathcal{H}$ constructed in the proof of Proposition 4.4.1. □

The last part of the proof of the result 4.1.2 is about the Matching problem for the variety $\mathcal{LRB}$. For technical reasons we will work with pattern equation systems.

First of all, we describe the solutions of a pattern equation system in the variety $\mathcal{SL}$. In fact, the free semilattice over the set $\mathcal{C}$ is the set of all non-empty subsets of $\mathcal{C}$ with the operation of union; we denote it $\mathcal{P}(\mathcal{C})$. So, a solution α of a system can be viewed as a solution in sets, i.e. $\alpha : \mathrm{X} \to \mathcal{P}(\mathcal{C})$. For two such mappings $\alpha, \beta : \mathrm{X} \to \mathcal{P}(\mathcal{C})$ we say that α is contained in β if and only if $\alpha(x) \subseteq \beta(x)$ for all $x \in \mathrm{X}$. The following fact about the maximum solution of a pattern equation system in sets is obvious.

Lemma 4.4.3. *The pattern equation system $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ is satisfiable in the variety $\mathcal{SL}$ if and only if the mapping $\gamma : \mathrm{X} \to \mathcal{P}(\mathcal{C}) \cup \{\emptyset\}$ defined by $\gamma(x) = \bigcap_{x \in \mathsf{c}(L_i)} \mathsf{c}(R_i)$ for every $x \in \mathrm{X}$ is a solution of S in sets. Moreover, any solution of the system S in sets is contained in γ.*

Remark 4.4.4. If we consider three mappings $\alpha, \beta, \gamma : \mathrm{X} \to \mathcal{P}(\mathcal{C})$ such that α is contained in β and β is contained in γ and α, γ are solutions of a pattern equation system S in sets, then β is also a solution of the system S in sets.

Now we are going to solve pattern equation systems in the variety $\mathcal{LRB}$ (recall $p \sim_{\mathcal{LRB}} q$ if and only if $\overrightarrow{p} = \overrightarrow{q}$). One can view a solution α of a pattern equation system as a solution in sets with orders on all sets $\alpha(x)$ as follows.

For a word $u \in \mathcal{C}^+$ and two distinct letters $a, b \in \mathsf{c}(u)$ we write $a <_u b$ if and only if the first occurrence of the letter a in u precedes all occurrences of the letter b in u. Note that $<_u$ is a strict linear order on the set $\mathsf{c}(u)$ and $u \sim_{\mathcal{LRB}} v$ if and only if $<_u$ and $<_v$ are the same relations on the set $\mathsf{c}(u) = \mathsf{c}(v)$.

For a mapping $\beta : \mathrm{X} \to \mathcal{P}(\mathcal{C})$, a letter $a \in \mathcal{C}$ and a word $L \in \mathrm{X}^+$ we denote by $f_a(L, \beta)$ the first letter x in L such that $a \in \beta(x)$ if such a letter exists.

Let $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ be a given pattern equation system. For a mapping $\beta : \mathrm{X} \to \mathcal{P}(\mathcal{C})$ we define a binary relation

$$\prec_x^\beta \subseteq \beta(x) \times \beta(x)$$

for all $x \in \mathrm{X}$ in this way:

$$a \prec_x^\beta b \quad \text{iff} \quad \exists i \in \{1, \ldots, n\} : f_a(L_i, \beta) = f_b(L_i, \beta) = x, a <_{R_i} b.$$

A mapping $\beta : \mathrm{X} \to \mathcal{P}(\mathcal{C})$ is called a *left-compatible* solution of the system S in sets if and only if the following conditions are satisfied:

(i) β is a solution in sets,

(ii) the transitive closure of $\prec_x^\beta$ is an irreflexive relation for all $x \in \mathrm{X}$,

(iii) $\forall i \in \{1, \ldots, n\}, \forall a, b \in \mathsf{c}(R_i) : f_a(L_i, \beta) <_{L_i} f_b(L_i, \beta) \implies a <_{R_i} b.$

Lemma 4.4.5. *Let $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ be a pattern equation system. Then S is satisfiable in the variety $\mathcal{LRB}$ if and only if there exists a left-compatible solution of S in sets.*

Proof. Let α be a solution of the system S in the variety $\mathcal{LRB}$. We put $\beta(x) = \mathsf{c}(\alpha(x))$. Such β is a left-compatible solution of S in sets. Indeed, (i) is trivial, (ii) follows from the observation $a \prec_x^\beta b \implies a <_{\alpha(x)} b$ and (iii) is clear because $f_a(L_i, \beta) <_{L_i} f_b(L_i, \beta)$ implies $a <_{\alpha(L_i)} b$.

Conversely, let β be a left-compatible solution of the system in sets. There is a linear order on $\beta(x)$ containing $\prec_x^\beta$ for each $x \in \mathrm{X}$, by condition (ii). We denote it $\ll_x$ and define $\alpha(x) = c_1 c_2 \cdots c_m$, where $\{c_1, c_2, \ldots, c_m\} = \beta(x)$ and $c_1 \ll_x c_2 \ll_x \cdots \ll_x c_m$. Such α is a solution of S in the variety $\mathcal{LRB}$. Really, assume for a moment that $a <_{\alpha(L_i)} b$ and $b <_{R_i} a$ for some $i \in \{1, \ldots, n\}$ and $a, b \in \mathcal{C}$. Then $f_a(L_i, \beta) = f_b(L_i, \beta)$ because $f_a(L_i, \beta) <_{L_i} f_b(L_i, \beta)$ implies $a <_{R_i} b$ by condition (iii) and $f_b(L_i, \beta) <_{L_i} f_a(L_i, \beta)$ is a contradiction with $a <_{\alpha(L_i)} b$. If we denote $x = f_a(L_i, \beta) = f_b(L_i, \beta)$ then we have $b \prec_x^\beta a$ and this implies $b <_{\alpha(x)} a$ which is a contradiction with $a <_{\alpha(L_i)} b$. ☐

Lemma 4.4.6. *It is decidable in polynomial time whether a given pattern equation system $S = \{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ has a left-compatible solution in sets.*

Proof. We will exhibit a polynomial algorithm which finds a left-compatible solution of S in sets if it exists. First of all, the algorithm computes (in polynomial time) the maximum solution γ of S in sets by Lemma 4.4.3 (if it exists). We obtain the statement if we apply the following claim to this solution γ of S in

sets. (Any left-compatible solution of S in sets is a solution in sets, so it is contained in the maximum solution of S in sets.)
Claim. Let γ be a solution of S in sets. It is decidable in polynomial time whether there exists a left-compatible solution in sets which is contained in γ. Moreover, if such a solution exists, then there is the maximum left-compatible solution of S in sets which is contained in γ and we can compute it in polynomial time.
Proof of Claim. We construct a polynomial algorithm which finds the maximum left-compatible solution of S in sets contained in γ if it exists. At the beginning let $\delta = \gamma$. In each step the algorithm improves the mapping δ until it becomes a left-compatible solution.

Assume that $\delta : \mathrm{X} \to \mathcal{P}(\mathcal{C}) \cup \{\emptyset\}$ is not a left-compatible solution of S in sets. If (i) is not satisfied for δ, then the algorithm answers that S has no left-compatible solution contained in γ. If (ii) is not satisfied, i.e. there is $x \in \mathrm{X}$ and $a_1 \prec_x^\delta a_2 \prec_x^\delta \dots \prec_x^\delta a_m \prec_x^\delta a_1$, then the algorithm replaces δ with the mapping δ' such that $\delta'(z) = \delta(z)$ for all variables $z \neq x$ and $\delta'(x) = \delta(x) \setminus \{a_1, \dots, a_n\}$. If (iii) is not satisfied, i.e. $x = f_a(L_i, \delta) <_{L_i} f_b(L_i, \delta)$, $b <_{R_i} a$ for some $i \in \{1, \dots, n\}$ and $a, b \in \mathcal{C}$, then the algorithm replaces δ with the mapping δ' such that $\delta'(z) = \delta(z)$ for all variables $z \neq x$ and $\delta'(x) = \delta(x) \setminus \{a\}$.

We must prove that our algorithm stops in polynomial time and that it returns the correct answer. In each step the algorithm computes all relations $\prec_x^\delta$ in polynomial time and it checks the conditions also in polynomial time. Then it reduces the set $\delta(x)$. Because $\delta(x)$ has lower bound $\emptyset$, the algorithm changes each $\delta(x)$ only polynomially many times. So, the algorithm stops in polynomial time.

We show by induction that an arbitrary left-compatible solution β of S in sets which is contained in γ is also contained in every mapping constructed by the algorithm. This is clearly enough to verify the correctness of the algorithm.

Let β be contained in δ which is considered in the algorithm. It is easy to see that δ satisfies (i) by Remark 4.4.4. We show that β is contained in δ'. In general, from the fact $\beta(x) \subseteq \delta(x)$ for all $x \in \mathrm{X}$ we have $f_a(L, \delta) = f_a(L, \beta)$ when $a \in \beta(f_a(L, \delta))$ and $f_a(L, \delta) <_L f_a(L, \beta)$ when $a \notin \beta(f_a(L, \delta))$ for all $L \in \mathrm{X}^+$ and $a \in \mathcal{C}$.

Assume that δ does not satisfy (ii), i.e. there is $x \in \mathrm{X}$ and $a_1 \prec_x^\delta a_2 \prec_x^\delta \dots \prec_x^\delta a_m \prec_x^\delta a_{m+1} = a_1$. Assume for a moment that there is $j \in \{1, \dots, m\}$ such that $a_j \notin \beta(x)$ and $a_{j+1} \in \beta(x)$. From the fact $a_j \prec_x^\delta a_{j+1}$ we can deduce the existence of $i \in \{1, \dots, n\}$ satisfying $f_{a_j}(L_i, \delta) = f_{a_{j+1}}(L_i, \delta) = x$ and $a_j <_{R_i} a_{j+1}$. Now

$$f_{a_{j+1}}(L_i, \beta) = f_{a_{j+1}}(L_i, \delta) = f_{a_j}(L_i, \delta) <_{L_i} f_{a_j}(L_i, \beta)$$

and this is a contradiction with condition (iii) for β. So, there are two possibilities

$\{a_1, \ldots, a_m\} \subseteq \beta(x)$ or $\{a_1, \ldots, a_m\} \cap \beta(x) = \emptyset$. The first one implies

$$a_1 \prec_x^\beta a_2 \prec_x^\beta \ldots \prec_x^\beta a_m \prec_x^\beta a_1$$

and this is a contradiction with condition (ii) for β. The second one means that β is contained in δ'.

Finally, assume that δ does not satisfy (iii), i.e.

$$x = f_a(L_i, \delta) <_{L_i} f_b(L_i, \delta),\ b <_{R_i} a \quad \text{for some} \quad i \in \{1, \ldots, n\} \quad \text{and} \quad a, b \in \mathcal{C}.$$

Assume for a moment that $a \in \beta(x)$. It is not hard to see that $f_a(L_i, \beta) = f_a(L_i, \delta) <_{L_i} f_b(L_i, \beta)$. Hence β does not satisfy condition (iii) and this is a contradiction. So, $a \notin \beta(x)$, i.e. β is contained in δ'. □

Proposition 4.4.7. *The $\mathcal{V}$-Matching problem is decidable in polynomial time for each variety $\mathcal{V} \subseteq \mathcal{RB}$.*

Proof. Let $S = \{L_1 = R_1, \ldots, L_n = R_n\}$ be a pattern equation system. The case $\mathcal{V} \subseteq \mathcal{NB}$ is solved in Proposition 4.4.2. Lemmas 4.4.5 and 4.4.6 verify the case $\mathcal{V} = \mathcal{LRB}$.

For $\mathcal{V} = \mathcal{LRB} \vee \mathcal{NB} = \mathcal{LRB} \vee \mathcal{RNB}$ we test whether the system S is satisfiable in $\mathcal{RNB}$. Then we compute the maximum left-compatible solution δ of S in sets by Lemma 4.4.6 and we check whether $\mathsf{t}(R_i) \in \delta(\mathsf{t}(L_i))$ for all $i \in \{1, \ldots, n\}$. This is sufficient because $\mathsf{t}(R_i) = \mathsf{t}(R_j)$ whenever $\mathsf{t}(L_i) = \mathsf{t}(L_j)$ thanks to the solvability in $\mathcal{RNB}$.

The cases $\mathcal{V} = \mathcal{RRB}$ and $\mathcal{V} = \mathcal{RRB} \vee \mathcal{NB}$ are dual to $\mathcal{LRB}$ and $\mathcal{LRB} \vee \mathcal{NB}$ respectively; we can consider the notion of a right-compatible solution of a system in sets.

For $\mathcal{V} = \mathcal{RB}$ we can consider the notion of a compatible solution of a system in sets (i.e. solution in sets which is left-compatible and at the same time right-compatible). We construct the algorithm which alternates the two algorithms for $\mathcal{LRB}$ and $\mathcal{RRB}$ in such a way that the left-compatible solution δ from the algorithm for $\mathcal{LRB}$ is the first mapping δ in the algorithm for $\mathcal{RRB}$ and vice-versa. So, the algorithm finds a solution of a solvable system in sets which is left-compatible and at the same time right-compatible. The correctness follows from the claim in the proof of Lemma 4.4.6. □

4.5 General Unification in Varieties of Bands

First of all, if we study the general $\mathcal{V}$-unification problem for a variety of bands $\mathcal{V}$, then we can try to apply Proposition 1.2.12 to obtain that the problem belongs to NP as we did in the case of the general AIU-unification problem in

Theorem 3.4.3. If the variety $\mathcal{V}$ contains all semilattices, i.e. $\mathcal{SL} \subseteq \mathcal{V} \subseteq \mathcal{B}$, then the condition $\mathsf{c}(\alpha(x)) \subseteq \{a_1, \ldots, a_n\}$ can be expressed by an equation in the same way as in Lemma 3.4.2. So, we immediately obtain the following result.

Lemma 4.5.1. *Let $\mathcal{V}$ be an arbitrary variety of bands containing all semilattices, i.e. $\mathcal{SL} \subseteq \mathcal{V} \subseteq \mathcal{B}$. Then the general $\mathcal{V}$-unification problem belongs to NP.*

In the case of the variety $\mathcal{SL}$ we have the NP-hardness result 1.2.24 and one can easily modify the original proof from the paper [28] for an arbitrary variety from the interval $[\mathcal{SL}, \mathcal{B}]$. We show a more sophisticated modification, because we want to use it also in a certain different situation later.

In the following proof we consider a signature $\mathcal{F}$ which contains an ai-symbol $\cdot$ and enough other free function symbols. As usual, for the symbol $\cdot$ we will use infix notation where the symbol $\cdot$ is often omitted.

Proposition 4.5.2. *Let $\mathcal{V}$ be an arbitrary variety of bands containing all semilattices, i.e. $\mathcal{SL} \subseteq \mathcal{V} \subseteq \mathcal{B}$. Then the general $\mathcal{V}$-unification problem and the general $\mathcal{V}$-matching problem are NP-complete.*

Note that the statement is a consequence of Theorems 4.1.1 and 4.1.2 in the case $\mathcal{V} \not\subseteq \mathcal{NB}$ for the general $\mathcal{V}$-unification problem and in the case $\mathcal{V} \not\subseteq \mathcal{RB}$ for the general $\mathcal{V}$-matching problem.

Proof. The problems are in NP by Lemma 4.5.1, so it is enough to show that the general $\mathcal{V}$-matching problem is NP-hard for every variety of bands $\mathcal{V}$ such that $\mathcal{SL} \subseteq \mathcal{V} \subseteq \mathcal{B}$.

We show a reduction from the 3-SAT problem. For a given formula $\Phi \equiv \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$ we construct a system of matching equations S with the following properties. If the formula Φ is satisfiable then the system S has a $\mathcal{B}$-unifier and if the system S has an $\mathcal{SL}$-unifier then the formula Φ is satisfiable. The rest is a consequence of Remark 1.2.2.

Let g be a free binary function symbol and 0, 1 be free constant function symbols. Let *Var* be the set of all boolean variables occurring in the formula Φ. For any $x \in \mathit{Var}$ we introduce a new variable $\overline{x}$ and denote $\overline{\mathit{Var}} = \{\overline{x} \mid x \in \mathit{Var}\}$.

For $x \in \mathit{Var}$ we define $\widetilde{x} = x$ and $\widetilde{\neg x} = \overline{x}$. For every clause $\Phi_j \equiv l_1 \vee l_2 \vee l_3$, where l_1, l_2 and l_3 are literals, we put to the system S the equation

$$0\widetilde{l_1}\widetilde{l_2}\widetilde{l_3}0 =^? 010. \tag{4.19}$$

And for each $x \in \mathit{Var}$ we add to the system S the equation

$$g(0,1)g(x,\overline{x})g(1,0) =^? g(0,1)g(1,0). \tag{4.20}$$

If the formula Φ is satisfied by a valuation v, we can define a $\mathcal{B}$-unifier σ in an obvious way: for $x \in \mathit{Var}$ we put $\sigma(x) = 1$ and $\sigma(\overline{x}) = 0$ if $v(x) = T$ and $\sigma(x) = 0$, $\sigma(\overline{x}) = 1$ if $v(x) = F$.

Conversely, let σ be an $\mathcal{SL}$-unifier of the system S and denote by E the theory of signature $\mathcal{F}$ corresponding to the variety $\mathcal{SL}$. From the equation (4.20) we can deduce that $\sigma(g(x, \overline{x}))$ is $=_{\mathrm{E}}$-related to $g(0, 1)$ or $g(1, 0)$ for any $x \in \mathit{Var}$. Hence $\sigma(x)$ and $\sigma(\overline{x})$ are $=_{\mathrm{E}}$-related to 0 or 1 and $\sigma(x) \neq_{\mathrm{E}} \sigma(\overline{x})$. Now we can correctly define a valuation v by the rule $v(x) = T$ iff $\sigma(x) =_{\mathrm{E}} 1$ (and $\sigma(\overline{x}) =_{\mathrm{E}} 0$ at the same time). From the equation (4.19) we can deduce that for every clause $\Phi_j \equiv l_1 \vee l_2 \vee l_3$ at least one of $\sigma(\widetilde{l_1})$, $\sigma(\widetilde{l_2})$ and $\sigma(\widetilde{l_3})$ is $=_{\mathrm{E}}$-related to 1. This ensures $v(\Phi_j) = T$ for any $j \in \{1, \ldots, n\}$. □

Remark 4.5.3. If we consider the theory E containing two independent ai-symbols (and enough constant symbols), then the following modification of the previous proof partly works. Let the function symbol g considered in the proof be also an ai-symbol. Then there are new solutions of the equation (4.20); for example $\sigma(x) = \sigma(\overline{x}) = g(0, 1)g(1, 0)$. If the symbol g is a trivial, left zero or right zero function symbol, then the proof does not work. Otherwise, i.e. if g is a $\mathcal{U}$-symbol for a variety of bands $\mathcal{U}$ containing all semilattices or all rectangular bands, the proof works if for each $x \in \mathit{Var}$ we add to the system S the equation

$$0x\overline{x}0 \quad =^? \quad 010. \tag{4.21}$$

We have to check the arguments which lead to the satisfiability of the formula Φ once again. For any solution σ of S and $x \in \mathit{Var}$ we can see from the equation (4.21) that both $\sigma(x)$ and $\sigma(\overline{x})$ are $=_{\mathrm{E}}$-related to one of the terms 0, 1, 01, 010, 10 and 101. So, we can deduce that $\sigma(x)$ and $\sigma(\overline{x})$ are $=_{\mathrm{E}}$-related to 0 or 1 from (4.20) again. The other arguments are the same.

To complete the description of the complexity of the general $\mathcal{V}$-unification and $\mathcal{V}$-matching problems it remains to discuss only the varieties of trivial, left zero, right zero and rectangular bands. There is nothing to do in the case of the trivial variety (see Remark 1.2.13). Also the case of the variety of left zero semigroups is simple.

Lemma 4.5.4. *The general $\mathcal{LZ}$-unification ($\mathcal{RZ}$-unification respectively) problem is decidable in polynomial time.*

Proof. Let $\cdot$ be an independent $\mathcal{LZ}$-symbol, i.e. $\mathrm{E} = \{x \cdot y \doteq x\}$. One can prove (by induction on the structure of terms) that for any term t there is a unique term s which does not contain the symbol $\cdot$ and satisfies $s =_{\mathrm{E}} t$. The term s can be constructed by replacing any product of subterms with the left subterm. Hence we can eliminate the symbol $\cdot$ from an instance of the problem. Also any

$\mathcal{LZ}$-unifier of the simpler instance can be written without the symbol $\cdot$, so it is a syntactic unifier because two $=_{\mathrm{E}}$-related terms without $\cdot$ are always equal. In this way we reduce the general $\mathcal{LZ}$-unification problem to the syntactic unification problem, which is decidable in polynomial time. □

We show that the general $\mathcal{RCB}$-unification problem is also tractable, although the proof of this fact is more complicated. Now we denote the independent $\mathcal{RCB}$-symbol by $*$ and use the infix notation again.

Proposition 4.5.5. *The general $\mathcal{RCB}$-unification problem is decidable in polynomial time.*

Proof. Our goal is a reduction of the general $\mathcal{RCB}$-unification problem to the syntactic unification problem. Polynomial algorithms solving the syntactic unification problem require a better representation of terms than strings of symbols or trees. In this proof we work with a representation of terms using directed acyclic multigraphs, whose nodes are labelled with function symbols or variables, whose outgoing edges from any node are ordered, and where the outdegree of any node labelled with a symbol f is equal to the arity of f (nodes labelled with variables have outdegree 0); the formal definition follows. In such a multigraph, each node has a natural interpretation as a term. The syntactic unification algorithm based on this representation was developed by Corbin and Bidoit [14]. The basic idea of this algorithm is to consider an instance as a multigraph without duplicate occurrences of variables, and to represent a most general unifier as a mapping of variables to nodes in the same multigraph.

Definition 4.5.6. A *term-graph* is a quintuple $\mathcal{G} = (G, a, \lambda, \alpha, \prec)$ where G is a finite non-empty set (of nodes), $a \in G$ is a distinguished node, $\lambda : G \to \mathcal{F} \cup X$ is a labelling of nodes, $\alpha : G \to \bigcup_{n=0}^{\infty} G^n$ is a mapping and $\prec$ is a binary relation on G such that:

i) $\alpha(b) = (b_1^\alpha, \dots, b_n^\alpha) \in G^n$ where n is the arity of the function symbol $\lambda(b)$ (for b such that $\lambda(b)$ is a constant symbol or a variable, $\alpha(b) \in G^0$ is not significant),

ii) $b \prec c$ iff there exists i such that $b = c_i^\alpha$,

iii) the transitive closure $<$ of the relation $\prec$ is irreflexive, i.e. a strict order.

Remark 4.5.7. The condition (ii) in the previous definition characterizes the binary relation $\prec$. This means that if α is given then there is only one binary relation $\prec$ on G satisfying (ii). Hence if we want to describe a term-graph, then it is enough to give a four-tuple (G, a, λ, α).

We define a term interpretation of a term-graph $\mathcal{G} = (G, a, \lambda, \alpha, \prec)$, in fact of any node in a term-graph $\mathcal{G}$, using a mapping $\tau : G \to \mathcal{T}(\mathcal{F}, X)$. We construct τ inductively with respect to the order $<$ in the following way

$$\tau(b) = \begin{cases} \lambda(b) & \text{if } \lambda(b) \text{ is a constant symbol or a variable,} \\ f(\tau(b_1^\alpha), \dots, \tau(b_n^\alpha)) & \text{if } \lambda(b) = f \text{ is an } n\text{-ary function symbol.} \end{cases}$$

Then we say that the term-graph $\mathcal{G}$ represents the term $\tau(a)$ and write $\tau(\mathcal{G}) = \tau(a)$.

Note that under this definition a given term has a lot of different term-graph representations as we can see in the following examples.

Example 4.5.8. The standard tree representation of the term t is an example of a term-graph representation. Here G is the set of all occurrences of function symbols and variables in t, a is the top symbol of t and the mappings λ and α are given in a natural way. It is easy to see that one can construct this representation in polynomial time.

Example 4.5.9. For a term t we can consider the set G of all subterms of t. Now, if for each subterm $s = f(s_1, \dots, s_n)$ of the term t we define $\lambda(s) = f$ and $\alpha(s) = (s_1, \dots, s_n)$, then we obtain a term-graph representation $\mathcal{G} = (G, t, \lambda, \alpha, \prec)$ of the term t. This representation is the smallest possible term-graph representation of the term t.

One can see that there are terms whose term-graph representation of Example 4.5.9 is exponentially smaller than their tree representation (or representation as a string of symbols). The following proposition shows that the syntactic unification problem is decidable in polynomial time even if its instances are given using arbitrary term-graph representations.

Proposition 4.5.10 ([14]). *There exists a polynomial algorithm deciding for given term-graphs $\mathcal{G}$ and $\mathcal{H}$ whether the corresponding terms $\tau(\mathcal{G})$ and $\tau(\mathcal{H})$ are syntactically unifiable.*

We have defined an abbreviate representation of terms and now we return to our equational theory. We denote by E the theory of a single independent rectangular symbol and enough other free function symbols. We consider a single equation as an instance of our problem, because systems can be reduced to single equations by Remark 1.2.6.

Let t be a term and x be a variable. We say that x is a *left variable in the term t* if every occurrence of x in t is a left argument of a product, i.e. every occurrence of x is in a subterm $x * s$ of t where s is a certain term. Dually, x is a *right variable in the term t* if every occurrence of x in t is a right argument of a product. We say that $x \in X$ is a left variable (right variable) in an equation

$s =^? t$ if and only if x is a left variable (a right variable respectively) in both terms s and t. An equation (a term respectively) is called *with one-sided variables* if each variable is a left variable or a right variable in the equation (the term respectively).

Lemma 4.5.11. *For any equation $s =^? t$ there exists an equation $s' =^? t'$ with one-sided variables such that $s =^? t$ is* E*-satisfiable if and only if $s' =^? t'$ is* E*-satisfiable. Moreover, such an equation $s' =^? t'$ can be constructed in polynomial time.*

Proof. For any variable $x \in X$ we introduce two new variables x_l and x_r. We construct the equation $s' =^? t'$ in the following way: for any variable x we replace any occurrence of the variable x in $s =^? t$ with the term $x_l * x_r$.

It is not hard to see that the construction can be done in polynomial time. Moreover, if σ is an E-unifier of $s =^? t$, then we can define an E-unifier σ' of $s' =^? t'$ by putting $\sigma'(x_l) = \sigma'(x_r) = \sigma(x)$ for any $x \in X$. On the other hand, if σ' is an E-unifier of $s' =^? t'$ then we can define an E-unifier σ of $s =^? t$ by letting $\sigma(x) = \sigma'(x_l) * \sigma'(x_r)$ for any $x \in X$. □

From this moment we can concentrate on equations with one-sided variables.

We will work with a special form of terms, so-called alternating terms, which have the property that the ai-symbol $*$ alternates with the other symbols; a more precise definition follows.

We define two types of *alternating* terms in a signature $\mathcal{F}$ inductively.

i) Any constant symbol and any variable is an odd alternating term.

ii) If s and t are odd alternating terms, then the term $s*t$ is an even alternating term.

iii) If f is an n-ary function symbol different from $*$ and $t_1, \ldots, t_n$ are even alternating terms, then the term $f(t_1, \ldots, t_n)$ is an odd alternating term.

A term is *alternating* if it is odd alternating or even alternating.

Lemma 4.5.12. *Let s and t be different alternating terms such that $s =_{\mathrm{E}} t$. Then exactly one of the terms s and t is an even alternating term. In particular, if s is an even alternating term, t is an odd alternating term and $s =_{\mathrm{E}} t$, then $s = t * t$.*

Proof. Assume that there are two different alternating terms s and t such that $s =_{\mathrm{E}} t$ and both are even alternating terms or both are odd alternating terms. Consider the smallest such pair s, t of alternating terms.

If both terms are even alternating then we have $s = s_1 * s_2 =_{\mathrm{E}} t_1 * t_2 = t$, where terms s_1, s_2, t_1 and t_2 are odd alternating. Because the top symbols of s_1,

s_2, t_1 and t_2 are free, the equality $s_1 * s_2 =_{\mathrm{E}} t_1 * t_2$ implies the equalities $s_1 =_{\mathrm{E}} t_1$ and $s_2 =_{\mathrm{E}} t_2$. As the terms s and t were chosen as the smallest counterexample to the statement, we have $s_1 = t_1$ and $s_2 = t_2$, which implies $s = t$.

If both terms s, t are odd alternating then they are both the same variable or the same constant symbol or $s = f(s_1, \dots, s_m) =_{\mathrm{E}} g(t_1, \dots, t_n) = t$ where f, g are free function symbols of arities m, n and $s_1, \dots, t_n$ are even alternating terms. From this equality we deduce $f = g$, $m = n$ and also $s_1 =_{\mathrm{E}} t_1, \dots, s_m =_{\mathrm{E}} t_m$. Again, because we have started with the smallest counterexample, we have $s_1 = t_1, \dots, s_m = t_m$. Hence we obtain $s = t$, which is a contradiction.

Let s be an even alternating term, t be an odd alternating term and $s =_{\mathrm{E}} t$. Then the top symbol of s is $*$ and the top symbol of t is a free function symbol. This means that $s = s_1 * s_2$, where $s_1 =_{\mathrm{E}} s_2 =_{\mathrm{E}} t$ are even alternating terms. The equalities $s_1 = s_2 = t$ follow by the first part of the proof. □

We show that alternating terms can be used for solving of the identity problem of the theory E.

Lemma 4.5.13. *For an arbitrary term t there exists a unique even alternating term s such that $t =_{\mathrm{E}} s$. Moreover, there exists a term-graph representation of the term s which can be computed from t in polynomial time.*

Note that a representation of the term s as a string of symbols could be exponential in the size of the term t, which is why we have introduced the term-graph representation.

Proof. The uniqueness of the even alternating term s follows from the previous lemma. Now we describe a polynomial construction of a term-graph representation of the term s, which in particular proves that this term really exists.

Let as start by stating a sufficient condition for a term-graph to represent an alternating term: a term-graph $\mathcal{G} = (G, a, \lambda, \alpha, \prec)$ represents an alternating term if for each pair $b \prec c$ in $\mathcal{G}$ exactly one of $\lambda(b)$ and $\lambda(c)$ is the symbol $*$; if in addition $\lambda(a) = *$, then $\mathcal{G}$ represents an even alternating term.

Let $\mathcal{G} = (G, a, \lambda, \alpha, \prec)$ be a term-graph representation of the term t. We rewrite $\mathcal{G}$ to a term-graph representation of the term s in the following way. First we add enough nodes labelled by $*$. We take $\overline{G} = \{\overline{b} \mid b \in G\}$ a disjoint copy of G and we construct a term-graph $\mathcal{G}' = (G \cup \overline{G}, \overline{a}, \lambda', \alpha', \prec')$ in this way: for any $b \in G$ we put

$$\lambda'(b) = \lambda(b), \quad \lambda'(\overline{b}) = *, \quad \alpha'(\overline{b}) = (b, b), \quad \alpha'(b) = (\overline{b^\alpha_1}, \dots, \overline{b^\alpha_n}),$$

and the relation $\prec'$ is given by the condition (ii) in the definition of the term-graph (see Remark 4.5.7). Let τ and τ' be the term interpretations of the term-graphs $\mathcal{G}$ and $\mathcal{G}'$ respectively. One can show (inductively with respect to $<'$, i.e. transitive closure of $\prec'$) for any $b \in G : \tau'(\overline{b}) =_{\mathrm{E}} \tau'(b) =_{\mathrm{E}} \tau(b)$, in particular $\tau'(\overline{a}) =_{\mathrm{E}} \tau(a)$.

Now for any $c, d \in G \cup \overline{G}$ such that $c \prec' d$ at least one of c and d belongs to $\overline{G}$, i.e. it is labelled by $*$. So, it is enough to eliminate pairs $c \prec' d$ such that $\lambda'(c) = \lambda'(d) = *$.

We redefine α' step by step in such a way that in any step we remove one pair from the set

$$\{(c, d) \mid c \prec' d,\ \lambda'(c) = \lambda'(d) = *\} \subseteq (G \cup \overline{G}) \times (G \cup \overline{G}).$$

If $e \in G \cup \overline{G}$ is a node such that $\lambda'(e) = \lambda'(e_1^{\alpha'}) = *$ and $\lambda'((e_1^{\alpha'})_1^{\alpha'}) \neq *$, then we redefine $\alpha'(e) = (e_1^{\alpha'}, e_2^{\alpha'})$ to $((e_1^{\alpha'})_1^{\alpha'}, e_2^{\alpha'})$. Dually, if e is a node such that $\lambda'(e) = \lambda'(e_2^{\alpha'}) = *$ and $\lambda'((e_2^{\alpha'})_2^{\alpha'}) \neq *$, then we redefine $\alpha'(e)$ to $(e_1^{\alpha'}, (e_2^{\alpha'})_2^{\alpha'})$. This means that we are replacing $e_i^{\alpha'} \prec' e$ with $(e_i^{\alpha'})_i^{\alpha'} \prec' e$. It is easy to see that such a replacement decreases the number of pairs $c \prec' d$ satisfying $\lambda'(c) = \lambda'(d) = *$ and if there is no such e then there is no pair $c \prec' d$, $\lambda'(c) = \lambda'(d) = *$. So, at the end we obtain a term-graph representing an alternating term. Moreover, one can see that the new term $\tau'(e)$ is $=_{\mathrm{E}}$-related to the previous one because we have equalities $(x * y) * z =_{\mathrm{E}} x * z$ and $x * (y * z) =_{\mathrm{E}} x * z$. Hence the resulting term-graph represents an alternating term $=_{\mathrm{E}}$-related to the original term t, i.e. the unique possible term s.

The whole construction can be done in polynomial time. $\square$

Remark 4.5.14. Note that if a variable x is a left (right) variable in the term t, then it is also a left (right respectively) variable in the even alternating term s such that $s =_{\mathrm{E}} t$.

Again, we extend the notion of alternating terms to equations; an equation $s =^? t$ is called *alternating* iff both terms s and t are even alternating.

A substitution σ is called *alternating* if for every $x \in X$ the term $\sigma(x)$ is an odd alternating term. Note that for an even (odd) alternating term t with one-sided variables and for an alternating substitution σ, the term $\sigma(t)$ is an even (odd) alternating term.

The following lemma shows how one can solve alternating equations with one-sided variables.

Lemma 4.5.15. *Let $s =^? t$ be an alternating equation with one-sided variables. Then $s =^? t$ is* E*-satisfiable if and only if $s =^? t$ is syntactically satisfiable, i.e $\emptyset$-satisfiable.*

Proof. We show that the equation $s =^? t$ has an E-unifier if and only it has an $\emptyset$-unifier. The "if" part is trivial (see Remark 1.2.2).

Let σ be an E-unifier of the equation $s =^? t$. We construct an alternating E-unifier σ' of $s =^? t$ first. Let x be a variable and $t_1 * t_2$ be the unique even alternating term $=_{\mathrm{E}}$-related to $\sigma(x)$. If x is a left variable in the equation $s =^? t$, then we put $\sigma'(x) = t_1$ and if x is a right variable in $s =^? t$, then we put $\sigma'(x) = t_2$.

Because the equation $s =^? t$ is alternating with one-sided variables, we have $\sigma'(s) =_{\mathrm{E}} \sigma(s)$ and $\sigma'(t) =_{\mathrm{E}} \sigma(t)$. Hence σ' is an E-unifier. It is easy to see that σ' is an alternating substitution, hence both $\sigma'(s)$ and $\sigma'(t)$ are even alternating terms. The equality $\sigma'(s) =_{\mathrm{E}} \sigma'(t)$ implies $\sigma'(s) = \sigma'(t)$ by Lemma 4.5.12. This means that σ' is a syntactic unifier of the equation $s =^? t$. □

Let us resume the proof of the proposition. By Lemma 4.5.11 we can exchange an instance of the general $\mathcal{RCB}$-unification problem for an equation $s =^? t$ with one-sided variables which is E-satisfiable if and only if the original equation is E-satisfiable. By Lemma 4.5.13 we replace both sides of the equation $s =^? t$ with $=_{\mathrm{E}}$-related even alternating terms represented by term-graphs. The resulting equation is still with one-sided variables by Remark 4.5.14 and it is E-satisfiable if and only if it has a syntactic unifier by Lemma 4.5.15. This question is decidable by Proposition 4.5.10. The whole construction can be performed in polynomial time. □

We can summarize the previous results to the following dichotomy theorem.

Theorem 4.5.16. *Let $\mathcal{V}$ be a variety of bands. Then the general $\mathcal{V}$-unification problem and the general $\mathcal{V}$-matching problem are decidable in polynomial time if $\mathcal{V} \subseteq \mathcal{RCB}$ and they are NP-complete otherwise, i.e. if $\mathcal{SL} \subseteq \mathcal{V}$.*

In the rest of the section we will consider theories consisting of more independent ai-symbols and free symbols. Again, as a consequence of Proposition 1.2.12 and the previous theorem, such equational unification problems are in NP. We have already solved a lot of particular cases. For example, in the proof of Lemma 4.5.4 we saw that any independent left zero (right zero respectively) function symbol can be eliminated from an instance and unifiers, hence each of our equational unification problems can be reduced to an equational unification problem without these symbols. Therefore it is enough to consider only ai-symbols which are not trivial, left zero or right zero. We call them *non-zero ai-symbols*. In Remark 4.5.3 we have solved the case of two non-zero ai-symbols such that at least one of them is not rectangular.

Lemma 4.5.17. *Let $\mathcal{F}$ contain at least two independent non-zero ai-symbols and let one of them be a non-rectangular function symbol. Then the corresponding equational unification problem with constants and the equational matching problem are NP-hard.*

Hence, the last case which we have to discuss is a combination of rectangular function symbols. We show that the unification problem with constants is NP-complete if the theory contains two independent rectangular symbols. We will use symbols $\cdot$ and $+$ and a more transparent infix notation where $\cdot$ is omitted and applied first as usual. For example, by $0x + 01$ we mean the term $+(\cdot(0,x), \cdot(0,1))$.

Proposition 4.5.18. *Let $\mathcal{F}$ contain at least two free constant symbols* 0 *and* 1 *and two independent rectangular symbols $\cdot$ and $+$, i.e.* $\mathrm{E} = \{x(yz) \doteq (xy)z, xx \doteq x, (xy)z \doteq xz, x + (y + z) \doteq (x + y) + z, x + x \doteq x, (x + y) + z \doteq x + z\}$. *Then the* E*-unification problem and the* E*-matching problem are NP-complete.*

Proof. We start with the solution of the identity problem for the theory E. We use for this an easy term rewriting system (see [4] for a definition and basic properties of term rewriting systems).

Lemma 4.5.19. *The term rewriting system consisting of the rules*

$$\begin{array}{rclrclrcl} (xy)z & \to & xz, & x(yz) & \to & xz, & xx & \to & x, \\ (x+y)+z & \to & x+z, & x+(y+z) & \to & x+z, & x+x & \to & x, \end{array}$$

is confluent, terminating and solves the identity problem for the theory E*, i.e. for two terms $s, t \in \mathcal{T}(\mathcal{F}, X)$ we have $s =_{\mathrm{E}} t$ if and only if s and t have the same normal form with respect to $\to$.* □

If a term s is in normal form with respect to $\to$ and we want to point out this fact, then we write $\underline{s}$. Note that for two terms s and t the equality $\underline{s} =_{\mathrm{E}} \underline{t}$ implies $s = t$. We present some consequence of the previous lemma.

Lemma 4.5.20. *Let $\circ$ be one of the symbols $\cdot$ and $+$ and s, t, u be terms in normal form with respect to $\to$ such that $\circ$ is not a top symbol of u. If $s \circ t =_{\mathrm{E}} u$ then either $s = u$ or there exists a term v in normal form such that $s = u \circ v$ and either $t = u$ or there exists a term w in normal form such that $t = w \circ u$.* □

If we use our notation, we can write the statement of the lemma in the following way

$$\underline{s} \circ \underline{t} =_{\mathrm{E}} \underline{u} \quad \Longrightarrow \quad (s = \underline{u} \ \vee \ (\exists v) s = \underline{u \circ v}) \ \wedge \ (t = \underline{u} \ \vee \ (\exists w) t = \underline{w \circ u}).$$

The case when u has the top symbol $\circ$ can be also solved by Lemma 4.5.19.

Lemma 4.5.21. *Let $\circ$ be one of the symbols $\cdot$ and $+$. Let s_1, s_2, and $t_1 \circ t_2$ be terms in normal form with respect to $\to$. Then*

$$\begin{array}{rcl} \underline{s_1} \circ \underline{s_2} =_{\mathrm{E}} \underline{t_1 \circ t_2} & \Longrightarrow & (s_1 = \underline{t_1} \ \vee \ (\exists v) s_1 = \underline{t_1 \circ v}) \ \wedge \\ & & (s_2 = \underline{t_2} \ \vee \ (\exists w) s_2 = \underline{w \circ t_2}). \end{array}$$

□

We return to the proof of the proposition. We know that the problems belong to NP.

To prove their NP-hardness we show a reduction from the NP-complete problem 1-in-3-SAT (it is called the *exactly-one-in-three satisfiability problem* in the literature). An instance of the 1-in-3-SAT problem is the same as in the case of the 3-SAT problem, i.e. a conjunction $\Phi \equiv \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$ of clauses where each clause Φ_i, $1 \leq i \leq n$, is of the form $l_1 \vee l_2 \vee l_3$ where l_j, $1 \leq j \leq 3$, is a *literal* (i.e. l_j is a *variable* from the set *Var*, possibly negated). The question is whether there exists a valuation v such that exactly one of $v(l_1)$, $v(l_2)$ and $v(l_3)$ is T for any clause $\Phi_i \equiv l_1 \vee l_2 \vee l_3$ in the formula Φ.

So, let Φ be an instance of the 1-in-3-SAT problem. As usually, $\overline{\mathit{Var}} = \{\overline{x} \mid x \in \mathit{Var}\}$ is a disjoint copy of the set *Var* and for $x \in \mathit{Var}$ we define $\widetilde{x} = x$ and $\widehat{x} = \overline{x}$ and $\widetilde{\neg x} = \overline{x}$, $\widehat{\neg x} = x$.

We want to code negations and disjunctions of literals using matching equations for the theory E. We start with solving of the following equation with two variables $x, \overline{x} \in X = \mathit{Var} \cup \overline{\mathit{Var}}$:

$$(0x + 01)(0 + 0\overline{x}) + 0 =^? 0. \tag{4.22}$$

It is not hard to see that the substitutions σ_1 and σ_0 given by the rules $\sigma_1(x) = x1$, $\sigma_1(\overline{x}) = \overline{x}0$ and $\sigma_0(x) = x0$, $\sigma_0(\overline{x}) = \overline{x}1$ are E-unifiers of the equation (4.22).

Lemma 4.5.22. *The set $\{\sigma_1, \sigma_0\}$ is a minimal complete set of* E*-unifiers of the equation (4.22).*

Proof. Let σ be an E-unifier of the equation (4.22) and denote $\underline{s} =_{\mathrm{E}} \sigma(x)$ and $\underline{s'} =_{\mathrm{E}} \sigma(\overline{x})$. We will decompose the equality

$$(0s + 01)(0 + 0s') + 0 =_{\mathrm{E}} 0$$

using Lemma 4.5.20 repeatedly. First of all, we have

$$(0s + 01)(0 + 0s') =_{\mathrm{E}} 0 \quad \text{or} \quad (0s + 01)(0 + 0s') =_{\mathrm{E}} \underline{0 + t}$$

for a (reduced) term t.

In the first case we deduce $0s + 01 =_{\mathrm{E}} 0$ or $0s + 01 =_{\mathrm{E}} \underline{0u}$ for some term u. The equality $0s + 01 =_{\mathrm{E}} 0$ is impossible and equality $0s + 01 =_{\mathrm{E}} \underline{0u}$ implies $01 =_{\mathrm{E}} \underline{0u}$ by Lemma 4.5.20. Hence we have $u = 1$ and then from $0s + 01 =_{\mathrm{E}} 01$ we deduce that $0s =_{\mathrm{E}} 01$ or $0s =_{\mathrm{E}} \underline{01 + v}$ for some term v. Again, the equality $0s =_{\mathrm{E}} \underline{01 + v}$ is impossible by Lemma 4.5.20. Hence we have $0s =_{\mathrm{E}} 01$ and we see that $s = 1$ or there is a term $\tilde{s}$ such that $s = \underline{\tilde{s}1}$. Altogether, there is a term r such that $s =_{\mathrm{E}} r1$.

Analogically we describe s'. We have $0 + 0s' =_{\mathrm{E}} 0$ or $(0 + 0s') =_{\mathrm{E}} \underline{u'0}$ for some term u', where the equality $0 + 0s' =_{\mathrm{E}} \underline{u'0}$ is impossible by Lemma 4.5.20. The equality $0 + 0s' =_{\mathrm{E}} 0$ implies $0s' =_{\mathrm{E}} 0$. Hence we see that $s' = 0$ or there is a term $\tilde{s}'$ such that $s' = \underline{\tilde{s}'0}$. We can conclude that there is a term r' such that $s' =_{\mathrm{E}} r'0$. In the first case we have deduced $\sigma = \rho\sigma_1$ where ρ is given by the rules $\rho(x) = r$ and $\rho(\overline{x}) = r'$.

In the second case one can deduce that $0s + 01 =_{\mathrm{E}} 0 + 0s' =_{\mathrm{E}} \underline{0 + t}$. From the equality $0s + 01 =_{\mathrm{E}} \underline{0 + t}$ we derive $01 =_{\mathrm{E}} \underline{t}$ or $01 =_{\mathrm{E}} \underline{v + t}$ for some term v by Lemma 4.5.21. The second equality is impossible, so we deduce $t = 01$. Now, from $0s + 01 =_{\mathrm{E}} 0 + 01$ we have $0s =_{\mathrm{E}} 0$ or $0s =_{\mathrm{E}} \underline{0 + w}$ for some term w, again by Lemma 4.5.21. The letter equality is impossible, hence we have $0s =_{\mathrm{E}} 0$ which implies $s = 0$ or $s = \underline{\tilde{s}0}$ for a suitable term $\tilde{s}$. Analogically, from $0 + 0s' =_{\mathrm{E}} 0 + 01$ we deduce that $s' = 1$ or $s' = \underline{\tilde{s}'1}$ for a suitable term $\tilde{s}'$. The second case is covered by the unifier σ_0. $\square$

Because we can exchange the function symbols $\cdot$ and $+$, we know all solutions of the equation

$$((0 + x)(0 + 1) + 0(0 + \overline{x}))0 =^? 0. \tag{4.23}$$

We can put both equations together to obtain the following.

Lemma 4.5.23. *The system consisting of the equations (4.22) and (4.23) has exactly two (up to $=_{\mathrm{E}}$) unifiers σ_1 and σ_0 given by the rules $\sigma_1(x) = 1$, $\sigma_1(\overline{x}) = 0$ and $\sigma_0(x) = 0$, $\sigma_0(\overline{x}) = 1$.* $\square$

Finally, we consider the following system S. We put to the system the equations (4.22) and (4.23) for every variable $x \in \mathit{Var}$ and for every clause $\Phi_i = l_1 \vee l_2 \vee l_3$ we add the equation

$$(0\widetilde{l_1} + \widetilde{l_2}\widehat{l_2})(0 + \widetilde{l_2}\widetilde{l_3}) + 0 =^? 0. \tag{4.24}$$

Now it is not hard to check that there exists a solution σ of the system S if and only if there exists a valuation v of the formula Φ such that exactly one of the literals in any clause is valuated to T. The correspondence between solutions and valuations is expressed by the equivalence $\sigma(x) =_{\mathrm{E}} 1 \iff v(x) = T$. $\square$

We extend Theorems 4.1.1, 4.1.2 and 4.5.16 to the following theorem in which we consider more ai-symbols.

Theorem 4.5.24. *Let* E *be a theory of independent non-trivial ai-symbols, enough free constant symbols and other free function symbols of an arbitrary arity.*

- *i) If* E *contains only one non-zero ai-symbol $\cdot$ and all free symbols are constant, then the* E*-unification problem is decidable in polynomial time if the symbol $\cdot$ satisfies the identity $x \cdot y \cdot z \cdot x \doteq x \cdot z \cdot y \cdot x$ and the* E*-unification* *problem is NP-complete otherwise, the* E*-matching problem is decidable in polynomial time if the symbol* $\cdot$ *satisfies the identity* $x \cdot y \cdot x \cdot z \cdot x \doteq x \cdot y \cdot z \cdot x$ *and the* E*-matching problem is NP-complete otherwise.*

ii) If E *contains only one non-zero ai-symbol* $\cdot$ *and some non-constant free symbol, then the* E*-unification problem and the* E*-matching problem are decidable in polynomial time if the symbol* $\cdot$ *satisfies the identity* $x \cdot y \cdot z \doteq x \cdot z$*, and both problems are NP-complete otherwise.*

iii) If E *contains at least two non-zero ai-symbols, then the* E*-unification problem and the* E*-matching problem are NP-complete.*

Chapter 5

Satisfiability of Equations over Bands

In this chapter we look at the problem of solving of equations from another point of view. We will consider the problem of solving of systems of equations over a concrete fixed finite semigroup, especially over a finite band. Trivially, this problem is decidable by a non-deterministic polynomial algorithm, which guesses a mapping and checks whether this mapping is a solution of the system. So, the basic question is a description of semigroups for which the problem is NP-complete and semigroups for which the problem is tractable.

All results contained in this chapter are based on our cooperation with Pascal Tesson and Denis Thérien. One can also find some of these results in the PhD thesis of Pascal Tesson [47] in a more general setting. Here we only present the proofs which are related to bands.

The precise definition of the problem of solving of equations and basic observations are contained in the first section. Equations with constant right hand sides over a band are studied in Section 5.2 and Section 5.3 deals with general equations. The last section only summarizes our results about complexity of solving of equations in semigroups.

5.1 Description of the Problem

We follow the basic definitions of word equations and solutions and modify them to the case of equations over a given finite semigroup.

Let S be a finite semigroup and X be a finite set of variables, $S \cap X = \emptyset$. The elements of $(S \cup X)^+$ are called *S-words* or simply *words*. An *S-equation* (also an *equation over the semigroup S*, or simply an *equation*) $L =^? R$ is a pair $(L, R) \in (S \cup X)^+ \times (S \cup X)^+$. We call an equation $L =^? R$ a *target equation* if

$R \in S^+$; in this case we can assume that $R \in S$. A *system of S-equations* is a finite set of equations over the semigroup S, i.e. $\{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ for $n \geq 0$. A system is called *target* if it consists of target equations only.

Any mapping $\alpha : X \to S$ can be uniquely extended to a homomorphism $\alpha' : (S \cup X)^+ \to S$ which behaves as an identity on the semigroup S. We will use the same symbol α for the mapping and its unique extension. Such a mapping $\alpha : X \to S$ is a *solution* of the system $\{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ if and only if $\alpha(L_i) = \alpha(R_i)$ for all $i = 1, \ldots, n$. Note that there is no need of a special definition for the case when S is a monoid, because we can use the neutral element of the monoid S instead of the empty S-word from $(S \cup X)^*$.

Definition 5.1.1. Let S be a fixed finite semigroup. An instance of the EQN_S problem is a system of S-equations $\{L_1 =^? R_1, \ldots, L_n =^? R_n\}$ and the question is whether there exists a solution of this system. The $\mathrm{T{-}EQN}_S$ problem is the modification of the EQN_S problem where only target systems are considered.

As we have mentioned in the introduction to this chapter, these problems are trivially in NP, because for any variable there are only constantly many possible values from S.

Lemma 5.1.2. *For any finite semigroup S, the* EQN_S *and* $\mathrm{T{-}EQN}_S$ *problems are in NP.* □

It is not surprising that there is a substantial difference between solving of a system and solving of a single equation. We are interested in the EQN_S problem, but we should mention known results about complexity of solving of one equation. We will denote the corresponding problems $1{-}\mathrm{EQN}_S$ and $1\mathrm{T{-}EQN}_S$. (An instance of the $1{-}\mathrm{EQN}_S$ problem and the $1\mathrm{T{-}EQN}_S$ problem is an equation and a target equation respectively.) The $1\mathrm{T{-}EQN}_S$ problem was studied in [9] in the case of monoids. There was given an example of a six-element monoid, namely the Brandt monoid[1] BA, for which the $1\mathrm{T{-}EQN}_{BA}$ is NP-complete. On the other hand, in the same paper the authors showed that the $1\mathrm{T{-}EQN}_M$ is decidable in polynomial time for any monoid M from the class **DA**. This result was extended by P. Tesson [47] to the following.

Proposition 5.1.3 ([47]). *The* $1\mathrm{T{-}EQN}_M$ *problem is decidable in polynomial time for any monoid* $M \in \mathbf{DO} \cap \overline{\mathbf{G}}_{\mathbf{NIL}}$.

Note that the class $\mathbf{DO} \cap \overline{\mathbf{G}}_{\mathbf{NIL}}$ contains some important classes of finite monoids, for example all commutative monoids, all idempotent monoids and the class **DA**. We should mention that the technique of the proof is based on the

[1]Brandt monoid is defined by the presentation $BA = \langle a, b \mid a^2 = b^2 = 0, aba = a, bab = b \rangle$ and it has elements $0, a, ab, b, ba, 1$.

fact that M is a monoid. Note also that, it was generalized to the $1-\mathrm{EQN}_M$ problem in [47].

We are able to establish the same result for bands. We generalize Lemma 3.1.1 where equations in free bands were studied. The idea is the same: if an equation is solvable then it has a constant solution.

Similarly to the case of free bands we consider for an S-word u the longest prefix of u which is an element of S^*, we denote it $p_S(u)$. We define $u_0 = \mathrm{eval}_S(p_S(u))$, where $\mathrm{eval}_S : S^* \to S^1$ is a canonical surjective homomorphism which maps a sequence of semigroup elements to their product in S^1. We define dually $u_1 = \mathrm{eval}_S(s_S(u)) \in S^1$, where $s_S(u)$ is the longest suffix of u belonging to S^*.

We will discuss the problems $1\mathrm{T}-\mathrm{EQN}_S$ and $1-\mathrm{EQN}_S$ separately. We call an equation $L =^? R$ *trivial* if $L, R \in S^+$. (A trivial equation $L =^? R$ has a solution if and only if $\mathrm{eval}_S(L) = \mathrm{eval}_S(R)$.) The $1\mathrm{T}-\mathrm{EQN}_S$ problem is solved by the following lemma which gives a condition equivalent to the solvability of a target equation. We are able to check this condition in polynomial time.

Lemma 5.1.4. *Let S be a band and let $L =^? R$ be a non-trivial target equation over S, i.e. $\mathsf{c}(L) \cap X \neq \emptyset$ and $R \in S$. Then there exists a solution of the equation $L =^? R$ if and only if $R \leq_{\mathcal{R}} L_0$, $R \leq_{\mathcal{L}} L_1$ and $R \leq_{\mathcal{J}} s$ for any $s \in \mathsf{c}(L) \cap S$.*

Proof. Let α be a solution of the equation $L =^? R$. Then $\alpha(L) = L_0\alpha(L') = R$ for a certain suffix L' of L. This implies $R \leq_{\mathcal{R}} L_0$. In the same way we can deduce $R \leq_{\mathcal{L}} L_1$. Also, for any $s \in \mathsf{c}(L) \cap S$ we see that $\alpha(L) = R$ implies $R \leq_{\mathcal{J}} s$.

Conversely, assume that $L =^? R$ satisfies $R \leq_{\mathcal{R}} L_0$, $R \leq_{\mathcal{L}} L_1$ and $R \leq_{\mathcal{J}} s$ for every $s \in \mathsf{c}(L) \cap S$. Then we define the mapping $\alpha : X \to S$ by the rule $\alpha(x) = R$ for any $x \in X$. Now $\alpha(L) = L_0RtRL_1$ for a certain $t \in S$. Because $\alpha(x) = R$ for any $x \in X$ and $R \leq_{\mathcal{J}} s$ for any $s \in \mathsf{c}(L) \cap S$, we can see that $R \leq_{\mathcal{J}} t$. From Lemma 1.1.8 we have $L_0R = R$, $RL_1 = R$ and $RtR = R$, which concludes the proof. $\square$

A similar condition can be formulated in the general case of a single equation over bands.

Lemma 5.1.5. *Let S be a band and let $L =^? R$ be a non-target equation over S, i.e. $\mathsf{c}(L) \cap X \neq \emptyset$ and $\mathsf{c}(R) \cap X \neq \emptyset$. Then there exists a solution of the equation $L =^? R$ if and only if there exists an element $s \in S$ such that $s \leq_{\mathcal{R}} L_0$, $s \leq_{\mathcal{R}} R_0$, $s \leq_{\mathcal{L}} L_1$ and $s \leq_{\mathcal{L}} R_1$.*

Proof. Let α be a solution of the equation $L =^? R$ and denote $s = \alpha(L)$. Then $s = \alpha(L) = L_0\alpha(L')L_1 = R_0\alpha(R')R_1 = \alpha(R)$ for certain factors L' and R' of L and R respectively. This gives the conditions.

Conversely, assume that for $L =^? R$ there exists s satisfying the conditions. Denote by e some $\mathcal{J}$-minimal element in S, e.g. some product of all elements of S. Now we define the mapping $\alpha : X \to S$ by the rule $\alpha(x) = ses$ for any $x \in X$. We can write $\alpha(L) = L_0 sestsesL_1$ for a certain $t \in S$ and analogically $\alpha(R) = R_0 sest'sesR_1$ for a certain $t' \in S$. From Lemma 1.1.8 we have $L_0 s = s$, $sL_1 = s$, $R_0 s = s$, $sR_1 = s$ and $ere = e$ for any $r \in S$. These equalities imply $\alpha(L) = ses = \alpha(R)$. $\square$

These basic observations lead to the following result.

Proposition 5.1.6. *Let S be a band. Then the* $1\mathrm{T}-\mathrm{EQN}_S$ *problem and the* $1-\mathrm{EQN}_S$ *problem are decidable in polynomial time.*

5.2 Target Systems for Bands

There is a nice dichotomy result for target systems. We will first mention known results and then explain our contribution.

Proposition 5.2.1 ([36]). *If S is a regular band then the* $\mathrm{T}-\mathrm{EQN}_S$ *problem is in P.*

The polynomial algorithm is based on the idea that there is a $\mathcal{J}$-minimal solution of any target system. In fact, such a minimal solution can be obtained as a product of all possible solutions of the system. The reason why this idea works is contained in the following technical lemma.

Lemma 5.2.2 ([36]). *Let S be a regular band and suppose $s_1 \cdots s_k = s$ and $t_1 \cdots t_l = s$ for some $s_i, t_i, s \in S$. For all shuffles[2] K of $s_1 \cdots s_k$ with $t_1 \cdots t_l$, we have $K = s$.*

The NP-completeness of the $\mathrm{T}-\mathrm{EQN}_M$ problem for non-regular idempotent monoids M was proved in [36]. The original technique of the proof is improved by the following new result.

Proposition 5.2.3. *Let S be a non-regular band. Then the* $\mathrm{T}-\mathrm{EQN}_S$ *problem is NP-complete.*

Proof. There are $A, B, C \in S$ such that $ABACA \neq ABCA$. We consider such A, B, C that $ABCA$ is maximal with respect to $\leq_{\mathcal{J}}$. We can assume without loss of generality that $ABACA$ is not $\mathcal{R}$-related to $ABCA$ for otherwise $ABACA$ is not $\mathcal{L}$-related to $ABCA$ and what follows can be done dually.

[2] A shuffle of $s_1 \cdots s_k$ with $t_1 \cdots t_l$ is the product of an arbitrary sequence of elements $s_1, \ldots, s_k, t_1, \ldots, t_l$ which respects the orders $s_1 \cdots s_k$ and $t_1 \cdots t_l$.

Claim: There exist elements $a, b, c \in S$ such that $ab = b$, $ba = a$ (i.e. $a \mathcal{R} b$), $ca = c$, $ac \neq bc$, $c <_{\mathcal{J}} a$ and satisfying the following condition

$$\forall s \in S : \quad c <_{\mathcal{J}} s \leq_{\mathcal{J}} a \implies as = bs. \tag{5.1}$$

Proof of the claim: Let A, B, C are as above. We put $a = ABA$, $b = AB$, $c = CABA$. Then we have $ab = b$, $ba = a$, $ca = c$. In other words, $a \mathcal{R} b$ and $c \leq_{\mathcal{L}} a$. Now,

$$abaca = ac = ABACABA \mathcal{R} ABACA,$$
$$abca = bc = ABCABA \mathcal{R} ABCA,$$

which means that ac and bc are not $\mathcal{R}$-related, in particular $ac \neq bc$. If we assume that $c \mathcal{J} a$, then a, b and c are elements of the rectangular band $\mathcal{J}_a$, so $abaca = a = abca$ which is a contradiction with $ac \neq bc$. This implies $c <_{\mathcal{J}} a$.

Now, let $s \in S$ be such that $c <_{\mathcal{J}} s \leq_{\mathcal{J}} a$. Because $absa \mathcal{J} s$ and we start with $\mathcal{J}$-maximal $ABCA \in \mathcal{J}_c$, we have $abasa = absa$. Since $ab = b$ and $ba = a$, we can derive $asa = bsa$. Because $s \leq_{\mathcal{J}} a$, i.e. $sas = s$, we can compute $as = asas = bsas = bs$. This concludes the proof of the claim.

We exhibit a reduction from the 3–SAT problem to the $\mathrm{T{-}EQN}_S$ problem using this triple of elements.

Let $\Phi \equiv \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$ be an instance of the 3–SAT problem. For any boolean variable $x \in \mathit{Var}$, we introduce three variables x, $\overline{x}$ and y_x; so, we have

$$X = \mathit{Var} \cup \{\overline{x} \mid x \in \mathit{Var}\} \cup \{y_x \mid x \in \mathit{Var}\}.$$

We construct a system of equations, which has a solution α satisfying the condition

$$\forall z \in X : \quad \alpha(z) \leq_{\mathcal{J}} a \tag{5.2}$$

if and only if the formula Φ is satisfiable. This is enough because a condition $s \leq_{\mathcal{J}} a$ is equivalent to $sas = s$ and one can replace all occurrences of a variable z in the system with zaz.

We construct the system of equations as follows: for any $x \in \mathit{Var}$ we take the equations

$$\begin{aligned}
cx &=^? c, && (5.3)\\
c\overline{x} &=^? c, && (5.4)\\
y_x a &=^? a, && (5.5)\\
axa\overline{x} &=^? ac, && (5.6)\\
bxb\overline{x} &=^? bc, && (5.7)\\
y_x xac &=^? ac, && (5.8)\\
y_x \overline{x} bc &=^? bc, && (5.9)
\end{aligned}$$

and for any element $q \in S$ which is $\mathcal{R}$-related to a we add the equations

$$qxqc =^? qc, \tag{5.10}$$
$$q\overline{x}qc =^? qc. \tag{5.11}$$

As usual, we define $\widetilde{x} = x$ if x is a positive literal, $\widetilde{\neg x} = \overline{x}$ if $\neg x$ is a negative literal. For all clauses $\Phi_i \equiv l_1 \vee l_2 \vee l_3$ in an instance $\Phi \equiv \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$ of the 3–SAT problem we take the equations

$$a\widetilde{l_1}a\widetilde{l_2}a\widetilde{l_3} =^? ac, \tag{5.12}$$
$$b\widetilde{l_1}b\widetilde{l_2}b\widetilde{l_3} =^? bc. \tag{5.13}$$

Now one can verify that if the 3–SAT instance is satisfiable, i.e. there exists a valuation v such that $v(\Phi) = T$, then the system of equations (5.3 – 5.13) can be solved by letting $\alpha(x) = c$, $\alpha(\overline{x}) = a$, $\alpha(y_x) = a$ whenever $v(x) = T$, and $\alpha(x) = a$, $\alpha(\overline{x}) = c$, $\alpha(y_x) = b$ whenever $v(x) = F$.

Conversely, suppose that the system of equations (5.3 – 5.13) has a solution α satisfying (5.2). The core of our construction is the following.
Claim: Exactly one of $\alpha(x)$ and $\alpha(\overline{x})$ is $\mathcal{J}$-related to the element c and the other one is strictly $\mathcal{J}$-above c.
Proof of the claim: From (5.3 – 5.4) we have $c \leq_{\mathcal{J}} \alpha(x), \alpha(\overline{x})$.

If both $\alpha(x)$ and $\alpha(\overline{x})$ are strictly $\mathcal{J}$-above c, then $a\alpha(x) = b\alpha(x)$ and $a\alpha(\overline{x}) = b\alpha(\overline{x})$ by the property (5.1). Hence

$$ac = \alpha(axa\overline{x}) = a\alpha(x)a\alpha(\overline{x}) = b\alpha(x)b\alpha(\overline{x}) = \alpha(bxb\overline{x}) = bc$$

by (5.6 – 5.7), which is a contradiction.

On the other hand, if both $\alpha(x)$ and $\alpha(\overline{x})$ are $\mathcal{J}$-related to c, then $c \;\mathcal{L}\; \alpha(x)$ and $c \;\mathcal{L}\; \alpha(\overline{x})$ from (5.3 – 5.4). Hence, for any q satisfying $c \leq_{\mathcal{J}} q$ we have $\alpha(x)qc = \alpha(x)$ and $\alpha(\overline{x})qc = \alpha(\overline{x})$, in particular $\alpha(x)ac = \alpha(x)$, $\alpha(\overline{x})bc = \alpha(\overline{x})$. Moreover, for any q which is $\mathcal{R}$-related to a we obtain from (5.10 – 5.11):

$$q\alpha(x) = q\alpha(x)qc = qc \quad \text{and} \quad q\alpha(\overline{x}) = q\alpha(\overline{x})qc = qc.$$

Because $\alpha(y_x)$ is $\mathcal{R}$-related to a by (5.5) and (5.2), due to (5.8 – 5.9) we have

$$ac = \alpha(y_x xac) = \alpha(y_x)\alpha(x)ac = \alpha(y_x)\alpha(x) = \alpha(y_x)c$$

and

$$bc = \alpha(y_x\overline{x}bc) = \alpha(y_x)\alpha(\overline{x})bc = \alpha(y_x)\alpha(\overline{x}) = \alpha(y_x)c.$$

This is a contradiction and we have finished the proof of the claim.

So, we set $v(x) = T$ whenever $\alpha(x) \;\mathcal{J}\; c$ and $v(x) = F$ whenever $\alpha(\overline{x})\mathcal{J}c$. Now the instance Φ is satisfied by v, because if for a clause $\Phi_i = l_1 \vee l_2 \vee l_3$ each of

$\alpha(\widetilde{l_1})$, $\alpha(\widetilde{l_2})$ and $\alpha(\widetilde{l_3})$ is strictly $\mathcal{J}$-above c, then $a\alpha(\widetilde{l_1}) = b\alpha(\widetilde{l_1})$, $a\alpha(\widetilde{l_2}) = b\alpha(\widetilde{l_2})$, $a\alpha(\widetilde{l_3}) = b\alpha(\widetilde{l_3})$ by (5.1) and the equality

$$a\alpha(\widetilde{l_1})a\alpha(\widetilde{l_2})a\alpha(\widetilde{l_3}) = b\alpha(\widetilde{l_1})b\alpha(\widetilde{l_2})b\alpha(\widetilde{l_3})$$

follows, in contradiction with (5.12 – 5.13). □

5.3 Systems of Equations for Bands

The dichotomy theorem for solving of equations in idempotent monoids was shown in the paper [36].

Proposition 5.3.1 ([36]). *If M is a commutative idempotent monoid, then the* EQN_M *problem is decidable in polynomial time.*

The solution of the problem is based on the fact that any solvable system has a $\mathcal{J}$-minimal solution. (Again, it can be viewed as a product of all solutions of the system.)

Proposition 5.3.2 ([36]). *If a monoid M is idempotent but is not commutative, then* EQN_M *is NP-complete.*

The case of semigroups is more complicated.

Proposition 5.3.3. *If S is a band which is not a normal band, then the* EQN_S *problem is NP-complete.*

Proof. For the sake of completeness we present a proof of the following technical lemma, which says that locally commutative bands are exactly normal bands.

Lemma 5.3.4. *A band S is normal if and only if the submonoid eSe is commutative for any $e \in S$.*

Proof. Let S be a normal band, i.e. satisfying the identity $xyzx \doteq xzyx$, and a, b, e be arbitrary elements of S. Then

$$eaebe = e(ae)(be)e = e(be)(ae)e = ebeae.$$

Hence the monoid eSe is commutative.

Let S be a band such that the monoid eSe is commutative for any $e \in S$ and let a, b, c be arbitrary elements of S. Then

$$abca = (ab)^3(ca)^3 = (aba)(aba)(abca)(aca)(aca) = (aba)(aca)(abca)(aba)(aca),$$

where the last equality holds because the multiplication in the monoid aSa is commutative. The last product is equal to $(aba)(aca)$ using Lemma 1.1.4. Hence we have $abca = abaca$ and this is equal to $acaba = acba$ due to the commutativity of aSa. This means that $abca = acba$, i.e. S is a normal band. □

Let us return to the proof of the proposition. Let S be a non-normal band. Assume that e is an element of S such that eSe is not commutative (such an element e exists by the previous lemma). Now any system over the monoid eSe is equivalent to a system over S if we replace any variable x with exe. So, the NP-completeness of the EQN_S problem is a consequence of Proposition 5.3.2. □

A slight modification of the polynomial algorithm for idempotent monoids works for semilattices too.

Lemma 5.3.5 ([47]). *If S is a semilattice then* EQN_S *is in P.*

The following result shows that there is also a normal band S with the NP-hard EQN_S problem.

Proposition 5.3.6. *There exists a left normal band S such that* EQN_S *is NP-complete.*

Proof. We will define a left normal band with the required property as a strong semilattice of left zero semigroups. Let $Y = \{*, f, s, t, l, r, v, 2, 4, 7, 8\}$ be a semilattice ordered as Figure 5.1 shows. Now we define the sets S_e for $e \in Y$ in

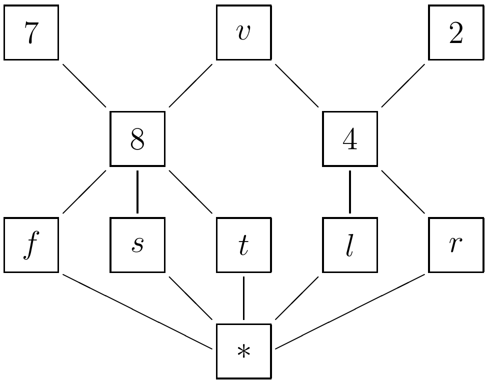

Figure 5.1: Semilattice Y.

this way: $S_* = \{*\}$, $S_f = S_s = S_t = S_l = S_r = S_v = \{0, 1\}$, $S_4 = S_v \times S_v$, $S_8 = S_v \times S_v \times S_v$, $S_7 = S_8 \setminus \{(0, 0, 0)\}$ and $S_2 = \{(0, 1), (1, 0)\}$. As the sets S_e should be pairwise disjoint, we will write relevant indices if necessary. The mappings $\varphi_{e,f}$ are defined in the following way. The mappings $\varphi_{7,8}$ and $\varphi_{2,4}$ are inclusions, and the mappings $\varphi_{v,8}$ and $\varphi_{v,4}$ are diagonal maps, i.e. $\varphi_{v,8}(0) = (0, 0, 0)$, $\varphi_{v,8}(1) = (1, 1, 1)$, $\varphi_{v,4}(0) = (0, 0)$ and $\varphi_{v,4}(1) = (1, 1)$. Further, the mappings $\varphi_{8,f}$, $\varphi_{8,s}$ and $\varphi_{8,t}$ are projections to the first, second and third coordinate respectively, i.e. $\varphi_{8,f}(0, 1, 0) = 0_f$ for example, and $\varphi_{4,l}$ and $\varphi_{4,r}$ are projections to the left and right coordinate respectively. Finally, the mappings $\varphi_{e,*}$ for

$e \in \{f, s, t, l, r\}$ are constant mappings to the one-element set $\{*\}$. The other mappings are defined in the only possible way to satisfy the conditions in the definition of a strong semilattice of left zero semigroups. Denote the resulting left normal band by S.

We will reduce the 3–SAT problem to the EQN_S problem. We show that we are able to encode valuations as solutions of equations over S. We use equations to characterize negations in Lemma 5.3.7 and to describe disjunctions of literals in Lemma 5.3.8.

For a triple of variables $x, \overline{x}, y_x$ we define the following system $T(x, \overline{x}, y_x)$ of equations:

$$
\begin{aligned}
0_v x &=^? 0_v, && (5.14)\\
0_v \overline{x} &=^? 0_v, && (5.15)\\
(0,1)_2 y_x &=^? (0,1)_2, && (5.16)\\
x 0_l &=^? y_x 0_l, && (5.17)\\
\overline{x} 0_r &=^? y_x 0_r. && (5.18)
\end{aligned}
$$

Lemma 5.3.7. *The system $T(x, \overline{x}, y_x)$ of equations (5.14 – 5.18) has exactly two solutions, namely α_0 and α_1 defined by the rules $\alpha_0(x) = 0_v$, $\alpha_0(\overline{x}) = 1_v$, $\alpha_0(y_x) = (0,1)_2$ and $\alpha_1(x) = 1_v$, $\alpha_1(\overline{x}) = 0_v$, $\alpha_1(y_x) = (1,0)_2$ respectively.*

Proof. Let α be a solution of the system $T(x, \overline{x}, y_x)$. Then we can see that $\alpha(x) \in S_v$ by (5.14) and analogically $\alpha(\overline{x}) \in S_v$, $\alpha(y_x) \in S_2$ by (5.15 – 5.16). The equations (5.17 – 5.18) ensure that $\alpha(y_x) = (\alpha(x), \alpha(\overline{x}))$ because both mappings $\varphi_{v,l} : S_v \to S_l$ and $\varphi_{v,r} : S_v \to S_r$ are identity maps. As the set S_2 has two elements, there are only two possible mappings, namely α_0 and α_1, which can be solutions. It is easy to see that α_0 and α_1 are really solutions of the system $T(x, \overline{x}, y_x)$. $\square$

For variables x_1, x_2, x_3 and z we consider the following system $U(x_1, x_2, x_3, z)$ of equations:

$$
\begin{aligned}
0_v x_1 &=^? 0_v, && (5.19)\\
0_v x_2 &=^? 0_v, && (5.20)\\
0_v x_3 &=^? 0_v, && (5.21)\\
(0,0,1)_7 z &=^? (0,0,1)_7, && (5.22)\\
x_1 0_f &=^? z 0_f, && (5.23)\\
x_2 0_s &=^? z 0_s, && (5.24)\\
x_3 0_t &=^? z 0_t. && (5.25)
\end{aligned}
$$

Lemma 5.3.8. *The system $U(x_1, x_2, x_3, z)$ of equations* (5.19 – 5.25) *has exactly seven solutions given by the condition $\alpha(z) = (\alpha(x_1), \alpha(x_2), \alpha(x_3)) \in S_7$.*

Proof. Let α be a solution of the system (5.19 – 5.25). Then $\alpha(z) \in S_7$ by (5.22). For $i = 1, 2, 3$ we have $\alpha(x_i) \in S_v$ by (5.19 – 5.21). For all $e \in \{f, s, t\}$ the mappings $\varphi_{v,e}$ are identity maps, hence $\alpha(z) = (\alpha(x_1), \alpha(x_2), \alpha(x_3))$ by the equations (5.23 – 5.25).

On the other hand, it is easy to see that all seven mappings are solutions of the system $U(x_1, x_2, x_3, z)$. □

Let $\Phi \equiv \Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$ be an instance of the 3-SAT problem with the set of variables *Var*. As usual, we put $\overline{Var} = \{\overline{x} \mid x \in Var\}$ and $\widetilde{x} = x$ if x is a positive literal, $\widetilde{\neg x} = \overline{x}$ if $\neg x$ is a negative literal. We denote

$$X = Var \cup \overline{Var} \cup \{y_x \mid x \in Var\} \cup \{z_i \mid i = 1, \ldots, n\}.$$

We consider the system of equations consisting of the following subsystems: for any $x \in Var$ we put to the system the subsystem $T(x, \overline{x}, y_x)$ and for any clause $\Phi_i = l_1 \vee l_2 \vee l_3$ we add the subsystem $U(\widetilde{l_1}, \widetilde{l_2}, \widetilde{l_3}, z_i)$. Now the formula Φ is satisfiable if and only if the system has a solution — it is an immediate consequence of Lemma 5.3.7 and Lemma 5.3.8. This means that the EQN_S problem is NP-complete. □

On the other hand, there are (left) normal bands for which systems of equations can be solved in polynomial time — for example free normal bands (Proposition 4.4.2). It seems that a description of the class of normal bands with EQN_S in P is quite complicated. We have proved in the paper [33] that any Constraint Satisfiability Problem for a finite relation is equivalent to the EQN_S problem for a certain left normal band S (the idea of the reduction is similar to the one used in the proof of Proposition 5.3.6). And, no descriptions of the class of finite relations with tractable Constraint Satisfiability Problem is known.

5.4 General Case

The results presented in this concluding section are proved in the paper [33] where we have generalized the results of [36] to arbitrary monoids and completely regular semigroups. These statements use also a result which has answered the case of groups [20]. We present here these results without proofs.

Proposition 5.4.1 ([33, 47]).

i) *For any finite monoid M, the* EQN_M *problem lies in P if M belongs to* $\mathbf{SL} \vee \mathbf{AB}$ *and* EQN_M *is NP-complete otherwise.*

ii) *For any finite monoid M, the* $\mathrm{T{-}EQN}_M$ *problem lies in P if M belongs to* $\mathbf{RB} \vee \mathbf{AB}$ *and* $\mathrm{T{-}EQN}_M$ *is NP-complete otherwise.*

iii) *For any finite completely regular semigroup S, the* $\mathrm{T{-}EQN}_S$ *problem lies in P if S belongs to* $\mathbf{RB} \vee \mathbf{AB}$ *and* $\mathrm{T{-}EQN}_S$ *is NP-complete otherwise.*

iv) *Let S be a finite completely regular semigroup. If S does not belong to* $\mathbf{NB} \vee \mathbf{AB}$ *then* EQN_S *is NP-complete. If S belongs to* $\mathbf{SL} \vee \mathbf{AB}$ *then* EQN_S *is in P.*

Polynomial algorithms use the structure theorem for considered semigroups, i.e. strong regular bands of abelian groups, and combine the two polynomial algorithms for groups and regular bands (normal and commutative bands respectively). The proofs of NP-hardness are similar to those used in this thesis. The complete proof of Proposition 5.4.1 can be found also in the PhD thesis of Pascal Tesson [47].

We hope that we will be able to extend the results to the case of regular semigroups too. On the other hand, we did not establish dichotomy theorems outside regular semigroups. Let us just mention that the class of nilpotent semigroups with the tractable EQN_S problem (and $\mathrm{T{-}EQN}_S$ problem respectively) is not a pseudovariety [33].

Chapter 6

Conclusion

The main result of the thesis is the following theorem, which speaks about the problem of solving of systems of equations in free idempotent monoids and bands.

Theorem 1 (3.4.1). *The* AIU-UNIFICATION *and* AI-UNIFICATION *problems are NP-complete.*

The same result holds for the special case of systems with variables occurring only on the left hand sides of equations.

Theorem 2 (2.1.1). *The* AIU-MATCHING *problem and the* AI-MATCHING *problem are NP-complete.*

If we assume that our function symbol satisfies some additional identities, then we obtain the following dichotomy theorem.

Theorem 3 (4.1.1 and 4.1.2). *Let* E *be an equational theory of a single binary function symbol $\cdot$ which is associative and idempotent and satisfies some additional identities. The* E*-unification problem with constants is decidable in polynomial time if the function symbol $\cdot$ satisfies the identity $x \cdot y \cdot z \cdot x \doteq x \cdot z \cdot y \cdot x$ and it is NP-complete otherwise. The* E*-matching problem is decidable in polynomial time if the function symbol $\cdot$ satisfies the identity $x \cdot y \cdot x \cdot z \cdot x \doteq x \cdot y \cdot z \cdot x$ and it is NP-complete otherwise.*

If we consider theories which contain in addition arbitrary free function symbols, then we have another borderline between tractable and NP-complete unification problems.

Theorem 4 (4.5.16). *Let* E *be an equational theory of a binary function symbol $\cdot$ which is associative and idempotent and satisfies some additional identities. The general* E*-unification problem and the general* E*-matching problem are decidable in polynomial time if the function symbol $\cdot$ satisfies the identity $x \cdot y \cdot z \doteq x \cdot z$ and they are NP-complete otherwise.*

We have also studied systems of equations over fixed finite bands. It is surprising that the results concerning the complexity aspects are almost the same as in the case of equations in free bands, although there is no general relationship between complexities of these problems.

Theorem 5 (5.2.1, 5.2.3, 5.3.3 and 5.3.5). *Let S be a finite band. The problem of solving of equations with constant right hand sides is decidable in polynomial time if S satisfies the identity $xyxzx \doteq xyzx$ and it is NP-complete otherwise. The problem of solving of equations is decidable in polynomial time if S is commutative and it is NP-complete if S does not satisfy the identity $xyzx \doteq xzyx$.*

The case of non-commutative bands satisfying the identity $xyzx \doteq xzyx$, i.e. non-commutative normal bands, is quite difficult and we do not expect existence of any nice property which characterizes the bands having the problem tractable (in the case P$\neq$NP). Note that the case of idempotent monoids is fully established, because any normal idempotent monoid is commutative.

Bibliography

[1] Almeida J.: Finite Semigroups and Universal Algebra, (1995) World Scientific, Singapore.

[2] Baader F.: The Theory of Idempotent Semigroups is of Unification Type Zero, J. Automated Reasoning **2** (1986) 283–286.

[3] Baader F.: Unification in Varieties of Idempotent Semigroups, Semigroup Forum **36** (1987) 127–145.

[4] Baader F., Nipkow T.: Term Rewriting and *All That* (1998) Cambridge University Press.

[5] Baader F., Schulz K. U.: General A- and AX-unification via optimized combination procedures, In Proceedings of IWWERT '91, Vol. **677** of LNCS (1993) 23–40.

[6] Baader F., Schulz K. U.: Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures, J. Symbolic Computation **21** (1996) 211–243.

[7] Baader F., Siekmann J. H.: Unification Theory, In Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. **2** (1994), Oxford University Press.

[8] Baader F., Snyder W.: Unification Theory, In Handbook of Automated Reasoning (2001), Elsevier Science Publishers.

[9] Barrington D. M., McKenzie P., Moore C., Tesson P. Thérien D.: Equation Satisfiability and Program Satisfiability for Finite Monoids. In Proceedings of MFCS'00, Vol. **1893** of LNCS (2000), 172–181.

[10] Birjukov A. P.: Varieties of Idempotent Semigroups, Algebra i Logika **9** (1970) 255–273 (Russian).

[11] Book R., Otto F.: String-Rewriting Systems (1993) Springer–Verlag.

[12] Burckert H.-J.: Matching — a special case of unification?, J Symbolic Computation **8** (5) (1989) 532–536.

[13] Burris S., Sankappanavar H. P.: A Course in Universal Algebra (1981) Springer–Verlag.

[14] Corbin J., Bidoit M.: A rehabilitation of Robinson's unification algorithm, In Proceedings of IFIP'83, Elsevier Paris (1983) 909–914.

[15] Dowling W. F., Gallier J. H.: Linear-time Algorithms for Testing the satisfiability of Propositional Horn Formulae, J. Logic Program. **1** (1984) 267–284.

[16] Fages F., Huet G.: Complete Sets of Unifiers and Matchers in Equational Theories, In Proceedings of the 5th Colloquium on Automata, Algebra and Programming, Vol. **159** of LNCS (1983) 205–220.

[17] Fennemore C. F.: All Varieties of Bands I, II, Math. Nachr. **48** (1971) 237–252, 253–262.

[18] Gerhard J. A.: The Lattice of Equational Classes of Idempotent Semigroups, J. Algebra **15** (1970) 195–224.

[19] Gerhard J. A., Petrich M.: Varieties of Bands Revisited, Proc. Lond. Math. Soc. **58** (1989) 323–350.

[20] Goldmann M., Russell A.: The complexity of solving equations over finite groups. In Proceedings of 14th Conference on Computational Complexity (1999) 80–86.

[21] Green J. A., Rees D.: On Semigroups in which $x^r = x$, Proc. Camb. Phil. Soc. **48** (1952) 35–40.

[22] Hermann M., Kolatitis P. G.: On the complexity of unification and disunification in commutative idempotent semigroups, In Proceedings of the 3rd International Conference on Principles and Practice of Constraint Programming, Vol. **1330** of LNCS (1997) 283–297.

[23] Hodges W.: Logical Features of Horn Clauses, In Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. **1** (1993), Oxford University Press.

[24] Howie J. M.: Fundamentals of Semigroup Theory (1995) Oxford University Press.

[25] Jouannaud J.-P., Kirchner C.: Solving equations in abstract algebras: A rule-based survey of unification, In Computation Logic: Essays in Honor of A. Robinson (1991), MIT Press.

[26] Kapur D., Narendran P.: NP–completeness of the Set Unification and Matching Problems, In Proceedings of CADE'86, Vol. **230** of LNCS (1986) 489–495.

[27] Kapur D., Narendran P.: Matching, unification and complexity, SIGSAM Bulletin (1987).

[28] Kapur D., Narendran P.: Complexity of Unification Problems with Associative-commutative Operators, J. Automated Reasoning **9**, No.2, (1992) 261–288.

[29] Klíma O.: On the solvability of equations in semigroups with $x^r = x$, In Proceedings of 58th Workshop on General Algebra , Contributions to General Algebra **12** (2000) 237–246, Verlag Johannes Heyn.

[30] Klíma O.: Unification Modulo Associativity and Idempotency is NP-complete, In Proceedings of MFCS'02, Vol. **2420** of LNCS (2002) 423–432.

[31] Klíma O.: Complexity of unification and matching problems in the varieties of idempotent semigroups, 2003. To appear in International Journal of Algebra and Computation. Technical Report 2002-15 available at `http://www.fc.up.pt/cmup/`.

[32] Klíma O., Srba J.: Matching Modulo Associativity and Idempotency is NP-complete, In Proceedings of MFCS'00, Vol. **1893** of LNCS (2000) 456–466.

[33] Klíma O., Tesson P., Thérien D.: Notes on Systems of Equations over Finite Semigroups, (preprint).

[34] Knuth D. E., Bendix P.B.: Simple word problems in universal algebras, In J. Leech (ed.), Computational Problems in Abstract Algebra (1970) Pergamon Press, Oxford.

[35] Makanin G. S.: The Problem of Solvability of Equations in a Free Semigroup, Mat. Sbornik. **103(2)** (1977) 147–236. (In Russian) English translation in: Math. USSR Sbornik **32** (1977) 129–198.

[36] Moore C., Tesson P., Thérien D.: Satisfiability of systems of equations over finite monoids. In Proceedings of MFCS'01, Vol. **2136** of LNCS (2001) 537–547.

[37] Papadimitriou C. H.: Computational Complexity (1994), Addison-Wesley Publishing Company.

[38] Paterson M. S., Wegeman M. N.: Linear unification, J. Computer and System Science **16** (2), (1978) 158–167.

[39] Plandowski W.: Solvability of word equations with constants is in P-SPACE, in P. Beame (ed) Proceedings of FOCS (1999) IEEE Computer Society Press, New York.

[40] Plotkin G.: Building in Equational Theories, Machine Intelligence **7** (1972) 73–90.

[41] Polák L.: On Varieties of Completely Regular Semigroups I,II,III, Semigroup Forum **32, 36, 37** (1985,1987,1988) 97–123, 253–284, 1–30.

[42] Robinson J. A.: A Machine-oriented Logic Based on the Resolution Principle, J. Assoc. Comput. Mach. **12**, (1965) 23–41.

[43] Schmidt-Schauss M.: Unification under Associativity and Idempotence is of Type Nullary, J. of Automated Reasoning **2** (1986) 277–281.

[44] Siekmann J. H.: Unification of Commutative Terms, In Proceedings of EUROSAM '79, Vol. **72** of LNCS (1979) 531–545.

[45] Siekmann J. H.: Unificaton theory: A survey, J. Symbolic Computation **7**(3,4) (1989) 207–274.

[46] Siekmann J. H., Szabó P.: A Noetherian and Confluent Rewrite System for Idempotent Semigroups, Semigroup Forum **25** (1982) 83–110.

[47] Tesson P.: Computational Complexity Questions Related to Finite Monoids and Semigroups. Ph.D. Thesis `http://www.cs.mcgill.ca/~ptesso/`.

Index