UPMAIL Technical Report No. 126
June 4, 1996
ISSN 1100–0686

Uniform Representation of Recursively Enumerable Sets with Simultaneous Rigid E-Unification

Margus Veanes

Box 311, S-751 05 Uppsala, Sweden
Phone: +48−18−18 25 00
Fax: +46−18−51 19 25

Abstract

Recently it was proved that the problem of simultaneous rigid E-unification (SREU) is undecidable. Here we perform an in-depth investigation of this matter and obtain that one can use SREU to uniformly represent any recursively enumerable set. From the exact form of this representation follows that SREU is undecidable already for 6 rigid equations with ground left hand sides and 2 variables.

There is a close correspondence between solvability of SREU problems and provability of the corresponding formulas in intuitionistic first order logic with equality. Due to this correspondence we obtain a new (uniform) representation of the recursively enumerable sets in intuitionistic first order logic with equality with one binary function symbol and a countable set of constants. From this result follows the undecidability of the $\exists\exists$-fragment of intuitionistic logic with equality. This is an improvement of a recent result regarding the undecidability of the $\exists^*$-fragment in general.

Contents

Chapter 1

Introduction

Recently it was proved that the problem of simultaneous rigid E-unification (SREU) is undecidable [9]. This (quite unexpected) undecidability result has lead to a series of undecidability results in related areas, and is also the main motivation behind this work. We perform an in-depth investigation of the undecidable nature of SREU. As a result we obtain that one can use SREU to uniformly represent any recursively enumerable set. From this representation follows that SREU is undecidable already for 6 rigid equations with ground left hand sides and 2 variables.

As a corollary of this investigation we obtain a new characterization of the recursively enumerable sets in intuitionistic first order logic with equality with one function symbol, say f, and a countable set of constants. More precisely, we can conclude this result as follows. Let W be an r. e. set of strings over some subset of constants as the alphabet. If t is a term $f(c_1, f(c_2, \ldots, f(c_n, c_0)\ldots))$ let $\widehat{t}$ stand for the string $c_1c_2\ldots c_n$ where all the c_i are constants. There is a quantifier free formula $\varphi(z, x, y)$ of the form

$$(\psi_1 \Rightarrow s_1 = t_1) \wedge \cdots \wedge (\psi_7 \Rightarrow s_7 = t_7)$$

where all the ψ_i are closed conjunctions of equations and all the variables in all the s_i and t_i are among $\{x, y, z\}$, which is obtained effectively from the index of W such that for all ground terms t in L

$$\widehat{t} \in W \quad \Leftrightarrow \quad \vdash_{\mathrm{i}} \exists x \exists y \varphi(t, x, y),$$

where $\vdash_{\mathrm{i}}$ stands for intuitionistic provability.

1.1 Background of SREU

Simultaneous rigid E-unification was proposed by Gallier, Raatz and Snyder [17] as a method for automated theorem proving in classical logics with equality. It can be used in automatic proof methods, like semantic tableaux [14], the connection method [4] or the mating method [1], model elimination [27], and others that are based on the Herbrand theorem, and use the property that a formula is valid (i.e., its negation is unsatisfiable) iff all paths through one of its matrix are inconsistent. This property was first recognized by Prawitz [35] (for first order logic without equality) and later by Kanger [22] (for first order logic with equality). In first order logic with equality, the problem of checking the inconsistency of the paths results in SREU.

1.2 Outline of the Report

In Chapter 2 we explain the notations used in the report. We also explain some background material concerning deterministic finite automata and rewrite systems. We want to point out here that the reader is not assumed to be an expert on rewrite systems (the author is not). We only use very simple results and the report is completely selfcontained. What is important to understand from this chapter (that the succeding chapters will make heavy use of) is how the graph of the transition function of a deterministic finite automaton can be seen as a convergent rewrite system.

In Chapter 3 we define the basic components needed to represent computations. These are words and sentences. Words are simply representations of strings and sentences are representations of sequences of strings. The main result of this chapter is Theorem 3.2.4. It shows how one can construct a system of rigid equations that describes sentences with words in certain regular sets such that the sentence itself has a given regular pattern. This theorem is used in Chapter 5.

In Chapter 4 we describe a technique that can be used to express roughly that one sequence of strings is an encoding of pairwise adjacent strings of another sequence. This is stated as Theorem 4.2.1. This technique is the cornerstone of the main result of this report (Theorem 5.2.1).

In Chapter 5 we prove that one can use SREU to represent (uniformly) any recursively enumerable set. The main result is Theorem 5.2.1. The system that is obtained has a very simple structure. The undecidability of SREU follows directly from this result (2 variables and six rigid equations is enough). We also give an overview of some other proofs of the undecidability of SREU that have appeared in the literature.

In Chapter 6 we give a uniform charterization of all r.e. sets with certain simple formulas in intuitionistic first order logic with equality, using only one function symbol of arity 2 and some number of constants. The main result here is Theorem 6.1.1. (It is actually a simple corollary of Theorem 5.2.1.) In particular, the undecidability of the $\exists\exists$-fragment of intuitionistic logic follows from it, this is an improvement of the recent result regarding the undecidability of the $\exists^*$-fragment in general [11].

In Chapter 7 we summerize the current status and state some open problems regarding decidability questions about SREU.

Chapter 2

Preliminaries

Here we explain the notational conventions of this this report and give the main definitions concerning SREU. Also some of the background material used in this report, concerning rewrite systems and deterministic finite automata is presented here. We introduce some useful notions and state some simple but useful lemmas relating finite automata with convergent rewrite systems.

In particular, the graph of the transition function of any deterministic finite automaton corresponds directly to a convergent rewrite system. This property remains true under union of graphs of transition functions of deterministic finite automata with disjoint sets of states, which is used in proving Theorem 3.2.4 (the main theorem of next chapter).

2.1 Basic Notions

Throughout the report, the first order language that we are working with is designated by L. L has one binary function symbol $\centerdot$ and a countable set of constants L_0. We will use infix notation for $\centerdot$ and assume that it associates to the right, so $t_1 \centerdot t_2 \centerdot t_3$ stands for the term $\centerdot(t_1, \centerdot(t_2, t_3))$. In general we will use the letters t and s to stand for terms in L.

Formulas are defined as usual from atomic formulas, i.e., equations since there are no relation symbols other than '=', using the connectives $\wedge$, $\vee$, $\Rightarrow$ and quantifiers $\exists$ and $\forall$. The notions of free and bound variables is standard. For a formula φ we write $\varphi(\vec{x})$, where $\vec{x} = x_1, x_2, \ldots, x_n$ for some $n \in \mathbb{N}$, to indicate that all the free variables in φ are among the $\vec{x}$. For t a term, $\varphi(\ldots, x_{i-1}, t, x_{i+1}, \ldots)$ stands for the formula obtained by substituting t for x_i in $\varphi(\ldots, x_{i-1}, x_i, x_{i+1}, \ldots)$.

Substitutions are mappings from variables to terms, and are extended to arbitrary expressions in the usual manner. An application of a substitution θ on an expression X is written $X\theta$. A formula is said to be *closed* if it contains no free variables. A term is said to be *ground* if it contains no variables. We say also that an equation or a set of equations is ground if all the terms involved are ground. For a formula $\varphi(\vec{x})$ we write $\forall\varphi$ for a universal closure of φ, i.e., for $\forall\vec{x}\varphi$.

We write $X \subset Y$ to say that X is a *nonempty finite* subset of Y. We will often overload the mening of '=', we trust the reader to understand the exact meaning from the context.

2.2 Simultaneous Rigid E-Unification

Here we define the main notions concerning simultaneous rigid E-unification. A

- ▶ *rigid equation* is an expression of the form $E \models_\forall s = t$ where E is a finite set of equations and s and t are arbitrary terms. A *system* of rigid equations is a finite set of rigid equations.

A substitution θ is a

▶ *solution* of or *solves* a rigid equation $E \models_{\forall} s = t$ if

$$\vdash \quad \forall\, (\bigwedge_{e \in E} e\theta) \Rightarrow s\theta = t\theta.$$

θ solves a system of rigid equations if it solves each member of the system.

Here $\vdash$ is classical or intuitionistic provability (for this class of formulas they are the same). The problem of solvability of systems of rigid equations is called *simultaneous rigid E-unification* or SREU for short. Solvability of a single rigid equation is called *rigid E-unification*.

2.3 Convergent Rewrite Systems

In order to simplify certain proofs we will make use of some results from rewrite systems [13, 21]. We start by introducing some terminology and finally we state a lemma that will be used later. We don't require the reader to be familiar with rewrite systems.

Let $\longrightarrow$ be a binary relation on terms. We define first some well-known properties of $\longrightarrow$. The reflexive and transitive closure of $\longrightarrow$ is denoted by $\xrightarrow{*}$. We say that $\longrightarrow$ is

▶ *noetherian* if there is no infinite chain $t_1 \longrightarrow t_2 \longrightarrow \cdots \longrightarrow t_i \longrightarrow \cdots$,

▶ *confluent* if $s \xrightarrow{*} t_1$ and $s \xrightarrow{*} t_2$ imply that there is a t such that $t_1 \xrightarrow{*} t$ and $t_2 \xrightarrow{*} t$,

▶ a *rewrite relation* if $s \longrightarrow t$ implies that $u[s\theta] \longrightarrow u[t\theta]$ for all terms s, t and u, and substitutions θ,

where $u[t]$ stands for u with certain subterm occurence t. Let E be a finite set of equations. We say that E is a

▶ *rewrite system* with respect to an ordering $\succ$ on terms if we have $s \succ t$ or $t \succ s$ for all equations $s = t$ in E.

We sometimes write $E^{\succ}$ if E is a rewrite system with respect to $\succ$, to emphasize the ordering. We say that an equation $s = t$ of E is a

▶ *rule $s = t$ of $E^{\succ}$* if $s \succ t$,

▶ by $\longrightarrow_{E^{\succ}}$ or simply $\longrightarrow_E$ we denote the smallest rewrite relation for which $s \longrightarrow_E t$ whenever $s = t$ is a rule of E.

We sometimes write $\longrightarrow$ for $\longrightarrow_E$ if E is clear from the context. A term s is said to be in

▶ *normal form* or *irreducible* with respect to E if there is no term t such that $s \longrightarrow_E t$.

We say that a rewrite system E is noetherian (confluent) if the corresponding rewrite relation $\longrightarrow_E$ is noetherian (confluent), and we say that E is

▶ *convergent* if it is both noetherian and confluent.

Convergent systems enjoy the property that each term has a unique normal form. Furthermore, if we want to decide whether an equation $s = t$ logically follows from a set of equations E, and E is a convergent rewrite system, then it is enough to see if the normal forms of s and t with respect to E coincide (cf [13, Section 2.4]).

This is the main motivation behind the completion procedure [24] that attempts to construct a convergent rewrite system from a given set E of equations. The kernel of this procedure is based on the superposition algorithm in combination with the critical pair lemma [13] or [21, Lemma 3.1]. (See also [24, Corollary of Theorem 5] or [21, Theorem 3.2].) This lemma can be used to prove that certain rewrite systems are confluent. In particular we have the following case. A rewrite system E is called

- ▶ *left-reduced* if for every rule $s = t$ of E, s is irreducible with respect to $E \backslash \{s = t\}$.

Lemma 2.3.1 *A left-reduced and noetherian ground rewrite system is convergent.*

Proof. Follows from the superposition algorithm and Lemma 3.1 in Huet [21]. ⊠

That particular property is also pointed out by Bachmair and Ganzinger [3, Section 2.3].

We now let $\succ$ stand for the following fixed ordering between terms:

- ▶ $t \succ s$ iff t has more symbols that s.

On a couple of occasions $\succ$ will temporarily be extended so that $a \succ b$ holds between certain constants a and b. The important property that will not be violated in that case is that the set of all such a's is disjoint from the set of all such b's.

Note It is clear (even in the extended case) that if $E^{\succ}$ is a ground rewrite system then it is noetherian (in any reduction step either the number of symbols or the number of a's (as above) decreases).

2.4 Deterministic Finite Automata

We will make use the following definitions. We follow Hopcroft and Ullman [20]. Formally, a

- ▶ *deterministic finite automaton (DFA) M* is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
  - Q is a finite set of *states*,
  - Σ is a finite *input alphabet*,
  - $\delta : Q \times \Sigma \to Q$ is the *transition function* (δ can be partial, i.e., undefined for certain elements of $Q \times \Sigma$),
  - $q_0 \in Q$ is the *initial* state, and
  - $F \subseteq Q$ is the set of *final states*.

We assume that the states and the input alphabet are *disjoint* subsets of L_0. When we say that a constant *occurs* in M we mean that it is either in Q or in Σ. Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. We say that the

- ▶ *language accepted by M*, $L(M)$, is the set of all strings $a_1 a_2 \ldots a_n \in \Sigma^*$, $n \geq 0$, such that $\delta(q_{j-1}, a_j) = q_j$ for $1 \leq j \leq n$, where q_n is a final state.

We say that two DFAs are

- ▶ *state-disjoint* if their sets of states are disjoint.

We extend this notion to a family of DFAs in the usual manner. We will also make use of the following definition. Let $M = (Q, \Sigma, \delta, q_0, F)$ and M' be two DFAs. We say that

▶ *M' is the extension of M with (a, f)*, if a and f are distinct constants that don't occur in M and

$$M' = (Q \cup \{f\}, \quad \Sigma \cup \{a\}, \quad \delta \cup \{\, (q, a) \mapsto f \mid q \in F \,\}, \quad q_0, \quad \{f\}),$$

we write $M[a, f]$ for M',

The reason why the definition is useful is that $M[a, f]$ has exactly one final state f and that Lemma 2.4.1 holds.

Let r be a regular expression. When it is clear from the context, we write r for the regular set that it denotes. We sometimes also use regular sets in regular expressions. For example, if a is the regular expression consisting of just the constant a and R is a regular set then Ra stands for the regular set $\{\, wa \mid w \in R \,\}$.

Lemma 2.4.1 *Let M be a DFA. Then $L(M[a, f]) = L(M)a$.*

Proof. Immediate from the definition of $M[a, f]$. ⊠

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. We define $\mathcal{G}_M$ as the following set of equations and call it

▶ *the graph of M*, $\mathcal{G}_M = \{\, a \centerdot q = p \mid \delta(q, a) = p \,\}$,

i.e., $\mathcal{G}_M$ represents the graph of the transition function of M with the first two arguments reversed. We generalize the notion of graph to a (finite) family $\mathcal{M} = \{M_i\}_{i \in I}$ of DFAs as follows:

$$\mathcal{G}_{\mathcal{M}} = \bigcup_{i \in I} \mathcal{G}_{M_i}.$$

The following lemma will be the key property in many proofs.

Lemma 2.4.2 *Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA, then $\mathcal{G}_M$ is a convergent rewrite system.*

Proof. All the equations in $\mathcal{G}_M$ have the form $a \centerdot p = q$ where a, p and q are constants. So $\mathcal{G}_M^{\succ}$ is a rewrite system. Furthermore, $\mathcal{G}_M$ is left-reduced because δ is a function. The statement follows now by using Lemma 2.3.1 and the note after that lemma. ⊠

Let M be a DFA. What Lemma 2.4.2 tells us is that, whenever we want to prove that an equation logically follows from $\mathcal{G}_M$, it suffices to prove that both sides of that equation reduce to the same normal form with respect to $\mathcal{G}_M$ (and vice versa ofcource). We will also need the following stronger version of Lemma 2.4.2.

Lemma 2.4.3 *Let $\mathcal{M}$ be a state-disjoint (finite) family of DFAs, then $\mathcal{G}_{\mathcal{M}}$ is a convergent rewrite system.*

Proof. Obvious generalization of the proof of Lemma 2.4.2. Note that the union of the transition functions of the members of $\mathcal{M}$ is still a function because of the state-disjointness of $\mathcal{M}$. ⊠

Chapter 3

Words and Sentences

Here we prove some useful properties about representing regular sets. Words are certain terms of L that represent strings, and sentences are certain terms that represent sequences of strings.

The main theorem of this chapter is Theorem 3.2.4. Many properties of simultaneous rigid equations regarding representation of regular sets follow from it, for example Theorem 3.1.3. A result corresponding to Theorem 3.1.3 is stated also in [19, 33].[1] Also several theorems used in Plaisted [33, Theorems 8.2–8.11] can be stated as corollaries of Theorem 3.2.4.

3.1 Words

The basic components in our later constructions are words. Words are our choice of representing strings of characters, where characters are (represented by) just constants. Mostly we will use the letters v and w to stand for strings of constants. Formally, we say that a gound term t of L is a

- ▶ *q-word* or simply a *word* when it has the form $a_1 \centerdot a_2 \centerdot \cdots \centerdot a_n \centerdot q$ for some $n \in \mathbb{N}$ where all the a_i and q are constants. If $n = 0$ then t is said to be *empty*.

If t is a word $a_1 \centerdot a_2 \centerdot \cdots \centerdot a_n \centerdot q$ we mostly use the shorthand $v \centerdot q$ for t where v is the string $a_1 a_2 \cdots a_n$. We also say that t

- ▶ *represents* the string v, in symbols $\widehat{t} = v$.

Note that any constant q is an empty q-word and represents ϵ. Mostly we want to be more specific and talk about strings in certain regular sets. Let R be a regular set over some set of constants in L. We say a ground term t of L is a

- ▶ *word in R* if t is a word and it represents a string in R.

For any string v we write

- ▶ v^{r} for the reverse of v, and for a set of strings R we write R^{r} for $\{ v^{\mathrm{r}} \mid v \in R \}$.

It is well-known that if R is a regular set then so is R^{r}, see for example Hopcroft and Ullman [20, p 281]. The following two lemmas will be used to prove Theorem 3.1.3 and they will also be used in later sections.

Lemma 3.1.1 *Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA, let $M' = M[a, f]$, and let t be a q_0-word. The following statements are equivalent:*

[1] There is a minor technical mistake in [33, Theorem 8.5] (corresponding to our Theorem 3.1.3) where, given a regular set R, one has to consider a DFA that accepts the *reverse* of R, not R itself.

1. t is a word in $L(M)^{\mathrm{r}}$ (i.e., $\widehat{t}^{\mathrm{r}} \in L(M)$),

2. $t \xrightarrow{*}_{\mathcal{G}_M} q$ for some $q \in F$,

3. $a \centerdot t \xrightarrow{*}_{\mathcal{G}_{M'}} f$.

Proof. We have that $t = a_n \cdots a_2 a_1 \centerdot q_0$ for some $n \in \mathbb{N}$ where each a_i is a constant. Let $v = a_1 a_2 \cdots a_n$, i.e., $v = \widehat{t}^{\mathrm{r}}$.

[**Proof of '1 $\Rightarrow$ 2'**] Assume $v \in L(M)$. Thus $\delta(q_{i-1}, a_i) = q_i$ for $1 \leq i \leq n$ where $q_n \in F$. So $\{ a_i \centerdot q_{i-1} = q_i \mid 1 \leq i \leq n \} \subseteq \mathcal{G}_M$ and thus we can construct the reduction

$$a_n \cdots a_2 a_1 \centerdot q_0 \longrightarrow_{\mathcal{G}_M} a_n \cdots a_2 \centerdot q_1 \xrightarrow{*}_{\mathcal{G}_M} a_n \centerdot q_{n-1} \longrightarrow_{\mathcal{G}_M} q_n. \tag{3.1}$$

So $t \xrightarrow{*}_{\mathcal{G}_M} q_n$ and $q_n \in F$.

[**Proof of '2 $\Rightarrow$ 1'**] Assume $t \xrightarrow{*}_{\mathcal{G}_M} q_n$ for some $q_n \in F$. From the structure of t and the fact that all the rules in $\mathcal{G}_M$ have the form $c_1 \centerdot c_2 = c_3$ where $\{c_1, c_2, c_3\}$ are constants, it follows that the reduction must have the form (3.1) and $\{ a_i \centerdot q_{i-1} = q_i \mid 1 \leq i \leq n \} \subseteq \mathcal{G}_M$. It follows by definition of $\mathcal{G}_M$ that v is accepted by M.

[**Proof of '1 $\Leftrightarrow$ 3'**] Apply '1 $\Leftrightarrow$ 2' to M' (for M) and $a \centerdot t$ (for t) to obtain that $va \in L(M')$ iff $a \centerdot t \xrightarrow{*}_{\mathcal{G}_{M'}} f$. But $L(M') = L(M)a$ by Lemma 2.4.1. ⊠

Lemma 3.1.2 *Let Σ be a set of constants and q a constant not in Σ. There is a set of equations Word_Σ^q such that θ solves $\mathrm{Word}_\Sigma^q \models_\forall x = q$ iff $x\theta$ is a q-word in Σ^*.*

Proof. Let $\mathrm{Word}_\Sigma^q = \{ a \centerdot q = q \mid a \in \Sigma \}$. Note that Word_Σ^q is the graph of the trivial DFA that has one state q and accepts Σ^*. Since Word_Σ^q is convergent it is enough to show that $x\theta \xrightarrow{*}_{\mathrm{Word}_\Sigma^q} q$ iff $x\theta$ is a q-word in Σ^*, which is easy to prove. ⊠

Theorem 3.1.3 *Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. There is a system $S(x)$ of rigid equations such that θ solves $S(x)$ iff $x\theta$ is a q_0-word in $L(M)^{\mathrm{r}}$.*

Proof. Let

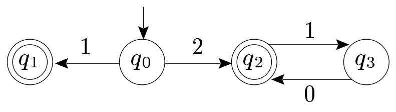
$$S(x) = \begin{cases} \mathrm{Word}_\Sigma^{q_0} & \models_\forall \; x = q_0, \\ \mathcal{G}_{M'} & \models_\forall \; a \centerdot x = f \end{cases}$$

where $\mathrm{Word}_\Sigma^{q_0}$ is given by Lemma 3.1.2 and $M' = M[a, f]$.

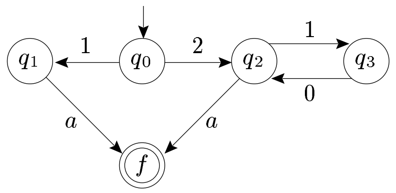
[**Proof of '$\Rightarrow$'**] Assume θ solves $S(x)$. By Lemma 3.1.2 $x\theta$ is a q_0-word in Σ^*, and by Lemma 2.4.2 $a \centerdot x\theta \xrightarrow{*}_{\mathcal{G}_{M'}} f$. Use now Lemma 3.1.1.

[**Proof of '$\Leftarrow$'**] Assume $x\theta$ is a q_0-word in $L(M)^{\mathrm{r}} \subseteq \Sigma^*$. So θ solves the first rigid equation according to Lemma 3.1.2, and $a \centerdot x\theta \xrightarrow{*}_{\mathcal{G}_{M'}} f$ follows from Lemma 3.1.1. So θ solves the second rigid equation by Lemma 2.4.2. ⊠

Example 3.1.4 To illustrate the construction above, consider the DFA M such that $L(M) = 1 + 2(10)^*$, with the following transition diagram:



The DFA $M' = M[a, f]$ has then the following transition diagram:



From these diagrams we can see that the graphs of M and M' are as follows:

$$\begin{array}{rcl} \mathcal{G}_M & = & \{1 \,.\, q_0 = q_1, \quad 2 \,.\, q_0 = q_2, \quad 1 \,.\, q_2 = q_3, \quad 0 \,.\, q_3 = q_2\} \\ \mathcal{G}_{M'} & = & \mathcal{G}_M \cup \{a \,.\, q_1 = f, \quad a \,.\, q_2 = f\} \end{array}$$

For example if $x\theta$ represents the string 012 then

$$\begin{array}{rcl} a012 \,.\, q_0 & \longrightarrow_{\mathcal{G}_M} & a01 \,.\, q_2 \\ & \longrightarrow_{\mathcal{G}_M} & a0 \,.\, q_3 \\ & \longrightarrow_{\mathcal{G}_M} & a \,.\, q_2 \\ & \longrightarrow_{\mathcal{G}_{M'}} & f \end{array}$$

which shows that $210 \in L(M)$, whereas if $x\theta$ represents the string 0012 then $x\theta$ reduces to $0 \,.\, q_2$ and there is no rule to reduce this further, so $a0 \,.\, q_2$ can't be reduced to f showing that $2100 \notin L(M)$. □

3.2 Sentences

Sentences are just representations of sequences of strings. Let us first choose a fixed constant [] ("nil") of L. Formally, a ground term t of L is called a

- ▶ *sentence* if it has the form $t_1 \,.\, t_2 \,.\, \cdots \,.\, t_n \,.\, []$ for some $n \in \mathbb{N}$ where each t_i is a word. If $n = 0$ then t is said to be *empty*.

We use $[t_1, t_2, \ldots, t_n]$ as a shorthand for the corresponding sentence. We say that a sentence $t = [t_1, t_2, \ldots, t_n]$

- ▶ *represents* the sequence of strings $\widehat{t} = (\widehat{t_1}, \widehat{t_2}, \ldots, \widehat{t_n})$.

Our aim here is to represent sequences of strings, where each string belongs to some member of a given family of regular sets, such that the sequence has some given regular pattern. For that purpose we introduce the following notion.

Let $\Sigma, \Gamma \subset L_0$ and let $\{R_q\}_{q \in \Gamma}$ be a family of regular sets over Σ and let R be a regular set over Γ. We say that a sentence $t = [t_1, t_2, \ldots, t_n]$ is a

- ▶ *sentence in* $\{R_q\}^R$ if each t_i is a q_i-word in R_{q_i} for some $q_i \in \Gamma$ and $q_1 q_2 \cdots q_n \in R$.

In other words, t is a sentence in $\{R_q\}^R$ iff any q-word of t is a word in R_q, and if we replace all the words of t with the corresponding empty words then the resulting term is a []-word in R. When all the members of the family are the same regular set then we drop the index in our notation.

Example 3.2.1 Some examples to illustrate the definition:

1. t is a sentence in $\{a^+\}^{(b+c)^*}$ means that t is a (possibly empty) sentence of b-words and c-words such that each one represents a nonempty string of a's. For example $[a \,.\, b, aaa \,.\, c]$ is such a sentence.

2. t is a sentence in $\{R\}^b$ means that t is a unit sentence $[s]$ where s is a b-word in R,

3. t is a sentence in $\{R_a, R_b, R_c\}^{ab^*c}$ means that the first word of t is an a-word in R_a, the last word of t is a c-word in R_c and the middle ones (if any) are b-words in R_b.

□

Let $\Gamma \subset L_0$. We write

- ▶ E_Γ for the set of equations $\{\, q = \text{choice}(\Gamma) \mid q \in \Gamma \setminus \{\text{choice}(\Gamma)\} \,\}$, where 'choice' is some fixed choice function for $\mathcal{P}_\omega(L_0)$.

The following two lemmas will be used in the proof of Theorem 3.2.4.

Lemma 3.2.2 *Let $\Sigma, \Gamma \subset L_0 \setminus \{[]\}$ be disjoint. There is a set of equations $\text{Sent}_\Sigma^\Gamma$ such that θ solves $\text{Sent}_\Sigma^\Gamma \models_\forall x = []$ iff $x\theta$ is a sentence in $\{\Sigma^*\}^{\Gamma^*}$.*

Proof. Let ε be the constant choice(Γ) and let $\text{Sent}_\Sigma^\Gamma$ be the following set of equations:

$$\text{Sent}_\Sigma^\Gamma \quad = \quad E_\Gamma \cup \text{Word}_\Sigma^\varepsilon \cup \{\varepsilon \,.\, [] = []\},$$

where $\text{Word}_\Sigma^\varepsilon$ is given by Lemma 3.1.2.

Extend the ordering $\succ$ so that $q \succ \varepsilon$ for all $q \in \Gamma \setminus \{\varepsilon\}$. Clearly, $(\text{Sent}_\Sigma^\Gamma)^\succ$ is left-reduced (check that for any rule $t = s$ in $\text{Sent}_\Sigma^\Gamma$, t can be reduced only with this rule, here the disjointness of Σ and Γ is needed). It follows from Lemma 2.3.1 and the note after that lemma that $\text{Sent}_\Sigma^\Gamma$ is convergent. It suffices therefore to prove the following statement:

$$x\theta \xrightarrow{*}_{\text{Sent}_\Sigma^\Gamma} [] \quad \Leftrightarrow \quad x\theta \text{ is a sentence in } \{\Sigma^*\}^{\Gamma^*}.$$

[**Proof of '$\Rightarrow$'**] By induction on the length of the reduction $x\theta \xrightarrow{*}_{\text{Sent}_\Sigma^\Gamma} []$.

[**Base case**] $x\theta = []$. Trivially, $[]$ is a sentence in $\{\Sigma^*\}^{\Gamma^*}$ because $\epsilon \in \Gamma^*$.

[**Induction case**] $x\theta \xrightarrow{*} \varepsilon \,.\, [] \longrightarrow []$. Thus $x\theta = t \,.\, s$ and $s \xrightarrow{*} []$ and $t \xrightarrow{*} \varepsilon$.

We prove first by induction on the length of the reduction of $t \xrightarrow{*} \varepsilon$ that t is a q-word in Σ^* for some $q \in \Gamma$.

[**Base case**] $t = \varepsilon$. Trivially ε is an ε-word in Σ^*.

[**Induction case**] We have two cases to consider, based on what the last rule of the reductions is.

- i. If $t \xrightarrow{*} q \longrightarrow \varepsilon$ by some rule $q = \varepsilon$ in E_Γ, then $t = q$ because q is the right hand side of no rule, and trivially q is a q-word in Σ^*.
- ii. Otherwise $t \xrightarrow{*} a \,.\, \varepsilon \longrightarrow \varepsilon$ by some rule $a \,.\, \varepsilon = \varepsilon$ in $\text{Word}_\Sigma^\varepsilon$. Then $t = a \,.\, t'$ and $t' \xrightarrow{*} \varepsilon$ because a is the right hand side of no rule. By the induction hypothesis t' is a q-word in Σ^* for some $q \in \Gamma$, and thus so is $a \,.\, t'$.

It follows that t is a q-word in Σ^* for some fixed $q \in \Gamma$.

By the induction hypothesis s is a sentence in $\{\Sigma^*\}^{\Gamma^*}$, so $s = [s_1, s_2, \ldots, s_m]$ for some $m \in \mathbb{N}$ where each s_i is a q_i-word in Σ^* for some $q_i \in \Gamma$.

Consequently $x\theta = [t, s_1, s_2, \ldots, s_m]$ is a sentence in $\{\Sigma^*\}^{\Gamma^*}$.

[**Proof of '$\Leftarrow$'**] Assume that $x\theta$ is a sentence in $\{\Sigma^*\}^{\Gamma^*}$. Thus $x\theta = [t_1, t_2, \ldots, t_n]$ for some $n \in \mathbb{N}$ where each t_i is a q_i-word in Σ^* for some $q_i \in \Gamma$.

1. For each t_i, reduce q_i in t_i to ε by using the rule $q_i = \varepsilon$ in E_Γ. Let us call the resulting term s_i. So each s_i is an ε-word in Σ^*.

2. Use the rules from $\text{Word}_\Sigma^\varepsilon$ to reduce each s_i to ε.

3. Finally, use the rule $\varepsilon \,.\, [] = []$ to reduce $[\varepsilon, \varepsilon, \ldots, \varepsilon]$ to $[]$.

From (1–3) follows that $x\theta \xrightarrow{*}_{\mathrm{Sent}_\Sigma^\Gamma} []$. $\boxtimes$

Lemma 3.2.3 *Let $\mathcal{M}$ be a state-disjoint family of DFA's and M a fixed member of $\mathcal{M}$. The following holds for all states q of M, all q-words t and all terms s. If $t \xrightarrow{*}_{\mathcal{G}_\mathcal{M}} s$ then $t \xrightarrow{*}_{\mathcal{G}_M} s$ and s is a p-word for some state p of M.*

Proof. By easy induction on the length of the reduction of $t \xrightarrow{*}_{\mathcal{G}_\mathcal{M}} s$. Let q be a state of M, t a q-word and s a term.

[**Base case**] If the length of the reduction is 0, i.e., $t = s$, then trivially $t \xrightarrow{*}_{\mathcal{G}_M} s$ and s is a q-word.

[**Induction case**] Assume $t \longrightarrow_{\mathcal{G}_\mathcal{M}} t' \xrightarrow{*}_{\mathcal{G}_\mathcal{M}} s$. Since t is a word, the rule that is used in the first step must be of the form $a \centerdot q = q'$ (for some constants a and q'), yielding a q'-word t'. But that rule must be in $\mathcal{G}_M$ because q is a state of M and the family is state-disjoint, i.e., $t \longrightarrow_{\mathcal{G}_M} t'$. So q' is a state of M. From the induction hypothesis follows now that $t \xrightarrow{*}_{\mathcal{G}_M} s$ and s is a p-word for some state p of M. $\boxtimes$

We can now state the main theorem of this section.

Theorem 3.2.4 *Let $\Sigma, \Gamma \subset L_0 \setminus \{[]\}$ be disjoint. Let $\{R_c\}_{c\in\Gamma}$ be a family of regular sets over Σ. Let R be a regular set over Γ. There exists a system $S(x)$ of rigid equations such that θ solves $S(x)$ iff $x\theta$ is a sentence in $\{R_c\}^R$.*

Proof. We start by constructing a state-disjoint family $\mathcal{M} = \{M_c\}_{c\in\Gamma\cup\{[]\}}$ of DFA's with the following properties. First let all the members M_c, $c \in \Gamma$, be such that

$$L(M_c) = {R_c}^{\mathrm{r}}, \quad M_c = (Q_c, \Sigma, \delta_c, c, F_c).$$

Let $F_\Gamma = \cup_{c\in\Gamma} F_c$. Let $M = (Q, F_\Gamma, \delta, [], F)$ be a DFA such that $q_m \cdots q_2 q_1 \in L(M)$ iff there exists $c_1 c_2 \cdots c_m \in R$ such that $q_i \in F_{c_i}$ for $1 \le i \le m$. (It is easy to construct M from a DFA for R^{r} by replacing any transition $(p, c) \mapsto p'$ in the latter with the set $\{\, (p, q) \mapsto p' \mid q \in F_c \,\}$ of transitions.) Finally, let $M_{[]} = M[a, f]$.

Assume also all the DFAs above to be such that $\mathcal{M}$ is state-disjoint.

Let now $S(x)$ be the following system of rigid equations:

$$S(x) \quad = \quad \left\{ \begin{array}{rcl} \mathrm{Sent}_\Sigma^\Gamma & \models_\forall & x = [], \\ \mathcal{G}_\mathcal{M} & \models_\forall & a \centerdot x = f \end{array} \right.$$

where $\mathrm{Sent}_\Sigma^\Gamma$ is given by Lemma 3.2.2. We will prove that θ solves $S(x)$ iff $x\theta$ is a sentence in $\{R_c\}^R$. First of all, we make the following observations:

- If θ solves $S(x)$ then $x\theta$ is a sentence in $\{\Sigma^*\}^{\Gamma^*}$ by Lemma 3.2.2.
- If $x\theta$ is a sentence in $\{R_c\}^R$, it is by definiton also a sentence in $\{\Sigma^*\}^{\Gamma^*}$ and solves therefore $\mathrm{Sent}_\Sigma^\Gamma \models_\forall x = []$ by Lemma 3.2.2.

Based on these observations and Lemma 2.4.3 it is sufficient to prove the following: if $x\theta$ is a sentence in $\{\Sigma^*\}^{\Gamma^*}$ then

$$a \centerdot x\theta \xrightarrow{*}_{\mathcal{G}_\mathcal{M}} f \quad \Leftrightarrow \quad x\theta \text{ is a sentence in } \{R_c\}^R.$$

So let $x\theta = [t_1, t_2 \ldots, t_m]$ be a sentence in $\{\Sigma^*\}^{\Gamma^*}$, where each t_j is an c_j-word in Σ^* for some $\{c_1, c_2, \ldots, c_m\} \subseteq \Gamma$.

[**Proof of '$\Rightarrow$'**] Assume that $a \centerdot x\theta \xrightarrow{*}_{\mathcal{G}_\mathcal{M}} f$. This reduction is possible only if

1) $t_j \xrightarrow{*}_{\mathcal{G}_\mathcal{M}} q_j$ for some fixed constant q_j for $1 \le j \le m$, and

2) $aq_1q_2\cdots q_m \centerdot [] \xrightarrow{*}_{\mathcal{G}_{\mathcal{M}}} f$, let $\vec{q} = q_1q_2\cdots q_m$.

Let j be fixed and consider (1). By applying Lemma 3.2.3 to (1) and that t_j is a c_j-word (note that c_j is a state of M_{c_j}) we obtain (1'). By applying Lemma 3.2.3 to (2) we obtain (2').

1') $t_j \xrightarrow{*}_{\mathcal{G}_{M_{c_j}}} q_j$ and q_j is a state of M_{c_j}, and

2') $a\vec{q} \centerdot [] \xrightarrow{*}_{\mathcal{G}_{M_{[]}}} f$.

From (2') follows, by using Lemma 3.1.1 (recall that $M_{[]} = M[a, f]$), that $\vec{q}^{\,\mathrm{r}} \in L(M)$. By definition of M this implies that $q_j \in F_{c'_j}$, $1 \le j \le m$, for some $c'_1c'_2\cdots c'_m \in R$, where, by state-disjointness and (1'), each $c'_j = c_j$. So

- $c_1c_2\cdots c_m \in R$.

From (1') and that $q_j \in F_{c_j}$ follows, by using Lemma 3.1.1 (recall that the constant c_j is the initial state of M_{c_j}), that

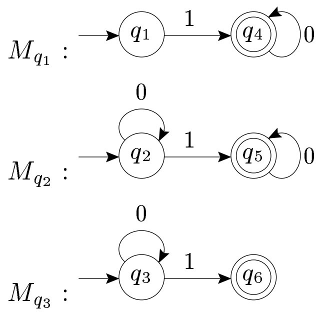
- t_j is a c_j-word in $L(M_{c_j})^{\mathrm{r}} = R_{c_j}$ for $1 \le j \le m$.

From these two points follows that $x\theta$ is a sentence in $\{R_c\}^R$.

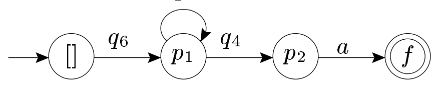
[Proof of '⇐'] Assume that $x\theta$ is a sentence in $\{R_c\}^R$, i.e., each t_j is a c_j-word in $R_{c_j} = L(M_{c_j})^{\mathrm{r}}$ and $c_1c_2\cdots c_m \in R$. By Lemma 3.1.1 $t_j \xrightarrow{*}_{\mathcal{G}_{M_{c_j}}} q_j$ for some $q_j \in F_{c_j}$. Let $\vec{q} = q_1q_2\cdots q_m$. So, by definition of M, $\vec{q} \in L(M)^{\mathrm{r}}$. Thus $a\vec{q} \centerdot [] \xrightarrow{*}_{\mathcal{G}_{M_{[]}}} f$ by Lemma 3.1.1. By putting the reductions together, we have that $a \centerdot x\theta \xrightarrow{*}_{\mathcal{G}_{\mathcal{M}}} f$. ⊠

Theorem 3.2.4 turns out to be quite useful. Many properties of simultaneous rigid equations regarding representation of regular sets follow from it, for example Theorem 3.1.3 – just consider unit sentences. Some other cases were mentioned in Example 3.2.1. We conclude this section with an example that illustrates the above construction.

Example 3.2.5 Let $\{R_c\}_{c\in\{q_1,q_2,q_3\}}$ be a family of regular sets over $\{0, 1\}$, where $R_{q_1} = 0^*1$, $R_{q_2} = 0^*10^*$ and $R_{q_3} = 10^*$. Let $R = q_1q_2^*q_3$. Assume that M_c for $c \in \{q_1, q_2, q_3\}$ have the transition diagrams below, so $L(M_c) = R_c{}^{\mathrm{r}}$.



From these and R we obtain a DFA $M_{[]}$ with the following transition diagram:



Let $\mathcal{M}$ be the family $\{M_{q_1}, M_{q_2}, M_{q_3}, M_{[]}\}$. We get that

$$\begin{aligned}
\mathcal{G}_{\mathcal{M}} \quad = \quad & \{1 \centerdot q_1 = q_4, \quad 0 \centerdot q_4 = q_4\} \cup \\
& \{0 \centerdot q_2 = q_2, \quad 1 \centerdot q_2 = q_5, \quad 0 \centerdot q_5 = q_5\} \cup \\
& \{0 \centerdot q_3 = q_3, \quad 1 \centerdot q_3 = q_6\} \cup \\
& \{q_6 \centerdot [] = p_1, \quad q_5 \centerdot p_1 = p_1, \quad q_4 \centerdot p_1 = p_2\} \cup \\
& \{a \centerdot p_2 = f\}.
\end{aligned}$$

Take for example

$$t = [001 \centerdot q_1, \quad 0010 \centerdot q_2, \quad 1 \centerdot q_2, \quad 10 \centerdot q_3].$$

Then t is clearly a sentence in $\{R_{q_1}, R_{q_2}, R_{q_3}\}^R$. We can also see that $a \centerdot t \xrightarrow{*}_{\mathcal{G}_{\mathcal{M}}} f$. First each q_i-word of t is reduced to a final state of M_{q_i}, so $t \xrightarrow{*} [q_4, q_5, q_5, q_6]$, then $[q_4, q_5, q_5, q_6] \xrightarrow{*} p_2$ and finally $a \centerdot p_2 \longrightarrow f$. So $a \centerdot t \xrightarrow{*} f$ showing that θ, such that $x\theta = t$, solves the corresponding system $S(x)$ constructed in the proof of Theorem 3.2.4. $\square$

Chapter 4

Shifted Pairing

The purpose of this chapter is to describe a technique, called shifted pairing, that can be used to construct a system of rigid equations, the solutions of which are sentences with certain interesting properties. Similar technique was used by Plaisted [33] and we have taken the term shifted pairing from there. The main result of this chapter is Theorem 4.2.1.

4.1 Encoding Pairs of Strings

Given a set of constants $\Sigma \subset L_0$, we want to encode pairs of strings over Σ in a simple manner. Let $\flat$ be a fixed constant in L_0 called a *blank*. We can assume without loss of generality that $\flat \in \Sigma$, $\Sigma \setminus \{\flat\}$ is nonempty and that we only wish to encode pairs of strings in Σ^* that don't end with a blank (otherwise just expand Σ with $\flat$). We say that a function $\langle\rangle : \Sigma \times \Sigma \to L_0$ is a

▶ *pairing function* for Σ if $\langle\rangle$ is injective and $\langle\Sigma\rangle = \{\, \langle a,b\rangle \mid a,b \in \Sigma \,\}$ is disjoint from Σ, and we associate the following sets of equations with $\langle\rangle$:

$$\begin{aligned}\Pi_1^{\langle\rangle} &= \{\, \langle a,b\rangle = a \mid a,b \in \Sigma \,\},\\ \Pi_2^{\langle\rangle} &= \{\, \langle a,b\rangle = b \mid a,b \in \Sigma \,\}.\end{aligned}$$

We will abbreviate $\Pi_1^{\langle\rangle}$ and $\Pi_2^{\langle\rangle}$ by Π_1 and Π_2, respectively.

Let $\langle\rangle$ be a pairing function for Σ. Consider

$$v = \langle a_1, b_1\rangle\langle a_2, b_2\rangle \cdots \langle a_k, b_k\rangle \in \langle\Sigma\rangle^*$$

for some $k \in \mathbb{N}$ and let $n, m \in \mathbb{N}$ be least such that $a_{n+1}, \ldots, a_k$ and $b_{m+1}, \ldots, b_k$ are blanks. We say that

▶ v *encodes* the pair $(a_1 a_2 \cdots a_n, b_1 b_2 \cdots b_m)$ of strings.

We will write $\langle v, w\rangle$ for any string that encodes the pair (v, w) of strings in $\Sigma^* \setminus \Sigma^*\flat$. Note that $\langle v, w\rangle$ has some arbitrary number of $\langle \flat, \flat\rangle$'s as suffix.

4.2 Shifted Pairing

We want to encode adjacent pairs of strings in a given sequence of strings. Let $\Sigma \subset L_0$. Assume $\flat \in \Sigma$, $\Sigma \setminus \{\flat\}$ is nonempty and let $\langle\rangle$ be a pairing function for Σ. Let $\vec{w} = (w_1, w_2, \ldots, w_n)$ be a *nonempty* sequence of strings in $\Sigma^* \setminus \Sigma^*\flat$. We say that a sequence $\vec{v} = (v_1, v_2, \ldots, v_n)$ of strings in $\langle\Sigma\rangle^*$ is a

▶ *shifted pairing* of $\vec{w}$ if v_i encodes the pair (w_i, w_{i+1}) for $1 \le i < n$ and v_n encodes the pair (w_n, ϵ), i.e., $\vec{v} = (\langle w_1, w_2\rangle, \langle w_2, w_3\rangle, \ldots, \langle w_{n-1}, w_n\rangle, \langle w_n, \epsilon\rangle)$.

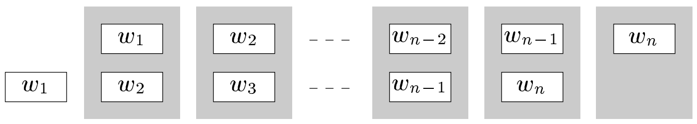


Figure 4.1: Shifted pairing.

This is illustrated in Figure 4.1.

Theorem 4.2.1 *Let $\Sigma \subset L_0$ be such that $\text{ƀ} \in \Sigma$, $\Sigma \setminus \{\text{ƀ}\}$ is nonempty, and let $\langle\rangle$ be a pairing function for Σ. Let also $\Gamma \subset L_0$. Assume that Σ, $\langle\Sigma\rangle$, Γ and $\{[]\}$ are all pairwise disjoint.*

Let $q \in \Sigma \setminus \{\text{ƀ}\}$. There is a system $\mathrm{SP}_q(z,x,y)$ of rigid equations such that

- *θ solves $\mathrm{SP}_q(z,x,y)$ iff*
- *$\widehat{y\theta}$ is a shifted pairing of $\widehat{x\theta}$ and the first string of $\widehat{x\theta}$ is $\widehat{qz\theta}$,*

for any substitution θ such that $z\theta$ is a c-word in $(\Sigma \setminus \{\text{ƀ}\})^*$ for some $c \in \Gamma$, $x\theta$ is a sentence in $\{\Sigma^+ \setminus \Sigma^*\text{ƀ}\}^{\Gamma^+}$ and $y\theta$ is a sentence in $\{\langle\Sigma\rangle^+\}^{\Gamma^+}$.

Proof. Let $\varepsilon = \mathrm{choice}(\Gamma)$. Assume that $z\theta$ is a word in $(\Sigma \setminus \{\text{ƀ}\})^*$ and let $t_w = q \cdot z\theta$ and $w = \widehat{t_w}$. Assume also that $x\theta = t$ and $y\theta = s$ are sentences in $\{\Sigma^+ \setminus \Sigma^*\text{ƀ}\}^{\Gamma^+}$ and $\{\langle\Sigma\rangle^+\}^{\Gamma^+}$, respectively. So

$$\begin{array}{rcll} t & = & [t_1, t_2, \ldots, t_n], & n \geq 1, \quad t_i \text{ is a word in } \Sigma^+ \setminus \Sigma^*\text{ƀ}, \\ s & = & [s_1, s_2, \ldots, s_m], & m \geq 1, \quad s_i \text{ is a word in } \langle\Sigma\rangle^+. \end{array}$$

Define $\mathrm{SP}_q(z,x,y)$ as the following system:

$$\mathrm{SP}_q(z,x,y) \quad = \quad \left\{ \begin{array}{lll} \underbrace{\Pi_1 \cup E_\Gamma \cup \{\text{ƀ} \cdot \varepsilon = \varepsilon\}}_{E_1} & \models_\forall & x = y, \\ \underbrace{\Pi_2 \cup E_\Gamma \cup \{\text{ƀ} \cdot \varepsilon = \varepsilon,\ \varepsilon \cdot [] = []\}}_{E_2} & \models_\forall & x = (q \cdot z) \cdot y \end{array} \right.$$

Extend the ordering $\succ$ so that $c \succ \varepsilon$ for all $c \in \Gamma \setminus \{\varepsilon\}$ and $\langle a,b\rangle \succ a$, $\langle a,b\rangle \succ b$ for all $a,b \in \Sigma$. From the assumption that the sets Σ, $\langle\Sigma\rangle$, Γ and $\{[]\}$ are pairwise disjoint, it follows that $E_1^\succ$ and $E_2^\succ$ are left-reduced. By using Lemma 2.3.1 and the note following that lemma, we get that E_1 and E_2 are convergent.

The only rules in E_1 or E_2 that can be used to reduce t are in E_Γ. Let t' be the normal form of t with respect to E_1 or E_2, so

$$t' = [w_1 \cdot \varepsilon, w_2 \cdot \varepsilon, \ldots, w_n \cdot \varepsilon]$$

where $w_i = \widehat{t_i}$ for $1 \leq i \leq n$. It is therefore sufficient to prove the statement:

$$s \xrightarrow{*}_{E_1} t' \text{ and } t_w \cdot s \xrightarrow{*}_{E_2} t' \text{ iff } \widehat{s} \text{ is a shifted pairing of } \widehat{t'} \text{ and } w_1 = w.$$

[**Proof of '$\Rightarrow$'**] Assume $s \xrightarrow{*}_{E_1} t'$ and $t_w \cdot s \xrightarrow{*}_{E_2} t'$. From $s \xrightarrow{*}_{E_1} t'$ follows that $n = m$ and

$$s_i \xrightarrow{*}_{E_1} w_i \cdot \varepsilon \quad (1 \leq i \leq n).$$

From $t_w \cdot s \xrightarrow{*}_{E_2} t'$ follows that $w_1 = w$ and $s \xrightarrow{*}_{E_2} [w_2 \cdot \varepsilon, \ldots, w_n \cdot \varepsilon]$. This last reduction implies that

$$s_i \xrightarrow{*}_{E_2} w_{i+1} \cdot \varepsilon \quad (1 \leq i < n), \quad s_n \cdot [] \xrightarrow{*}_{E_2} [].$$

Let $i \in \{1, \ldots, n-1\}$ be fixed. From $s_i \xrightarrow{*}_{E_1} w_i \mathbin{.} \varepsilon$ and $s_i \xrightarrow{*}_{E_2} w_{i+1} \mathbin{.} \varepsilon$ follows that $\widehat{s_i}$ encodes the pair (w_i, w_{i+1}). The reduction $s_n \mathbin{.} [] \xrightarrow{*}_{E_2} []$ is possible only if $s_n \xrightarrow{*}_{E_2} \varepsilon$. Together with $s_n \xrightarrow{*}_{E_1} w_n \mathbin{.} \varepsilon$ it follows that $\widehat{s_n}$ encodes the pair (w_n, ϵ). So $\widehat{s}$ is a shifted pairing of $\widehat{t'}$ and $w_1 = w$.

[**Proof of '$\Leftarrow$'**] Assume $\widehat{s}$ is a shifted pairing of $\widehat{t'}$ and $w_1 = w$. So $n = m$ and $\widehat{s_i}$ encodes the pair (w_i, w_{i+1}) for $1 \leq i \leq n$ where $w_{n+1} = \epsilon$.

We prove first that $s \xrightarrow{*}_{E_1} t'$, by proving that $s_i \xrightarrow{*}_{E_1} w_i \mathbin{.} \varepsilon$ for $1 \leq i \leq n$. Let i be fixed. Use first the rules in Π_1 and a rule in E_Γ to obtain the word $w_i \vec{b} \mathbin{.} \varepsilon$. Use then the rule $b \mathbin{.} \varepsilon = \varepsilon$ to remove the blanks.

We prove now that $t_w \mathbin{.} s \xrightarrow{*}_{E_2} t'$. Trivially $t_w \xrightarrow{*}_{E_\Gamma} w_1 \mathbin{.} \varepsilon$ since $w = w_1$. Let $i \in \{1, \ldots, n\}$ be fixed. To see that $s_i \xrightarrow{*}_{E_2} w_{i+1} \mathbin{.} \varepsilon$, use first the rules in Π_2 and a rule in E_Γ. Remove then the blanks with the rule $b \mathbin{.} \varepsilon = \varepsilon$. Finally use the rule $\varepsilon \mathbin{.} [] = []$ to get rid of the last ε. ⊠

This theorem is the kernel behind the main result of this report (Theorem 5.2.1).

Chapter 5

Uniform Characterization of RE with SREU

The main result of this chapter and the whole report is Theorem 5.2.1 which shows that any r. e. set can be represented by a system of simultaneous rigid equations which is obtained uniformly in the r. e. index of that set. The construction of the system is in fact easily seen to be primitive recursive.

In particular, this theorem and the way the system in the theorem is constructed imply that given any r. e. set W over some alphabet Σ and a string w over Σ, one can effectively construct a system of rigid equations, having ground left hand sides and only two variables, which has a solution iff $w \in W$. So SREU is undecidable already with ground left hand sides (which was also shown by Plaisted [33]) and only two variables (that is a new result).

We start by giving the formal definitions of the notions that are needed. The terminology is taken mainly from Hopcroft and Ullmann [20]. We then prove the main result, which is Theorem 5.2.1.

5.1 The Turing Machine Model

In this section we give the formal definition of the Turing machine model that we will use and some definitions of related concepts, in particular the notion of valid computations. Formally, a

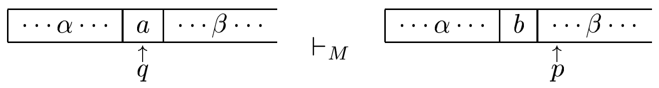
- *Turing machine* (TM) M is a 7-tuple $(Q, \Sigma_0, \Sigma_1, \delta, q_0, \text{\textbeta}, F)$, where
  - Q is the set of all *states* of M,
  - Σ_0 is the *input alphabet* not including $\bar{b}$,
  - $\Sigma_1 = \Sigma_0 \cup \{\bar{b}\}$,
  - $\delta : Q \times \Sigma_1 \to Q \times \Sigma_1 \times \{\mathrm{L}, \mathrm{R}\}$ is the *transition function*,
  - $q_0 \in Q$ is the *initial* state, and
  - $F \subseteq Q$ is the set of *final* states.

We also assume here, as we did with DFAs, that Q and Σ_1 are disjoint subsets of L_0. The constant $\bar{b}$ is still called a blank. Let $M = (Q, \Sigma_0, \Sigma_1, \delta, q_0, \bar{b}, F)$ be a TM. An

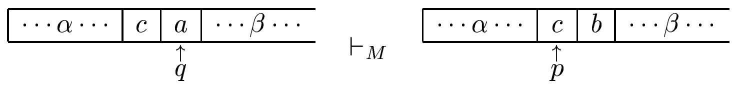
- *instantaneous description* (ID) of M is any string $\alpha q \beta$ where $q \in Q$ and $\alpha \in \Sigma_1^*$ and β is a string in Σ_1^* not ending with a blank.

The intended meaning of an ID $\alpha q \beta$ of M is to give a complete description of a possible execution state of M. There q is the state of the machine, α corresponds to the contents of the tape from the left edge of the tape to (but not including) the symbol pointed to by the tape head, and β is the rest of the contents of the tape terminated by the rightmost nonblank. So any "snapshot" of M during its computation is some ID (there can ofcource exist ID's that can never be reached by M). Let $\vec{b}$ stand for a string of 0 or more blanks. Define

- ▶ *move* as a pair (v, w) of ID's such that if $v\vec{b} = \alpha q a \beta$ and $\delta(q, a) = (p, b, \mathrm{R})$ then $w\vec{b} = \alpha b p \beta$,



  and if $v\vec{b} = \alpha c q a \beta$ and $\delta(q, a) = (p, b, \mathrm{L})$ then $w\vec{b} = \alpha p c b \beta$,



  i.e., w is obtained from v according to the next move function.

The binary relation of all moves of M is denoted by $\vdash_M$, as shown above already, and its transitive and reflexive closure by $\vdash_M^*$. The

- ▶ *language accepted by M*, $L(M)$, is the following set

$$L(M) = \{\, w \in \Sigma_0^* \mid q_0 w \vdash_M^* \alpha p \beta \text{ where } p \in F \text{ and } \alpha p \beta \text{ is an ID} \,\}.$$

Valid Computations

The notions of valid and invalid computations [20] of a TM are a powerful tool in proving undecidability results about context free languages. The technique that is used in our proof (in particular shifted pairing) bears certain similarities to the technique that is used to prove that the language of any TM is given by the intersection of two context free languages [20, Lemma 8.6]. A

- ▶ *valid computation* of M is a nonempty sequence $(w_1, w_2, \ldots, w_n)$ such that
  - each w_i is an ID of M, i.e., $w_i \in \Sigma_1^* Q(\Sigma_1^* \setminus \Sigma_1^* \mathfrak{b})$ for $1 \le i \le n$,
  - w_1 is the *initial* ID, one of the form $q_0 v$ where $v \in \Sigma_0^*$,
  - w_n is a *final* ID, $w_n \in \Sigma_1^* F(\Sigma_1^* \setminus \Sigma_1^* \mathfrak{b})$,
  - $w_i \vdash_M w_{i+1}$ for $1 \le i < n$, i.e., each pair (w_i, w_{i+1}) is a move of M.

We will use the following relationship between valid computations and the language of M without further notice: there is a valid computation of M with initial ID $q_0 v$ iff $v \in L(M)$.

5.2 The Main Theorem

Let M be a Turing machine. In the following theorem we will, effectively from M, construct a system $S_M(z, x, y)$ of rigid equations which represents the language accepted by M by the set of all solutions for z. The auxiliary variables x and y are such that, for any θ that solves $S_M(z, x, y)$, $\widehat{x\theta}$ is a valid computation of M with initial ID $q_0 \widehat{z\theta}$ and $\widehat{y\theta}$ is a shifted pairing of $\widehat{x\theta}$.

This theorem turns out to have some far-reaching consequences. We mention some of them at the the end of this section. The results of the following chapter are also based on this theorem.

Theorem 5.2.1 *Let $M = (Q, \Sigma_0, \Sigma_1, \delta, q_0, \mathfrak{b}, F)$ be a TM and let ε be constant not in Q or Σ_1. There is a system $S_M(z,x,y)$ of rigid equations such that for any substitution θ that solves $S_M(z,x,y)$, $z\theta$ is an ε-word and*

$$L(M) = \{\, \widehat{z\theta} \mid \theta \textit{ solves } S_M(z,x,y)\,\}.$$

Proof. Let $\Sigma = \Sigma_1 \cup Q$ and let $\langle\rangle$ be a pairing function for Σ. Assume also, without loss of generality, that all the sets Σ, $\langle\Sigma\rangle$, $\Gamma = \{\varepsilon, \varepsilon_1\}$ and $\{[]\}$ are pairwise disjoint and that $\varepsilon = \mathrm{choice}(\Gamma)$. Let R_{id}, R_{fin} and R_{mv} be the following regular sets:

$$\begin{array}{rcl} R_{\mathrm{id}} & = & \Sigma_1^* Q(\Sigma_1^* \setminus \Sigma_1^*\mathfrak{b}), \\ R_{\mathrm{fin}} & = & \Sigma_1^* F(\Sigma_1^* \setminus \Sigma_1^*\mathfrak{b}), \\ R_{\mathrm{mv}} & = & \Upsilon^*\Delta\Upsilon^*, \end{array}$$

where

$$\begin{array}{rcl} \Upsilon & = & \{\, \langle a,a\rangle \mid a \in \Sigma_1 \,\}, \\ \Delta & = & \{\, \langle q,b\rangle\langle a,p\rangle \mid \delta(q,a) = (p,b,\mathrm{R}) \,\} \cup \\ & & \{\, \langle c,p\rangle\langle q,c\rangle\langle a,b\rangle \mid \delta(q,a) = (p,b,\mathrm{L}), c \in \Sigma_1 \,\}. \end{array}$$

So $R_{\mathrm{id}} \subseteq \Sigma^+$ and $R_{\mathrm{fin}} \subseteq \Sigma^+$, are the sets of IDs and final IDs, respectively. A string v is in $R_{\mathrm{mv}} \subseteq \langle\Sigma\rangle^+$ iff v encodes a move of M. Use now Theorem 3.2.4 to obtain the systems $S_{\mathrm{id}}(x)$ and $S_{\mathrm{mv}}(y)$ and Lemma 3.1.2 to obtain the system $S_{\mathrm{in}}(z)$, such that

1. θ solves $S_{\mathrm{id}}(x)$ iff $x\theta$ is a sentence in $\{\varepsilon \mapsto R_{\mathrm{id}}, \varepsilon_1 \mapsto R_{\mathrm{fin}}\}^{\varepsilon^*\varepsilon_1}$,
2. θ solves $S_{\mathrm{mv}}(y)$ iff $y\theta$ is a sentence in $\{\varepsilon \mapsto R_{\mathrm{mv}}, \varepsilon_1 \mapsto \langle\Sigma\rangle^+\}^{\varepsilon^*\varepsilon_1}$, and
3. θ solves $S_{\mathrm{in}}(z)$ iff $z\theta$ is an ε-word in Σ_0^*.

Let $\mathrm{SP}_{q_0}(z,x,y)$ be the system obtained from Theorem 4.2.1. So, for any substitution θ that solves the systems given in (1–3) we have in particular that $z\theta$ is an ε-word in $(\Sigma_0 \cup Q)^*$, $x\theta$ is a sentence in $\{\Sigma^+ \setminus \Sigma^*\mathfrak{b}\}^{\Gamma^+}$, and $y\theta$ is a sentence in $\{\langle\Sigma\rangle^+\}^{\Gamma^+}$. Theorem 4.2.1 tells us then that, for any such θ,

4. θ solves $\mathrm{SP}_{q_0}(z,x,y)$ iff $q_0\widehat{z\theta}$ is the first string of $\widehat{x\theta}$ and $\widehat{y\theta}$ is a shifted pairing of $\widehat{x\theta}$.

Define now $S_M(z,x,y)$ as follows:

$$S_M(z,x,y) = S_{\mathrm{id}}(x) \cup S_{\mathrm{mv}}(y) \cup S_{\mathrm{in}}(z) \cup \mathrm{SP}_{q_0}(z,x,y).$$

If there is a substitution θ that solves $S_M(z,x,y)$ then $z\theta$ is an ε-word by item 3. It remains to prove prove that $\{\, \widehat{z\theta} \mid \theta \text{ solves } S_M(z,x,y)\,\} = L(M)$.

[**Proof of '$\subseteq$'**] Assume that θ solves $S_M(z,x,y)$. By item 1, and the definitions of R_{id} and R_{fin},

$$\widehat{x\theta} = (v_1, v_2, \ldots, v_n), \quad v_i \text{ is an ID of } M \text{ for } 1 \le i \le n,\ v_n \text{ is final.}$$

By item 2 and the definition of R_{mv},

$$\widehat{y\theta} = (\langle w_1, w_2'\rangle, \langle w_2, w_3'\rangle, \ldots, \langle w_{m-1}, w_m'\rangle, \_), \quad w_i \vdash_M w_{i+1}' \text{ for } 1 \le i < m.$$

By item 4 and the definition of shifted pairing, $n = m$, $v_1 = q_0\widehat{z\theta}$, and

$$\begin{array}{rcll} v_i & = & w_i & (1 \le i \le n-1), \\ v_i & = & w_i' & (2 \le i \le n). \end{array}$$

So $\widehat{x\theta}$ is a valid computation of M and thus $\widehat{z\theta} \in L(M)$.

[Proof of '$\supseteq$'] Let $w \in L(M)$. So there is a valid computation $(v_1, v_2, \ldots, v_n)$ where $v_1 = q_0 w$. Let θ be such that

$$\begin{array}{rcl} x\theta & = & [v_1 \mathbin{\bullet} \varepsilon, \ldots \quad v_{n-1} \mathbin{\bullet} \varepsilon, \quad v_n \mathbin{\bullet} \varepsilon_1], \\ y\theta & = & [\langle v_1, v_2\rangle \mathbin{\bullet} \varepsilon, \ldots \quad \langle v_{n-1}, v_n\rangle \mathbin{\bullet} \varepsilon, \quad \langle v_n, \epsilon\rangle \mathbin{\bullet} \varepsilon_1], \\ z\theta & = & w \mathbin{\bullet} \varepsilon. \end{array}$$

From items (1–3) follows immediately that θ solves the system $S_{\mathrm{id}}(x) \cup S_{\mathrm{mv}}(y) \cup S_{\mathrm{in}}(z)$. From item 4 follows that θ solves the system system $\mathrm{SP}_{q_0}(z, x, y)$. ⊠

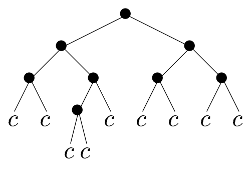
Corollary 5.2.2 *SREU is undecidable.*

Proof. Let M be a TM and w a string over its input alphabet. Let $S(x, y)$ be the system $S_M(w \mathbin{\bullet} \varepsilon, x, y)$ given by Theorem 5.2.1. The construction of $S(x, y)$ is clearly effective and $S(x, y)$ is solvable iff $w \in L(M)$. ⊠

Corollary 5.2.3 *SREU is undecidable even when restricted to ground equations on the left hand side and allowing only two variables, in any first order language with at least one binary function symbol and one constant.*

Proof. The system $S(x, y)$ in the preceding corollary contains only ground equations on the left hand side and has two variables x and y.

Furthermore, one can easily simulate any number of constants with just one constant c and $\bullet$, e.g., as follows. If at most 2^k constants for some $k \in \mathbb{N}$ are required then the i'th constant can be simulated by the term correspoding to the perfectly balanced binary tree of depth $k + 1$ and with $2^k + 1$ leaves such that the i'th vertex at level k is internal and all the others are external. For example if $k = 3$ then the third simulated constant is the term



Because of the way words and sentences are defined, all the statements in this report remain intact even if all constants are simulated. ⊠

This corollary shows that an even smaller subclass of SREU is undecidable than known before. Plaisted [33] has a proof for ground left hand sides and three variables on the right hand sides (and his proof uses several function symbols of arity 1 and 2).

5.3 Undecidability Proofs of SREU

Here we outline the main points of some of the undecidability proofs of SREU that have emerged since the problem was first [9] found to be undecidable. The different proofs reflect the undecidable nature of SREU more or less directly. The most transparent proof is probably by reduction of second order unification, which shows how close these problems really are to each other. The proof by reduction of Hilberts 10'th is less transparent and reveals that one can express certain derivations with a system of rigid equations. The least transparent proof, revealing more or less completely the undecidable nature of SREU, is ofcourse the one preseted above.

5.3.1 Reduction of Monadic Semi-unification

The first proof of the udecidability of SREU [9] was by reduction of the monadic semi-unification to SREU. This proof has its roots in [6] where it is proved that the variable-bounded semi-unification[1] can be reduced to SREU. Semi-unification was proven undecidable by Kfoury, Tiuryn and Urzyczyn [23] and the monadic semi-unification was proven undecidable by Baaz [2]. A *semi-unification problem* consists of a set of expressions $s_i \leq t_i$, $1 \leq i \leq n$, where s_i and t_i are terms. Its solution consists of a substitution σ and a set of substitutions τ_i, $1 \leq i \leq n$, such that $\tau_i \sigma s_i$ coincides with σt_i. In the *monadic* case each τ_i is either empty or involves exactly one variable.

The first step in reducing the monadic semi-unification to SREU is to give a uniform (in n) presentation of this problem by a finite set of (simpler) Φ-unification problems. A Φ-unification corresponds roughly to some particular permutation (or guess) of n variables invoved in the τ_i (there are at most $n!$ such guesses). It follows that Φ-unification is undecidable. A Φ-unification problem is then reduced to SREU. This reduction is rather technical, and it does not really reveal the nature of SREU that makes it undecidable.

5.3.2 Reduction of Second Order Unification

The second proof of the undecidability of SREU by Degtyarev and Voronkov [8, 11], and probably the most straightforward one, is by reducing second order unification to SREU. The undecidability of second order unification was proved by Goldfarb [18].

A *second order unification problem* is the problem of deciding if a finite set S of second order equations is unifiable. A *second order equation* is an expression $t = s$ where t and s are terms with possibly some (second order) variables in place of function symbols. One can without loss of generality assume that all the equations in S are such that

1. either all variables in t and s are first order, or
2. that s is $x(s_1, \ldots, s_m)$ where all variables in all the s_i and t are first order and x is a second order variable.

In the second case a second order substitution θ maps x to a term $x\theta$ where so called bound (first order) variables $\{w_1, \ldots, w_m\}$ (say $\vec{w}$) may occur, meaning that $x\theta$ corresponds to the λ-abstraction $\lambda \vec{w} x\theta$. For θ to be a unifier for $s = t$ it must be the case that $x\theta\{s_1\theta/w_1, \ldots, s_m\theta/w_m\}$ coincides with $t\theta$.

The set S is reduced to the following system of rigid equations [11, Theorem 1]. (Roughly speaking.) The first case is simply reduced to the rigid equation $\models_\forall t = s$. The second case is reduced to two rigid equations. The first one stating that x is a term possibly containing new "constants" from $\vec{w}$, and the second one stating that $\{w_1 = s_1, \ldots, w_m = s_m\} \models_\forall x = t$, where the w_i are constants.

Clearly, this is just a slight reformulation of the original problem, and one readily proves that S has a unifier if and only if this system of rigid equations is solvable [11, Lemma 5].

5.3.3 Reduction of Hilberts 10'th

In 1900 David Hilbert presented a list of 23 problems at a mathematics conference in Paris. The 10'th problem was, if there exists an algorithm that for each diophantine equation can decide whether it has an integer solution or not. A diophantine equation is an equation $p(x_1, \ldots, x_n) = 0$ where $p(\vec{x})$ is a polynomial in variables $\vec{x}$ with coefficients that are integers, e.g., $3x^3y^4 - 5xz + 3 = 0$ is a diophantine equation.

It took 70 years before Matiyasevič proved the problem to be undecidable [29]. As the third undecidability proof of SREU [10], Degtyarev and Voronkov showed how to reduce Hilberts 10'th to SREU. The proof is quite short and the key argument [10, Lemma 6] lies in representing multiplication with a system of rigid equations.

[1]It is not known if this problem is decidable.

The idea is to represent a multiplication of k and l as a list D of pairs, such that the first pair in D is (k_0, l_0) (some start values) the next $(k_0 + k, l_0 + 1)$, the one after that $(k_0 + 2k, l_0 + 2)$ and so on until the last element is $(k_0 + kl, l_0 + l)$. So if we denote such a list by $D_{k_0,l_0}(k,l)$ then

$$D_{k_0,l_0}(k,l) = \begin{cases} [(k_0, l_0)], & \text{if } l = 0; \\ [(k_0, l_0)|D_{k_0+k,l_0+1}(k, l-1)], & \text{otherwise.} \end{cases}$$

So the first element of the last pair in $D_{0,0}(k,l)$ is kl. This can be expressed by a sytem of rigid equations. (Using two lists, in the same spirit as shifted pairing.)

Similar technique is used by Voda and Komara [38] to claim (we did not check the details) the undecidability of the problem of Herbrand skeletons, i.e., given n and a formula $\psi = \exists\vec{x}\varphi(\vec{x})$ where φ is quantifier free, if the Herbrand skeleton of size n of ψ is solvable. (The Herbrand skeleton of size n of ψ is the disjunction of n variants of φ.) For $n = 1$ SREU is a special case of this problem.

5.3.4 Reduction of PCP

The Post's Correspondence Problem (PCP) over an alphabet Σ can be stated as follows. Given $(v_1, v_2, \ldots, v_k)$ and $(w_1, w_2, \ldots, w_k)$ as two sequences of strings over Σ, is there a sequence $i_1, i_2, \ldots, i_m$, $m \geq 1$, such that

$$w_{i_1} w_{i_2} \cdots w_{i_m} = v_{i_1} v_{i_2} \cdots v_{i_m}?$$

This is an undecidable problem [34]. To reduce PCP to SREU one uses the same basic technique that is used to reduce the membership problem above. This was done by Plaisted [33]. The proof gets more complex because it is not as straightforward to describe the legal transposition relation as it is to describe the moves of a TM.

Chapter 6

Uniform Representation of RE in Intuitionistic Logic with Equality

In this chapter we show that any r. e. set can be represented by a f.o. formula $\psi(z)$ of the form $\exists x \exists y \varphi(z,x,y)$, where φ is quantifier free and its only connectives are $\wedge$ and $\Rightarrow$, in a language (with equality) with just one binary function symbol and a countable set of constants. Furthermore, the construction of ψ is performed uniformly in the index of the r. e. set in question.

We obtain that the $\exists\exists$-fragment of intuitionistic logic is undecidable. This is an improvement of the undecidability result of the $\exists^*$-fragment in general shown recently by Degtyarev and Voronkov [10, Theorem 10] (or [11, Theorem 3]).

A closely related problem is the skeleton instantiation problem, i.e., the problem of existence of a derivation with a given skeleton. Voronkov shows that SREU is polynomially reducible to this problem [39, Theorem 3.12] (where the actual proof system under consideration is a sequent calculus $LJ^=$ for intuitionistic logic with equality). Moreover, the basic structure of the skeleton is determined by the number of variables in the SREU problem and the number of rigid equations in it. Our result implies that this problem is undecidable already for a fairly restricted class of skeletons.

6.1 The Main Theorem

Recall the following. L has one binary function symbol $\centerdot$ and a countable set of constants. The input alphabet of a Turing machine is assumed to be a subset of the constants in L. If t is the term $\centerdot(c_1, \centerdot(c_2, \ldots, \centerdot(c_n, \varepsilon)\ldots))$, where all the c_i and ε are constants, we call it an ε-word and we write $c_1c_2\ldots c_n \centerdot \varepsilon$ for t, and $\widehat{t}$ stands for the string $c_1c_2\ldots c_n$. Let $\vdash_{\mathrm{i}}$ stand for provability in intuitionistic predicate calculus with equality and let $\vdash_{\mathrm{c}}$ stand for provability in classical predicate calculus with equality.

Theorem 6.1.1 *Let M be a Turing machine. There is a formula $\varphi(z,x,y)$ and a constant ε in L such that the following statements are equivalent, for all ground terms t in L:*

1. *t is an ε-word in $L(M)$,*
2. *$\vdash_{\mathrm{i}} \exists x \exists y \varphi(t,x,y)$.*

Proof. Let $S_M(z,x,y)$ be the system of rigid equations given by Theorem 5.2.1. So

$$S_M(z,x,y) \quad = \quad \{\, E_j \vDash_{\forall} s_j = t_j \mid 1 \le j \le 7 \,\},$$

where each E_j is a set of (ground) equations, let

$$\varphi(z,x,y) \quad = \quad \bigwedge_{1\leq j\leq 7}((\bigwedge_{e\in E_j} e) \Rightarrow s_j = t_j).$$

[**Proof of** '$1 \Rightarrow 2$'] Assume $w \in L(M)$ and $t = w \,\centerdot\, \varepsilon$. By Theorem 5.2.1 there is a θ such that $z\theta = t$ that solves $S_M(z,x,y)$. By definition, this means that $\vdash_{\mathrm{c}} \varphi(t, x\theta, y\theta)$. But

$$\vdash_{\mathrm{c}} \varphi(t, x\theta, y\theta) \quad \Rightarrow \quad \vdash_{\mathrm{i}} \varphi(t, x\theta, y\theta)$$

for this particular class of formulas. Statement 2 follows now by $\exists$-introduction.

[**Proof of** '$2 \Rightarrow 1$'] By explicit definabilty of intuitionistic logic there are ground terms t_x and t_y such that $\vdash_{\mathrm{i}} \varphi(t, t_x, t_y)$. It follows that θ, such that $z\theta = t$, $x\theta = t_x$ and $y\theta = t_y$, solves the system $S_M(z,x,y)$. Thus t is an ε-word in $L(M)$ by Theorem 5.2.1. ⊠

Some comments The exact number of implications in the formula φ is not the least possible. The author believes that it is possible to give an equivalent formula with just 4 implications. This requires a slightly different formulation of Theorem 4.2.1. Technically the presentation becomes more cumbersome, although nothing substantial changes.

6.2 Undecidability of the $\exists\exists$-fragment of Intuitionistic Logic

This is an improvement of the undecidability result of the $\exists^*$-fragment in general shown recently by Degtyarev and Voronkov [10, Theorem 10] (or [11, Theorem 3]):

Corollary 6.2.1 *The class of formulas in intuitionistic logic with equality, of the form $\exists x \exists y \varphi$ where φ is quantifier free and the only connectives in φ are $\wedge$ and $\Rightarrow$, is undecidable.*

Proof. Note that the construction of the formula $\varphi(t,x,y)$ in Theorem 6.1.1 is effective. ⊠

Note that it is enough that the number of implications in φ is 6. This is because in the corresponding system of rigid equations, the rigid equation $S_{\mathrm{in}}(t)$ is ground and can be decided for example by using the Shostak congruence closure algorithm [36, 5].

Decidabilty problems for some other fragments of intuitionistic logic with and without equality were studied by Orevkov [31, 32], Mints [30] and Lifschitz [26]. More recently some new results have been obtained by Degtyarev and Voronkov [40, 39, 12, 7], and Tammet [37].

Another interesting question is the relationship between classical ($\vdash_{\mathrm{c}}$) and intuitionistic provability of the formulas φ_M in Theorem 6.1.1. For example, if it was the case that $\vdash_{\mathrm{c}} \exists x \exists y \varphi_M(t,x,y)$ always implies that $\widehat{t} \in L(M)$ then the problem of Herbrand skeletons would be undecidable already for a very restricted class of formulas. Voda and Komara have recently claimed that this problem is undecidable [38].

One should note that the classical and the intuitionistic provability of the "SREU formulas" is not the same in general even if all the left hand sides are ground. A simple counterexample is $\exists z((c = 0 \Rightarrow z = 1) \wedge (c = 1 \Rightarrow z = 0))$. This formula is obviously valid classically, whereas it is not true in a Kripke model consisting of three nodes formed as a V, where, except for trivial identities, only $c = 0$ is true in the left branch and only $c = 1$ is true in the right branch.

Chapter 7

Current Status and Open Problems

Decidability of rigid E-unification has been known for some time now [16], for a clear proof see for example De Kogel [5]. The current status about what is known about SREU and rigid E-unification is summarized below.

1. Rigid E-unification with ground lefthand side is NP-complete [25]. Rigid E-unification in general is NP-complete and there exist finite complete sets of unfiers [16, 15].
2. If all function symbols have arity ≤ 1 then SREU is PSPACE-hard [19]. If only one unary function symbol is allowed then the problem is decidable [7, 6]. If only constants are allowed then the problem is NP-complete [7] if there are at least two constants.
3. If there are more than one unary function symbol then the decidability is still an open question, it is known however that the word equation solving [28] (unification under associativity), which is an extremely hard problem (no interesting upper bounds for the complexity of this problem are yet known), can be reduced to SREU [6].
4. In general SREU is undecidable [9], already with ground left hand sides [33] and two variables (this report).

Some other decidable cases of SREU are also described by Plaisted [33]. It should also be noted that the decidability of SREU with just one variable is an open question and thus also the decidability of the $\exists$-fragment of intuitionistic logic with equality. Note that SREU is decidable when there are no variables, then each rigid equation can be decided for example by using the Shostak congruence closure algorithm [36, 5].

Acknowledgements

I want to thank Andrei Voronkov and Erik Palmgren for valuable comments on earlier versions of this report.

Bibliography

[1] P.B. Andrews. Theorem proving via general matings. *Journal of the Association for Computing Machinery*, 28(2):193–214, 1981.

[2] M. Baaz. Note on the existence of most general semi-unifiers. In *Arithmetic, Proof Theory and Computation Complexity*, volume 23 of *Oxford Logic Guides*, pages 20–29. Oxford University Press, 1993.

[3] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.

[4] W. Bibel. *Deduction. Automated Logic.* Academic Press, 1993.

[5] E. De Kogel. Rigid E-unification simplified. In P. Baumgartner, R. Hähnle, and J. Posegga, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, number 918 in Lecture Notes in Artificial Intelligence, pages 17–30, Schloß Rheinfels, St. Goar, Germany, May 1995.

[6] A. Degtyarev, Yu. Matiyasevich, and A. Voronkov. Simultaneous rigid E-unification is not so simple. UPMAIL Technical Report 104, Uppsala University, Computing Science Department, April 1995.

[7] A. Degtyarev, Yu. Matiyasevich, and A. Voronkov. Simultaneous rigid E-unification and related algorithmic problems. In *LICS'96*, pages 1–11, 1996.

[8] A. Degtyarev and A. Voronkov. Reduction of second-order unification to simultaneous rigid E-unification. UPMAIL Technical Report 109, Uppsala University, Computing Science Department, June 1995.

[9] A. Degtyarev and A. Voronkov. Simultaneous rigid E-unification is undecidable. UPMAIL Technical Report 105, Uppsala University, Computing Science Department, May 1995.

[10] A. Degtyarev and A. Voronkov. Simultaneous rigid E-unification is undecidable. In *Computer Science Logic Workshop*, pages 1–12, 1995.

[11] A. Degtyarev and A. Voronkov. The undecidability of simultaneous rigid e-unification (note). *Theoretical Computer Science*, pages 1–10, 1995.

[12] A. Degtyarev and A. Voronkov. Skolemization and decidability problems for fragments of intuitionistic logic. In *LICS'96*, pages 1–10, 1996.

[13] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. Van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Methods and Semantics, chapter 6, pages 243–309. North Holland, Amsterdam, 1990.

[14] M. Fitting. First-order modal tableaux. *Journal of Automated Reasoning*, 4:191–213, 1988.

[15] J. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid E-unification: NP-completeness and applications to equational matings. *Information and Computation*, 87(1/2):129–195, 1990.

[16] J.H. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid E-unification is NP-complete. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 338–346. IEEE Computer Society Press, July 1988.

[17] J.H. Gallier, S. Raatz, and W. Snyder. Theorem proving using rigid E-unification: Equational matings. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 338–346. IEEE Computer Society Press, 1987.

[18] Warren D. Goldfarb. The undecidability of the second-order unification problem. *Theoretical Computer Science*, 13:225–230, 1981.

[19] J. Goubault. Rigid $\vec{E}$-unifiability is DEXPTIME-complete. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*. IEEE Computer Society Press, 1994.

[20] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley Publishing Co., 1979.

[21] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the Association for Computing Machinery*, 27(4):797–821, October 1980.

[22] S. Kanger. A simplified proof method for elementary logic. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning. Classical Papers on Computational Logic*, volume 1, pages 364–371. Springer Verlag, 1983. Originally appeared in 1963.

[23] A.J. Kfoury, J. Tiuryn, and P. Urzyczyn. The undecidability of the semi-unification problem. *Information and Computation*, 102:83–101, 1993.

[24] D. Knuth and P. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, Oxford, 1970.

[25] D. Kozen. Positive first-order logic is NP-complete. *IBM J. of Research and Development*, 25(4):327–332, 1981.

[26] V. Lifschitz. The decidability problem for some constructive theories of equality (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967.

[27] D.W. Loveland. Mechanical theorem proving by model elimination. *Journal of the Association for Computing Machinery*, 15:236–251, 1968.

[28] G.S. Makanin. The problem of solvability of equations in free semigroups. *Mat. Sbornik (in Russian)*, 103(2):147–236, 1977. English Translation in American Mathematical Soc. Translations (2), vol. 117, 1981.

[29] Yu.V. Matiyasevič. The diophantiness of recursively enumerable sets (in Russian). *Soviet Mathematical Doklady*, pages 279–282, 1970.

[30] G.E. Mints. Collecting terms in the quantifier rules of the constructive predicate calculus (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967.

[31] V.P. Orevkov. The undecidability in the constructive predicate calculus of the class of formulas of the form $\neg\neg\forall\exists$ (in Russian). *Soviet Mathematical Doklady*, 163(3):581–583, 1965. The journal is cover-to-cover translated to English.

[32] V.P. Orevkov. Solvable classes of pseudo-prenex formulas (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 60:109–170, 1976. English translation in: Journal of Soviet Mathematics.

[33] D.A. Plaisted. Special cases and substitutes for rigid E-unification. Technical Report MPI-I-95-2-010, Max-Planck-Institut für Informatik, November 1995.

[34] E.L. Post. A variant of a recursively unsolvable problem. *Bull. Am. Math. Soc.*, 52(4):264–268, 1946.

[35] D. Prawitz. An improved proof procedure. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning. Classical Papers on Computational Logic*, volume 1, pages 162–201. Springer Verlag, 1983. Originally appeared in 1960.

[36] R. Shostak. An algorithm for reasoning about equality. *Communications of the ACM*, 21:583–585, July 1978.

[37] T. Tammet. A resolution theorem prover for intuitionistic logic. Unpublished manuscript, 1996.

[38] P.J. Voda and J. Komara. On Herbrand skeletons. Technical report, Institute of Informatics, Comenius University Bratislava, July 1995.

[39] A. Voronkov. On proof-search in intuitionistic logic with equality, or back to simultaneous rigid E-Unification. UPMAIL Technical Report 121, Uppsala University, Computing Science Department, January 1996.

[40] A. Voronkov. Proof-search in intuitionistic logic based on the constraint satisfaction. UPMAIL Technical Report 120, Uppsala University, Computing Science Department, January 1996.