Available online at www.sciencedirect.com
ScienceDirect
Journal of Symbolic Computation 42 (2007) 352–388
Journal of Symbolic Computation
www.elsevier.com/locate/jsc

Solving equations with sequence variables and sequence functions[☆]

Temur Kutsia[*]

Research Institute for Symbolic Computation, Johannes Kepler University, A-4040 Linz, Austria

Received 15 February 2005; accepted 28 December 2006
Available online 7 January 2007

Abstract

Term equations involving individual and sequence variables and sequence function symbols are studied. Function symbols can have either fixed or flexible arity. A sequence variable can be instantiated by any finite sequence of terms. A sequence function abbreviates a finite sequence of functions all having the same argument lists. It is proved that solvability of systems of equations of this form is decidable. A new unification procedure that enumerates a complete almost minimal set of solutions is presented, together with variations for special cases. The procedure terminates if the solution set is finite. Applications in various areas of artificial intelligence, symbolic computation, and programming are discussed.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Unification; Sequence variables; Sequence functions

1. Introduction

We study term equations with individual and sequence variables and function symbols. A sequence variable can be instantiated by any finite sequence of terms, including the empty sequence. A sequence function abbreviates a finite sequence of functions all having the same argument lists. Semantically, sequence functions can be interpreted as multi-valued functions. Individual variables and function symbols are just the ordinary ones.

Sequence variables add expressiveness and elegance to the language. For instance, the property of a function being "orderless" can be easily defined using sequence variables:

☆ Supported by the Austrian Science Foundation (FWF) under Project SFB F1302.

* Tel.: +43 732 2468 9982; fax: +43 732 2468 9930.
E-mail address: Temur.Kutsia@risc.uni-linz.ac.at.

0747-7171/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jsc.2006.12.002

$f(\overline{x}, x, \overline{y}, y, \overline{z}) = f(\overline{x}, y, \overline{y}, x, \overline{z})$ specifies that the order of arguments in terms with the head f and with any number of arguments does not matter. Here x and y are individual variables and the letters with the overbars are sequence variables. Without them one would need the permutation function to express the same property. Note that the function symbol f has a flexibly arity. Sequence variables are normally used with flexible arity function or predicate symbols.

List concatenation is another example. Here sequence variables help to avoid recursive definition: $\langle\overline{x}\rangle \asymp \langle\overline{y}\rangle = \langle\overline{x}, \overline{y}\rangle$. Furthermore, some proofs become simpler, e.g., associativity of concatenation can be proved without induction.

Sequence variables provide a natural way to formalize and implement sequent calculi. For instance, in the rule

$$\frac{\Gamma, A, B, \Delta \to \Lambda}{\Gamma, A \wedge B, \Delta \to \Lambda}$$

Γ, Δ, and Λ can be implemented as sequence variables and A and B as individual variables.

Sequence variables in programming help to write an elegant, short code. The following rule-based implementation of bubble sort is a good example:

$$\mathtt{sort}(\langle\overline{\mathtt{x}}, \mathtt{x}, \overline{\mathtt{y}}, \mathtt{y}, \overline{\mathtt{z}}\rangle) := \mathtt{sort}(\langle\overline{\mathtt{x}}, \mathtt{y}, \overline{\mathtt{y}}, \mathtt{x}, \overline{\mathtt{z}}\rangle) \ \mathtt{if}\ \mathtt{x} > \mathtt{y}$$
$$\mathtt{sort}(\langle\overline{\mathtt{x}}\rangle) := \langle\overline{\mathtt{x}}\rangle.$$

Sequence variables can be used to query semistructured data. In particular, they can be useful in XML querying and processing.

Bringing sequence functions into the language allows Skolemization over sequence variables: Let x, y be individual variables, $\overline{x}$ be a sequence variable, and p be a flexible arity predicate symbol. Then $\forall x \forall y \exists \overline{x}.p(x, y, \overline{x})$ Skolemizes to $\forall x \forall y.p(x, y, \overline{f}(x, y))$, where $\overline{f}$ is a binary Skolem sequence function symbol. Another example, $\forall \overline{y} \exists \overline{x}.p(\overline{y}, \overline{x})$, where $\overline{y}$ is a sequence variable, after Skolemization introduces a flexible arity sequence function symbol $\overline{g}$: $\forall \overline{y}.p(\overline{y}, \overline{g}(\overline{y}))$. The integer division function $div(x, y)$ is an instance of a sequence function. It abbreviates the sequence of quotient and remainder functions: $q(x, y), r(x, y)$.

Note that sequence functions are interpreted as multi-valued functions where number of values is *not* fixed. Modeling functions with a fixed number n of values is trivial: we could just replace a "macro" $\overline{f}(t_1, \ldots, t_m)$ by an n-ary sequence $f_1(t_1, \ldots, t_m), \ldots, f_n(t_1, \ldots, t_m)$. Similarly, the length of possible values for a sequence variable is not fixed. Otherwise we could simply replace the sequence variable by a sequence of individual variables of the corresponding length.

Equation solving with sequence variables has applications in various areas of artificial intelligence, symbolic computation, and programming. At the end of the paper we briefly review some of the related work.

We contribute to this area by introducing a new unification procedure for solving equations in the free theory with individual and sequence variables and function symbols. Function symbols have either fixed or flexible arity. The procedure enumerates an almost minimal complete set of solutions and terminates if the set is finite. We prove that solvability of systems of equations of this form is decidable. Omitting the decision algorithm and adding extra rules for failure, we obtain a "lighter" version of the unification procedure. It is still sound and complete, easier to implement, but for some failing cases might not terminate. We implemented the "light" procedure in *Mathematica* (Wolfram, 2003).

It should be noted that some of the techniques we use are similar to those known from general associative unification (Plotkin, 1972) and word equations (e.g., Schulz (1993)). We discuss the relation to these problems in the section about the related work.

Equation solving in the free theory with individual and sequence variables and function symbols can be considered as a special case of order-sorted higher-order E-unification. However, it does not make the problem easier, because, to the best of our knowledge, order-sorted higher-order E-unification is a problem that still waits for its solution.

The paper is organized as follows. In Section 2 basic notions are introduced. In Section 3 decidability of unification with individual and sequence variables and function symbols is proved. In Section 4 the unification procedure is introduced and its soundness, completeness, and almost minimality are proved. The "light" procedure is introduced in Section 5 and its termination issues are addressed in Section 6. The implementation is briefly described in Section 7. A relation with order-sorted higher-order E-unification is discussed in Section 8. Some of the related work is reviewed in Section 9.

This work is an extension and a refinement of our previous results on unification with sequence variables (Kutsia, 2002a,b, 2004).

2. Preliminaries

We assume that the reader is familiar with the standard notions of unification theory (Baader and Snyder, 2001).

2.1. Syntax and substitutions

We assume fixed pairwise disjoint sets of symbols: individual variables $\mathcal{V}_I$, sequence variables $\mathcal{V}_S$, fixed arity individual function symbols $\mathcal{F}ix_I$, flexible arity individual function symbols $\mathcal{F}lex_I$, fixed arity sequence function symbols $\mathcal{F}ix_S$, flexible arity sequence function symbols $\mathcal{F}lex_S$. Each set of variables and sequence function symbols is countable. Each set of individual function symbols is finite or countable. Additionally, we define:

$$\begin{array}{lll} \mathcal{F}_I := \mathcal{F}ix_I \cup \mathcal{F}lex_I, & \mathcal{F}ix := \mathcal{F}ix_I \cup \mathcal{F}ix_S, & \mathcal{F} := \mathcal{F}_I \cup \mathcal{F}_S, \\ \mathcal{F}_S := \mathcal{F}ix_I \cup \mathcal{F}lex_S, & \mathcal{F}lex := \mathcal{F}lex_I \cup \mathcal{F}lex_S, & \mathcal{V} := \mathcal{V}_I \cup \mathcal{V}_S. \end{array}$$

The *arity* of $f \in \mathcal{F}ix$ is denoted by $\mathcal{A}r(f)$. A function symbol $c \in \mathcal{F}ix$ is called a *constant* if $\mathcal{A}r(c) = 0$.

If not otherwise stated, we use x, y, z for individual variables, $\overline{x}, \overline{y}, \overline{z}$ for sequence variables, f, g, h for individual function symbols, $\overline{f}, \overline{g}, \overline{h}$ for sequence function symbols, a, b, c for individual constants, and $\overline{a}, \overline{b}, \overline{c}$ for sequence constants. Moreover, v will be used for (individual or sequence) variables, and l (in some cases) for variables or sequence function symbols. The meta-variables may come with indices.

Terms over $\mathcal{F}$ and $\mathcal{V}$ are constructed using the following grammar:

$$t ::= it \mid st$$

where *it* is an *individual term* and *st* is a *sequence term*. They are constructed as follows:

$$it ::= x \mid f(it_1, \ldots, it_n) \mid g(t_1, \ldots, t_m)$$
$$st ::= \overline{x} \mid \overline{f}(it_1, \ldots, it_n) \mid \overline{g}(t_1, \ldots, t_m)$$

where $f, \overline{f} \in \mathcal{F}ix$ with $\mathcal{A}r(f) = \mathcal{A}r(\overline{f}) = n$, and $g, \overline{g} \in \mathcal{F}lex$.

We denote by $\mathcal{T}(\mathcal{F}, \mathcal{V})$, $\mathcal{T}_I(\mathcal{F}, \mathcal{V})$, and $\mathcal{T}_S(\mathcal{F}, \mathcal{V})$, respectively, the sets of all terms, all individual terms, and all sequence terms over $\mathcal{F}$ and $\mathcal{V}$.

Equations are defined as pairs of individual terms $\langle it_1, it_2\rangle$. We use more conventional notation for equations, writing $it_1 \approx it_2$ for $\langle it_1, it_2\rangle$. We do not define equations between sequence terms, because they can be encoded as equations between individual terms using flexible arity individual function symbols.

The *head* of an individual term $t = f(t_1, \dots, t_n)$ (resp. of a sequence term $t = \overline{f}(t_1, \dots, t_n)$), $n \geq 0$, denoted by $\mathcal{H}ead(t)$, is the individual function symbol f (resp. the sequence function symbol $\overline{f}$). For T being either a term, a sequence of terms, or a set of terms, we denote

- the set of all individual variables occurring in T by $\mathcal{V}_I(T)$;
- the set of all sequence variables occurring in T by $\mathcal{V}_S(T)$;
- the set $\mathcal{V}_I(T) \cup \mathcal{V}_S(T)$ by $\mathcal{V}(T)$;
- the set of all individual function symbols occurring in T by $\mathcal{F}_I(T)$;
- the set of all sequence function symbols occurring in T by $\mathcal{F}_S(T)$;
- the set of all fixed arity function symbols occurring in T by $\mathcal{F}ix(T)$;
- the set of all flexible arity function symbols occurring in T by $\mathcal{F}lex(T)$.

A term t is called *ground* if $\mathcal{V}(t) = \emptyset$. We use the letters s, t, r, and q, maybe with indices, for terms.

Below we do not distinguish between a singleton sequence and its sole element.

A *substitution* is a mapping from individual variables to individual terms, from sequence variables to finite, possibly empty sequences of terms, and from sequence function symbols to finite nonempty sequences of sequence function symbols, such that all but finitely many individual variables, sequence variables, and sequence function symbols are mapped to themselves, and sequence function symbol mapping preserves arity. The last condition means the following: If $\sigma(\overline{f}) = \ulcorner \overline{g_1}, \dots, \overline{g_n} \urcorner$[1] for a substitution σ, then $\mathcal{A}r(\overline{g_1}) = \cdots = \mathcal{A}r(\overline{g_n}) = \mathcal{A}r(\overline{f})$ if $\overline{f} \in \mathcal{F}ix$, and $\overline{g_1}, \dots, \overline{g_n} \in \mathcal{F}lex$ if $\overline{f} \in \mathcal{F}lex$.

We will use the traditional notation for substitutions representing them as finite sets of bindings

$$\{x_1 \mapsto it_1, \dots, x_n \mapsto it_n, \overline{x}_1 \mapsto \ulcorner s^1_1, \dots, s^1_{k_1} \urcorner, \dots, \overline{x}_m \mapsto \ulcorner s^m_1, \dots, s^m_{k_m} \urcorner,$$
$$\overline{f_1} \mapsto \ulcorner \overline{g^1_1}, \dots, \overline{g^1_{l_1}} \urcorner, \dots, \overline{f_r} \mapsto \ulcorner \overline{g^r_1}, \dots, \overline{g^r_{l_r}} \urcorner\}.$$

Lower case Greek letters are used to denote substitutions. The empty substitution is denoted by ε.

Substitutions are extended to terms as follows:

$$\begin{aligned} x\sigma &= \sigma(x), & f(t_1, \dots, t_n)\sigma &= f(t_1\sigma, \dots, t_n\sigma), \\ \overline{x}\sigma &= \sigma(\overline{x}), & \overline{f}(t_1, \dots, t_n)\sigma &= \sigma(\overline{f})[t_1\sigma, \dots, t_n\sigma], \end{aligned}$$

where the notation $\ulcorner \overline{g_1}, \dots, \overline{g_m} \urcorner [t_1, \dots, t_n]$ is just a shortcut for the sequence $\ulcorner \overline{g_1}(t_1\sigma, \dots, t_n\sigma), \dots, \overline{g_m}(t_1\sigma, \dots, t_n\sigma) \urcorner$. For a term t and a substitution σ, we call $t\sigma$ an *instance* of t with respect to σ. Substitutions are extended to sequences of terms, sequences of sequence function symbols, and equations in the standard way.

Example 1. Let $\sigma = \{x \mapsto a,\ y \mapsto f(\overline{x}),\ \overline{x} \mapsto \ulcorner\urcorner,\ \overline{y} \mapsto \ulcorner a, f(\overline{x}), \overline{b} \urcorner,\ \overline{g} \mapsto \ulcorner \overline{g_1}, \overline{g_2} \urcorner\}$. Then $f(x, \overline{x}, \overline{g}(y), \overline{y})\sigma = f(a, \overline{g_1}(f(\overline{x})), \overline{g_2}(f(\overline{x})), a, f(\overline{x}), \overline{b})$.

[1] For better readability we write sequences between the symbols $\ulcorner$ and $\urcorner$.

A nonstandard feature with term instances is that a ground term can be further instantiated. For example, $f(\overline{g}(a))\{\overline{g} \mapsto \ulcorner \overline{g_1}, \overline{g_2} \urcorner\} = f(\overline{g_1}(a), \overline{g_2}(a))$. However, such an instantiation only "splits" sequence terms and the instance obtained remains ground.

The *domain* of a substitution σ is the set of variables and sequence function symbols $\mathcal{D}om(\sigma) := \{l \mid l\sigma \neq l\}$. The *codomain* of σ, denoted as $\mathcal{C}od(\sigma)$, is the set of terms and sequence function symbols defined as follows:

$$\begin{aligned}\mathcal{C}od(\sigma) = \{t \mid {} & \text{there exists } x \in \mathcal{D}om(\sigma) \text{ such that } t = x\sigma, \text{ or} \\ & \text{there exist } \overline{x} \in \mathcal{D}om(\sigma) \text{ and terms } t_1, \ldots, t_n,\ n \geq 0, \\ & \text{such that } \ulcorner t_1, \ldots, t, \ldots, t_n \urcorner = \overline{x}\sigma\} \cup \\ \{\overline{f} \mid {} & \text{there exist } \overline{g} \in \mathcal{D}om(\sigma) \text{ and sequence function symbols} \\ & \overline{f}_1, \ldots, \overline{f}_n, n \geq 0, \text{ such that } \ulcorner \overline{f}_1, \ldots, \overline{f}, \ldots, \overline{f}_n \urcorner = \overline{g}\sigma\}.\end{aligned}$$

For instance, $\mathcal{C}od(\{\overline{x} \mapsto \ulcorner\urcorner\}) = \emptyset$ and $\mathcal{C}od(\{x \mapsto f(a), \overline{x} \mapsto \ulcorner a, a, \overline{b} \urcorner, \overline{a} \mapsto \ulcorner \overline{b}, \overline{c} \urcorner\}) = \{f(a), a, \overline{b}(), \overline{b}, \overline{c}\}$. Note that in codomains, omitting parentheses in sequence terms with the empty list of arguments might lead to confusion: One cannot distinguish such a term from its head (a sequence function symbol). To avoid this, in codomains we write such terms with parentheses. This is why we have both $\overline{b}()$ and $\overline{b}$ in the codomain in the second example above: $\overline{b}()$ comes from the binding for $\overline{x}$ and $\overline{b}$ comes from the binding for $\overline{a}$.

The *range* of σ is the set of variables $\mathcal{R}an(\sigma) := \mathcal{V}(\mathcal{C}od(\sigma))$. A substitution σ is called *ground* if $\mathcal{R}an(\sigma) = \emptyset$.

The *restriction* of a substitution σ to a set of variables and sequence function symbols S, denoted as $\sigma|_S$, is the substitution defined by $l\sigma|_S = l\sigma$ if $l \in S$, and $l\sigma|_S = l$ otherwise. Besides, we define $\mathcal{VD}om(\sigma) := \mathcal{D}om(\sigma) \cap \mathcal{V}$ and $\mathcal{FD}om(\sigma) := \mathcal{D}om(\sigma) \cap \mathcal{F}$. We write the *composition* of two substitutions σ and ϑ as $\sigma\vartheta$. The following example illustrates composition of substitutions:

$$\begin{aligned}\sigma &= \{x \mapsto y,\ \overline{x} \mapsto \ulcorner \overline{y}, \overline{x} \urcorner, \overline{y} \mapsto \ulcorner f(a,b), y, \overline{g}(\overline{x}) \urcorner, \overline{f} \mapsto \ulcorner \overline{g}, \overline{h} \urcorner\}. \\ \vartheta &= \{y \mapsto x,\ \overline{y} \mapsto \overline{x},\ \overline{x} \mapsto \ulcorner\urcorner, \overline{g} \mapsto \ulcorner \overline{g_1}, \overline{g_2} \urcorner\}. \\ \sigma\vartheta &= \{y \mapsto x,\ \overline{y} \mapsto \ulcorner f(a,b), x, \overline{g_1}(), \overline{g_2}() \urcorner, \overline{f} \mapsto \ulcorner \overline{g_1}, \overline{g_2}, \overline{h} \urcorner, \overline{g} \mapsto \ulcorner \overline{g_1}, \overline{g_2} \urcorner\}.\end{aligned}$$

2.2. Technical notions

Given a set E of equations over $\mathcal{F}$ and $\mathcal{V}$ we denote by $\approx_E$ the least congruence relation on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ that is closed under substitution application and contains E. To be more precise, $\approx_E$ contains E, satisfies reflexivity, symmetry, transitivity, congruence, and a special form of substitutivity: For all $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, if $s \approx_E t$ and $s\sigma, t\sigma \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ for some σ, then $s\sigma \approx_E t\sigma$. Substitutivity in this form only affects situations where $s\sigma$ and $t\sigma$ are terms. The set $\approx_E$ is called an *equational theory* defined by E. We will also call the set E an equational theory or an E-theory. The *signature* of E, denoted as $\mathcal{S}ig(E)$, is the set of all individual function symbols occurring in E.

In the rest of the paper, if not otherwise stated, E stands for an equational theory, $\mathcal{X}$ for a finite set of variables, and $\mathcal{Q}$ for a finite set of sequence function symbols.

Definition 2. A substitution σ is called *erasing* on $\mathcal{X}$ modulo E if either there exist $f \in \mathcal{S}ig(E)$ and $v \in \mathcal{X}$ such that $f(v)\sigma \approx_E f()$ or there exists $\overline{x} \in \mathcal{X}$ such that $\overline{x}\sigma = \ulcorner\urcorner$. We say that σ is *non-erasing* on $\mathcal{X}$ modulo E if σ is not erasing on $\mathcal{X}$ modulo E.

The notion of erasing substitution will be needed in defining the notions of minimal and almost minimal sets of substitutions later.

Example 3. (1) Let $E = \emptyset$ and $\mathcal{X} = \{x, \overline{x}\}$. Then any substitution that maps $\overline{x}$ to the empty sequence is erasing on $\mathcal{X}$ modulo E.

(2) Let $E = \{f(\overline{x}, f(\overline{y}), \overline{z}) \approx f(\overline{x}, \overline{y}, \overline{z})\}$ and $\mathcal{X} = \{x, \overline{x}\}$. Then any substitution that maps x to $f()$, or maps $\overline{x}$ to a (possibly empty) sequence of $f()$'s is erasing on $\mathcal{X}$ modulo E.

Definition 4. A substitution σ *agrees* with a substitution ϑ on $\mathcal{X}$ and $\mathcal{Q}$ modulo E, denoted as $\sigma =_E^{\mathcal{X},\mathcal{Q}} \vartheta$, if

(1) for all $\overline{x} \in \mathcal{X}$, there exist $t_1, \ldots, t_n, s_1, \ldots, s_n \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, $n \geq 0$, such that $\overline{x}\sigma = \ulcorner t_1, \ldots, t_n \urcorner$, $\overline{x}\vartheta = \ulcorner s_1, \ldots, s_n \urcorner$, and $t_i \approx_E s_i$ for each $1 \leq i \leq n$;
(2) for all $x \in \mathcal{X}$, $x\sigma \approx_E x\vartheta$;
(3) for all $\overline{f} \in \mathcal{Q}$, $\overline{f}\sigma = \overline{f}\vartheta$.

Definition 5. A substitution φ *matches* a substitution σ with a substitution ϑ on $\mathcal{X}$ and $\mathcal{Q}$ modulo E, if $\sigma\varphi =_E^{\mathcal{X},\mathcal{Q}} \vartheta$.

Example 6. Let $\sigma = \{\overline{x} \mapsto \overline{a}\}$, $\vartheta = \{\overline{x} \mapsto \ulcorner \overline{b}, \overline{c} \urcorner, \overline{a} \mapsto \ulcorner \overline{b}, \overline{c} \urcorner\}$ and $\varphi = \{\overline{x} \mapsto \ulcorner \overline{b}, \overline{c} \urcorner, \overline{a} \mapsto \ulcorner \overline{b}, \overline{c} \urcorner\}$. Let also $\mathcal{X} = \{\overline{x}\}$, $\mathcal{Q} = \{\overline{a}\}$, and $E = \emptyset$. Then $\sigma\varphi =_E^{\mathcal{X},\mathcal{Q}} \vartheta$.

Definition 7. A substitution σ is *more general* (resp. *strongly more general*) than a substitution ϑ on sets $\mathcal{X}$ and $\mathcal{Q}$ modulo E, denoted as $\sigma \leq_E^{\mathcal{X},\mathcal{Q}} \vartheta$ (resp. $\sigma \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \vartheta$), if there exists a substitution (resp. a substitution non-erasing on $\mathcal{X}$ modulo E) φ such that $\sigma\varphi =_E^{\mathcal{X},\mathcal{Q}} \vartheta$.

Example 8. Let $\sigma = \{\overline{x} \mapsto \overline{y}\}$, $\vartheta = \{\overline{x} \mapsto \ulcorner a, b \urcorner, \overline{y} \mapsto \ulcorner a, b \urcorner\}$, $\eta = \{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner\urcorner\}$.

(1) If $\mathcal{X} = \{\overline{x}, \overline{y}\}$, then $\sigma \leq_\emptyset^{\mathcal{X},\emptyset} \vartheta$, $\sigma \trianglelefteq_\emptyset^{\mathcal{X},\emptyset} \vartheta$, $\sigma \leq_\emptyset^{\mathcal{X},\emptyset} \eta$, $\sigma \ntrianglelefteq_\emptyset^{\mathcal{X},\emptyset} \eta$.
(2) If $\mathcal{X} = \{\overline{x}\}$, then $\sigma \leq_\emptyset^{\mathcal{X},\emptyset} \vartheta$, $\sigma \trianglelefteq_\emptyset^{\mathcal{X},\emptyset} \vartheta$, $\sigma \leq_\emptyset^{\mathcal{X},\emptyset} \eta$, $\sigma \trianglelefteq_\emptyset^{\mathcal{X},\emptyset} \eta$.

From Definition 7 it follows that $\trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \subseteq \leq_E^{\mathcal{X},\mathcal{Q}}$. A substitution ϑ is an *E-instance* (resp. *strong E-instance*) of a substitution σ on $\mathcal{X}$ and $\mathcal{Q}$ if $\sigma \leq_E^{\mathcal{X},\mathcal{Q}} \vartheta$ (resp. $\sigma \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \vartheta$). The equivalence associated with $\leq_E^{\mathcal{X},\mathcal{Q}}$ (resp. with $\trianglelefteq_E^{\mathcal{X},\mathcal{Q}}$) is denoted by $\doteq_E^{\mathcal{X},\mathcal{Q}}$ (resp. by $\triangleq_E^{\mathcal{X},\mathcal{Q}}$). If $\sigma\varphi =_E^{\mathcal{X},\mathcal{Q}} \vartheta$, then $\sigma\varphi \doteq_E^{\mathcal{X},\mathcal{Q}} \vartheta$. If, in addition, φ is non-erasing on $\mathcal{X}$ and $\mathcal{Q}$ modulo E, then $\sigma\varphi \triangleq_E^{\mathcal{X},\mathcal{Q}} \vartheta$.

Definition 9. A set of substitutions S is called *minimal* (resp. *almost minimal*) with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E if two distinct elements of S are incomparable with respect to $\leq_E^{\mathcal{X},\mathcal{Q}}$ (resp. $\trianglelefteq_E^{\mathcal{X},\mathcal{Q}}$), i.e., for all σ, $\vartheta \in S$, $\sigma \leq_E^{\mathcal{X},\mathcal{Q}} \vartheta$ (resp. $\sigma \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \vartheta$) implies $\sigma = \vartheta$.

Minimality implies almost minimality, but not vice versa: A counterexample is provided by the set $\{\sigma, \eta\}$ and $\mathcal{X} = \{\overline{x}, \overline{y}\}$ from Example 8.

Definition 10. A set of substitutions S is called *disjoint* (resp. *almost disjoint*) with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E if two distinct elements of S have no common E-instance (resp. strong E-instance) on $\mathcal{X}$ and $\mathcal{Q}$, i.e., for all $\sigma, \vartheta \in S$, if there exists φ such that $\sigma \leq_E^{\mathcal{X},\mathcal{Q}} \varphi$ (resp. $\sigma \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \varphi$) and $\vartheta \leq_E^{\mathcal{X},\mathcal{Q}} \varphi$ (resp. $\vartheta \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \varphi$), then $\sigma = \vartheta$.

Disjointness implies almost disjointness, but not vice versa: Consider again the set $\{\sigma, \eta\}$ and $\mathcal{X} = \{\overline{x}, \overline{y}\}$ in Example 8.

Proposition 11. *If a set of substitutions S is disjoint (resp. almost disjoint) with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E, then it is minimal (resp. almost minimal) with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E.*

Proof. Assume that $\sigma, \vartheta \in S$ and $\sigma \leq_E^{\mathcal{X},\mathcal{Q}} \vartheta$. Since $\vartheta \leq_E^{\mathcal{X},\mathcal{Q}} \vartheta$, by disjointness of S with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E we get $\sigma = \vartheta$, which implies minimality of S with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E. Almost minimality can be proved in the same way. □

However, almost disjointness does not imply minimality: Again, consider the set $\{\sigma, \eta\}$ and $\mathcal{X} = \{\overline{x}, \overline{y}\}$ in Example 8. On the other hand, minimality does not imply almost disjointness: Let $\sigma = \{x \mapsto f(a, y)\}$, $\vartheta = \{x \mapsto f(y, b)\}$, $\mathcal{X} = \{x\}$, $\mathcal{Q} = \emptyset$, and $E = \emptyset$. Then $\{\sigma, \vartheta\}$ is minimal but not almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E, because $\sigma \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \varphi$ and $\vartheta \trianglelefteq_E^{\mathcal{X},\mathcal{Q}} \varphi$, where $\varphi = \{x \mapsto f(a, b)\}$, but $\sigma \neq \vartheta$. The same example can be used to show that almost minimality does not imply almost disjointness either. From these observations we can also conclude that neither minimality nor almost minimality imply disjointness.

Definition 12. A substitution σ is *disjointness preserving* (resp. *almost-disjointness preserving*) with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E if for any two substitutions ϑ_1 and ϑ_2, disjointness (resp. almost disjointness) of the set $\{\vartheta_1, \vartheta_2\}$ with respect to the set of variables $\cup_{v \in \mathcal{X}} \mathcal{V}(v\sigma)$ and sequence function symbols $\cup_{\overline{f} \in \mathcal{Q}} \mathcal{F}_S(\overline{f}\sigma)$ modulo E implies disjointness (resp. almost disjointness) of the set $\{\sigma\vartheta_1, \sigma\vartheta_2\}$ with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E.

2.3. Unification problems

Solving equations in an equational theory E is called *E-unification*. The fact that the equation $s \approx t$ has to be solved in an E-theory is written as $s \approx_E^? t$.

First, we define the notion of substitution linearizing away from a set of sequence function symbols. As we will see later, it plays an important role in defining the notion of a solution of an equation. Roughly, it will be used to guarantee that term equations with two ground sides that are not E-equal to each other cannot be solved in the E-theory.

Definition 13. A substitution σ is called *linearizing away* from a finite set of sequence function symbols $\mathcal{Q}$ if the following three conditions hold:

(1) $\mathcal{C}od(\sigma) \cap \mathcal{Q} = \emptyset$.
(2) For all $\overline{f}, \overline{g} \in \mathcal{FD}om(\sigma) \cap \mathcal{Q}$, if $\overline{f} \neq \overline{g}$, then $\{\overline{f}\sigma\} \cap \{\overline{g}\sigma\} = \emptyset$.
(3) If $\overline{f}\sigma = \ulcorner \overline{g_1} \dots, \overline{g_n} \urcorner$ and $\overline{f} \in \mathcal{Q}$, then $\overline{g_i} \neq \overline{g_j}$ for all $1 \leq i < j \leq n$.

(A remark about the notation $\{\overline{f}\sigma\}$: If $\overline{f}\sigma = \ulcorner \overline{f}_1, \dots, \overline{f}_n \urcorner$, then $\{\overline{f}\sigma\}$ is a set of sequence function symbols $\{\overline{f}_1, \dots, \overline{f}_n\}$.) Intuitively, a substitution linearizing away from $\mathcal{Q}$ either leaves a sequence function symbol in $\mathcal{Q}$ "unchanged" or "moves it away from" $\mathcal{Q}$, binding it with a sequence of distinct sequence function symbols that do not occur in $\mathcal{Q}$, and maps different sequence function symbols to disjoint sequences.

Definition 14. Let E be an equational theory and let $\mathcal{F}$ contain $\mathcal{S}ig(E)$. An *E-unification problem* over $\mathcal{F}$ is a finite set of equations $\Gamma = \{s_1 \approx_E^? t_1, \dots, s_n \approx_E^? t_n\}$ over $\mathcal{F}$ and $\mathcal{V}$. An *E-quasi-unifier*, or an *E-quasi-solution*, of Γ is a substitution σ such that $s_i\sigma \approx_E t_i\sigma$ for all $1 \leq i \leq n$. The set of all E-quasi-unifiers of Γ is denoted by $\mathcal{QU}_E(\Gamma)$. An *E-unifier*, or an *E-solution*, of Γ is an E-quasi-unifier of Γ that is linearizing away from $\mathcal{F}_S(\Gamma)$. The set of all E-unifiers of Γ is denoted by $\mathcal{U}_E(\Gamma)$, and Γ is *E-unifiable*, or *E-solvable*, if $\mathcal{U}_E(\Gamma) \neq \emptyset$.

Note that if $\Gamma = \{s_1 \approx^?_E t_1, \dots, s_n \approx^?_E t_n\}$ is an E-unification problem, then all $s_1, \dots, s_n, t_1, \dots, t_n \in \mathcal{T}_{\mathcal{I}}(\mathcal{F}, \mathcal{V})$. The equational theory E is often not mentioned explicitly if the context makes it clear.

Note that if we did not require bindings of sequence function symbols in substitutions to be *nonempty*, the unification problem $\{f(\overline{x}) \approx^?_\emptyset f(\overline{g}(\overline{x}))\}$ would be solvable (with $\{\overline{x} \mapsto \ulcorner\urcorner, \overline{g} \mapsto \ulcorner\urcorner\}$), which is in contrast to the fact that the corresponding positive sentence $\exists \overline{x}\, \forall \overline{y}\, f(\overline{x}) \approx f(\overline{y})$ (from which $f(\overline{x}) \approx^?_\emptyset f(\overline{g}(\overline{x}))$ is obtained by Skolemization) is not valid; see Kutsia and Buchberger (2004).

The following example shows the importance of the condition that an unifier of an E-unification problem Γ should be linearizing away from $\mathcal{F}_S(\Gamma)$.

Example 15. Let $\Gamma = \{f(\overline{a}) \approx^?_\emptyset f(\overline{b})\}$. Then $\mathcal{U}_\emptyset(\Gamma) = \emptyset$. Intuitively, it is justified, because $f(\overline{a})$ and $f(\overline{b})$ are two ground terms such that $f(\overline{a}) \not\approx_\emptyset f(\overline{b})$, or, equivalently, because the corresponding positive sentence $\forall \overline{x}\, \forall \overline{y}\, f(\overline{x}) \approx f(\overline{y})$ is not valid. Note that $\mathcal{QU}_\emptyset(\Gamma) \neq \emptyset$, e.g., $\{\overline{a} \mapsto \overline{b}\} \in \mathcal{QU}_\emptyset(\Gamma)$.

Definition 16. Let Γ be a E-unification problem over $\mathcal{F}$, $\mathcal{X} = \mathcal{V}(\Gamma)$, and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. A *complete set of E-unifiers* of Γ is a set S of substitutions such that

(1) $S \subseteq \mathcal{U}_E(\Gamma)$, i.e., each element of S is an E-unifier of Γ,
(2) for each $\vartheta \in \mathcal{U}_E(\Gamma)$ there exists $\sigma \in S$ such that $\sigma \leq^{\mathcal{X},\mathcal{Q}}_E \vartheta$.

The set S is a *minimal* (resp. *almost minimal*) *complete set of E-unifiers* of Γ if it is a complete set that is minimal (resp. almost minimal) with respect to $\mathcal{X}$ and $\mathcal{Q}$ modulo E.

A minimal (resp. almost minimal) complete set of E-unifiers of Γ, if it exists, is unique up to the equivalence $\doteq^{\mathcal{X},\mathcal{Q}}_E$ (resp. $\triangleq^{\mathcal{X},\mathcal{Q}}_E$), where $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. That is, if S_1 and S_2 are minimal (resp. almost minimal) complete sets of E-unifiers of Γ, then for each $\sigma_1 \in S_1$ there exists exactly one $\sigma_2 \in S_2$ such that $\sigma_1 \doteq^{\mathcal{X},\mathcal{Q}}_E \sigma_2$ (resp. $\sigma_1 \triangleq^{\mathcal{X},\mathcal{Q}}_E \sigma_2$). We will use this fact and denote by $mcu_E(\Gamma)$ (resp. by $amcu_E(\Gamma)$) a minimal (resp. almost minimal) complete set of unifiers, and interpret an equality $mcu_E(\Gamma) = S$ (resp. $amcu_E(\Gamma) = S$) as equality up to the equivalence $\doteq^{\mathcal{X},\mathcal{Q}}_E$ (resp. $\triangleq^{\mathcal{X},\mathcal{Q}}_E$), where $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$.

A substitution σ is a *most general E-unifier* of a unification problem Γ if $mcu_E(\Gamma) = \{\sigma\}$.

Proposition 17. *An E-unification problem Γ has an almost minimal complete set of E-unifiers if and only if it has a minimal complete set of E-unifiers.*

Proof. (⇒) Let $\mathcal{X} = \mathcal{V}(\Gamma)$, $\mathcal{Q} = \mathcal{F}_S(\Gamma)$, and let S be an almost minimal complete set of E-unifiers of Γ. Then the set

$$S \setminus \{\vartheta \mid \vartheta \in S \text{ and there exists } \sigma \in S, \text{ with } \sigma \neq \vartheta \text{ and } \sigma \leq^{\mathcal{X},\mathcal{Q}}_E \vartheta\}$$

is a minimal complete set of E-unifiers of Γ.

(⇐) Every minimal complete set of E-unifiers of Γ is itself an almost minimal complete set of E-unifiers of Γ. □

Example 18. Let $E = \emptyset$.

(1) $\Gamma = \{f(\overline{x}) \approx^?_E f(\overline{y})\}$. Then

$$mcu_E(\Gamma) = \{\{\overline{x} \mapsto \overline{y}\}\}.$$
$$amcu_E(\Gamma) = \{\{\overline{x} \mapsto \overline{y}\}, \{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner\urcorner\}\}.$$

(2) $\Gamma = \{f(\overline{x}, x, \overline{y}) \approx^?_E f(f(\overline{x}), x, a, b)\}$. Then

$$\begin{aligned} mcu_E(\Gamma) &= \{\{x \mapsto f(), \overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner f(), a, b\urcorner\}\}. \\ amcu_E(\Gamma) &= mcu_E(\Gamma). \end{aligned}$$

(3) $\Gamma = \{f(\overline{x}, x, \overline{y}) \approx^?_E f(a, x, b)\}$. Then

$$\begin{aligned} mcu_E(\Gamma) &= \{\{\overline{x} \mapsto a, \overline{y} \mapsto b\}, \{x \mapsto a, \overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner a, b\urcorner\}, \\ &\qquad \{x \mapsto b, \overline{x} \mapsto \ulcorner a, b\urcorner, \overline{y} \mapsto \ulcorner\urcorner\}\}. \\ amcu_E(\Gamma) &= mcu_E(\Gamma). \end{aligned}$$

(4) $\Gamma = \{f(a, \overline{x}) \approx^?_E f(\overline{x}, a)\}$. Then

$$\begin{aligned} mcu_E(\Gamma) &= \{\{\overline{x} \mapsto \ulcorner\urcorner\}, \{\overline{x} \mapsto a\}, \{\overline{x} \mapsto \ulcorner a, a\urcorner\}, \ldots\} \\ amcu_E(\Gamma) &= mcu_E(\Gamma). \end{aligned}$$

(5) $\Gamma = \{f(\overline{x}, \overline{y}, x) \approx^?_E f(\overline{c}, a)\}$. Then

$$\begin{aligned} mcu_E(\Gamma) &= \{\{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \overline{c}, x \mapsto a\}, \{\overline{x} \mapsto \overline{c}, \overline{y} \mapsto \ulcorner\urcorner, x \mapsto a\}, \\ &\qquad \{\overline{x} \mapsto \overline{c_1}, \overline{y} \mapsto \overline{c_2}, x \mapsto a, \overline{c} \mapsto \ulcorner \overline{c_1}, \overline{c_2}\urcorner\}\}. \\ amcu_E(\Gamma) &= mcu_E(\Gamma). \end{aligned}$$

(6) $\Gamma = \{f(\overline{a}) \approx^?_E f(\overline{b})\}$. Then $mcu_E(\Gamma) = amcu_E(\Gamma) = \emptyset$.

Definition 19. Let E be an equational theory and let Γ be an E-unification problem over $\mathcal{F}$. The problem Γ has *type unitary* (*finitary*, *infinitary*) if it has a minimal complete set of E-unifiers of cardinality 1 (finite cardinality, infinite cardinality). If Γ does not have a minimal complete set of E-unifiers, then it is of *type zero*, or *nullary*. We abbreviate type unitary by 1, type finitary by ω, type infinitary by ∞, and type nullary by 0, and order these types as follows: $1 < \omega < \infty < 0$.

The unification type of E with respect to $\mathcal{F}$ is the maximal type of an E-unification problem over $\mathcal{F}$.

Definition 20. Let E be an equational theory, Γ be an E-unification problem over $\mathcal{F}$, and $\mathcal{F}_I$ be the set of individual function symbols in $\mathcal{F}$.

(1) Γ is an *elementary* E-unification problem if $\mathcal{F}_I = \mathcal{S}ig(E)$.
(2) Γ is an E-unification problem *with constants* if $\mathcal{F}_I \setminus \mathcal{S}ig(E)$ is a set of individual constant symbols (called *free* constants).
(3) Γ is a *general* E-unification problem if $\mathcal{F}_I \setminus \mathcal{S}ig(E)$ contains arbitrary individual function symbols (called *free* function symbols).

The equational theory $E = \emptyset$ is called the *free theory with individual and sequence variables and function symbols*. We call unification in the free theory the *syntactic sequence unification.*

Three main questions that arise in unification theory are:

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

Elementary syntactic sequence unification and syntactic sequence unification with constants are trivially decidable unitary problems, which can be solved simply by the Robinson unification algorithm (Robinson, 1965). Therefore, in the rest of the paper we try to answer these questions only for general syntactic sequence unification. We assume that the set of individual function symbols $\mathcal{F}_I$ is countable. Decidability is shown in Section 3, and the questions about the procedure and the type are addressed in Section 4.

The equational theory $E = \{f(\overline{x}, f(\overline{y}), \overline{z}) \approx f(\overline{x}, \overline{y}, \overline{z})\}$, which we encountered in Example 3, is called the *flat theory with individual and sequence variables and function symbols*, where $f \in \mathcal{F}lex_I$ is called a flat symbol. We call unification in the flat theory F-unification. Below we use certain properties of the flat theory in proving decidability of the general syntactic sequence unification.

3. Decidability

To show decidability of a general syntactic sequence unification problem we design a rule-based decision algorithm that nondeterministically performs at most four steps. Each of these steps preserves solvability. On the first step the problem is reduced to another general syntactic sequence unification problem containing no sequence function symbols. The second step gets rid of all free flexible arity functions, obtaining an F-unification problem whose signature consists of fixed arity individual functions and one flat flexible arity individual function. The third step replaces all sequence variables with individual variables. On the fourth step the F-unification problem is represented as a combination of word equations and Robinson unification whose decidability is proved by the Baader–Schulz combination method (Baader and Schulz, 1996).

We start with two lemmata that characterize solutions of general syntactic sequence unification problems.

Lemma 21. *If a general syntactic sequence unification problem Γ is solvable, then there exists a solution σ of Γ such that $\mathcal{FD}om(\sigma) = \emptyset$.*

Proof. Let ϑ_0 be a solution of Γ and let $\overline{f} \mapsto \ulcorner \overline{g_1}, \ldots, \overline{g_n} \urcorner \in \vartheta_0$. Assume without loss of generality that $\overline{f} \in \mathcal{F}_S(\Gamma)$. Since ϑ_0 is linearizing away from $\mathcal{F}_S(\Gamma)$, we have that the sequence function symbols $\overline{g_1}, \ldots, \overline{g_n} \notin \mathcal{F}_S(\Gamma)$, they are all distinct, and do not appear in ϑ_0 in the bindings of any other sequence function symbol. Let ϑ_1 be a substitution obtained from ϑ_0 by

- deleting the binding $\overline{f} \mapsto \ulcorner \overline{g_1}, \ldots, \overline{g_n} \urcorner$ from ϑ_0,
- replacing each binding $x \mapsto t$ in ϑ_0 with $x \mapsto s$, where the term s is obtained from t by deleting all subterms of the form $\overline{g}_i(r_1, \ldots, r_m)$, $2 \leq i \leq n$, $m \geq 0$, and replacing all occurrences of the sequence function symbol $\overline{g}_1$ with $\overline{f}$,
- replacing each binding $\overline{x} \mapsto \ulcorner t_1, \ldots, t_k \urcorner$, $k \geq 1$, in ϑ_0 with the binding $\overline{x} \mapsto \ulcorner s_1, \ldots, s_l \urcorner$, $l \geq 0$, where the sequence $\ulcorner s_1, \ldots, s_l \urcorner$ is obtained from the sequence $\ulcorner t_1, \ldots, t_k \urcorner$ by deleting all subterms of the form $\overline{g}_i(r_1, \ldots, r_m)$, $2 \leq i \leq n$, $m \geq 0$, and replacing all occurrences of the sequence function symbol $\overline{g}_1$ with $\overline{f}$.

The substitution ϑ_1 contains no occurrences of $\overline{g_1}, \ldots, \overline{g_n}$. Intuitively, what the transformation from ϑ_0 to ϑ_1 does is to "undo" splitting $\overline{f}$ into $\overline{g_1}, \ldots, \overline{g_n}$. Since the $\overline{g}$'s occur neither in Γ nor in ϑ_1, we can conclude that ϑ_1 is still a solution of Γ that contains no binding for $\overline{f}$. Repeating this transformation for each binding for sequence function symbols in ϑ_1, we arrive at the substitution σ with the property $\mathcal{FD}om(\sigma) = \emptyset$ and σ is a solution of Γ. □

Lemma 22. *If a general syntactic sequence unification problem Γ is solvable, then there exists a solution of Γ that introduces no new function symbol and does not instantiate any sequence function symbol.*

Proof. Let ϑ be a solution of Γ. By Lemma 21 we can assume that $\mathcal{FD}om(\vartheta) = \emptyset$. If we replace in ϑ every individual (resp. sequence) term whose head does not occur in Γ with a new individual (resp. sequence) variable we get a substitution σ that still is a solution of Γ, but does not introduce any new function symbol. □

Now we start defining inference rules for the decision algorithm. We will formulate them for general syntactic sequence unification problems consisting of a single equation only. For general unification it is not a restriction, because the problems $\{s_1 \approx^?_\emptyset t_1, \ldots, s_n \approx^?_\emptyset t_n\}$ and $\{f(s_1, \ldots, s_n) \approx^?_\emptyset f(t_1, \ldots, t_n)\}$, where $f \in \mathcal{F}_I$, have the same set of solutions, and we can always take such an f. Below we will use the unification problem

$$\{f_0(\overline{x}, x, \overline{y}, \overline{z}, f_0(\overline{u}, x)) \approx^?_\emptyset f_0(g_0(\overline{x}), x, \overline{a}, f_0(g_0(\overline{x}), \overline{u}))\} \tag{1}$$

as an example to demonstrate the steps of the decision algorithm.

The first inference rule eliminates sequence function symbols:

SFE: Sequence Function Elimination
$\{s \approx^?_\emptyset t\} \Longrightarrow \{f(s', x_1, \ldots, x_n) \approx^?_\emptyset f(t', r_1, \ldots, r_n)\}$,
where

- s or t contains sequence function symbols,
- f is a new $n+1$-ary individual function symbol,
- s' and t' are terms obtained respectively from s and t by replacing each sequence function symbol $\overline{g}$ with a new individual function symbol $g_{\overline{g}}$ that has the same arity as $\overline{g}$,
- $x_1, \ldots, x_n$ is an enumeration of all individual variables in $s \approx^?_\emptyset t$,
- each r_i is either a new individual constant c, an individual term of the form $h(y_1, \ldots, y_m)$, or an individual term of the form $h(\overline{y})$, where $h \in \mathcal{F}_I(s, t)$, y_i's are fresh distinct individual variables, and $\overline{y}$ is a fresh sequence variable. $h(y_1, \ldots, y_m)$ is used when h is m-ary, and $h(\overline{y})$ is used when h has a flexible arity.

We assume that for the given s and t we have the function symbol f, the terms s' and t', the enumeration of variables $x_1, \ldots, x_n$, and the constant c fixed. However, we have a nondeterministic choice of the function symbols that define the r_i's, which makes SFE a nondeterministic rule.

In general, there are $(k+1)^n$ different ways to apply SFE on a unification problem Γ, where k is the number of elements in $\mathcal{F}_I(\Gamma)$, and n is the number of elements in $\mathcal{V}_I(\Gamma)$. We can restrict this choice in particular cases. For instance, if Γ is $\{f(s_1, \ldots, s_m) \approx^?_\emptyset f(t_1, \ldots, t_l)\}$, where f occurs neither in s_i's nor in t_i's, then there is no point in considering it as a head of one of the r_i's. In this case we will have k^n alternatives for applying SFE. One can come up with more ways to restrict applications of the SFE rule, but it is not in the scope of this paper.

Example 23. From the unification problem (1), by the rule SFE, we obtain the following three unification problems:

$$\{f_1(f_0(\overline{x}, x, \overline{y}, \overline{z}, f_0(\overline{u}, x)), x) \approx^?_\emptyset f_1(f_0(g_0(\overline{x}), x, a_{\overline{a}}, f_0(g_0(\overline{x}), \overline{u})), c_1)\} \tag{2}$$

$$\{f_1(f_0(\overline{x}, x, \overline{y}, \overline{z}, f_0(\overline{u}, x)), x) \approx^?_\emptyset f_1(f_0(g_0(\overline{x}), x, a_{\overline{a}}, f_0(g_0(\overline{x}), \overline{u})), g_0(\overline{v}_1))\} \tag{3}$$

$$\{f_1(f_0(\overline{x}, x, \overline{y}, \overline{z}, f_0(\overline{u}, x)), x) \approx^?_\emptyset f_1(f_0(g_0(\overline{x}), x, a_{\overline{a}}, f_0(g_0(\overline{x}), \overline{u})), f_0(\overline{v}_1))\}. \tag{4}$$

We may use the rule name abbreviation as a subscript. For instance, we may write $\Gamma \Longrightarrow_{\mathsf{SFE}} \Delta$ to indicate that Δ is obtained from Γ by an application of the SFE rule. The next lemma shows that SFE preserves solvability.

Lemma 24. *Let Γ be a general syntactic sequence unification problem that contains sequence function symbols. Then Γ is solvable if and only if there exists a general syntactic sequence unification problem Δ without sequence function symbols such that $\Gamma \Longrightarrow_{\mathsf{SFE}} \Delta$ and Δ is solvable.*

Proof. ($\Rightarrow$) Let Γ be $\{s \approx^?_\emptyset t\}$ and let σ be a solution of Γ. By Lemma 22 we can assume that σ does not bind any sequence function symbol and does not introduce any new function symbol. Let σ' be a substitution obtained from σ by replacing each sequence function symbol $\overline{g}$ by the corresponding $g_{\overline{g}}$. We take Δ of the form $\{f(s', x_1, \dots, x_n) \approx^?_\emptyset f(t', r_1, \dots, r_n)\}$ that is obtained from Γ by the SFE rule with r_i's selected in the following way: If $x_i\sigma'$ is a variable, then $r_i = c$. If $x_i\sigma' = h(s_1, \dots, s_m)$, where $m \geq 0$, then $r_i = h(y_1, \dots, y_m)$ if h is an m-ary individual function symbol, and $r_i = h(\overline{y})$ if h is a flexible arity individual function symbol. All y_i's are distinct fresh individual variables, and $\overline{y}$ is a fresh sequence variable. Δ does not contain sequence function symbols.

We now construct a solution of Δ. It is easy to see that $s'\sigma' \approx_\emptyset t'\sigma'$. Let now ϑ be a substitution defined as follows: For each x_i, if $x_i\sigma' = y$ for some y, then $y \mapsto c \in \vartheta$; if $x_i\sigma'$ has a form $h(s_1, \dots, s_m)$, where $m \geq 0$, then depending on whether h is an m-ary or a flexible arity symbol we have two cases: If h is m-ary, then $y_j \mapsto s_j \in \vartheta$ for all $1 \leq j \leq m$, where $h(y_1, \dots, y_m) = r_i$. If h has a flexible arity, then $\overline{y} \mapsto \ulcorner s_1, \dots, s_m \urcorner \in \vartheta$, where $h(\overline{y}) = r_i$. We take $\vartheta' = \vartheta\vartheta$. Then $s'\sigma'\vartheta' \approx_\emptyset t'\sigma'\vartheta'$ and $x_i\sigma'\vartheta' \approx_\emptyset r_i\sigma'\vartheta'$ for each x_i. This implies that $\sigma'\vartheta'$ is a solution of Δ.[2]

($\Leftarrow$) From a solution of Δ we can get a solution of Γ replacing each individual function symbol $g_{\overline{g}}$ introduced by the rule SFE by the corresponding symbol $\overline{g} \in \mathcal{F}_S(\Gamma)$. □

Remark 25. Note that had we formulated SFE as $\{s \approx^?_\emptyset t\} \Longrightarrow \{s' \approx^?_\emptyset t'\}$ (with the same conditions on s, t, s', and t' as in SFE), Lemma 24 would not hold. A simple counterexample is the unsolvable problem $\{f(x) \approx^?_\emptyset f(\overline{a})\}$ that in this case would have been transformed into $\{f(x) \approx^?_\emptyset f(a_{\overline{a}})\}$ that is solved by $\{x \mapsto a_{\overline{a}}\}$.

Remark 26. If $\Gamma \Longrightarrow_{\mathsf{SFE}} \Delta$, then, in general, there is no one-to-one correspondence between the sets $amcu_\emptyset(\Gamma)$ and $amcu_\emptyset(\Delta)$ or between the sets $mcu_\emptyset(\Gamma)$ and $mcu_\emptyset(\Delta)$. For instance, if $\Gamma = \{g(\overline{x}, \overline{y}) \approx^?_\emptyset g(\overline{a})\}$, then $\Delta = \{f(g(\overline{x}, \overline{y})) \approx^?_\emptyset f(g(a_{\overline{a}}))\}$ and

$$\begin{aligned} mcu_\emptyset(\Gamma) = amcu_\emptyset(\Gamma) = \{&\{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \overline{a}\}, \{\overline{x} \mapsto \overline{a}, \overline{y} \mapsto \ulcorner\urcorner\},\\ &\{\overline{x} \mapsto \overline{a_1}, \overline{y} \mapsto \overline{a_2}, \overline{a} \mapsto \ulcorner \overline{a_1}, \overline{a_2} \urcorner\}\}.\\ mcu_\emptyset(\Delta) = amcu_\emptyset(\Delta) = \{&\{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto a_{\overline{a}}\}, \{\overline{x} \mapsto a_{\overline{a}}, \overline{y} \mapsto \ulcorner\urcorner\}\}. \end{aligned}$$

The next inference rule eliminates free flexible arity symbols and reduces a general syntactic sequence unification problem to an F-unification problem.

[2] Note that taking ϑ instead of ϑ' is not enough because ϑ can contain bindings $y \mapsto c$ and $y_j \mapsto s_j$ with $y \in \mathcal{V}_I(s_j)$, and therefore, $\sigma'\vartheta$ cannot be a solution of Δ. Just take $s' = f(x)$, $t' = f(g(g(y)))$, $\Delta = \{h(f(x), x, y) \approx^?_\emptyset h(f(g(g(y))), g(y_1), c)\}$, $\sigma' = \{x \mapsto g(g(y))\}$, and $\vartheta = \{y \mapsto c, y_1 \mapsto g(y)\}$ as a counterexample.

FlexE: **Flexible Arity Function Elimination**

$\{s \approx^?_\emptyset t\} \Longrightarrow \{f(s', x_1, \ldots, x_n) \approx^?_F f(t', r_1, \ldots, r_n)\}$,
where

- s and t contain no sequence function symbols,
- s or t contains (free) flexible arity function symbols,
- f is a new $n + 1$-ary individual function symbol,
- s' and t' are terms obtained respectively from s and t by recursively replacing each term $g(s_1, \ldots, s_m)$, where $g \in \mathcal{F}lex$ and is free, with a term $h_g(seq(s_1, \ldots, s_m))$, where $h_g \in \mathcal{F}ix$ is unary and $seq \in \mathcal{F}lex$ is flat. Neither h_g nor seq occurs in s or in t,
- $x_1, \ldots, x_n$ is an enumeration of all individual variables in $s \approx^?_\emptyset t$,
- each r_i is either a new individual constant c or an individual term of the form $h(y_1, \ldots, y_m)$, where $h \in \mathcal{F}_I(s', t') \setminus \{f, seq\}$ is m-ary and y_i's are fresh distinct individual variables.

Like we did for SFE, we assume also for FlexE that for the given s and t the function symbol f, the terms s' and t', the enumeration of variables $x_1, \ldots, x_n$, and the constant c are fixed, as well as the function symbol seq. However, we have a nondeterministic choice of the function symbols that define the r_i's, which makes FlexE a nondeterministic rule. If $\Gamma \Longrightarrow_{\mathsf{FlexE}} \Delta$, then all function symbols except seq that occur in Δ are fixed arity individual function symbols. All sequence variables in Δ are arguments of terms whose head is seq.

In general, there are $(k + 1)^n$ different ways to transform a unification problem Γ into another unification problem Δ by FlexE, where k is the number of elements in $\mathcal{F}_I(\Gamma)$ and n is the number of elements in $\mathcal{V}_I(\Gamma)$. Like for SFE, the number of alternatives for FlexE can be reduced in particular cases.

Example 27. Applying the rule FlexE to the unification problem (2), we obtain the following F-unification problems:

$$\{f_2(s', x) \approx^?_F f_2(t', c_2)\} \tag{5}$$
$$\{f_2(s', x) \approx^?_F f_2(t', a_{\overline{a}})\} \tag{6}$$
$$\{f_2(s', x) \approx^?_F f_2(t', c_1)\} \tag{7}$$
$$\{f_2(s', x) \approx^?_F f_2(t', h_{g_0}(y_2))\} \tag{8}$$
$$\{f_2(s', x) \approx^?_F f_2(t', h_{f_0}(y_2))\} \tag{9}$$
$$\{f_2(s', x) \approx^?_F f_2(t', f_1(y_2, z_2))\} \tag{10}$$

where

$$s' = f_1(h_{f_0}(seq(\overline{x}, x, \overline{y}, \overline{z}, h_{f_0}(seq(\overline{u}, x)))), x),$$
$$t' = f_1(h_{f_0}(seq(h_{g_0}(seq(\overline{x})), x, a_{\overline{a}}, h_{f_0}(seq(h_{g_0}(seq(\overline{x})), \overline{u})))), c_1).$$

Similarly, from (3) by FlexE we obtain:

$$\{f_2(s', x) \approx^?_F f_2(t', c_2)\} \tag{11}$$
$$\{f_2(s', x) \approx^?_F f_2(t', a_{\overline{a}})\} \tag{12}$$
$$\{f_2(s', x) \approx^?_F f_2(t', h_{g_0}(y_2))\} \tag{13}$$
$$\{f_2(s', x) \approx^?_F f_2(t', h_{f_0}(y_2))\} \tag{14}$$
$$\{f_2(s', x) \approx^?_F f_2(t', f_1(y_2, z_2))\} \tag{15}$$

where

$$s' = f_1(h_{f_0}(seq(\overline{x}, x, \overline{y}, \overline{z}, h_{f_0}(seq(\overline{u}, x)))), x),$$
$$t' = f_1(h_{f_0}(seq(h_{g_0}(seq(\overline{x})), x, a_{\overline{a}}, h_{f_0}(seq(h_{g_0}(seq(\overline{x})), \overline{u})))), h_{g_0}(seq(\overline{v}_1))).$$

The equations obtained from (4) by FlexE are similar to the Eqs. (11)–(15) with the difference that t' there is

$$f_1(h_{f_0}(seq(h_{g_0}(seq(\overline{x})), x, a_{\overline{a}}, h_{f_0}(seq(h_{g_0}(seq(\overline{x})), \overline{u})))), h_{f_0}(seq(\overline{v}_1))).$$

Lemma 28. *Let Γ be a general syntactic sequence unification problem with flexible arity functions but without sequence functions. Then Γ is solvable if and only if there exists an F-unification problem Δ without sequence functions and free flexible arity functions such that $\Gamma \Longrightarrow_{\mathsf{FlexE}} \Delta$ and Δ is solvable.*

Proof. ($\Rightarrow$) Let Γ be $\{s \approx^?_\emptyset t\}$ and let σ be a solution of Γ. By Lemma 22 we can assume that σ does not introduce any new function symbol. Let σ' be a substitution obtained from σ by (recursively) replacing each term $g(s_1, \dots, s_m)$, where $g \in \mathcal{F}lex$ and is free, by the corresponding term $h_g(seq(s_1, \dots, s_m))$ and by replacing each binding $\overline{x} \mapsto \ulcorner s_1, \dots, s_m \urcorner$ by $\overline{x} \mapsto seq(s_1, \dots, s_m)$. We take Δ of the form $\{f(s', x_1, \dots, x_n) \approx^?_F f(t', r_1, \dots, r_n)\}$ that is obtained from Γ by the FlexE rule with r_i's selected in the following way: If $x_i\sigma'$ is a variable, then $r_i = c$. If $x_i\sigma' = h(s_1, \dots, s_m)$, where $m \geq 0$, then $r_i = h(y_1, \dots, y_m)$ if h is an m-ary individual function symbol, and $r_i = h_h(seq(\overline{y}))$ if h is a flexible arity individual function symbol. All y_i's are distinct fresh individual variables, and $\overline{y}$ is a fresh sequence variable. Δ does not contain flexible arity function symbols except flat *seq*. There are no sequence function symbols in Δ.

We now construct a solution of Δ. It is easy to see that $s'\sigma' \approx_F t'\sigma'$. Let now ϑ be a substitution defined as follows: For each x_i, if $x_i\sigma' = y$ for some y, then $y \mapsto c \in \vartheta$; if $x_i\sigma'$ has a form $h(seq(s_1, \dots, s_m))$, then $\overline{y} \mapsto seq(s_1, \dots, s_m) \in \vartheta$, where $h(seq(\overline{y})) = r_i$; otherwise, if $x_i\sigma'$ has a form $h(s_1, \dots, s_m)$, where $m \geq 0$, then $y_j \mapsto s_j \in \vartheta$ for all $1 \leq j \leq m$, where $h(y_1, \dots, y_m) = r_i$. We take $\vartheta' = \vartheta\vartheta$. Then $s'\sigma'\vartheta' \approx_F t'\sigma'\vartheta'$ and $x_i\sigma'\vartheta' \approx_F r_i\sigma'\vartheta'$ for each x_i. This implies that $\sigma'\vartheta'$ is a solution of Δ.[3]

($\Leftarrow$) From a solution of Δ we get a solution of Γ replacing each unary symbol h_g and each term $seq(s_1, \dots, s_m)$ introduced by the rule FlexE by the corresponding symbol $g \in \mathcal{F}lex(\Gamma)$ and by the sequence $s_1, \dots, s_m$, respectively. □

Remark 29. Like for the rule SFE above, it is not enough to formulate FlexE as $\{s \approx^?_\emptyset t\} \Longrightarrow \{s' \approx^?_F t'\}$ where s, t, s', and t' satisfy the conditions from FlexE. In this case the unsolvable problem $\{f(x) \approx^?_\emptyset f(a, b)\}$ would be transformed into $\{h_f(seq(x)) \approx^?_F h_f(seq(a, b))\}$, that can be solved by the substitution $\{x \mapsto seq(a, b)\}$ because $h_f(seq(seq(a, b))) \approx_F h_f(seq(a, b))$.

The next inference rule replaces sequence variables with individual variables:

SVR: **Sequence Variable Replacement**

$\Gamma \Longrightarrow \Delta$,
where

[3] Again, it is not enough to take just ϑ instead of ϑ'. See the footnote in the proof of Lemma 24.

- Γ is a F-unification problem,
- Γ contains no sequence function symbols,
- the only flexible arity function symbol that occurs in Γ is flat *seq*,
- Γ contains sequence variables,
- Δ is obtained from Γ by replacing each sequence variable $\overline{x}$ with a new individual variable x_Γ.

We assume that the choice of individual variables x_Γ is fixed for each Γ. This assumption makes SVR a deterministic rule: There is only one way to get Δ from Γ by SVR. If $\Gamma \Longrightarrow_{\mathsf{SVR}} \Delta$, then Δ, like Γ, is an F-unification problem that contains no sequence function symbols and no flexible arity function symbols except *seq*. Moreover, unlike in Γ, there are no sequence variables in Δ.

Example 30. Here we only show the result of application of the rule SVR to the problem (13):

$$\begin{aligned}\{&f_2(f_1(h_{f_0}(seq(z_{\overline{x}}, x, z_{\overline{y}}, z_{\overline{z}}, h_{f_0}(seq(z_{\overline{u}}, x)))), x), x) \approx^?_F \\ &f_2(f_1(h_{f_0}(seq(h_{f_0}(seq(z_{\overline{x}})), x, a_{\overline{a}}, h_{f_0}(seq(h_{g_0}(seq(z_{\overline{x}})), z_{\overline{u}})))), \\ &\qquad h_{g_0}(seq(z_{\overline{v}_1}))), h_{g_0}(y_2))\}\end{aligned} \tag{16}$$

Lemma 31. *Let Γ be an F-unification problem with sequence variables but without sequence function symbols whose only flexible arity function symbol is flat seq and let Δ be obtained from Γ by* SVR. *Then Γ is solvable if and only if Δ is solvable.*

Proof. ($\Rightarrow$) From a solution of Γ we obtain a solution of Δ in two steps: First, we replace each sequence variable $\overline{x}$ with the corresponding x_Γ. Second, we replace each expression $x_\Gamma \mapsto \ulcorner s_1, \ldots, s_n \urcorner$ (if there are any) with the binding $x_\Gamma \mapsto seq(s_1, \ldots, s_n)$.

($\Leftarrow$) Replacing each individual variable x_Γ with $\overline{x}$ in a solution of Δ yields a solution of Γ. $\square$

The last rule makes the decision step:

DS: **Decision Step**

$\Gamma \Longrightarrow \Delta$,
where

- Γ is a F-unification problem,
- the only flexible arity function symbol that occurs in Γ is flat *seq*,
- Γ contains no sequence function symbols and sequence variables,
- Δ is $\top$ if Γ is solvable; otherwise, Δ is $\bot$.

To justify DS we need the combination method:

Theorem 32 (*Combination Method (Baader and Schulz, 1996)*). *Let $E_1, \ldots, E_n$ be equational theories over disjoint signatures such that solvability of E_i-unification problems with linear constant restrictions is decidable for each $1 \leq i \leq n$. Then solvability of elementary unification problems is decidable for the combined theory $E_1 \cup \cdots \cup E_n$.*

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

Lemma 33. *Let Γ be an F-unification problem without sequence function symbols and sequence variables whose only flexible arity function symbol is flat seq. Then solvability of Γ is decidable.*

Proof. Let $\mathcal{F}_1 = \{seq\}$ and $\mathcal{F}_2 = \mathcal{F}ix(\Gamma)$ be two disjoint signatures. Let E_1 be a flat theory over $\mathcal{F}_1$ and $\mathcal{V}_I$ and let E_2 be a free theory over $\mathcal{F}_2$ and $\mathcal{V}_I$. Then Γ can be considered as an elementary unification problem in the combined theory $E_1 \cup E_2$. Then, by Theorem 32, we need to prove that solvability of E_1- and E_2-unification problems with LCV is decidable. E_1-unification problems are, in fact, word equations, while E_2-unification is the Robinson unification. Decidability of word equations with LCV, and of Robinson unification with LCV was proved by Baader and Schulz (1991). $\square$

Lemma 33 shows that the rule DS always gives the output: for any Γ that fulfils the conditions of DS, application of DS yields either $\top$ or $\bot$.

Example 34. The rule DS gives $\top$ when applied to (16). This problem, in fact, has an infinite minimal complete set of F-unifiers. One of the unifiers is $\{z_{\overline{x}} \mapsto seq(), x \mapsto h_{g_0}(seq()), z_{\overline{y}} \mapsto h_{g_0}(seq()), z_{\overline{z}} \mapsto a_{\overline{a}}, z_{\overline{u}} \mapsto seq(), z_{\overline{v}_1} \mapsto seq(), y_2 \mapsto seq()\}$, from which we can reconstruct a solution of (1): $\{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto g_0(), \overline{y} \mapsto g_0(), \overline{z} \mapsto \overline{a}, \overline{u} \mapsto \ulcorner\urcorner\}$.

The *decision algorithm* $\mathfrak{D}$ takes a general syntactic sequence unification problem Γ as an input, turns it into a single equation problem if necessary, and uses the inference rules SFE, FlexE, SVR, and DS in all possible ways to generate a *decision tree* whose root is labeled with Γ, internal nodes are labeled with unification problems (obtained from their ancestors by SFE, FlexE, or SVR), and leaves are labeled either with $\top$ or with $\bot$ (obtained from their ancestors by DS). The decision tree is finite: The conditions of inference rules guarantee that, first, the depth of the tree is maximum four (on each branch there is maximum one application of each rule). Second, it is finitely branching: Each rule can be applied only finitely many times. Lemmas 24, 28, 31 and 33, together with the construction of the decision tree guarantee soundness and completeness of $\mathfrak{D}$. A unification problem Γ is solvable if a decision tree with the root Γ contains a leaf labeled with $\top$, and Γ is unsolvable if all leaves are labeled with $\bot$. This implies the main result of this section:

Theorem 35 (*Decidability*). *General syntactic sequence unification is decidable.*

4. Unification procedure

In the rest of the paper, unless otherwise stated, the term "unification problem" stands for general syntactic sequence unification problem.

We now present inference rules for deriving solutions for unification problems. A *system* is either the symbol $\bot$ (representing failure) or a pair $\langle\Gamma;\ \sigma\rangle$, where Γ is a unification problem and σ is a substitution. The inference system $\mathfrak{I}$ consists of the transformation rules on systems listed below. In the Splitting rule $\overline{f_1}$ and $\overline{f_2}$ are new sequence function symbols of the same arity as $\overline{f}$ in the same rule. We assume that the indices $n, m, k, l \geq 0$.

P: **Projection**

$\langle\Gamma;\ \sigma\rangle \Longrightarrow \langle\Gamma\vartheta;\ \sigma\vartheta\rangle$,
where $\vartheta \neq \varepsilon$, $\mathcal{D}om(\vartheta) \subseteq \mathcal{V}_S(\Gamma)$, and $\mathcal{C}od(\vartheta) = \emptyset$.

T: **Trivial**

$\langle\{s \approx^?_\emptyset s\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \langle\Gamma';\ \sigma\rangle$.

O1: **Orient 1**

$\langle \{s \approx^?_\emptyset x\} \cup \Gamma'; \sigma\rangle \Longrightarrow \langle \{x \approx^?_\emptyset s\} \cup \Gamma'; \sigma\rangle$, if $s \notin \mathcal{V}_I$.

O2: **Orient 2**

$\langle \{f(s, s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{x}, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$
$\Longrightarrow \langle \{f(\overline{x}, t_1, \ldots, t_m) \approx^?_\emptyset f(s, s_1, \ldots, s_n)\} \cup \Gamma';\ \sigma\rangle$, if $s \notin \mathcal{V}_S$.

S: **Solve**

$\langle \{x \approx^?_\emptyset t\} \cup \Gamma'; \sigma\rangle \Longrightarrow \langle \Gamma'\vartheta;\ \sigma\vartheta\rangle$, if $x \notin \mathcal{V}_I(t)$ and $\vartheta = \{x \mapsto t\}$.

TD: **Total Decomposition**

$\langle \{f(s_1, \ldots, s_n) \approx^?_\emptyset f(t_1, \ldots, t_n)\} \cup \Gamma';\ \sigma\rangle$
$\Longrightarrow \langle \{s_1 \approx^?_\emptyset t_1, \ldots, s_n \approx^?_\emptyset t_n\} \cup \Gamma';\ \sigma\rangle$,

if $f(s_1, \ldots, s_n) \neq f(t_1, \ldots, t_n)$ and $s_i, t_i \in \mathcal{T}_I(\mathcal{F}, \mathcal{V})$ for all $1 \leq i \leq n$.

PD1: **Partial Decomposition 1**

$\langle \{f(s_1, \ldots, s_n) \approx^?_\emptyset f(t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow$
$\langle \{s_1 \approx^?_\emptyset t_1, \ldots, s_{k-1} \approx^?_\emptyset t_{k-1}, f(s_k, \ldots, s_n) \approx^?_\emptyset f(t_k, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$,

if $f(s_1, \ldots, s_n) \neq f(t_1, \ldots, t_m)$, for some $1 < k \leq \min(n, m)$, either $s_k \in \mathcal{T}_S(\mathcal{F}, \mathcal{V})$ or $t_k \in \mathcal{T}_S(\mathcal{F}, \mathcal{V})$, and $s_i, t_i \in \mathcal{T}_I(\mathcal{F}, \mathcal{V})$ for all $1 \leq i < k$.

PD2: **Partial Decomposition 2**

$\langle \{f(\overline{f}(r_1, \ldots, r_k), s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{f}(q_1, \ldots, q_l), t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow$
$\langle \{f(r_1, \ldots, r_k) \approx^?_\emptyset f(q_1, \ldots, q_l), f(s_1, \ldots, s_n) \approx^?_\emptyset f(t_1, \ldots, t_m)\} \cup \Gamma'; \sigma\rangle$,

if $f(\overline{f}(r_1, \ldots, r_k), s_1, \ldots, s_n) \neq f(\overline{f}(q_1, \ldots, q_l), t_1, \ldots, t_m)$.

SVE1: **Sequence Variable Elimination 1**

$\langle \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{x}, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$
$\Longrightarrow \langle \{f(s_1, \ldots, s_n) \approx^?_\emptyset f(t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$,

if $f(\overline{x}, s_1, \ldots, s_n) \neq f(\overline{x}, t_1, \ldots, t_m)$.

SVE2: **Sequence Variable Elimination 2**

$\langle \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$
$\Longrightarrow \langle \{f(s_1, \ldots, s_n)\vartheta \approx^?_\emptyset f(t_1, \ldots, t_m)\vartheta\} \cup \Gamma'\vartheta;\ \sigma\vartheta\rangle$,

if $\overline{x} \notin \mathcal{V}_S(t)$ and $\vartheta = \{\overline{x} \mapsto t\}$.

W1: **Widening 1**

$\langle \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$
$\Longrightarrow \langle \{f(\overline{x}, s_1\vartheta, \ldots, s_n\vartheta) \approx^?_\emptyset f(t_1\vartheta, \ldots, t_m\vartheta)\} \cup \Gamma'\vartheta;\ \sigma\vartheta\rangle$,

if $\overline{x} \notin \mathcal{V}_S(t)$ and $\vartheta = \{\overline{x} \mapsto \ulcorner t, \overline{x}\urcorner\}$.

W2: **Widening 2**

$\langle \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{y}, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle$
$\Longrightarrow \langle \{f(s_1\vartheta, \ldots, s_n\vartheta) \approx^?_\emptyset f(\overline{y}, t_1\vartheta, \ldots, t_m\vartheta)\} \cup \Gamma'\vartheta;\ \sigma\vartheta\rangle$,

where $\vartheta = \{\overline{y} \mapsto \ulcorner \overline{x}, \overline{y}\urcorner\}$.

Sp: **Splitting**

$$\langle\{f(\overline{x}, s_1, \dots, s_n) \approx^?_\emptyset f(\overline{f}(r_1, \dots, r_k), t_1, \dots, t_m)\} \cup \Gamma';\ \sigma\rangle$$
$$\Longrightarrow \langle\{f(s_1, \dots, s_n)\vartheta \approx^?_\emptyset f(\overline{f_2}(r_1, \dots, r_k), t_1, \dots, t_m)\vartheta\} \cup \Gamma'\vartheta;\ \sigma\vartheta\rangle,$$

if $\overline{x} \notin \mathcal{V}_S(\overline{f}(r_1, \dots, r_k))$ and $\vartheta = \{\overline{x} \mapsto \overline{f_1}(r_1, \dots, r_k)\}\{\overline{f} \mapsto \ulcorner\overline{f_1}, \overline{f_2}\urcorner\}$.

We write $\vartheta = \{\overline{x} \mapsto \overline{f_1}(r_1, \dots, r_k)\}\{\overline{f} \mapsto \ulcorner\overline{f_1}, \overline{f_2}\urcorner\}$ in the Sp rule because $r_1, \dots, r_k$ can contain $\overline{f}$. In the rule PD2 we replace $\overline{f}$ with f to guarantee that the transformation yields a system again. Besides using the rule name abbreviations as subscripts, we may also write $\langle\Gamma_1;\ \sigma_1\rangle \Longrightarrow_{\mathsf{BT}} \langle\Gamma_2;\ \sigma_2\rangle$ to indicate that $\langle\Gamma_1;\ \sigma_1\rangle$ was transformed to $\langle\Gamma_2;\ \sigma_2\rangle$ by some *basic transformation* (i.e., non-projection) rule. We denote the transitive closure of $\Longrightarrow$ by $\Longrightarrow^+$.

The Projection can be applied to the same system in (finitely many) different ways.

Example 36. Projection rule transforms the system $\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\}; \varepsilon\rangle$ in the following three different ways:

$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{P}} \langle\{f(a) \approx^?_\emptyset f(a, \overline{y})\}; \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle.$$
$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{P}} \langle\{f(\overline{x}, a) \approx^?_\emptyset f(a)\}; \{\overline{y} \mapsto \ulcorner\urcorner\}\rangle.$$
$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{P}} \langle\{f(a) \approx^?_\emptyset f(a)\}; \{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner\urcorner\}\rangle.$$

The rules SVE2, W1, W2, and Sp can be applied to the same equation.

Example 37. SVE2 and W1 transform the system $\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\};\ \varepsilon\rangle$:

$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{SVE2}} \langle\{f(a) \approx^?_\emptyset f(\overline{y})\}; \{\overline{x} \mapsto a\}\rangle.$$
$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{W1}} \langle\{f(\overline{x}, a) \approx^?_\emptyset f(\overline{y})\}; \{\overline{x} \mapsto \ulcorner a, \overline{x}\urcorner\}\rangle.$$

SVE2, W1, and Sp transform the system $\langle\{f(\overline{x}, \overline{y}, \overline{a}) \approx^?_\emptyset f(\overline{a}, \overline{z})\};\ \varepsilon\rangle$:

$$\langle\{f(\overline{x}, \overline{y}, \overline{a}) \approx^?_\emptyset f(\overline{a}, \overline{z})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{SVE2}} \langle\{f(\overline{y}, \overline{a}) \approx^?_\emptyset f(\overline{z})\}; \{\overline{x} \mapsto \overline{a}\}\rangle.$$
$$\langle\{f(\overline{x}, \overline{y}, \overline{a}) \approx^?_\emptyset f(\overline{a}, \overline{z})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{W1}} \langle\{f(\overline{x}, \overline{y}, \overline{a}) \approx^?_\emptyset f(\overline{z})\}; \{\overline{x} \mapsto \ulcorner\overline{a}, \overline{x}\urcorner\}\rangle.$$
$$\langle\{f(\overline{x}, \overline{y}, \overline{a}) \approx^?_\emptyset f(\overline{a}, \overline{z})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{Sp}} \langle\{f(\overline{y}, \overline{a_1}, \overline{a_2}) \approx^?_\emptyset f(\overline{a_2}, \overline{z})\};$$
$$\{\overline{x} \mapsto \overline{a_1}, \overline{a} \mapsto \ulcorner\overline{a_1}, \overline{a_2}\urcorner\}\rangle.$$

SVE2, W1, and W2 transform the system $\langle\{f(\overline{x}, a) \approx^?_\emptyset f(\overline{z}, \overline{y})\};\ \varepsilon\rangle$:

$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(\overline{z}, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{SVE2}} \langle\{f(a) \approx^?_\emptyset f(\overline{y})\};\ \{\overline{x} \mapsto \overline{z}\}\rangle.$$
$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(\overline{z}, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{W1}} \langle\{f(\overline{x}, a) \approx^?_\emptyset f(\overline{y})\};\ \{\overline{x} \mapsto \ulcorner\overline{z}, \overline{x}\urcorner\}\rangle.$$
$$\langle\{f(\overline{x}, a) \approx^?_\emptyset f(\overline{z}, \overline{y})\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{W2}} \langle\{f(a) \approx^?_\emptyset f(\overline{z}, \overline{y})\};\ \{\overline{z} \mapsto \ulcorner\overline{x}, \overline{z}\urcorner\}\rangle.$$

A *derivation* is a sequence $\langle\Gamma_1; \sigma_1\rangle \Longrightarrow \langle\Gamma_2; \sigma_2\rangle \Longrightarrow \cdots$ of system transformations. A *selection strategy* $\mathcal{S}$ is a function which given a derivation $\langle\Gamma_1; \sigma_1\rangle \Longrightarrow \cdots \Longrightarrow \langle\Gamma_n; \sigma_n\rangle$ returns an equation, called a *selected equation*, from Γ_n. A derivation is *via* a selection strategy $\mathcal{S}$ if in the derivation all choices of selected equations, being transformed by the transformation rules, are performed according to $\mathcal{S}$.

In the definition below we need two versions of the decision algorithm $\mathfrak{D}$. One, denoted by $\mathfrak{D}_m$, calls in the DS step (as one of the ingredients of the combination method that it uses) the decision algorithm for solving equations in a free monoid (Abdulrab and Pécuchet, 1990). The other one, denoted as $\mathfrak{D}_s$, uses the decision algorithm for solving equations in a free semigroup (Makanin, 1977).

Definition 38. A *syntactic sequence unification procedure* $\mathfrak{U}$ is any program that takes a system $\langle \Gamma;\ \varepsilon\rangle$ and a selection strategy $\mathcal{S}$ as an input and uses the transformation rules of the inference system $\mathfrak{I}$ to generate a tree of derivations via $\mathcal{S}$, called the *unification tree for Γ via $\mathcal{S}$*, in the following way:

(1) The root of the tree is labeled with $\langle \Gamma;\ \varepsilon\rangle$.
(2) Each branch of the tree is a derivation via $\mathcal{S}$ of the form

$$\langle \Gamma;\ \varepsilon\rangle \Longrightarrow_{\diamond} \langle \Gamma_1;\ \sigma_1\rangle \Longrightarrow_{\mathsf{BT}} \langle \Gamma_2;\ \sigma_2\rangle \Longrightarrow_{\mathsf{BT}} \cdots,$$

where $\diamond$ is either P or BT. The nodes in the tree are systems.
(3) If different instances of the projection rule are applicable to the root node in the tree, they are applied concurrently.
(4) If several basic transformation rules are applicable to the selected equation in a node in the tree, they are applied concurrently.
(5) The decision algorithm $\mathfrak{D}_m$ is applied to the root. If the problem Γ in the root is unsolvable, then the branch is extended by $\langle \Gamma; \varepsilon\rangle \Longrightarrow_{\mathsf{DAm}} \bot$ and the procedure stops with failure. Otherwise, the decision algorithm $\mathfrak{D}_s$ is applied to the root and to each node generated by P, SVE2, W1, W2, and Sp, to decide whether the node contains a unification problem that can be solved without replacing any sequence variable with the empty sequence. If Δ in a node $\langle \Delta;\ \delta\rangle$ cannot be solved under this restriction, then the branch is extended only by $\langle \Delta;\ \delta\rangle \Longrightarrow_{\mathsf{DAs}} \bot$.

The unification tree for Γ via $\mathcal{S}$, generated by $\mathfrak{U}$, is denoted as $\mathcal{UT}^{\mathcal{S}}_{\mathfrak{U}}(\Gamma)$. We will often omit $\mathcal{S}$ and write just $\mathcal{UT}_{\mathfrak{U}}(\Gamma)$.

The leaves of $\mathcal{UT}_{\mathfrak{U}}(\Gamma)$ are labeled either with the systems of the form $\langle \emptyset;\ \sigma\rangle$ or with the system $\bot$. The branches of $\mathcal{UT}_{\mathfrak{U}}(\Gamma)$ that end with leaves of the form $\langle \emptyset;\ \sigma\rangle$ are called *successful branches*, and those with the leaves $\bot$ are *failed branches*. We denote by $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ the solution set for Γ generated by $\mathfrak{U}$, i.e., the set of all substitutions σ such that $\langle \emptyset;\ \sigma\rangle$ is a leaf of $\mathcal{UT}_{\mathfrak{U}}(\Gamma)$.

Example 39. Let $\Gamma = \{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\}$. DAm reports that Γ is solvable. Then the unification procedure generates the following seven derivations:

$$\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle \Longrightarrow_{\mathsf{DAs}} \bot$$

$$\begin{aligned}
&\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle\\
&\quad\Longrightarrow_{\mathsf{P}}\quad \langle\{f(x) \approx^?_\emptyset f(f(), x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner\urcorner\}\rangle\\
&\quad\Longrightarrow_{\mathsf{DAs}}\quad \bot
\end{aligned}$$

$$\begin{aligned}
&\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle\\
&\quad\Longrightarrow_{\mathsf{P}}\quad \langle\{f(\overline{x}, x) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \{\overline{y} \mapsto \ulcorner\urcorner\}\rangle\\
&\quad\Longrightarrow_{\mathsf{DAs}}\quad \bot
\end{aligned}$$

$$\begin{aligned}
&\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle\\
&\quad\Longrightarrow_{\mathsf{P}}\quad \langle\{f(x, \overline{y}) \approx^?_\emptyset f(f(), x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle\\
&\quad\Longrightarrow_{\mathsf{PD1}}\ \langle\{x \approx^?_\emptyset f(), f(\overline{y}) \approx^?_\emptyset f(x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle\\
&\quad\Longrightarrow_{\mathsf{S}}\quad \langle\{f(\overline{y}) \approx^?_\emptyset f(f(), a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f()\}\rangle\\
&\quad\Longrightarrow_{\mathsf{SVE2}}\ \langle\{f() \approx^?_\emptyset f(a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto f()\}\rangle\\
&\quad\Longrightarrow_{\mathsf{DAs}}\quad \bot
\end{aligned}$$

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{P}} & \langle\{f(x, \overline{y}) \approx^?_\emptyset f(f(), x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{PD1}} & \langle\{x \approx^?_\emptyset f(), f(\overline{y}) \approx^?_\emptyset f(x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{S}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(f(), a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f()\}\rangle \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), \overline{y}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{SVE2}} & \langle\{f() \approx^?_\emptyset f(b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{DAs}} & \bot
\end{array}$$

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{P}} & \langle\{f(x, \overline{y}) \approx^?_\emptyset f(f(), x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{PD1}} & \langle\{x \approx^?_\emptyset f(), f(\overline{y}) \approx^?_\emptyset f(x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{S}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(f(), a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f()\}\rangle \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), \overline{y}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a, \overline{y}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{SVE2}} & \langle\{f() \approx^?_\emptyset f()\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a, b\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{T}} & \langle\emptyset;\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a, b\urcorner\}\rangle
\end{array}$$

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, x, \overline{y}) \approx^?_\emptyset f(f(\overline{x}), x, a, b)\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{P}} & \langle\{f(x, \overline{y}) \approx^?_\emptyset f(f(), x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{PD1}} & \langle\{x \approx^?_\emptyset f(), f(\overline{y}) \approx^?_\emptyset f(x, a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{S}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(f(), a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f()\}\rangle \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(a, b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), \overline{y}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{y}) \approx^?_\emptyset f(b)\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a, \overline{y}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{y}) \approx^?_\emptyset f()\};\ \{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a, b, \overline{y}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{DAs}} & \bot
\end{array}$$

Therefore, $\mathcal{S}ol_{\mathfrak{U}}(\Gamma) = \{\{\overline{x} \mapsto \ulcorner\urcorner, x \mapsto f(), \overline{y} \mapsto \ulcorner f(), a, b\urcorner\}\}$.

Example 40. Let $\Gamma = \{f(\overline{x}, a, \overline{x}) \approx^?_\emptyset f(a, \overline{x}, a)\}$. Then the unification procedure generates the following derivations:

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, a, \overline{x}) \approx^?_\emptyset f(a, \overline{x}, a)\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{P}} & \langle\{f(a) \approx^?_\emptyset f(a, a)\}; \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{DAs}} & \bot
\end{array}$$

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, a, \overline{x}) \approx^?_\emptyset f(a, \overline{x}, a)\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{SVE2}} & \langle\{f(a, a) \approx^?_\emptyset f(a, a)\}; \{\overline{x} \mapsto a\}\rangle \\
\Longrightarrow_{\mathsf{T}} & \langle\emptyset; \{\overline{x} \mapsto a\}\rangle
\end{array}$$

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, a, \overline{x}) \approx^?_\emptyset f(a, \overline{x}, a)\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{W1}} & \langle\{f(\overline{x}, a, a, \overline{x}) \approx^?_\emptyset f(a, \overline{x}, a)\}; \{\overline{x} \mapsto \ulcorner a, \overline{x}\urcorner\}\rangle \\
\Longrightarrow_{\mathsf{DAs}} & \bot
\end{array}$$

Therefore, $\mathcal{S}ol_{\mathfrak{U}}(\Gamma) = \{\{\overline{x} \mapsto a\}\}$.

Example 41. Let $\Gamma = \{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{x})\}$. Then the unification procedure generates infinitely many derivations:

$$\begin{array}{ll}
\multicolumn{2}{l}{\langle\{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{x})\};\ \varepsilon\rangle} \\
\Longrightarrow_{\mathsf{P}} & \langle\{f(a) \approx^?_\emptyset f(a)\}; \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle
\end{array}$$

$$\Longrightarrow_{\mathsf{T}} \quad \langle \emptyset;\ \{\overline{x} \mapsto \ulcorner\urcorner\}\rangle$$

$$\begin{aligned}
&\langle \{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{x})\};\ \varepsilon\rangle\\
&\quad \Longrightarrow_{\mathsf{SVE2}} \langle \{f(a) \approx^?_\emptyset f(a)\};\ \{\overline{x} \mapsto a\}\rangle\\
&\quad \Longrightarrow_{\mathsf{T}} \quad \langle \emptyset;\ \{\overline{x} \mapsto a\}\rangle
\end{aligned}$$

$$\begin{aligned}
&\langle \{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{x})\};\ \varepsilon\rangle\\
&\quad \Longrightarrow_{\mathsf{W1}} \quad \langle \{f(\overline{x}, a) \approx^?_\emptyset f(a, \overline{x})\};\ \{\overline{x} \mapsto \ulcorner a, \overline{x}\urcorner\}\rangle\\
&\quad \Longrightarrow_{\mathsf{SVE2}} \langle \{f(a) \approx^?_\emptyset f(a)\};\ \{\overline{x} \mapsto \ulcorner a, a\urcorner\}\rangle\\
&\quad \Longrightarrow_{\mathsf{T}} \quad \langle \emptyset;\ \{\overline{x} \mapsto \ulcorner a, a\urcorner\}\rangle
\end{aligned}$$

and so on. $\mathcal{S}ol_{\mathfrak{U}}(\Gamma) = \{\{\overline{x} \mapsto \ulcorner\urcorner\}, \{\overline{x} \mapsto a\}, \{\overline{x} \mapsto \ulcorner a, a\urcorner\}, \ldots\}$.

4.1. Soundness

In this subsection we will show soundness of $\mathfrak{U}$: for any general syntactic sequence unification problem Γ, every substitution in the solution set $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is a syntactic unifier of Γ.

Below Γ and Δ are general syntactic sequence unification problems.

Lemma 42. *If $\mathcal{QU}_\emptyset(\Gamma) = \mathcal{QU}_\emptyset(\Delta)$, then $\mathcal{QU}_\emptyset(\Gamma\vartheta) = \mathcal{QU}_\emptyset(\Delta\vartheta)$ for any substitution ϑ.*

Proof. $\sigma \in \mathcal{QU}_\emptyset(\Gamma\vartheta)$ if and only if $\vartheta\sigma \in \mathcal{QU}_\emptyset(\Gamma)$ if and only if $\vartheta\sigma \in \mathcal{QU}_\emptyset(\Delta)$ if and only if $\sigma \in \mathcal{QU}_\emptyset(\Delta\vartheta)$. □

Lemma 43. *If $\langle \Gamma;\ \sigma\rangle \Longrightarrow \langle \Delta;\ \sigma\vartheta\rangle$, then $\mathcal{QU}_\emptyset(\Delta) = \mathcal{QU}_\emptyset(\Gamma\vartheta)$.*

Proof. The nontrivial cases concern the rules S, SVE2, W1, W2, and Sp.

S: If $x \notin \mathcal{V}_I(t)$, then $x\theta \approx t\theta$ for $\theta = \{x \mapsto t\}$, and $\Gamma\theta = \{x\theta \approx^?_\emptyset t\theta\} \cup \Gamma'\theta$ and $\Delta = \Gamma'\theta$ have the same set of quasi-unifiers. SVE2 is similar to S.

W1: Let $\Gamma = \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma'$ such that $\overline{x} \notin \mathcal{V}_S(t)$. By W1 we get $\Delta = \{f(\overline{x}, s_1\vartheta, \ldots, s_n\vartheta) \approx^?_\emptyset f(t_1\vartheta, \ldots, t_m\vartheta)\} \cup \Delta'$, where $\vartheta = \{\overline{x} \mapsto \ulcorner t, \overline{x}\urcorner\}$. Then $\Gamma\vartheta$ and Δ have exactly the same set of quasi-unifiers. W2 is similar to W1.

Sp: Let $\Gamma = \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{g}(r_1, \ldots, r_k), t_1, \ldots, t_m)\} \cup \Gamma'$ such that $\overline{x} \notin \mathcal{V}_S(\overline{g}(r_1, \ldots, r_k))$. Then by Sp we get a new problem $\Delta = \{f(s_1, \ldots, s_n)\vartheta \approx^?_\emptyset f(\overline{g_2}(r_1, \ldots, r_k), t_1, \ldots, t_m)\vartheta\} \cup \Delta'$, where $\vartheta = \{\overline{x} \mapsto \overline{g_1}(r_1, \ldots, r_k)\}\{\overline{g} \mapsto \ulcorner \overline{g_1}, \overline{g_2}\urcorner\}$. Then $\Gamma\theta$ and Δ have exactly the same set of quasi-unifiers. □

Lemma 44. *If $\langle \Gamma;\ \sigma\rangle \Longrightarrow^+ \langle \Delta;\ \sigma\vartheta\rangle$, then $\mathcal{QU}_\emptyset(\Delta) = \mathcal{QU}_\emptyset(\Gamma\vartheta)$.*

Proof. By induction on the derivation length. Lemma 43 proves the case when the length is 1. Now assume that the lemma holds for the derivation length n. We have to show that it holds for the length $n + 1$. Let the derivation have a form $\langle \Gamma;\ \sigma\rangle \Longrightarrow \langle \Delta_1;\ \sigma\delta_1\rangle \Longrightarrow \cdots \Longrightarrow \langle \Delta_{n+1};\ \sigma\delta_1\delta_2\cdots\delta_{n+1}\rangle$ where $\vartheta = \delta_1\delta_2\cdots\delta_{n+1}$. By the induction hypothesis $\mathcal{QU}_\emptyset(\Delta_n) = \mathcal{QU}_\emptyset(\Gamma\delta_1\cdots\delta_n)$. By Lemma 43 $\mathcal{QU}_\emptyset(\Delta_{n+1}) = \mathcal{QU}_\emptyset(\Delta_n\delta_{n+1})$. Therefore, by Lemma 42 we obtain $\mathcal{QU}_\emptyset(\Delta_{n+1}) = \mathcal{QU}_\emptyset(\Gamma\delta_1\cdots\delta_n\delta_{n+1}) = \mathcal{QU}_\emptyset(\Gamma\vartheta)$. □

Lemma 45. *If $\langle \Gamma;\ \varepsilon\rangle \Longrightarrow^+ \langle \emptyset;\ \vartheta\rangle$, then $\vartheta \in \mathcal{U}_\emptyset(\Gamma)$.*

Proof. By Lemma 44 we have $\mathcal{QU}_\emptyset(\emptyset) = \mathcal{QU}_\emptyset(\Gamma\vartheta)$. Since $\varepsilon \in \mathcal{QU}_\emptyset(\emptyset)$, we get $\varepsilon \in \mathcal{QU}_\emptyset(\Gamma\vartheta)$ and, hence, $\vartheta \in \mathcal{QU}_\emptyset(\Gamma)$. Moreover, ϑ is linearizing away from $\mathcal{F}_S(\Gamma)$, because all the bindings for sequence function symbols introduced during the derivation (by Sp) introduce fresh distinct sequence function symbols. Hence, $\vartheta \in \mathcal{U}_\emptyset(\Gamma)$. □

From Lemma 45 and Definition 38 we immediately get soundness of $\mathfrak{U}$:

Theorem 46 (*Soundness of* $\mathfrak{U}$). *Let Γ be a general syntactic sequence unification problem. Then $\mathcal{S}ol_{\mathfrak{U}}(\Gamma) \subseteq \mathcal{U}_{\emptyset}(\Gamma)$.*

4.2. Completeness

Proving completeness is more involved than the soundness proof. In this section we prove completeness by showing that for any solution ϑ of a unification problem Γ there exists a derivation from $\langle \Gamma;\ \varepsilon \rangle$ that terminates with success and the substitution in the last system of the derivation is strongly more general than ϑ. For the termination proof, we need to define a complexity measure on the systems, introduce a well-founded ordering on the measures, and show that every step in the derivation strictly decreases the measure.

First, we introduce notions needed later to define complexity measures.

Definition 47. The *length of the image* of a set of variables $\mathcal{X}$ with respect to a substitution σ, denoted as $\mathcal{L}en(\mathcal{X}, \sigma)$, is defined as $\sum_{v \in \mathcal{X}} len(v\sigma)$, where $len(v\sigma)$ is 1 if $v\sigma$ is a single term, and is n if $v\sigma = \ulcorner t_1, \ldots, t_n \urcorner$ for some terms $t_1, \ldots, t_n$, $n \geq 0$.

The following lemma is an easy consequence of Definitions 47 and 7:

Lemma 48. *For all σ, ϑ, $\mathcal{X}$, and $\mathcal{Q}$, if $\mathcal{R}an(\sigma) \subseteq \mathcal{X}$ and $\sigma \trianglelefteq_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$, then $\mathcal{L}en(\mathcal{X}, \sigma) \leq \mathcal{L}en(\mathcal{X}, \vartheta)$.*

We denote by $\mathcal{D}if(\mathcal{X}, \vartheta, \sigma)$ the *length difference* $\mathcal{L}en(\mathcal{X}, \vartheta) - \mathcal{L}en(\mathcal{X}, \sigma)$. Obviously, $\mathcal{D}if(\mathcal{X}, \vartheta, \sigma) = \mathcal{D}if(\mathcal{V}_S(\mathcal{X}), \vartheta, \sigma)$.

Lemma 49. *Let Γ be a unification problem, $\mathcal{X} = \mathcal{V}(\Gamma)$, $\mathcal{Q} = \mathcal{F}_S(\Gamma)$, and $\vartheta \in \mathcal{U}_{\emptyset}(\Gamma)$. Let $\mathcal{S}$ be a selection strategy. If ϑ is non-erasing on $\mathcal{X}$, then there exists a derivation via $\mathcal{S}$ of the form $\langle \Gamma_1;\ \sigma_1 \rangle \Longrightarrow_{\mathsf{BT}}^{+} \langle \emptyset, \sigma_k \rangle$ with $\Gamma_1 = \Gamma$ and $\sigma_1 = \varepsilon$ such that $\sigma_k \trianglelefteq_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$.*

Proof. We prove the lemma in two steps. First, we construct a derivation of the form $\langle \Gamma_1;\ \sigma_1 \rangle \Longrightarrow_{\mathsf{BT}} \langle \Gamma_2;\ \sigma_2 \rangle \Longrightarrow_{\mathsf{BT}} \cdots$ with $\Gamma_1 = \Gamma$ and $\sigma_1 = \varepsilon$ such that $\sigma_i \trianglelefteq_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$ for all $i \geq 1$, and then we show that it terminates with success.

Step 1: Construction. We construct the derivation recursively. We take $\Gamma_1 = \Gamma$, $\sigma_1 = \varepsilon$, and start the derivation from $\langle \Gamma_1;\ \sigma_1 \rangle$. Obviously, $\sigma_1 \trianglelefteq_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$.

We assume that $\langle \Gamma_n;\ \sigma_n \rangle$, $n \geq 1$, $\Gamma_n \neq \emptyset$ belongs to the derivation. We have to find a system $\langle \Gamma_{n+1};\ \sigma_{n+1} \rangle$ such that $\langle \Gamma_n;\ \sigma_n \rangle \Longrightarrow_{\mathsf{BT}} \langle \Gamma_{n+1};\ \sigma_{n+1} \rangle$ and $\sigma_{n+1} \trianglelefteq_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$.

Since $\langle \Gamma_n;\ \sigma_n \rangle$ belongs to the derivation, we have $\sigma_n \trianglelefteq_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$, i.e., there exists φ, non-erasing on $\mathcal{X}$, such that $\sigma_n \varphi =_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$. Let σ'_n be the restriction of σ_n to $\mathcal{X}$ and $\mathcal{Q}$. Moreover, we can assume without loss of generality that $\mathcal{FD}om(\varphi) \subseteq \mathcal{F}_S(\Gamma_n)$, because any sequence function symbol in $\mathcal{F}_S(\Gamma_n)$ either belongs to $\mathcal{Q}' = \mathcal{Q} \setminus \mathcal{FD}om(\sigma'_n)$ or is introduced by applying σ'_n to an element of $\mathcal{Q}$. Then $\sigma'_n \varphi =_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$ holds. Our goal is first to prove that $\varphi \in \mathcal{U}_{\emptyset}(\Gamma_n)$ and then to extend the derivation with the help of φ.

From $\sigma'_n \varphi =_{\emptyset}^{\mathcal{X},\mathcal{Q}} \vartheta$ we have $\sigma'_n \varphi \in \mathcal{U}_{\emptyset}(\Gamma_1) \subseteq \mathcal{QU}(\Gamma_1)$ and therefore $\varphi \in \mathcal{QU}_{\emptyset}(\Gamma_1 \sigma'_n)$. Moreover, by Lemma 44 we have $\mathcal{QU}_{\emptyset}(\Gamma_1 \sigma'_n) = \mathcal{QU}_{\emptyset}(\Gamma_n)$. Therefore, $\varphi \in \mathcal{QU}_{\emptyset}(\Gamma_n)$.

Now we show that φ is linearizing away from $\mathcal{F}_S(\Gamma_n)$ which will imply $\varphi \in \mathcal{U}_{\emptyset}(\Gamma_n)$. Assume by contradiction that it is not. Then we have the following three cases:

(1) $\mathcal{C}od(\varphi) \cap \mathcal{F}_S(\Gamma_n) \neq \emptyset$. Assume $\overline{f} \mapsto \ulcorner \dots, \overline{g}, \dots \urcorner \in \varphi$ with $\overline{g} \in \mathcal{F}_S(\Gamma_n)$. Since $\mathcal{FD}om(\varphi) \subseteq \mathcal{F}_S(\Gamma_n)$, and any sequence function symbol in $\mathcal{F}_S(\Gamma_n)$ either belongs to $\mathcal{Q}'$ or is introduced by applying σ_n' to an element of $\mathcal{Q}$, we have one of the possible four cases: (i) $\overline{f}, \overline{g} \in \mathcal{Q}'$, (ii) $\overline{f} \in \mathcal{C}od(\sigma_n')$ and $\overline{g} \in \mathcal{Q}'$, (iii) $\overline{f} \in \mathcal{Q}'$ and $\overline{g} \in \mathcal{C}od(\sigma_n')$, or (iv) $\overline{f}, \overline{g} \in \mathcal{C}od(\sigma_n')$. In the first two cases $\mathcal{C}od(\sigma_n'\varphi) \cap \mathcal{Q} \neq \emptyset$, which contradicts the fact that $\sigma_n'\varphi$ is linearizing away from $\mathcal{Q}$ (because $\sigma_n'\varphi \in \mathcal{U}_\emptyset(\Gamma_1)$). In the last two cases there exist $\overline{h_1}, \overline{h_2} \in \mathcal{FD}om(\sigma_n'\varphi) \cap \mathcal{Q}$ such that $\overline{h_1}\sigma_n'\varphi$ and $\overline{h_2}\sigma_n'\varphi$ share a common sequence function symbol, which also violates the condition on $\sigma_n'\varphi$ being linearizing away from $\mathcal{Q}$.

(2) There exist two distinct sequence function symbols $\overline{f}, \overline{g} \in \mathcal{FD}om(\varphi) \cap \mathcal{F}_S(\Gamma_n)$ such that the sequences $\overline{f}\varphi$ and $\overline{g}\varphi$ share a common sequence function symbol. Then there exist $\overline{h_1}, \overline{h_2} \in \mathcal{FD}om(\sigma_n'\varphi) \cap \mathcal{Q}$ such that $\overline{h_1}\sigma_n'\varphi$ and $\overline{h_2}\sigma_n'\varphi$ share a common sequence function symbol, which is a contradiction to the fact that $\sigma_n'\varphi$ is linearizing away from $\mathcal{Q}$.

(3) There exists $\overline{f} \in \mathcal{F}_S(\Gamma_n)$ such that $\overline{f}\varphi = \ulcorner \overline{g_1}, \dots, \overline{g_m} \urcorner$ and $\overline{g_i} = \overline{g_j}$ for some $1 \leq i < j \leq m$. Then there exists $\overline{h} \in \mathcal{Q}$ such that the sequence $\overline{h}\sigma_n'\varphi$ contains two equal elements $\overline{g}_i$ and $\overline{g}_j$. Again a contradiction.

Hence, all three cases contradict the fact that $\sigma_n'\varphi$ is linearizing away from $\mathcal{Q}$. This implies that φ is linearizing away from $\mathcal{F}_S(\Gamma_n)$ and, therefore, $\varphi \in \mathcal{U}_\emptyset(\Gamma_n)$.

Let $s \approx^?_\emptyset t$ be an equation in Γ_n selected by $\mathcal{S}$. We represent Γ_n as $\{s \approx^?_\emptyset t\} \cup \Gamma_n'$. Depending on the form of the pair $\langle s, t\rangle$, we have the following four cases:

Case 1: $\langle s, t\rangle$ is a pair of identical terms. We extend the derivation with the step $\langle \Gamma_n;\ \sigma_n\rangle \Longrightarrow_{\mathsf{T}} \langle \Gamma_n';\ \sigma_n\rangle$. Therefore, $\sigma_{n+1} = \sigma_n \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$.

Case 2: $\langle s, t\rangle$ is a pair of distinct individual variables. Let $s = x$ and $t = y$. Then $x\varphi \approx_\emptyset y\varphi$. Let $\psi = \{x \mapsto y\}$. Then $\psi\varphi = \varphi$. Thus, $\sigma_n\psi\varphi =^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$, which implies $\sigma_n\psi \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$. Therefore, we can take $\Gamma_{n+1} = \Gamma_n'\psi$, $\sigma_{n+1} = \sigma_n\psi$, and extend the derivation with the step $\langle \Gamma_n;\ \sigma_n\rangle \Longrightarrow_{\mathsf{S}} \langle \Gamma_{n+1};\ \sigma_{n+1}\rangle$.

Case 3: $\langle s, t\rangle$ is a pair of an individual variable and a non-variable term. If $s = x$ and t is a non-variable term that does not contain x, then we proceed as in the previous case, extending the derivation with the rule S. Note that t cannot contain x, because otherwise it would lead to the contradiction $\varphi \in \emptyset$. If $t = x$ and s is a non-variable term, then we take $\Gamma_{n+1} = \{x \approx^?_\emptyset s\} \cup \Gamma_n'$, $\sigma_{n+1} = \sigma_n$, and extend the derivation with the step $\langle \Gamma_n;\ \sigma_n\rangle \Longrightarrow_{\mathsf{O1}} \langle \Gamma_{n+1};\ \sigma_{n+1}\rangle$.

Case 4: $\langle s, t\rangle$ is a pair of distinct non-variable terms. Assume $s = f(s_1, \dots, s_k)$ and $t = f(t_1, \dots, t_m)$. If neither s_1 nor t_1 is a sequence variable, then we extend the derivation with the step $\langle \Gamma_n;\ \sigma_n\rangle \Longrightarrow_\diamond \langle \Gamma_n';\ \sigma_n\rangle$, where $\diamond$ is either TD, $\mathsf{PD1}$, or $\mathsf{PD2}$. Therefore, $\sigma_{n+1} = \sigma_n \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$. If both s_1 and t_1 are sequence variables with $s_1 = t_1$, then we extend the derivation with the step $\langle \Gamma_n;\ \sigma_n\rangle \Longrightarrow_{\mathsf{SVE1}} \langle \Gamma_n';\ \sigma_n\rangle$ and $\sigma_{n+1} = \sigma_n \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$. If t_1 is a sequence variable and s_1 is not, then the derivation is extended with the step $\langle \Gamma_n;\ \sigma_n\rangle \Longrightarrow_{\mathsf{O2}} \langle \Gamma_n';\ \sigma_n\rangle$ and, again, $\sigma_{n+1} = \sigma_n \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$.

The only remaining case is when $s_1 \in \mathcal{V}_S$, $m > 0$ and $s_1 \notin \mathcal{V}(t_1)$. Let s_1 be $\overline{x}$. We have the following three cases depending on t_1:

If t_1 is a sequence variable $\overline{y}$, then we define substitutions ψ and ρ in three different ways as follows. (i) If $f(\overline{x})\varphi \approx_\emptyset f(\overline{y})\varphi$, then $\psi = \{\overline{x} \mapsto \overline{y}\}$ and $\rho = \varphi$. In this case the rule $\mathsf{SVE2}$ will be used. (ii) If there exists a nonempty sequence of terms $r_1, \dots, r_l$ such that $f(\overline{x}\varphi) \approx_\emptyset f(\overline{y}\varphi, r_1, \dots, r_l)$, then $\psi = \{\overline{x} \mapsto \ulcorner \overline{y}, \overline{x} \urcorner\}$ and $\rho = \{\overline{x} \mapsto \ulcorner r_1, \dots, r_l \urcorner\} \cup \varphi^-$, where $\varphi^- = \varphi|_{\mathcal{D}om(\varphi)\setminus\{\overline{x}\}}$. We will use the rule $\mathsf{W1}$. (iii) If there exists a nonempty sequence of terms $r_1, \dots, r_l$ such that $f(\overline{y}\varphi) \approx_\emptyset f(\overline{x}\varphi, r_1, \dots, r_l)$, then $\psi = \{\overline{y} \mapsto \ulcorner \overline{x}, \overline{y} \urcorner\}$ and

$\rho = \{\overline{y} \mapsto \ulcorner r_1, \dots, r_l \urcorner\} \cup \varphi^-$, where $\varphi^- = \varphi|_{\mathcal{D}om(\varphi)\setminus\{\overline{y}\}}$. We will use W2 in this case. Since $\varphi \in \mathcal{U}_\emptyset(\Gamma_n)$, these three cases for ψ are the only possibilities for getting Γ_{n+1} from Γ_n via the selection strategy $\mathcal{S}$. Thus, we can take $\sigma_{n+1} = \sigma_n\psi$ and extend the derivation either by SVE2, W1 or W2 rules, depending on the cases for ψ. Since $\psi\rho = \varphi$, we have $\sigma_n\psi\rho =_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$ and therefore $\sigma_{n+1}\rho =_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$. Moreover, ρ is non-erasing on the set $\mathcal{X}$, which implies $\sigma_{n+1} \trianglelefteq_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$.

If t_1 is an individual term, then we can proceed as in the previous case (where t_1 was a sequence variable $\overline{y}$). The only difference is that now ψ can have only two alternatives instead of three. Therefore, we can extend the derivation either by SVE2 or W1 rules ensuring $\sigma_{n+1} = \sigma_n\psi \trianglelefteq_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$.

If t_1 is a sequence term $\overline{f}(r_1, \dots, r_l)$, then we define substitutions ψ and ρ in three different ways. (i) If $f(\overline{x})\varphi \approx_\emptyset f(t_1)\varphi$, then $\psi = \{\overline{x} \mapsto t_1\}$ and $\rho = \varphi$. We will use SVE2 here. (ii) If there exists a nonempty sequence of terms $q_1, \dots, q_j$ such that $f(\overline{x})\varphi \approx_\emptyset f(t_1\varphi, q_1, \dots, q_j)$, then $\psi = \{\overline{x} \mapsto \ulcorner t_1, \overline{x}\urcorner\}$ and $\rho = \{\overline{x} \mapsto \ulcorner q_1, \dots, q_j\urcorner\} \cup \varphi^-$, where $\varphi^- = \varphi|_{\mathcal{D}om(\varphi)\setminus\{\overline{x}\}}$. The rule W1 will be used in this case. (iii) If $f(\overline{x})\varphi = f(\overline{f_1}(r_1, \dots, r_l))\varphi$ and $f(\overline{f}(r_1, \dots, r_l))\varphi = f(\overline{f_1}(r_1, \dots, r_l), \overline{f_2}(r_1, \dots, r_l))\varphi$ for some sequence function symbols $\overline{f_1}$ and $\overline{f_2}$, then $\psi = \{\overline{x} \mapsto \overline{f_1}(r_1, \dots, r_l)\}\{\overline{f} \mapsto \ulcorner \overline{f_1}, \overline{f_2}\urcorner\}$ and $\rho = \varphi|_{\mathcal{D}om(\varphi)\setminus\{\overline{f}\}}$. In this case Sp will be used. Since $\varphi \in \mathcal{U}_\emptyset(\Gamma_n)$, these three cases for ψ are the only possibilities for getting Γ_{n+1} from Γ_n via the selection strategy $\mathcal{S}$. Thus, we can take $\sigma_{n+1} = \sigma_n\psi$ and get Γ_{n+1} from Γ_n either by SVE2, W1 or Sp rules, depending on the case for ψ. Since $\psi\rho = \varphi$, we have $\sigma_n\psi\rho =_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$ and therefore $\sigma_{n+1}\rho =_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$. Moreover, ρ is non-erasing on the set $\mathcal{X}$, which implies that $\sigma_{n+1} \trianglelefteq_\emptyset^{\mathcal{X},\mathcal{Q}} \vartheta$. Hence, Case 4 is proved.

Since ϑ is non-erasing on $\mathcal{X}$, none of the cases above involves the projection rule. This implies that the derivation constructed consists of basic transformation (BT) steps only. This concludes Step 1 of the proof.

Step 2: Termination. We have to show that the derivation constructed terminates with success. We define a complexity measure (with respect to a given substitution and a set of variables) on systems as a 7-tuple of integers, ordered by the lexicographic ordering on tuples of integers as follows: the tuple $\langle m_1, m_2, m_3, m_4, m_5, m_6, m_7\rangle$ is a complexity measure of a system $\langle \Delta;\ \sigma\rangle$ with respect to a substitution λ and a set of variables $\mathcal{Y}$, if

$m_1 =$ the number of distinct variables in Δ;
$m_2 = \mathcal{D}if(\mathcal{Y}, \lambda, \sigma)$;
$m_3 =$ the number of symbols in Δ;
$m_4 =$ the number of occurrences of sequence function symbols in Δ;
$m_5 =$ the number of subterms in Δ of the form $f(s_1, \dots, s_n)$, where s_1 is not a sequence term;
$m_6 =$ the number of equations in Δ of the form $t \approx_\emptyset^? x$, where t is not an individual variable;
$m_7 =$ the number of equations in Δ that have the form $f(s, s_1, \dots, s_n) \approx_\emptyset^? f(\overline{x}, t_1, \dots, t_m)$, where s is not a sequence variable.

All these numbers except m_2 are, indeed, natural numbers. As for m_2, depending on λ and σ, it can also be negative. But for each substitution σ_i in the derivation constructed above, and for the set $\mathcal{X} = \mathcal{V}(\Gamma)$, we have $\mathcal{R}an(\sigma_i) \subseteq \mathcal{X}$ because $\sigma_1 = \varepsilon$ and no rule in $\mathfrak{I}$ introduces a new variable. Therefore, by Lemma 48 we have $\mathcal{D}if(\mathcal{X}, \vartheta, \sigma_i) \geq 0$ for each i, i.e., m_2 is a natural number for each $\langle \Gamma_i;\ \sigma_i\rangle$. Thus, the ordering on complexity measures of systems (with respect to ϑ and $\mathcal{X}$) in the derivation is well-founded. Then, each step in the derivation strictly reduces the complexity measure: T and SVE1 do not increase m_1 and m_2 and decrease m_3. O1 decreases m_6 and does not increase the others. O2 decreases m_7 and does not increase the others. S, SVE2, and Sp decrease m_1. TD does not increase m_1 and m_2 and decreases m_3. PD1 does not increase m_1, m_2, m_3, and m_4 and decreases m_5. PD2 does not increase m_1, m_2, and m_3 and decreases m_4. W1 and W2 do not increase m_1 and decrease m_2. Since the case with W1 and W2 is not as obvious as with the other rules, we show the details for W1: Let the corresponding step in the derivation be $\langle \Gamma_i;\ \sigma_i\rangle \Longrightarrow_{\mathsf{W1}} \langle \Gamma_{i+1};\ \sigma_{i+1}\rangle$, where $\Gamma_i = \{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma'_i$, $\Gamma_{i+1} = \{f(\overline{x}, s_1\varphi, \ldots, s_n\varphi) \approx^?_\emptyset f(t_1\varphi, \ldots, t_m\varphi)\} \cup \Gamma'_i\varphi$ and $\sigma_{i+1} = \sigma_i\varphi$, with $\overline{x} \notin \mathcal{V}_S(t)$ and $\varphi = \{\overline{x} \mapsto \ulcorner t, \overline{x}\urcorner\}$. It is clear that the step does not enlarge the number of distinct variables in the systems. Moreover, $\overline{x} \in \mathcal{X}$, because $\overline{x} \in \mathcal{V}_S(\Gamma_i)$ and $\mathcal{V}_S(\Gamma_i) \subseteq \mathcal{X}$. (The last inclusion follows from the fact that no rule in the inference system $\mathfrak{I}$ introduces a new variable.) For all $\overline{y} \in \mathcal{X} \setminus \{\overline{x}\}$ we have $len(\overline{y}\sigma_{i+1}) \geq len(\overline{y}\sigma_i)$. As for $\overline{x}$ itself, if $\overline{x} \in \mathcal{VD}om(\sigma_i)$, then $\overline{x} \in \{\overline{x}\sigma_i\}$ (otherwise it would have been impossible to have $\overline{x}$ in $\mathcal{V}_S(\Gamma_{i+1})$) and therefore $len(\overline{x}\sigma_{i+1}) > len(\overline{x}\sigma_i)$. If $\overline{x} \notin \mathcal{VD}om(\sigma_i)$, then $len(\overline{x}\sigma_{i+1}) = 2 > 1 = len(\overline{x}\sigma_i)$. This means that $\mathcal{L}en(\mathcal{X}, \sigma_{i+1}) > \mathcal{L}en(\mathcal{X}, \sigma_i)$, i.e., $\mathcal{D}if(\mathcal{X}, \vartheta, \sigma_i) > \mathcal{D}if(\mathcal{X}, \vartheta, \sigma_{i+1})$.

Hence, the derivation terminates. Let $\langle \Gamma_k, \sigma_k\rangle$ be the last system in the derivation. Then, on the one hand, $\sigma_k \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$. On the other hand, $\Gamma_k = \emptyset$ (otherwise we could make another step), which finishes the proof of the lemma. □

Lemma 50. *Let Γ be a unification problem, $\mathcal{X} = \mathcal{V}(\Gamma)$, $\mathcal{Q} = \mathcal{F}_S(\Gamma)$, and $\vartheta \in \mathcal{U}_\emptyset(\Gamma)$. Let $\mathcal{S}$ be a selection strategy. If ϑ is erasing on $\mathcal{X}$, then there exists a derivation via $\mathcal{S}$ of the form $\langle \Gamma_0;\ \sigma_0\rangle \Longrightarrow_{\mathsf{P}} \langle \Gamma_1;\ \sigma_1\rangle \Longrightarrow^+_{\mathsf{BT}} \langle \emptyset;\ \sigma_n\rangle$ with $\Gamma_0 = \Gamma$ and $\sigma_0 = \varepsilon$ such that $\sigma_n \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$.*

Proof. Assume $\overline{x_1}, \ldots, \overline{x_k} \in \mathcal{X}$, $k > 0$, are all the variables in $\mathcal{X}$ that ϑ maps to the empty sequence. Let $\sigma_1 = \{\overline{x_1} \mapsto \ulcorner\urcorner, \ldots, \overline{x_k} \mapsto \ulcorner\urcorner\}$, $\Gamma_1 = \Gamma_0\sigma_1$, and make the projection rule the first step of derivation: $\langle \Gamma_0;\ \sigma_0\rangle \Longrightarrow_{\mathsf{P}} \langle \Gamma_1;\ \sigma_1\rangle$. We have $\sigma_1\sigma_1 = \sigma_1$. Let φ be $\vartheta|_{\mathcal{D}om(\vartheta)\setminus\mathcal{D}om(\sigma_1)}$. Then $\vartheta = \sigma_1\varphi = \sigma_1\sigma_1\varphi \in \mathcal{U}_\emptyset(\Gamma_0)$. Therefore, $\sigma_1\varphi \in \mathcal{U}_\emptyset(\Gamma_0\sigma_1)$, i.e., $\vartheta \in \mathcal{U}_\emptyset(\Gamma_1)$. Moreover, ϑ is non-erasing on $\mathcal{V}(\Gamma_1)$. Then, by Lemma 49, there exists a derivation via $\mathcal{S}$ $\langle \Delta_1;\ \delta_1\rangle \Longrightarrow^+_{\mathsf{BT}} \langle \emptyset;\ \delta_n\rangle$ such that $\Delta_1 = \Gamma_1$, $\delta_1 = \varepsilon$, and $\delta_n \trianglelefteq^{\mathcal{X}_1,\mathcal{Q}_1}_\emptyset \vartheta$, where $\mathcal{X}_1 = \mathcal{V}(\Gamma_1)$ and $\mathcal{Q}_1 = \mathcal{F}_S(\Gamma_1)$. Hence, there exists a substitution ψ that is non-erasing on $\mathcal{X}_1$ such that $\delta_n\psi =^{\mathcal{X}_1,\mathcal{Q}_1}_\emptyset \vartheta$. Since $\mathcal{Q}_1 = \mathcal{Q}$, we have, in fact, $\delta_n\psi =^{\mathcal{X}_1,\mathcal{Q}}_\emptyset \vartheta$. Moreover, $\mathcal{VD}om(\delta_n) \subseteq \mathcal{X}_1$ and $\mathcal{R}an(\delta_n) \subseteq \mathcal{X}_1$ since the rules in $\mathfrak{I}$ do not introduce new variables. From $\mathcal{R}an(\delta_n) \subseteq \mathcal{X}_1$ we can assume that ψ is non-erasing on any finite superset of $\mathcal{X}_1$ and in particular, on $\mathcal{X}$. From $\mathcal{VD}om(\delta_n) \subseteq \mathcal{X}_1$ we have $\sigma_1\delta_n = \sigma_1 \cup \delta_n$ and, finally, $\sigma_1\delta_n\psi =^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$.

For all $i \geq 1$, if $\langle \Delta_{i+1};\ \delta_{i+1}\rangle$ is obtained from $\langle \Delta_i;\ \delta_i\rangle$ by a rule of $\mathfrak{I}$, then $\langle \Delta_{i+1};\ \sigma_1\delta_{i+1}\rangle$ can be obtained from $\langle \Delta_i;\ \sigma_1\delta_i\rangle$ by the same rule. Thus, taking $\Gamma_i = \Delta_i$ and $\sigma_i = \sigma_1\delta_i$ for all $1 < i \leq n$, we get the derivation $\langle \Gamma;\ \varepsilon\rangle \Longrightarrow_{\mathsf{P}} \langle \Gamma_1;\ \sigma_1\rangle \Longrightarrow_{\mathsf{BT}} \langle \Gamma_2;\ \sigma_2\rangle \Longrightarrow_{\mathsf{BT}} \cdots \Longrightarrow_{\mathsf{BT}} \langle \emptyset;\ \sigma_n\rangle$ such that $\sigma_n \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta$. □

From Theorem 46, Lemmas 49 and 50, and the fact that $\trianglelefteq^{\mathcal{X},\mathcal{Q}}_E \subseteq \leq^{\mathcal{X},\mathcal{Q}}_E$, by Definitions 38 and 16 we get the completeness theorem:

Theorem 51 (*Completeness of $\mathfrak{U}$*). *Let Γ be a general syntactic sequence unification problem. Then $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is a complete set of solutions of Γ.*

4.3. *Almost minimality*

The solution set $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$, in general, is not minimal with respect to $\mathcal{V}(\Gamma)$ and $\mathcal{F}_S(\Gamma)$. Just consider $\Gamma = \{f(\overline{x}) \approx^?_\emptyset f(\overline{y})\}$: We have $\mathcal{S}ol_{\mathfrak{U}}(\Gamma) = \{\{\overline{x} \mapsto \overline{y}\},\ \{\overline{x} \mapsto \ulcorner\urcorner, \overline{y} \mapsto \ulcorner\urcorner\}\}$. However, it can be shown that $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is almost minimal with respect to $\mathcal{V}(\Gamma)$ and $\mathcal{F}_S(\Gamma)$. In fact, we will prove a stronger statement: $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is almost disjoint with respect to $\mathcal{V}(\Gamma)$ and $\mathcal{F}_S(\Gamma)$.

Before proceeding we need to establish a few notational conventions. We denote by $\mathcal{BS}ub(\Gamma, Eq)$ the set of substitutions obtained by performing a basic transformation step on a general syntactic sequence unification problem $\Gamma = \{Eq\} \cup \Gamma'$ with Eq as the selected equation: $\mathcal{BS}ub(\Gamma, Eq) = \{\delta \mid \langle\{Eq\} \cup \Gamma';\ \varepsilon\rangle \Longrightarrow_{\mathsf{BT}} \langle\Delta;\ \delta\rangle$ for some $\Delta\}$. By $\mathcal{P}roj(\Gamma)$ we denote the set of projecting substitutions $\{\pi \mid \langle\Gamma;\ \varepsilon\rangle \Longrightarrow_{\mathsf{P}} \langle\Gamma\pi;\ \pi\rangle\}$. Finally, $\mathcal{S}ub(\Gamma, Eq)$ denotes $\mathcal{P}roj(\Gamma) \cup \mathcal{BS}ub(\Gamma, Eq)$.

To prove that $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is almost disjoint with respect to $\mathcal{V}(\Gamma)$ and $\mathcal{F}_S(\Gamma)$ we will show that for any $Eq \in \Gamma$, the set $\mathcal{S}ub(\Gamma, Eq)$ is almost disjoint with respect to the sets $\mathcal{V}(\Gamma)$ and $\mathcal{F}_S(\Gamma)$, and preserves almost disjointness.

Lemma 52. *Let Γ be a unification problem and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then $\mathcal{P}roj(\Gamma)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. Assume by contradiction that $\mathcal{P}roj(\Gamma)$ is not almost disjoint. Then there exist $\sigma, \vartheta \in \mathcal{P}roj(\Gamma)$, $\sigma \neq \vartheta$, and a φ such that $\sigma \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$ and $\vartheta \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$. Since $\sigma \neq \vartheta$, we assume without loss of generality that there exists $\overline{x} \in \mathcal{X}$ such that $\overline{x} \mapsto \ulcorner\urcorner \in \vartheta \setminus \sigma$. But then $\overline{x} \mapsto \ulcorner\urcorner \in \varphi$, and therefore $\overline{x} \mapsto \ulcorner\urcorner \in \delta$ for any δ such that $\sigma\delta =^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$. This contradicts the fact that $\sigma \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$. $\square$

Lemma 53. *Let Γ be a unification problem, Eq be an equation in Γ, and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then $\mathcal{S}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$ if and only if $\mathcal{BS}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. ($\Rightarrow$) Since $\mathcal{S}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$, any subset of $\mathcal{S}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.

($\Leftarrow$) Since, by the assumption and Lemma 52, $\mathcal{BS}ub(\Gamma, Eq)$ and $\mathcal{P}roj(\Gamma)$ are almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$, what remains to show is that no $\sigma \in \mathcal{BS}ub(\Gamma, Eq)$ and $\vartheta \in \mathcal{P}roj(\Gamma)$ can have a common strong $\emptyset$-instance on $\mathcal{X}$ and $\mathcal{Q}$. Assume by contradiction that there exists φ such that $\sigma \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$ and $\vartheta \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$. Then there exists $\overline{x} \in \mathcal{X}$ such that $\overline{x} \mapsto \ulcorner\urcorner \in \vartheta$ and $\overline{x} \mapsto \ulcorner\urcorner \notin \sigma$. But then $\overline{x} \mapsto \ulcorner\urcorner \in \varphi$, and therefore $\overline{x} \mapsto \ulcorner\urcorner \in \delta$ for any δ such that $\sigma\delta =^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$. This contradicts the fact that $\sigma \trianglelefteq^{\mathcal{X},\mathcal{Q}}_\emptyset \varphi$. $\square$

Lemma 54. *Let Γ be a unification problem, Eq be an equation in Γ, and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then $\mathcal{BS}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. We consider only the cases when $\mathcal{BS}ub(\Gamma, Eq)$ contains more than one element. (Otherwise the lemma is trivial.) This leads to assuming that Eq has a form $\{f(s, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\}$, where s is a sequence variable and t is a term different from s. Depending on t, we have the following two cases:

Case 1. Let $s = \overline{x}$ and $t = \overline{y}$. Then $\mathcal{BS}ub(\Gamma, Eq) = \{\sigma_1, \sigma_2, \sigma_3\}$, where $\sigma_1 = \{\overline{x} \mapsto \overline{y}\}$, $\sigma_2 = \{\overline{x} \mapsto \ulcorner\overline{y}, \overline{x}\urcorner\}$, and $\sigma_3 = \{\overline{y} \mapsto \ulcorner\overline{x}, \overline{y}\urcorner\}$. Assume by contradiction that $\mathcal{BS}ub(\Gamma, Eq)$ is not almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Then there must exist i and j with $1 \leq i < j \leq 3$ such that σ_i and σ_j have a common strong instance, i.e., there exist ϑ and φ such that $\sigma_i\vartheta =_{\emptyset}^{\mathcal{X},\mathcal{Q}} \sigma_j\varphi$, and ϑ and φ are non-erasing on $\mathcal{X}$.

Assume that $i = 1$ and $j = 2$. Then we have $\overline{x}\sigma_1\vartheta = \overline{y}\vartheta$, $\overline{x}\sigma_2\varphi = \ulcorner\overline{y}\varphi, \overline{x}\varphi\urcorner$, $\overline{y}\sigma_1\vartheta = \overline{y}\vartheta$, $\overline{y}\sigma_2\varphi = \overline{y}\varphi$. This implies that $\overline{x}\varphi = \ulcorner\urcorner$, i.e., $\overline{x} \mapsto \ulcorner\urcorner \in \varphi$, but contradicts the fact that φ is non-erasing on $\mathcal{X}$.

Assume that $i = 1$ and $j = 3$. Then $\overline{x}\sigma_1\vartheta = \overline{y}\vartheta$, $\overline{x}\sigma_3\varphi = \overline{x}\varphi$, $\overline{y}\sigma_1\vartheta = \overline{y}\vartheta$, $\overline{y}\sigma_3\varphi = \ulcorner\overline{x}\varphi, \overline{y}\varphi\urcorner$. This implies that $\overline{y}\varphi = \ulcorner\urcorner$, i.e., $\overline{y} \mapsto \ulcorner\urcorner \in \varphi$, but contradicts the fact that φ is non-erasing on $\mathcal{X}$.

Assume that $i = 2$ and $j = 3$. Then $\overline{x}\sigma_2\vartheta = \ulcorner\overline{y}\vartheta, \overline{x}\vartheta\urcorner$, $\overline{x}\sigma_3\varphi = \overline{x}\varphi$, $\overline{y}\sigma_2\vartheta = \overline{y}\vartheta$, $\overline{y}\sigma_3\varphi = \ulcorner\overline{x}\varphi, \overline{y}\varphi\urcorner$. This implies that $\overline{y}\varphi = \ulcorner\urcorner$ and $\overline{x}\vartheta = \ulcorner\urcorner$, i.e., $\overline{y} \mapsto \ulcorner\urcorner \in \varphi$ and $\overline{x} \mapsto \ulcorner\urcorner \in \vartheta$, but contradicts the fact that φ and ϑ are non-erasing on $\mathcal{X}$. Hence, in Case 1 $\mathcal{BS}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.

Case 2. If s occurs in t, then $\mathcal{BS}ub(\Gamma, Eq) = \emptyset$ is trivially almost disjoint. Otherwise, if t is not a sequence term, we can proceed like for Case 1, having only two elements in $\mathcal{BS}ub(\Gamma, Eq)$. If t is a non-variable sequence term, assume that $s = \overline{x}$ and $t = \overline{f}(r_1, \dots, r_k)$ for $k \geq 0$. Then $\mathcal{BS}ub(\Gamma, Eq) = \{\sigma_1, \sigma_2, \sigma_3\}$, where $\sigma_1 = \{\overline{x} \mapsto \overline{f}(r_1, \dots, r_k)\}$, $\sigma_2 = \{\overline{x} \mapsto \ulcorner\overline{f}(r_1, \dots, r_k), \overline{x}\urcorner\}$, and $\sigma_3 = \{\overline{x} \mapsto \overline{f_1}(r_1, \dots, r_k)\}\{\overline{f} \mapsto \ulcorner\overline{f_1}, \overline{f_2}\urcorner\}$. The substitutions σ_1 and σ_2 cannot have a common strong instance. This can be shown in the same way as in the case $i = 1$, $j = 2$ above. The substitutions σ_1 and σ_3, and also σ_2 and σ_3 cannot have even a common instance, because $\overline{f_2}$ cannot be eliminated from σ_3. Hence, $\mathcal{BS}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. □

Lemmas 52–54 imply almost disjointness of $\mathcal{S}ub(\Gamma, Eq)$:

Lemma 55. *Let Γ be a unification problem, Eq be an equation in Γ, and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then $\mathcal{S}ub(\Gamma, Eq)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Now we show that substitutions from $\mathcal{S}ub(\Gamma, Eq)$ preserve almost disjointness. First, we prove two auxiliary lemmata.

Lemma 56. *Let Γ be a unification problem and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then every $\sigma \in \mathcal{P}roj(\Gamma)$ is almost-disjointness preserving with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. Let ϑ_1 and ϑ_2 be two substitutions such that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to the set of variables $\cup_{v\in\mathcal{X}} \mathcal{V}(v\sigma) = \mathcal{X} \setminus \mathcal{VD}om(\sigma)$ and the set of sequence function symbols $\cup_{\overline{f}\in\mathcal{Q}} \mathcal{F}_S(\overline{f}\sigma) = \mathcal{Q}$. Assume by contradiction that $\{\sigma\vartheta_1, \sigma\vartheta_2\}$ is not almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Then there exist φ_1 and φ_2, both non-erasing on $\mathcal{X}$, such that $\sigma\vartheta_1\varphi_1 =_{\emptyset}^{\mathcal{X},\mathcal{Q}} \sigma\vartheta_2\varphi_2$. But then $\vartheta_1\varphi_1 =_{\emptyset}^{\mathcal{X}^-,\mathcal{Q}} \vartheta_2\varphi_2$, where $\mathcal{X}^- = \mathcal{X} \setminus \mathcal{VD}om(\sigma)$, and this contradicts the fact that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\mathcal{X}^-$ and $\mathcal{Q}$. □

Lemma 57. *Let Γ be a unification problem, Eq be an equation in Γ, and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then every $\sigma \in \mathcal{BS}ub(\Gamma, Eq)$ is almost-disjointness preserving with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. We prove the lemma by case distinction on the basic transformation rules applicable to Γ where *Eq* is the selected equation. For the rules T, O1, O2, TD, PD1, PD2, and SVE1 the set $\mathcal{BS}ub(\Gamma, Eq)$ consist of ε only and, therefore, the lemma trivially holds. The cases with the rules S, SVE2, W1, W2, and Sp are considered below.

S: We have $Eq = \{x \approx^?_\emptyset t\}$, $x \notin \mathcal{V}(t)$, and $\sigma = \{x \mapsto t\}$. Moreover, $\cup_{v\in\mathcal{X}} \mathcal{V}(v\sigma) = \mathcal{X} \setminus \{x\}$ and $\cup_{\overline{f}\in\mathcal{Q}} \mathcal{F}_S(\overline{f}\sigma) = \mathcal{Q}$. Let ϑ_1 and ϑ_2 be two substitutions such that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\mathcal{X} \setminus \{x\}$ and $\mathcal{Q}$. We have to show that $\{\sigma\vartheta_1, \sigma\vartheta_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Assume by contradiction that it is not. Then there exist substitutions φ_1 and φ_2 such that $\sigma\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset \sigma\vartheta_2\varphi_2$. This implies that $v\sigma\vartheta_1\varphi_1 = v\sigma\vartheta_2\varphi_2$ for all $v \in \mathcal{X} \setminus \{x\}$. But since $v\sigma = v$ for all $v \in \mathcal{X} \setminus \{x\}$, we get $v\vartheta_1\varphi_1 = v\vartheta_2\varphi_2$ for all $v \in \mathcal{X} \setminus \{x\}$, which contradicts almost disjointness of $\{\vartheta_1, \vartheta_2\}$ with respect to $\mathcal{X} \setminus \{x\}$ and $\mathcal{Q}$.

SVE2: We have $Eq = \{f(\overline{x}, s_1, \dots, s_n) \approx^?_\emptyset f(t, t_1, \dots, t_m)\}$ with $\overline{x} \notin \mathcal{V}_S(t)$, and $\sigma = \{\overline{x} \mapsto t\}$. We can proceed here in the same way as for S above.

W1: We have $Eq = \{f(\overline{x}, s_1, \dots, s_n) \approx^?_\emptyset f(t, t_1, \dots, t_m)\}$ with $\overline{x} \notin \mathcal{V}_S(t)$, and $\sigma = \{\overline{x} \mapsto \ulcorner t, \overline{x}\urcorner\}$. Assume t is an individual term. (The case when t is a sequence variable can be proved like for W2, and the case when t is a non-variable sequence term can be proved like for Sp; see below.) Let ϑ_1 and ϑ_2 be two substitutions such that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\cup_{v\in\mathcal{X}} \mathcal{V}(v\sigma) = \mathcal{X}$ and $\cup_{\overline{f}\in\mathcal{Q}} \mathcal{F}_S(\overline{f}\sigma) = \mathcal{Q}$ and assume by contradiction that $\{\sigma\vartheta_1, \sigma\vartheta_2\}$ is not. Then there exist φ_1 and φ_2, both non-erasing on $\mathcal{X}$, such that $\sigma\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset \sigma\vartheta_2\varphi_2$. But then $\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta_2\varphi_2$, which contradicts almost disjointness of $\{\vartheta_1, \vartheta_2\}$.

W2: We have $Eq = \{f(\overline{x}, s_1, \dots, s_n) \approx^?_\emptyset f(\overline{y}, t_1, \dots, t_m)\}$ and $\sigma = \{\overline{y} \mapsto \ulcorner\overline{x}, \overline{y}\urcorner\}$. Let ϑ_1 and ϑ_2 be two substitutions such that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\cup_{v\in\mathcal{X}} \mathcal{V}(v\sigma) = \mathcal{X}$ and $\cup_{\overline{f}\in\mathcal{Q}} \mathcal{F}_S(\overline{f}\sigma) = \mathcal{Q}$ and assume by contradiction that $\{\sigma\vartheta_1, \sigma\vartheta_2\}$ is not. Then there exist φ_1 and φ_2, both non-erasing on $\mathcal{X}$, such that $\sigma\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset \sigma\vartheta_2\varphi_2$. This implies, in particular, that $\overline{x}\vartheta_1\varphi_1 = \overline{x}\vartheta_2\varphi_2$ (since $\overline{x}\sigma = \overline{x}$) and $\overline{y}\vartheta_1\varphi_1 = \overline{y}\vartheta_2\varphi_2$ (following from the equalities $\overline{y}\sigma\vartheta_1\varphi_1 = \overline{y}\sigma_2\vartheta_2\varphi_2$, $\overline{y}\sigma\vartheta_1\varphi_1 = \ulcorner\overline{x}\vartheta_1\varphi_1, \overline{y}\vartheta_1\varphi_1\urcorner$, $\overline{y}\sigma_2\vartheta_2\varphi_2 = \ulcorner\overline{x}\vartheta_2\varphi_2, \overline{y}\vartheta_2\varphi_2\urcorner$, and $\overline{x}\vartheta_1\varphi_1 = \overline{x}\vartheta_2\varphi_2$). But then $\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset \vartheta_2\varphi_2$, which contradicts almost disjointness of $\{\vartheta_1, \vartheta_2\}$.

Sp: We have $Eq = \{f(\overline{x}, s_1, \dots, s_n) \approx^?_\emptyset f(\overline{f_1}(r_1, \dots, r_k), t_1, \dots, t_m)\}$, $\overline{x} \notin \mathcal{V}_S(\overline{f_1}(r_1, \dots, r_k))$, and $\sigma = \{\overline{x} \mapsto \overline{f_1}(r_1, \dots, r_k)\}\{\overline{f} \mapsto \ulcorner\overline{f_1}, \overline{f_2}\urcorner\}$. Let ϑ_1 and ϑ_2 be two substitutions such that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\cup_{v\in\mathcal{X}} \mathcal{V}(v\sigma) = \mathcal{X} \setminus \{\overline{x}\}$ and $\cup_{\overline{f}\in\mathcal{Q}} \mathcal{F}_S(\overline{f}\sigma) = \mathcal{Q} \cup \{\overline{f_1}, \overline{f_2}\} \setminus \{\overline{f}\}$. Assume by contradiction that $\{\sigma\vartheta_1, \sigma\vartheta_2\}$ is not almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Then there exist φ_1 and φ_2, both non-erasing on $\mathcal{X}$, such that $\sigma\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset \sigma\vartheta_2\varphi_2$. But then $\vartheta_1\varphi_1 =^{\mathcal{X}',\mathcal{Q}'}_\emptyset \vartheta_2\varphi_2$, where $\mathcal{X}' = \mathcal{X} \setminus \{\overline{x}\}$ and $\mathcal{Q}' = \mathcal{Q} \cup \{\overline{f_1}, \overline{f_2}\} \setminus \{\overline{f}\}$, which contradicts the fact that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\mathcal{X} \setminus \{\overline{x}\}$ and $\mathcal{Q} \cup \{\overline{f_1}, \overline{f_2}\} \setminus \{\overline{f}\}$. □

Lemmas 56 and 57 imply that $\mathcal{S}ub(\Gamma, Eq)$ preserves almost disjointness:

Lemma 58. *Let Γ be a unification problem and Eq be an equation in Γ. Then every substitution in $\mathcal{S}ub(\Gamma, Eq)$ is almost-disjointness preserving with respect to $\mathcal{V}(\Gamma)$ and $\mathcal{F}_S(\Gamma)$.*

We need one more technical result:

Lemma 59. *Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1, \sigma_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$ such that $\mathcal{R}an(\sigma_1) \subseteq \mathcal{X}$ and $\mathcal{R}an(\sigma_2) \subseteq \mathcal{X}$. Let ϑ_1 and ϑ_2 be two non-erasing substitutions on $\mathcal{X}$ with $\mathcal{R}an(\vartheta_1) \subseteq \mathcal{X}$ and $\mathcal{R}an(\vartheta_2) \subseteq \mathcal{X}$. Then $\{\sigma_1\vartheta_1, \sigma_2\vartheta_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. Assume by contradiction that $\{\sigma_1\vartheta_1, \sigma_2\vartheta_2\}$ is not almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Then there exist substitutions φ_1 and φ_2, both non-erasing on $\mathcal{X}$, such that $\sigma_1\vartheta_1\varphi_1 =^{\mathcal{X},\mathcal{Q}}_\emptyset$

$\sigma_2\vartheta_2\varphi_2$. But this contradicts the fact that $\{\sigma_1, \sigma_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$, because $\vartheta_1\varphi_1$ and $\vartheta_2\varphi_2$ are non-erasing on $\mathcal{X}$. $\square$

Now we can prove the almost disjointness theorem:

Theorem 60 (*Almost Disjointness*). *Let Γ be a unification problem and let $\mathcal{X} = \mathcal{V}(\Gamma)$ and $\mathcal{Q} = \mathcal{F}_S(\Gamma)$. Then $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$.*

Proof. Let σ_1, σ_2 be two substitutions from $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ and let

$$\begin{aligned}
&\langle\Gamma, \varepsilon\rangle \Longrightarrow \langle\Gamma_1, \gamma_1\rangle \Longrightarrow \cdots \Longrightarrow \langle\Gamma_n, \gamma_1\cdots\gamma_n\rangle \Longrightarrow \\
&\qquad \Longrightarrow \langle\Delta_1, \delta_1\rangle \Longrightarrow \cdots \Longrightarrow \langle\Delta_m, \delta_m\rangle \Longrightarrow \langle\emptyset, \sigma_1\rangle, \\
&\langle\Gamma, \varepsilon\rangle \Longrightarrow \langle\Gamma_1, \gamma_1\rangle \Longrightarrow \cdots \Longrightarrow \langle\Gamma_n, \gamma_1\cdots\gamma_n\rangle \Longrightarrow \\
&\qquad \Longrightarrow \langle\Phi_1, \varphi_1\rangle \Longrightarrow \cdots \Longrightarrow \langle\Phi_k, \varphi_k\rangle \Longrightarrow \langle\emptyset, \sigma_2\rangle
\end{aligned}$$

be two derivations in $\mathfrak{U}$, leading respectively to $\langle\emptyset, \sigma_1\rangle$ and $\langle\emptyset, \sigma_2\rangle$ and having the common initial part $\langle\Gamma, \varepsilon\rangle \Longrightarrow \langle\Gamma_1, \gamma_1\rangle \Longrightarrow \cdots \Longrightarrow \langle\Gamma_n, \gamma_1\cdots\gamma_n\rangle$. Let $\gamma_0 = \varepsilon$. We prove the theorem in three steps.

Step 1. In the first step we show that for any set of substitutions $\{\vartheta_1, \vartheta_2\}$ that is almost disjoint with respect to $\mathcal{V}(\Gamma_n)$ and $\mathcal{F}_S(\Gamma_n)$, the set $\{\gamma_0\gamma_1\cdots\gamma_n\vartheta_1, \gamma_0\gamma_1\cdots\gamma_n\vartheta_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. We use induction on n. For $n = 0$ the claim is trivial. As the induction hypothesis, assume that for any set of substitutions $\{\vartheta_1, \vartheta_2\}$ that is almost disjoint with respect to $\mathcal{V}(\Gamma_{n_0})$ and $\mathcal{F}_S(\Gamma_{n_0})$, $n_0 \geq 0$, the set $\{\gamma_0\gamma_1\cdots\gamma_{n_0}\vartheta_1, \gamma_0\gamma_1\cdots\gamma_{n_0}\vartheta_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Now assume that $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\mathcal{V}(\Gamma_{n_0+1})$ and $\mathcal{F}_S(\Gamma_{n_0+1})$. Since $\mathcal{V}(\Gamma_{n_0+1}) \subseteq \mathcal{V}(\Gamma_{n_0}\gamma_{n_0+1}) = \cup_{v\in\mathcal{V}(\Gamma_{n_0})}\mathcal{V}(v\gamma_{n_0+1})$ and $\mathcal{F}_S(\Gamma_{n_0+1}) \subseteq \mathcal{F}_S(\Gamma_{n_0}\gamma_{n_0+1}) = \cup_{\overline{f}\in\mathcal{F}_S(\Gamma_{n_0})}\mathcal{F}_S(\overline{f}\gamma_{n_0+1})$, we, in fact, have that the set $\{\vartheta_1, \vartheta_2\}$ is almost disjoint with respect to $\cup_{v\in\mathcal{V}(\Gamma_{n_0})}\mathcal{V}(v\gamma_{n_0+1})$ and $\cup_{\overline{f}\in\mathcal{F}_S(\Gamma_{n_0})}\mathcal{F}_S(\overline{f}\gamma_{n_0+1})$. By Lemma 58, the substitution γ_{n_0+1} is almost-disjointness preserving with respect to $\mathcal{V}(\Gamma_{n_0})$ and $\mathcal{F}_S(\Gamma_{n_0})$, because $\gamma_{n_0+1} \in \mathcal{S}ub(\Gamma_{n_0}, Eq)$ for some $Eq \in \Gamma_{n_0}$. Therefore, by Definition 12, $\{\gamma_{n_0+1}\vartheta_1, \gamma_{n_0+1}\vartheta_2\}$ is almost disjoint with respect to $\mathcal{V}(\Gamma_{n_0})$ and $\mathcal{F}_S(\Gamma_{n_0})$. Instantiating the induction hypothesis with $\{\gamma_{n_0+1}\vartheta_1, \gamma_{n_0+1}\vartheta_2\}$, we obtain that $\{\gamma_0\gamma_1\cdots\gamma_{n_0}\gamma_{n_0+1}\vartheta_1, \gamma_0\gamma_1\cdots\gamma_{n_0}\gamma_{n_0+1}\vartheta_2\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. This proves the first step.

Step 2. Now we will show that the set $\{\delta_1\cdots\delta_m, \varphi_1\cdots\varphi_k\}$ is almost disjoint with respect to $\mathcal{V}(\Gamma_n)$ and $\mathcal{F}_S(\Gamma_n)$. The substitutions δ_1 and φ_1 belong to $\mathcal{S}ub(\Gamma_n, Eq)$ for some $Eq \in \Gamma_n$ and, by Lemma 55, the set $\{\delta_1, \varphi_1\}$ is almost disjoint with respect to $\mathcal{V}(\Gamma_n)$ and $\mathcal{F}_S(\Gamma_n)$. The substitutions $\delta_2\cdots\delta_m$ and $\varphi_2\cdots\varphi_k$ are non-erasing on $\mathcal{V}(\Gamma_n)$. Moreover, $\mathcal{R}an(\delta_1) \subseteq \mathcal{V}(\Gamma_n)$, $\mathcal{R}an(\varphi_1) \subseteq \mathcal{V}(\Gamma_n)$, $\mathcal{R}an(\delta_2\cdots\delta_m) \subseteq \mathcal{V}(\Gamma_n)$, and $\mathcal{R}an(\varphi_2\cdots\varphi_k) \subseteq \mathcal{V}(\Gamma_n)$. Therefore, by Lemma 59, the set $\{\delta_1\cdots\delta_m, \varphi_1\cdots\varphi_k\}$ is almost disjoint with respect to the sets $\mathcal{V}(\Gamma_n)$ and $\mathcal{F}_S(\Gamma_n)$.

Step 3. From the previous two steps we conclude that the set $\{\gamma_0\cdots\gamma_n\delta_1\cdots\delta_m, \gamma_0\cdots\gamma_n\varphi_1\cdots\varphi_k\}$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Since $\sigma_1 = \gamma_0\cdots\gamma_n\delta_1\cdots\delta_m$ and $\sigma_2 = \gamma_0\cdots\gamma_n\varphi_1\cdots\varphi_k$, we obtain that $\{\sigma_1, \sigma_2\}$ (and hence an arbitrary two-element subset of $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$) is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. Therefore, by Definition 10, $\mathcal{S}ol_{\mathfrak{U}}(\Gamma)$ is almost disjoint with respect to $\mathcal{X}$ and $\mathcal{Q}$. $\square$

Theorems 51 and 60, and Proposition 11 imply the main result about general syntactic sequence unification:

Theorem 61 (*Main Theorem*). *Let Γ be a general syntactic sequence unification problem. Then $\mathcal{S}ol_{\mathfrak{U}}(\Gamma) = amcu_{\emptyset}(\Gamma)$.*

The Main Theorem implies that every general syntactic sequence unification problem has an almost minimal complete set of unifiers. Then, by Proposition 17, every such problem has a minimal complete set of unifiers. Since for some problems (e.g., for $\{f(a, \overline{x}) \approx^?_\emptyset f(\overline{x}, a)\}$) this set is infinite, we obtain the following result about the unification type:

Theorem 62 (*Unification Type*). *The unification type for general syntactic sequence unification is infinitary.*

5. "Lighter" version of the unification procedure

Application of the decision algorithm $\mathfrak{D}$ can be a costly operation. The DS step in $\mathfrak{D}$ invokes the NP-hard decidability test for word equations with linear constant restrictions. Below we describe a "lighter" version of the unification procedure that does not call the decision algorithm. It uses the inference system $\mathfrak{I}$ extended with rules to detect some (but not all) failing cases. We denote the extended inference system by $\mathfrak{I}_{\mathrm{Ext}}$. The rules that detect failure are the following ones:

IVOC: **Individual Variable Occurrence Check**

$\langle\{x \approx^?_\emptyset t\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp,\qquad$ if $x \neq t$ and $x \in \mathcal{V}_I(t)$.

SC1: **Symbol Clash 1**

$\langle\{f(s_1, \ldots, s_n) \approx^?_\emptyset g(t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp,\qquad$ if$f \neq g$.

SC2: **Symbol Clash 2**

$\langle\{f(s, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp,$

if $\mathcal{H}ead(s), \mathcal{H}ead(t) \in \mathcal{F}_S$ and $\mathcal{H}ead(s) \neq \mathcal{H}ead(t)$.

AD: **Arity Disagreement**

$\langle\{f(s_1, \ldots, s_n) \approx^?_\emptyset f(t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp,$

if $n \neq m$ and $s_i \notin \mathcal{V}_S$ and $t_j \notin \mathcal{V}_S$ for all $1 \leq i \leq n$ and for all $1 \leq j \leq m$.

E1: **Empty 1**

$\langle\{f() \approx^?_\emptyset f(t, t_1, \ldots, t_n)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp.$

E2: **Empty 2**

$\langle\{f(s, s_1, \ldots, s_n) \approx^?_\emptyset f()\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp.$

SVOC: **Sequence Variable Occurrence Check**

$\langle\{f(\overline{x}, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp,$
if $\overline{x} \neq t$ and $\overline{x} \in \mathcal{V}_S(t)$.

PF: **Prefix Failure**

$\langle\{f(s, s_1, \ldots, s_n) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\} \cup \Gamma';\ \sigma\rangle \Longrightarrow \perp,$

if $s \in \mathcal{T}_S^-(\mathcal{F}, \mathcal{V})$ and $t \in \mathcal{T}_I(\mathcal{F}, \mathcal{V})$, or $s \in \mathcal{T}_I(\mathcal{F}, \mathcal{V})$ and $t \in \mathcal{T}_S^-(\mathcal{F}, \mathcal{V})$, where $\mathcal{T}_S^-(\mathcal{F}, \mathcal{V}) = \mathcal{T}_S(\mathcal{F}, \mathcal{V}) \setminus \mathcal{V}_S$.
Hence, $\mathfrak{I}_{\mathrm{Ext}} = \mathfrak{I} \cup \{\mathsf{IVOC}, \mathsf{SC1}, \mathsf{SC2}, \mathsf{AD}, \mathsf{E1}, \mathsf{E2}, \mathsf{SVOC}, \mathsf{PF}\}$.

One way of refining the unification procedure $\mathfrak{U}$ (see Definition 38) is to use $\mathfrak{I}_{\mathrm{Ext}}$ instead of $\mathfrak{I}$ and to retain the decision algorithm $\mathfrak{D}$. In this case instead of immediately applying $\mathfrak{D}$ on a unification problem in a unification tree first the failure detection rules of $\mathfrak{I}_{\mathrm{Ext}}$ are tried. If they cannot detect failure, then the decision algorithm is used to decide whether the problem is solvable or not. In this way we can tailor the failure detection rules in $\mathfrak{U}$ as a pre-filter before applying the costly decision algorithm.

Another way is to use $\mathfrak{I}_{\mathrm{Ext}}$ instead of $\mathfrak{I}$ and omit $\mathfrak{D}$ completely. We call the unification procedure obtained from $\mathfrak{U}$ in this way the "light" unification procedure and denote it by $\mathfrak{U}_{\mathrm{Light}}$. Obviously, soundness and completeness theorems hold for $\mathfrak{U}_{\mathrm{Light}}$ as well. However, there are cases when $\mathfrak{U}$ stops with failure, but $\mathfrak{U}_{\mathrm{Light}}$ can go on forever. This is because the failure rules in $\mathfrak{I}_{\mathrm{Ext}}$ do not detect all failing cases, even if a fair selection strategy is used. For instance, none of them apply to an unsolvable unification problem $\{f(\overline{x}) \approx^?_{\emptyset} f(a, \overline{x})\}$.

6. Termination without the decision algorithm

In this section we consider three special cases when omitting the application of the decision algorithm $\mathfrak{D}$ does not lead to nontermination.

6.1. Equations in unification problems have at least one ground side

Unification procedure $\mathfrak{U}_{\mathrm{Light}}$ terminates if equations in the unification problem have at least one ground side. It can be proved by showing that every rule in the inference system $\mathfrak{I}_{\mathrm{Ext}}$ strictly decreases a complexity measure, a 5-tuple of natural numbers $\langle n_1, n_2, n_3, n_4, n_5\rangle$, associated with a system $\langle \Delta, \sigma\rangle$ where:

n_1 = the number of distinct variables in Δ;
n_2 = the total number of symbols in the ground sides of equations in Δ;
n_3 = the number of subterms in Δ of the form $f(s_1, \ldots, s_n)$, where s_1 is not a sequence term;
n_4 = the number of equations in Δ of the form $t \approx^?_{\emptyset} x$, where t is not an individual variable;
n_5 = the number of equations in Δ that have the form $f(s, s_1, \ldots, s_n) \approx^?_{\emptyset} f(\overline{x}, t_1, \ldots, t_m)$, where s is not a sequence variable,

and measures are compared lexicographically.

This result, in particular, implies that general syntactic matching with sequence variables and sequence function symbols in finitary.

6.2. Unification problems with linear shallow sequence variables

Unification problems with linear shallow sequence variables are problems where every sequence variable occurs only once and the occurrence happens at the top level, like, for instance, in $\{f(\overline{x_1}, x, \overline{y_1}) \approx^?_{\emptyset} f(g(x), g(h(y))),\ f(\overline{x_2}, x) \approx^?_{\emptyset} f(\overline{y_2}, g(h(a)))\}$. The problems like $\{f(a, \overline{x}) \approx^?_{\emptyset} x,\ f(\overline{x}, a) \approx^?_{\emptyset} x\}$ or $\{f(x, x) \approx^?_{\emptyset} f(g(a, \overline{x}), g(\overline{x}, a))\}$ do not fall into this class. Although the restriction might look too strong, it is common in formalizing and implementing sequent calculi (Paulson, 1990).

Termination of $\mathfrak{U}_{\mathrm{Light}}$ for unification problems with linear shallow sequence variables is not hard to establish. We can consider a complexity measure for a system $\langle \Delta, \sigma\rangle$, a 6-tuple of natural numbers $\langle n_1, n_2, n_3, n_4, n_5, n_6\rangle$, where

n_1 = the number of distinct variables in Δ;

$n_2 =$ the number of symbols in Δ;
$n_3 =$ the number of occurrences of sequence function symbols in Δ;
$n_4 =$ the number of subterms in Δ of the form $f(s_1, \ldots, s_n)$, where s_1 is not a sequence term;
$n_5 =$ the number of equations in Δ of the form $t \approx^?_\emptyset x$, where t is not an individual variable;
$n_6 =$ the number of equations in Δ that have the form $f(s, s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{x}, t_1, \ldots, t_m)$, where s is not a sequence variable.

Measures are compared lexicographically. It is easy to show that each rule in the inference system $\mathfrak{I}_{\mathrm{Ext}}$ strictly decreases the measure, and it implies termination.

6.3. Sequence variables occur only in the last argument positions in terms

This is another interesting case. As it turns out, it makes unification unitary and application of the decision algorithm obsolete.

We start with modifying the inference system. First, we introduce rules that take into account the occurrence restriction for sequence variables. These are the following:

E1m: **Empty 1, modified**

$\langle\{f() \approx^?_\emptyset f(t, t_1, \ldots, t_n)\};\ \sigma\rangle \Longrightarrow \bot, \qquad$ if $t \notin \mathcal{V}_S$.

E2m: **Empty 2, modified**

$\langle\{f(s, s_1, \ldots, s_n) \approx^?_\emptyset f()\};\ \sigma\rangle \Longrightarrow \bot, \qquad$ if $s \notin \mathcal{V}_S$.

SVOCm: **Sequence Variable Occurrence Check, modified**

$\langle\{f(\overline{x}) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\};\ \sigma\rangle \Longrightarrow \bot,$
if $\overline{x} \neq t$ and $\overline{x} \in \mathcal{V}_S(t, t_1, \ldots, t_m)$.

SVD1: **Sequence Variable Deletion 1**

$\langle\{f(\overline{x}) \approx^?_\emptyset f()\};\ \sigma\rangle \Longrightarrow \langle\emptyset;\ \sigma\vartheta\rangle, \qquad$ where $\vartheta = \{\overline{x} \mapsto \ulcorner\urcorner\}$.

SVD2: **Sequence Variable Deletion 2**

$\langle\{f() \approx^?_\emptyset f(\overline{x})\};\ \sigma\rangle \Longrightarrow \langle\emptyset;\ \sigma\vartheta\rangle, \qquad$ where $\vartheta = \{\overline{x} \mapsto \ulcorner\urcorner\}$.

SVEm: **Sequence Variable Elimination, modified**

$\langle\{f(\overline{x}) \approx^?_\emptyset f(t, t_1, \ldots, t_m)\};\ \sigma\rangle \Longrightarrow \langle\emptyset;\ \sigma\vartheta\rangle,$
if $\overline{x} \notin \mathcal{V}_S(t, t_1, \ldots, t_m)$ and $\vartheta = \{\overline{x} \mapsto \ulcorner t, t_1, \ldots, t_m\urcorner\}$.
We define the modified inference system $\mathfrak{I}_{\mathrm{Mod}}$ as the set of inference rules:

$$\mathfrak{I}_{\mathrm{Mod}} = \{\mathsf{T}, \mathsf{O1}, \mathsf{O2}, \mathsf{S}, \mathsf{TD}, \mathsf{PD1}, \mathsf{PD2}, \mathsf{IVOC}, \mathsf{SC1}, \mathsf{SC2}, \mathsf{AD}, \mathsf{PF}, \\ \mathsf{E1m}, \mathsf{E2m}, \mathsf{SVOCm}, \mathsf{SD1}, \mathsf{SD2}, \mathsf{SVEm}\}.$$

The modified unification procedure $\mathfrak{U}_{\mathrm{Mod}}$ is obtained from $\mathfrak{U}_{\mathrm{Light}}$ by replacing the inference system $\mathfrak{I}_{\mathrm{Light}}$ by $\mathfrak{I}_{\mathrm{Mod}}$.

Remark 63. Note that only one rule in $\mathfrak{I}_{\mathrm{Mod}}$ is applicable to a selected equation. No rules can be applied on $\bot$ and $\langle\emptyset;\ \sigma\rangle$. Hence, if Γ is unifiable, then the solution set $\mathcal{S}ol_{\mathfrak{U}_{\mathrm{Mod}}}(\Gamma)$ generated by $\mathfrak{U}_{\mathrm{Mod}}$ is a singleton.

Termination of any transformation sequence in $\mathfrak{I}_{\mathrm{Mod}}$ can be shown in the standard way. First, we define a complexity measure for a system $\langle \Delta, \sigma \rangle$ as a 6-tuple of natural numbers $\langle n_1, n_2, n_3, n_4, n_5, n_6 \rangle$, where

n_1 = the number of distinct variables in Δ;
n_2 = the number of symbols in Δ;
n_3 = the number of occurrences of sequence function symbols in Δ;
n_4 = the number of subterms in Δ of the form $f(s_1, \ldots, s_n)$, where s_1 is not a sequence term;
n_5 = the number of equations in Δ of the form $t \approx^?_\emptyset x$, where t is not an individual variable;
n_6 = the number of equations in Δ of the form $f(s_1, \ldots, s_n) \approx^?_\emptyset f(\overline{x})$, where $n \geq 0$ and s_1 is not a sequence variable.

Measures are compared lexicographically. Then, we prove that each rule in $\mathfrak{I}_{\mathrm{Mod}}$ reduces the complexity measure. This is pretty straightforward.

Two other important properties of $\mathfrak{U}_{\mathrm{Mod}}$, soundness and completeness, can be formulated and proved similarly to the soundness and completeness theorems for $\mathfrak{U}$. We do not give details of the proofs here. Rather, we point out that completeness of $\mathfrak{U}_{\mathrm{Mod}}$, together with the fact that the solution set $\mathfrak{U}_{\mathrm{Mod}}(\Gamma)$ is a singleton for an unifiable Γ, implies that $\mathfrak{U}_{\mathrm{Mod}}$ calculates a most general unifier for unification problems where sequence variables occur as the last arguments. The unification type for this case is unitary, since any unifiable problem has a most general unifier.

7. Implementation

We implemented the "light" unification procedure in *Mathematica* on the basis of a rule-based programming system *ρLog*[4] (Marin and Kutsia, 2003). A rule in *ρLog* is a specification of a nondeterministic and partially defined computation. The system has primitive operators for defining elementary rules and for computing with unions, compositions, reflexive–transitive closures, rewriting, and normal forms of rules. With these tools the "light" syntactic sequence unification procedure (with bounded depth) was implemented quite easily. Within the bounded depth, the procedure, by default, uses the depth-first search method with backtracking provided with *ρLog*. Options allow the user to modify the depth bound, use iterative deepening instead of the depth-first method, and stop computation after obtaining a certain number of solutions.

8. Relation with order-sorted higher-order unification

Syntactic sequence unification can be considered as a special case of order-sorted higher-order E-unification. Here we show the corresponding encoding in the framework described in Kohlhase (1994).

We consider simply typed λ-calculus with the types i and o. The set of base sorts consists of `ind`, `seq`, `seqc`, `o` such that the type of `o` is o and the type of the other sorts is i. Individual and sequence variables are treated as first-order variables, while sequence function symbols are encoded as second-order variables. We define a context C (a function that assigns sorts to

[4] Available from http://www.score.is.tsukuba.ac.jp/~mmarin/RhoLog/.

variables) such that $C(x) = \mathtt{ind}$ for all $x \in \mathcal{V}_I$, $C(\overline{x}) = \mathtt{seq}$ for all $\overline{x} \in \mathcal{V}_S$, $C(\overline{f}) = \mathtt{seq} \to \mathtt{seqc}$ for each $\overline{f} \in \mathcal{F}lex_S$, and $C(\overline{f}) = \mathtt{ind} \to \cdots \to \mathtt{ind} \to \mathtt{seqc}$ (with n arrows) for each $\overline{f} \in \mathcal{F}ix_S$ with $\mathcal{A}r(f) = n$. Individual function symbols are treated as constants. We assign to each $f \in \mathcal{F}lex_I$ a functional sort $\mathtt{seq} \to \mathtt{ind}$ and to each $f \in \mathcal{F}ix_I$ with $\mathcal{A}r(f) = n$ a functional sort $\mathtt{ind} \to \cdots \to \mathtt{ind} \to \mathtt{ind}$ (with n arrows). We assume equality constants $\approx_{\mathtt{s}}$ for every sort $\mathtt{s}$. In addition, we have two function symbols: binary $\ulcorner\urcorner$ of the sort $\mathtt{seq} \to \mathtt{seq} \to \mathtt{seq}$ and a constant $[\,]$ of the sort $\mathtt{seq}$. Sorts are partially ordered as $\mathtt{ind} \leq \mathtt{seqc}$ and $\mathtt{seqc} \leq \mathtt{seq}$. The equational theory is an AU-theory, asserting associativity of $\ulcorner\urcorner$ with $[\,]$ as left and right unit. We consider unification problems for terms of the sort $\mathtt{ind}$, where terms are in $\beta\eta$-normal form containing no bound variables, and terms whose head is $\ulcorner\urcorner$ are flattened. For a given unification problem in this theory, we are looking for unifiers that obey the following restrictions: If a unifier σ binds a second-order variable $\overline{f}$ of the sort $\mathtt{seq} \to \mathtt{seqc}$, then $\overline{f}\sigma = \lambda\overline{x}.\ulcorner\overline{g_1}(\overline{x}), \ldots, \overline{g_m}(\overline{x})\urcorner$. If σ binds a second-order variable $\overline{f}$ of the sort $\mathtt{ind} \to \cdots \to \mathtt{ind} \to \mathtt{seqc}$ (with n arrows), then $\overline{f}\sigma = \lambda x_1.\ldots x_n.\ulcorner\overline{g_1}(x_1, \ldots, x_n), \ldots, \overline{g_m}(x_1, \ldots, x_n)\urcorner$. In both cases $m > 1$ and $\overline{g_1}, \ldots, \overline{g_m}$ are fresh variables of the same sort as $\overline{f}$.

Hence, syntactic sequence unification can be considered as order-sorted second-order AU-unification with additional restrictions. Order-sorted higher-order syntactic unification was investigated by Kohlhase (1994), but we are not aware of any work done on order-sorted higher-order equational unification.

9. Related work

Solving equations involving sequence variables has applications in various fields, like artificial intelligence, knowledge management, programming, rewriting, program schemata, XML processing, and theorem proving. In this section we briefly review just some of the methods related to our work. Note that in the literature flexible arity symbols are also called "variadic", "polyadic", "variable arity", "varying-arity", or "multiple arity" symbols.

Sequence variables are part of Common Logic (Common Logic Working Group, 2003) and SCL, a simplified version of Common Logic. These are languages designed for use in the interchange of knowledge among disparate computer systems. Moreover, sequence variables occur in a "concrete" instance of the Common Logic language, called Knowledge Interchange Format, KIF (Genesereth et al., 1998) and in a simplified version of KIF, called SKIF (Hayes and Menzel, 2001). In SKIF, sequence variables are called row variables. Hayes and Menzel (2001) point out that unification with row variables is very difficult, because two expressions can have infinitely many most general unifiers. They also remark that allowing row variables only in the last argument positions guarantees that unification patterns that create the difficulties with infinitely many most general unifiers cannot arise. Unification procedures for these languages are not discussed.

Probably the first attempt to design and implement unification with sequence variables (without sequence functions) was made in the MVL system (Ginsberg, 1991). The implementation of unification was incomplete because of restricted use of the widening technique. The restriction was imposed intentionally, for efficiency reasons. No theoretical study of the unification algorithm of MVL, to the best of our knowledge, was undertaken.

Word equations (Siekmann, 1975; Abdulrab and Pécuchet, 1990; Jaffar, 1990; Schulz, 1993) and associative unification (Plotkin, 1972) can be modeled by syntactic sequence unification using constants, sequence variables, and one flexible arity function symbol. In a similar way we can imitate unification for path logics closed under right identity and associativity (Schmidt, 1998).

Equations for which the length of the values of sequence variables is bounded have finitely many solutions. This fact is used in, for instance, *Prolog* III (Colmerauer, 1990) and in transformation schemata for Prolog programs (Richardson and Fuchs, 1997). In *Prolog* III, a restricted form of word unification is incorporated for reasoning with lists with several subparts of unknown length. Solving such equations is delayed until the length of those subparts becomes known. In Richardson and Fuchs (1997), a unification algorithm with vector variables is described. Vector variables are similar to sequence variables, but come with their possible length attached, which makes unification finitary. The algorithm was implemented and used for schema-based logic program transformation, but its properties have never been investigated.

Extensions of logic and functional programming, integrated in the *RelFun* system (Boley, 1999), permit sequence variables in the last argument positions of flexible arity symbols. Unification for such terms is unitary. *RelFun* allows multiple-valued functions as well.

Implementation of first-order logic in *Isabelle* (Paulson, 1990) is based on sequent calculus formulated using sequence variables (on the meta-level). Sequence meta-variables are used to denote sequences of formulae and individual meta-variables denote single formulae. Since in every such unification problem no sequence meta-variable occurs more than once and all of them occur only on the top level, *Isabelle*, in fact, deals with a finitary case of sequence unification.

The *Set-Var* prover (Bledsoe and Feng, 1993) has a construct called the vector of (Skolem) functions that resembles our sequence functions. For instance, a vector of functions denoted by $\overline{g}(a, \overline{s})$, where $\overline{s}$ is a vector of variables, abbreviates a sequence of functions $g_1(a, \overline{s}), \ldots, g_m(a, \overline{s})$. However, splitting vectors of functions between variables is not allowed in unification: such a vector of functions either entirely unifies with a variable or with another vector of functions.

The programming language of *Mathematica* has a built-in pattern matching mechanism, which supports sequence variables (represented as identifiers with “triple blanks”, e.g., `x___`) and flexible arity function symbols. The behavior of the matching algorithm is explained in examples in the *Mathematica* book (Wolfram, 2003). Our procedure (without sequence function symbols) can imitate this behavior. For a given matching problem, the output of the procedure would be identical to the set of all possible matchers *Mathematica* matching algorithm computes. On the other hand, when *Mathematica* tries to match patterns to some expression, it tries first those matchers that assign the shortest sequences of arguments to the first triple blanks that appear in the pattern and returns the first matcher it finds. We can simulate this behavior also, imposing an order of choosing successors in the Projection rule, applying Sequence Variable Elimination 2 before Widening 1, and stopping the procedure whenever the first solution appears. In exactly the same way we can model the minimal sequence matching algorithm described in (Hamana, 1997), where it was used to define rewriting with sequences and study rewriting semantics of *Mathematica/R* (Buchberger, 1996). *Mathematica/R* is the rewriting part of the *Mathematica* programming language.

Marin and Ţepeneu (2003) provided a more advanced mechanism for controlling pattern matching with sequence variables in *Mathematica*. Their package *Sequentica* allows users to specify the sequence variable instantiation order, and the lengths of term sequences sequence variables can be instantiated with.

Hamada and Ida (1997) extended *Mathematica* symbolic computation capabilities with higher-order lazy narrowing calculi. The extension itself did not involve sequence variables, but the authors indicated that properly used sequence variables enhance clarity of programs and emphasized the need for clear semantics of sequence variables.

Coelho and Florido (2004) developed a constraint logic programming language CLP(Flex) over the domain of terms with sequence variables and flexible arity symbols. Constraint solving in CLP(Flex) is based on a version of our unification procedure without sequence functions (Kutsia, 2002b). CLP(Flex) is applied to XML processing, where XML documents are abstracted by terms with flexible arity symbols. It gives a highly declarative model for XML processing yielding a substantial degree of flexibility in programming.

Buchberger introduced sequence variables and sequence functions in the *Theorema* system (Buchberger et al., 2000). Kutsia and Buchberger (2004) studied the meta-mathematical implications of introducing sequence variables in predicate logic. The equational prover of *Theorema* (Kutsia, 2003) supports proving by unfailing completion for unit equalities with sequence variables in the last argument positions and proving by rewriting with unrestricted occurrences of sequence variables. The unification procedure implemented in the prover follows the procedure $\mathfrak{U}_{\mathrm{Mod}}$ described in this paper.

10. Conclusions

We proved that general unification in the free theory with individual and sequence variables and function symbols is decidable and has the infinitary type. We developed a rule-based unification procedure and proved its soundness, completeness, and almost minimality. The procedure uses the decision algorithm to cut failing branches in the unification tree. A “lighter” version of the procedure replaces the decision algorithm with extra rules for detecting failure. It is still sound and complete, easier to implement, but for some failing cases might not terminate. We also showed a relation between general syntactic sequence unification and order-sorted higher-order equational unification.

Under certain restrictions sequence unification problems have finitely many solutions: sequence variables in the last argument positions, unification problems with at least one ground side (matching as an instance), all sequence variables on the top level with maximum one occurrence. It would be interesting to identify more cases with finite or finitely representable solution sets.

Acknowledgements

I thank Bruno Buchberger who introduced me to the subject, Mircea Marin for interesting discussions, Manuel Kauers for useful LaTeX macros, and the anonymous referees for helpful comments.

References

Abdulrab, H., Pécuchet, J.-P., 1990. Solving word equations. J. Symbolic Comput. 8 (5), 499–522.

Baader, F., Schulz, K.U., 1991. General A- and AX-unification via optimized combination procedures. In: Abdulrab, H., Pécuchet, J.-P. (Eds.), Proc. of the 2nd Int. Workshop on Word Equations and Related Topics. In: LNCS, vol. 677. Springer, pp. 23–42.

Baader, F., Schulz, K.U., 1996. Unification in the union of disjoint equational theories: Combining decision procedures. J. Symbolic Comput. 21 (2), 211–244.

Baader, F., Snyder, W., 2001. Unification theory. In: Robinson, A., Voronkov, A. (Eds.), Handbook of Automated Reasoning., vol. I. Elsevier Science, pp. 445–532 (Chapter 8).

Bledsoe, W.W., Feng, G., 1993. Set-Var. J. Automat. Reason. 11 (3), 293–314.

Boley, H., 1999. A Tight, Practical Integration of Relations and Functions. In: LNAI, vol. 1712. Springer.

Buchberger, B., 1996. *Mathematica* as a rewrite language. In: Ida, T., Ohori, A., Takeichi, M. (Eds.), Proc. of the 2nd Fuji Int. Workshop on Functional and Logic Programming. World Scientific, Shonan Village Center, Japan, pp. 1–13.

Buchberger, B., Dupré, C., Jebelean, T., Kriftner, F., Nakagawa, K., Vasaru, D., Windsteiger, W., 2000. The *Theorema* project: A progress report. In: Kerber, M., Kohlhase, M. (Eds.), Proc. of Calculemus'2000 Conference. pp. 98–113.

Coelho, J., Florido, M., 2004. CLP(Flex): Constraint logic programming applied to XML processing. In: Meersman, R., Tari, Z. (Eds.), On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE (Proc. of Confederated Int. Conferences). In: LNCS, vol. 3291. Springer, pp. 1098–1112.

Colmerauer, A., 1990. An introduction to *Prolog* III. Commu. ACM 33 (7), 69–91.

Common Logic Working Group, 2003. Common logic: Abstract syntax and semantics. http://cl.tamu.edu/docs/cl/1.0/cl-1.0.pdf.

Genesereth, M.R., Petrie, C., Hinrichs, T., Hondroulis, A., Kassoff, M., Love, N., Mohsin, W., 1998. Knowledge interchange format, draft proposed American national standard (dpANS). Tech. Rep. NCITS.T2/98-004. http://logic.stanford.edu/kif/dpans.html.

Ginsberg, M.L., 1991. The MVL theorem proving system. SIGART Bull. 2 (3), 57–60.

Hamada, M., Ida, T., 1997. Implementation of lazy narrowing calculi in *Mathematica*. Technical Report 97-02, RISC. Johannes Kepler University, Linz, Austria.

Hamana, M., 1997. Term rewriting with sequences. In: Proc. of the First Int. *Theorema* Workshop. Technical report 97–20, RISC. Johannes Kepler University, Linz, Austria.

Hayes, P., Menzel, C., 2001. Semantics of Knowledge Interchange Format, http://reliant.teknowledge.com/IJCAI01/HayesMenzel-SKIF-IJCAI2001.pdf.

Jaffar, J., 1990. Minimal and complete word unification. J. ACM 37 (1), 47–85.

Kohlhase, M., 1994. A mechanization of sorted higher-order logic based on the resolution principle. Ph.D. Thesis. Saarland University, Saarbrücken, Germany.

Kutsia, T., 2002a. Solving and proving in equational theories with sequence variables and flexible arity symbols. Ph.D. Thesis. Johannes Kepler University, Linz, Austria.

Kutsia, T., 2002b. Unification with sequence variables and flexible arity symbols and its extension with pattern-terms. In: Calmet, J., Benhamou, B., Caprotti, O., Henocque, L., Sorge, V. (Eds.), Artificial Intelligence, Automated Reasoning and Symbolic Computation. (Proc. of Joint AISC'2002—Calculemus'2002 Conference). In: LNAI, vol. 2385. Springer, pp. 290–304.

Kutsia, T., 2003. Equational prover of *Theorema*. In: Nieuwenhuis, R. (Ed.), Proc. of the 14th Int. Conference on Rewriting Techniques and Applications. In: LNCS, vol. 2706. Springer, pp. 367–379.

Kutsia, T., 2004. Solving equations involving sequence variables and sequence functions. In: Buchberger, B., Campbell, J.A. (Eds.), Artificial Intelligence and Symbolic Computation (Proc. of AISC'04 Conference). In: LNAI, vol. 3249. Springer, pp. 157–170.

Kutsia, T., Buchberger, B., 2004. Predicate logic with sequence variables and sequence function symbols. In: Asperti, A., Bancerek, G., Trybulec, A. (Eds.), Proc. of the 3rd Int. Conference on Mathematical Knowledge Management. In: LNCS, vol. 3119. Springer, pp. 205–219.

Makanin, G.S., 1977. The problem of solvability of equations in a free semigroup. Math. USSR Sbornik 32 (2), 129–198.

Marin, M., Ţepeneu, D., 2003. Programming with sequence variables: The *Sequentica* package. In: Mitic, P., Ramsden, P., Carne, J. (Eds.), Challenging the Boundaries of Symbolic Computation (Proc. of 5th Int. *Mathematica* Symposium). Imperial College Press, London, pp. 17–24.

Marin, M., Kutsia, T., 2003. On the implementation of a rule-based programming system and some of its applications. In: Konev, B., Schmidt, R. (Eds.), Proc. of the 4th Int. Workshop on the Implementation of Logics. pp. 55–68.

Paulson, L., 1990. *Isabelle*: The next 700 theorem provers. In: Odifreddi, P. (Ed.), Logic and Computer Science. Academic Press, pp. 361–386.

Plotkin, G., 1972. Building in equational theories. In: Meltzer, B., Michie, D. (Eds.), Machine Intelligence, vol. 7. Edinburgh University Press, pp. 73–90.

Richardson, J., Fuchs, N.E., 1997. Development of correct transformation schemata for *Prolog* programs. In: Fuchs, N.E. (Ed.), Proc. of the 7th Int. Workshop on Logic Program Synthesis and Transformation. In: LNCS, vol. 1463. Springer, pp. 263–281.

Robinson, J.A., 1965. A machine-oriented logic based on the resolution principle. J. ACM 12 (1), 23–41.

Schmidt, R., 1998. *E*-Unification for subsystems of S4. In: Nipkow, T. (Ed.), Proc. of the 9th Int. Conference on Rewriting Techniques and Applications. In: LNCS, vol. 1379. Springer, pp. 106–120.

Schulz, K.U., 1993. Word unification and transformation of generalized equations. J. Automated Reasoning 11 (2), 149–184.

Siekmann, J., 1975. String unification. Research paper, Essex University.

Wolfram, S., 2003. The Mathematica Book, 5th edn. Wolfram Media.