Unification in a Free Theory with Sequence Variables and Flexible Arity Symbols and its Extensions*

Temur Kutsia

Research Institute for Symbolic Computation
Johannes Kepler University Linz
A-4040, Linz, Austria
kutsia@risc.uni-linz.ac.at

Software Competence Center Hagenberg
Hauptstrasse, 99
A-4232, Hagenberg, Autsria
teimuraz.kutsia@scch.at

Abstract. A minimal and complete unification procedure for a theory with individual and sequence variables, free constants and free fixed and flexible arity function symbols is described. The theory is extended in two ways: with patterns-terms and with sequence variables as arguments of terms with fixed arity head. A minimal and complete unification procedure for the pattern-term extension is given.

1 Introduction

We design a unification procedure for a theory with individual and sequence variables, fixed and flexible arity function symbols. The subject of this research was proposed by B. Buchberger in [7] and in a couple of personal discussions. The research described in this paper is a part of the author's PhD thesis.

We refer to unification in a theory with individual and sequence variables, free constants and free fixed and flexible arity function symbols shortly as unification with sequence variables and flexible arity symbols, underlining the importance of these two constructs. Sequence variables are variables which can be instantiated by an arbitrary finite sequence of terms. Flexible arity function symbols can take arbitrary finite number of arguments. In the literature the symbols with similar property are also referred as "variable arity", "variadic" or "multiple arity" symbols.

Languages with sequence variables and variable arity symbols have been used in different areas. Here we enumerate some of them:

– Knowledge management - Knowledge Interchange Format KIF ([13]) and its version SKIF ([24]) are extensions of first order language with (among the other constructs) sequence variables and variable arity function symbols. KIF is used to interchange knowledge among disparate computer systems. Another example of using sequence variables and variable arity symbols in knowledge systems is Ontolingua ([11]) - a tool which provides a distributed collaborative environment to browse, create, edit, modify, and use ontologies.

* Supported by the Austrian Science Foundation (FWF) under Project SFB F1302 and by Software Competence Center Hagenberg (Austria) under MathSoft project.

- Databases - sequences and sequence variables provide flexibility in data representation and manipulation for genome or text databases, where much of the data has an inherently sequential structure. Numerous formalisms involving sequences and sequence variables, like Sequence Logic ([14]), Alignment Logic ([15]), Sequence Datalog ([22]), String Calculus ([16],[6]), have been developed for this field.
- Rewriting - variable arity symbols used in rewriting usually come from flattening terms with associative top function symbol. Sequences and sequence variables (sometimes called also patterns), which are used together with variable arity symbols, make the syntax more flexible and expressive, and increase the performance of a rewriting system (see [30], [17]).
- Programming languages - variable arity symbols are supported by many programming languages. The programming language of Mathematica ([31]) is one of such examples, which uses the full expressive power of sequence variables as well. A relation of Mathematica programming language and rewrite rule languages, and the role of sequence variables in this relation is discussed in [7].
- Theorem proving - a package Epilog ([12]) can be used in programs that manipulate information encoded in Standard Information Format (SIF) - a subset of KIF ([13]) language, containing sequence variables and variable arity symbols. Among the other routines, Epilog includes pattern matchers of various sorts, and an inference procedure based on model elimination.

These applications involve (and in some cases, essentially depend on) solving equations with sequence variables and variable arity symbols. The mostly used solving technique is matching. However, for some applications, like theorem proving or completion/rewriting, more powerful solving techniques (unification, for instance) are needed.

The problem whether Knuth-Bendix completion procedure ([20]) can be extended to handle term rewriting systems with function symbols of variable arity, sequences and sequence variables (patterns) is stated as an open problem in [30]. The primary reason why it is an open problem is the absence of a corresponding unification algorithm.

In this paper, we make the first step towards solving this problem, providing a unification procedure with individual and sequence variables, fixed and flexible arity function symbols and its extension with patterns. Sequence variables and patterns can be seen as particular examples of the pattern construct of [30]. The term “flexible arity” was suggested by Buchberger ([8]) instead of “variable arity” or “variadic”, mainly because of the following reason: variable arity symbols, as they are understood in theorem proving or rewriting, are flattened associative symbols, i.e. flat symbols which take at least two arguments, while we allow flexible arity symbols to have zero or one argument as well and not to be necessarily flat, i.e. to have more “flexibility”. Non-flatness of flexible arity symbols makes one of main differences between unification with sequence variables and flexible arity symbols and associative unification: the unification problem $f(x, f(y, z)) \stackrel{?}{=} f(f(a, b), c)$, with the variables x, y, z and constants a, b, c, has no unifier, if f is a flexible arity symbol, but admits a unifier $\{x \leftarrow a, y \leftarrow b, z \leftarrow c\}$ for an associative f. Even in the case of a flat flexible arity f the problem would not be equivalent to A-unification: the substitution $\{x \leftarrow f(a), y \leftarrow f(b, c), z \leftarrow f()\}$ is a unifier for a flat f, but not for an associative f.

Unification with sequence variables and flexible arity symbols is a quite hard problem: its Siekmann hierarchy ([29]) type is infinitary.

Designing a unification procedure with sequence variables and variable arity symbols is a part of studying equational theories (like, for instance, free, flat, orderless, flat-orderless theories) with sequence variables and flexible arity symbols. This problem was posed by Buchberger [8] and was inspired by Mathematica's usage of sequence variables and flexible arity symbols.

It should be mentioned that in the theorem proving context quantification over sequence variables naturally introduces flexible arity symbols and patterns. For instance, skolemizing the expression $\forall \overline{x} \exists y \Phi[\overline{x}, y]$, where $\overline{x}$ is a sequence variable, y is an individual variable and $\Phi[\overline{x}, y]$ is a formula which depends on $\overline{x}$ and y, introduces a flexible arity Skolem function f: $\forall \overline{x} \Phi[\overline{x}, f(\overline{x})]$. On the other hand, skolemizing the expression $\forall x \exists \overline{y} \Phi[x, \overline{y}]$ introduces a pattern $h_{1,n(x)}(x)$, which can be seen as an abbreviation of a sequence of terms $h_1(x), \ldots, h_{n(x)}(x)$ of unknown length, where $h_1, \ldots, h_{n(x)}$ are (unary) Skolem functions.

The procedure can be used in theorem proving purpose in the similar way as Plotkin showed in [25]: building in equational theories. Although unification with sequence variables and flexible arity symbols is infinitary, special cases can be identified when the procedure terminates.

We show that unification with sequence variables and flexible arity symbols is decidable. Based on the decision procedure, a constraint-based approach to theorem proving with sequence variables and flexible arity symbols can be developed (compare [26]).

Particular instances of unification with sequence variables and flexible arity symbols are word equations ([1],[18], [28]), equations over free semigroups ([21]), equations over lists of atoms with concatenation ([10]), pattern matching.

We have implemented the unification procedure (without decision algorithm) and its extension (where sequence variables are allowed in terms with fixed arity heads, described in the Section 7 of this report) as a Mathematica package and incorporated it into the Theorema system [9], which aims at extending computer algebra systems by facilities for supporting mathematical proving. Currently the package is used in the Theorema Equational Prover. It makes Theorema probably the only system being able to handle solving and proving equations which involve sequence variables and flexible arity symbols. The package also enhances Mathematica solving capabilities, considering unification as a solving method. We used the package, for instance, to find matches for S-polynomials in non-commutative Gröbner basis algorithm [23]. The extension where sequence variables are allowed in terms with fixed arity heads depends on solving systems of linear Diophantine equations over non-negative integers. We use a Mathematica package called Omega ([3]), developed by G. E. Andrews and the RISC combinatorics group, to solve such systems.

The paper is organized as follows. In the Section 2 preliminaries and definitions are given. In the Section 3 definitions and examples related to equational theories with sequence variables and flexible arity symbols are given. Decidability of the free unification is considered in Section 4. Unification procedure for the free theory is described in Section 5. Extension with patterns and the corresponding unification procedure are considered in Section 6. Another extension of E-unification - by allowing sequence variables in arguments of terms with fixed arity heads - is given in Section 7. Section 8 is about implementation. Section 9 is the conclusion.

2 Preliminaries

The set of individual variables IV is a denumerable set of words consisting of an English letter and subsequent letters or digits, starting with x, y or z. The set of sequence variables SV is a denumerable set of words with overline, consisting of an English letter and subsequent letters or digits, starting with x, y or z.

Definition 1 (Constants). *$(CONST, FFIX, FFLEX, \doteq, AR)$ is domain of constants iff*

- *$CONST$ is a set of symbols ("the set of object constants"),*
- *$FFIX$ is a set of symbols ("the set of function constants of fixed arity"),*
- *$FFLEX$ is a set of symbols ("the set of function constants of flexible arity"),*
- *$\doteq$ is the "equality relation constant",*
- *$AR : (FFIX \cup \{\doteq\}) \to \mathbb{N}$ ("the arity function"),*
- *$AR(\doteq) = 2$,*
- *$CONST, FFIX, FFLEX, \{\doteq\}$ are pairwise disjoint and disjoint from IV and SV.*

Let now V stand for (IV, SV) (variables), C for $(CONST, FFIX, FFLEX, \doteq, AR)$ (a domain of constants) and P - for a set $\{(,),,\}$ ("parentheses and comma"). We define terms and equations over (V, C, P).

Definition 2 (Term). *The set of terms (over (V, C, P)) is the smallest set of strings over (V, C, P) that satisfies the following conditions:*

- *If $v \in IV \cup SV$ then v is a term.*
- *If $c \in CONST$ then c is a term.*
- *If $f \in FFIX$, $AR(f) = n$, $n \geq 0$ and $t_1, \ldots, t_n$ are terms such that for all $1 \leq i \leq n$, $t_i \notin SV$, then $f(t_1, \ldots, t_n)$ is a term. f is called the head of $f(t_1, \ldots, t_n)$.*
- *If $f \in FFLEX$ and $t_1, \ldots, t_n$ $(n \geq 0)$ are terms, then $f(t_1, \ldots, t_n)$ is a term. f is called the head of $f(t_1, \ldots, t_n)$.*

Definition 3 (Equation). *The set of equations (over the alphabet (V, C, P)) is the smallest set of strings over (V, C, P) that satisfies the following condition:*

– *If t_1 and t_2 are terms over (V, C, P) such that $t_1 \notin SV$ and $t_2 \notin SV$, then $\doteq (t_1, t_2)$ is an equation over (V, C, P). $\doteq$ is called the head of the equation $\doteq (t_1, t_2)$.*

For equation we will use infix notation and write $t_1 \doteq t_2$ for $\doteq (t_1, t_2)$. If not otherwise stated, we will use x, y and z as metavariables ranging over individual variables, $\overline{x}$, $\overline{y}$ and $\overline{z}$ - metavariables over sequence variables, v and u - over (individual or sequence) variables, c - over object constants, $ffix$, $gfix$ and $hfix$ - over function symbols of fixed arity, $fflex$, $gflex$ and $hflex$ - over function symbols of flexible arity, f, g and h - over (fixed or flexible arity) function symbols, s and t - over terms, eq - over equations. We might use indices with these symbols as well.

Definition 4 (Substitution). *A substitution is a finite set*

$$\{x_1 \leftarrow s_1, \ldots, x_n \leftarrow s_n, \overline{x}_1 \leftarrow t_1^1, \ldots, t_{k_1}^1, \ldots, \overline{x}_m \leftarrow t_1^m, \ldots, t_{k_m}^m\}$$

where

– $n \geq 0$, $m \geq 0$ *and for all* $1 \leq i \leq m$, $k_i \geq 0$,
– $x_1, \ldots, x_n$ *are distinct individual variables,*
– $\overline{x}_1, \ldots, \overline{x}_m$ *are distinct sequence variables,*
– *for all* $1 \leq i \leq n$, s_i *is a term,* $s_i \notin SV$ *and* $s_i \neq x_i$,
– *for all* $1 \leq i \leq m$, $t_1^i, \ldots, t_{k_i}^i$ *is a sequence of terms and if* $k_i=1$ *then* $t_{k_i}^i \neq \overline{x}_i$.

Each $x_i \leftarrow s_i$ ($\overline{x}_i \leftarrow t_1^i, \ldots, t_{k_i}^i$) *is called a binding for* x_i ($\overline{x_i}$).

The substitution is called ground iff all $s_1, \ldots, s_n, t_1^1, \ldots, t_{k_1}^1, t_1^m, \ldots, t_{k_m}^m$ are variable-free terms. The substitution is called empty iff $n = 0$ and $m = 0$. Greek letters are used to denote substitutions. The letter ε denotes the empty substitution.

Definition 5 (Instance). *Given a substitution θ, we define an instance of a term or equation with respect to θ recursively as follows:*

– $x\theta = \begin{cases} s \text{ if } x \leftarrow s \in \theta, \\ x \text{ otherwise} \end{cases}$
– $\overline{x}\theta = \begin{cases} s_1, \ldots, s_m \text{ if } x \leftarrow s_1, \ldots, s_m \in \theta, \ m \geq 0 \\ \overline{x} \qquad\qquad \text{otherwise} \end{cases}$
– $f(s_1, \ldots, s_n)\theta = f(s_1\theta, \ldots, s_n\theta)$
– $(s_1 \doteq s_2)\theta = s_1\theta \doteq s_2\theta$

Definition 6 (Domain). *The domain of a substitution σ is defined as*

$$Dom(\sigma) = \{v \in IV \cup SV \mid v\sigma \neq v\}.$$

Definition 7 (Codomain). *The codomain of a substitution σ is defined as*

$$Cod(\sigma) = \{v\sigma \mid v \in Dom(\sigma)\}.$$

Let $VarSet(t_1, \ldots, t_n)$, $n \geq 0$, be the set of variables occurring in a sequence of terms $t_1, \ldots, t_n$.

Definition 8 (Range). *The range of a substitution σ is defined as*

$$Ran(\sigma) = \cup_{v \in Dom(\sigma)} VarSet(v\sigma).$$

We extend the notion of instance of a term to the notion of instance of a sequence in a straightforward way: given a sequence of terms $t_1, \ldots, t_n$, $n \geq 0$ and a substitution θ, the instance of $t_1, \ldots, t_n$ with respect to θ, denoted as $(t_1, \ldots, t_n)\theta$, is the sequence $t_1\theta, \ldots, t_n\theta$.

Definition 9 (Composition of Substitutions). *Let*

$$\theta = \{x_1 \leftarrow s_1, \ldots, x_n \leftarrow s_n, \overline{x_1} \leftarrow t_1^1, \ldots, t_{k_1}^1, \ldots, \overline{x_m} \leftarrow t_1^m, \ldots, t_{k_m}^m\}$$

and

$$\lambda = \{y_1 \leftarrow d_1, \ldots, y_n \leftarrow d_l, \overline{y_1} \leftarrow e_1^1, \ldots, e_{q_1}^1, \ldots, \overline{y_r} \leftarrow e_1^r, \ldots, e_{q_r}^r\}$$

be two substitutions. Then the composition of θ and λ is the substitution, denoted by $\theta \circ \lambda$, obtained from the set

$$\begin{aligned}\{\ & x_1 \leftarrow s_1\lambda, \ldots, x_n \leftarrow s_n\lambda, \overline{x_1} \leftarrow t_1^1\lambda, \ldots, t_{k_1}^1\lambda, \ldots, \overline{x_m} \leftarrow t_1^m\lambda, \ldots, t_{k_m}^m\lambda,\\ & y_1 \leftarrow d_1, \ldots, y_l \leftarrow d_l, \overline{y_1} \leftarrow e_1^1, \ldots, e_{q_1}^1, \ldots, \overline{y_r} \leftarrow e_1^r, \ldots, e_{q_r}^r\}\end{aligned}$$

by deleting

- *all the elements $x_i \leftarrow s_i\lambda$ $(1 \leq i \leq n)$ for which $x_i = s_i\lambda$,*
- *all the elements $\overline{x_i} \leftarrow t_1^i\lambda, \ldots, t_{k_i}^i\lambda$ $(1 \leq i \leq m)$ for which $k_i = 1$ and $\overline{x_i} = t_1^i\lambda$,*
- *all the elements $y_i \leftarrow d_i$ $(1 \leq i \leq l)$ such that $y_i \in \{x_1, \ldots, x_n\}$,*
- *all the elements $\overline{y_i} \leftarrow e_1^i, \ldots, e_{q_i}^i$ $(1 \leq i \leq r)$ such that $\overline{y_i} \in \{\overline{x_1}, \ldots, \overline{x_m}\}$.*

Example 1. Let

$$\theta = \{x \leftarrow f(y),\ \overline{x} \leftarrow \overline{y}, \overline{x},\ \overline{y} \leftarrow \overline{y}, \overline{z}\}$$

and

$$\lambda = \{y \leftarrow g(c,c),\ z \leftarrow f(c), \overline{x} \leftarrow c,\ \overline{z} \leftarrow\}.$$

Then

$$\theta \circ \lambda = \{x \leftarrow f(g(c,c)),\ y \leftarrow g(c,c),\ z \leftarrow f(c),\ \overline{x} \leftarrow \overline{y}, c,\ \overline{z} \leftarrow\}.$$

Theorem 1. *For a term t and substitutions θ and λ*

$$t\theta \circ \lambda = t\theta\lambda.$$

Proof. We prove the theorem by induction on the structure of t.

1. $t \in IV$.

(a) $t \leftarrow s \in \theta \circ \lambda$, for some s. Then $t\theta \circ \lambda = s$. We show that $t\theta\lambda = s$. We have the following two cases:
   i. There exists an r such that $t \leftarrow r \in \theta$ and $s = r\lambda$. Then
$$t\theta\lambda = r\lambda = s.$$
   ii. $t \leftarrow s \in \lambda$ and θ does not contain a binding for t. Then
$$t\theta\lambda = t\lambda = s.$$
(b) $\theta \circ \lambda$ does not contain a binding for t. Then $t\theta \circ \lambda = t$. We show that $t\theta\lambda = t$. We have the following two cases:
   i. Neither θ nor λ contain a binding for t. Then
$$t\theta\lambda = t\lambda = t.$$
   ii. There exists an r such $t \leftarrow r \in \theta$ and $r\lambda = t$. Then
$$t\theta\lambda = r\lambda = t.$$

2. $t \in SV$.
(a) $t \leftarrow s_1, \ldots, s_n \in \theta \circ \lambda$ for some $s_1, \ldots, s_n$, $n \geq 0$. Then $t\theta \circ \lambda = s_1, \ldots, s_n$. We show that $t\theta\lambda = s_1, \ldots, s_n$. We have the following two cases:
   i. There exist $r_1, \ldots, r_m$, $m \geq 0$, such that $t \leftarrow r_1, \ldots, r_m \in \theta$ and $r_1\lambda, \ldots, r_m\lambda = s_1, \ldots, s_n$. Then
$$t\theta\lambda = (r_1, \ldots, r_m)\lambda = r_1\lambda, \ldots, r_m\lambda = s_1, \ldots, s_n.$$
   ii. $t \leftarrow s_1, \ldots, s_n \in \lambda$ and θ does not contain a binding for t. Then
$$t\theta\lambda = t\lambda = s_1, \ldots, s_m.$$
(b) $\theta \circ \lambda$ does not contain a binding for t. The $t\theta \circ \lambda = t$. We show that $t\theta\lambda = t$. We have the following two cases:
   i. Neither θ nor λ contain a binding for t. Then
$$t\theta\lambda = t\lambda = t.$$
   ii. There exist $r_1, \ldots, r_m$, $m \geq 0$, such that $t \leftarrow r_1, \ldots, r_m \in \theta$ and $(r_1, \ldots, r_m)\lambda = t$. Then
$$t\theta\lambda = (r_1, \ldots, r_m)\lambda = t.$$

3. t is a compound term $f(s_1, \ldots, s_n)$, $f \in FFIX \cup FFLEX$. Then
$$\begin{array}{ll} f(s_1, \ldots, s_n)\theta \circ \lambda & = \text{(by Definition 5)} \\ f(s_1\theta \circ \lambda, \ldots, s_n\theta \circ \lambda) & = \text{(by induction hypothesis)} \\ f(s_1\theta\lambda, \ldots, s_n\theta\lambda). & \end{array}$$
and
$$\begin{array}{ll} f(s_1, \ldots, s_n)\theta\lambda & = \text{(by Definition 5)} \\ f(s_1\theta, \ldots, s_n\theta)\lambda & = \text{(by Definition 5)} \\ f(s_1\theta\lambda, \ldots, s_n\theta\lambda). & \end{array}$$

Corollary 1. *If* $t_1, \ldots, t_n$ *is a sequence of terms and* θ *and* λ *are substitutions, then*

$$(t_1 \ldots, t_n)\theta \circ \lambda = ((t_1 \ldots, t_n)\theta)\lambda.$$

Theorem 2. *For any substitutions* θ, λ *and* σ,

$$(\theta \circ \lambda) \circ \sigma = \theta \circ (\lambda \circ \sigma).$$

Proof. ($\subseteq$) Let $v \leftarrow \overline{r} \in (\theta \circ \lambda) \circ \sigma$, where $\overline{r}$ is a sequence of terms, if $v \in SV$ and is a single term, if $v \in IV$. We show that

$$v \leftarrow \overline{r} \in \theta \circ (\lambda \circ \sigma). \tag{1}$$

To prove 1, by Definition 5 it suffices to show

$$v\theta \circ (\lambda \circ \sigma) = \overline{r}. \tag{2}$$

We prove 2 as follows:

$$\begin{array}{ll}
v\theta \circ (\lambda \circ \sigma) & = (\text{by Definition } 5) \\
v\theta(\lambda \circ \sigma) & = (\text{by Definition } 5) \\
v\theta\lambda\sigma & = (\text{by Theorem1}) \\
v(\theta \circ \lambda)\sigma & = (\text{by Theorem1}) \\
v\theta \circ (\lambda \circ \sigma) & = \\
\overline{r}. &
\end{array}$$

($\supseteq$) Similarly to ($\subseteq$).

3 Equational Theory with Sequence Variables and Flexible Arity Symbols

A set of equations E (called representation) defines an equational theory, i.e. the equality of terms induced by E. We use the term E-theory for the equational theory defined by E. We will write $s \doteq_E t$ for $s \doteq t$ modulo E.

Solving equations in an E-theory is called E-unification. The fact that the equation $s \doteq_E t$ has to be solved is written as $s \overset{?}{\doteq}_E t$. A finite system of equations $\langle s_1 \overset{?}{\doteq}_E t_1, \ldots, s_n \overset{?}{\doteq}_E t_n \rangle$ is called an E-unification problem.

Some examples of equational theories with sequence variables and flexible arity symbols are:

1. Free theory: $E = \emptyset$;
2. Flat theory (a theory with flexible arity flat symbol(s)):

$$E = \{fflex(\overline{x}, fflex(\overline{y}), \overline{z}) \doteq fflex(\overline{x}, \overline{y}, \overline{z})\}.$$

3. Restricted flat theory (a restricted theory with flexible arity flat symbol(s)):

$$E = \{fflex(\overline{x}, fflex(\overline{y_1}, x, \overline{y_2}), \overline{z}) \doteq fflex(\overline{x}, \overline{y_1}, x, \overline{y_2}, \overline{z})\}.$$

4. Orderless theory (a theory with orderless flexible arity flat symbol(s)):

$$E = \{fflex(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq fflex(\overline{x}, y, \overline{y}, x, \overline{z})\}.$$

5. Flat-orderless theory (a theory with flat-orderless flexible arity flat symbol(s)):

$$\begin{aligned} E = \{\ & fflex(\overline{x}, fflex(\overline{y}), \overline{z}) \doteq fflex(\overline{x}, \overline{y}, \overline{z}), \\ & fflex(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq fflex(\overline{x}, y, \overline{y}, x, \overline{z})\}. \end{aligned}$$

6. Restricted flat-orderless theory (a restricted theory with flat-orderless flexible arity flat symbol(s)):

$$\begin{aligned} E = \{\ & fflex(\overline{x}, fflex(\overline{y_1}, x, \overline{y_2}), \overline{z}) \doteq fflex(\overline{x}, \overline{y_1}, x, \overline{y_2}, \overline{z}), \\ & fflex(\overline{x}, x, \overline{y}, y, \overline{z}) \doteq fflex(\overline{x}, y, \overline{y}, x, \overline{z})\}. \end{aligned}$$

Free, flat, restricted flat, orderless and restricted flat-orderless theories are referred respectively as $\emptyset$- F-, RF-, O-, FO- and ROF-theory.

Definition 10 (Unifier). *A substitution θ is called an E-unifier (or E-solution) of an E-unification problem $\langle s_1 \stackrel{?}{\doteq}_E t_1, \ldots, s_n \stackrel{?}{\doteq}_E t_n \rangle$ iff $s_i\theta \doteq_E t_i\theta$ for all $1 \leq i \leq n$. An E-unification problem Γ is called E-unifiable (or E-solvable) iff there exists an E-unifier of Γ.*

By solving an E-unification problem Γ we mean finding an E-unifier of Γ.

Definition 11 (More General Substitution). *A substitution θ is more general than a substitution σ on a finite set of variables Var modulo a theory E (denoted $\theta \ll_E^{Var} \sigma$) iff there exists a substitution λ such that*

- *for all $\overline{x} \in Var$,*
  - *$\overline{x} \leftarrow \ \notin \lambda$;*
  - *there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma = t_1, \ldots, t_n$, $\overline{x}\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$;*
- *for all $x \in Var$, $x\sigma \doteq_E x\theta \circ \lambda$.*

Note that for the substitutions without sequence variables this definition coincides with the standard definition of more general substitution. We will write $\sigma \gg_E^{Var} \theta$ iff $\theta \ll_E^{Var} \sigma$.

Example 2. Let $\theta = \{\overline{x} \leftarrow \overline{y}\}$. Then $\theta \ll_{\emptyset}^{\{\overline{x},\overline{y}\}} \sigma$, where $\sigma = \{\overline{x} \leftarrow a, \overline{z},\ \overline{y} \leftarrow a, \overline{z}\}$, but not $\theta \ll_{\emptyset}^{\{\overline{x},\overline{y}\}} \eta$, where $\eta = \{\overline{x} \leftarrow, \overline{y} \leftarrow\}$.

Definition 12 (Minimal Set of Substitutions). *A set of substitutions Σ is called E-minimal with respect to a set of variables Var iff for all $\theta, \sigma \in \Sigma$, $\theta \ll_E^{Var} \sigma$ implies $\theta = \sigma$.*

Definition 13 (The Minimal Complete Set of Unifiers). *The minimal complete set of E-unifiers of Γ, denoted $MCU_E(\Gamma)$, is an E-minimal set of substitutions with respect to the set of variables Var of Γ, satisfying the following conditions:*

E-Correctness - for all $\theta \in MCU_E(\Gamma)$, θ is an E-unifier of Γ.
E-Completeness - for any E-unifier σ of Γ there exists $\theta \in MCU_E(\Gamma)$ such that $\theta \ll_E^{Var} \sigma$.

Example 3. Compute the minimal complete set of unifiers in the free, flat and restricted flat theories (f and g are free flexible arity function symbols, h is a flat flexible arity function symbol, rh - restricted flat flexible arity symbols):

1. $MCU_\emptyset(\langle f(\overline{x}, y) \stackrel{?}{=}_\emptyset f(a, b, c)\rangle) = \{\{\overline{x} \leftarrow a, b,\ y \leftarrow c\}\}$.
2. $MCU_\emptyset(\langle f(\overline{x}, \overline{y}) \stackrel{?}{=}_\emptyset f(a, b)\rangle) = \{\{\overline{x} \leftarrow,\ \overline{y} \leftarrow a, b\},\ \{\overline{x} \leftarrow a, \overline{y} \leftarrow b\},\ \{\overline{x} \leftarrow a, b, \overline{y} \leftarrow\}\}$.
3. $MCU_\emptyset(\langle f(\overline{x}, a) \stackrel{?}{=}_\emptyset f(a, \overline{x})\rangle) = \{\{\overline{x} \leftarrow\},\ \{\overline{x} \leftarrow a\},\ \{\overline{x} \leftarrow a, a\}, \ldots\}$.
4. $MCU_\emptyset(\langle f(g(a, g(\overline{y}, c)), \overline{x}) \stackrel{?}{=}_\emptyset f(\overline{u},\ g(b, \overline{v}))\rangle) = \{\{\overline{u} \leftarrow g(a, \overline{x}),\ \overline{y} \leftarrow b,\ \overline{v} \leftarrow c\},\ \{\overline{x} \leftarrow, \overline{u} \leftarrow g(a),\ \overline{y} \leftarrow b,\ \overline{v} \leftarrow c\},\ \{\overline{u} \leftarrow g(a, \overline{x}),\ \overline{y} \leftarrow b, \overline{y},\ \overline{v} \leftarrow \overline{y}, c\},\ \{\overline{x} \leftarrow, \overline{u} \leftarrow g(a),\ \overline{y} \leftarrow b, \overline{y},\ \overline{v} \leftarrow \overline{y}, c\}\}$.
5. $MCU_F(\langle x \stackrel{?}{=}_F h(x)\rangle) = \{\{x \leftarrow h(x)\}\}$.
6. $MCU_F(\langle h(\overline{x}) \stackrel{?}{=}_F h(a)\rangle) = \{\{\overline{x} \leftarrow a\}, \{\overline{x} \leftarrow h(a)\}, \{\overline{x} \leftarrow a, h()\},\ \{\overline{x} \leftarrow h(a), h()\},\ \{\overline{x} \leftarrow h(), a\},\ \{\overline{x} \leftarrow h(), h(a)\},\ \{\overline{x} \leftarrow h(), a, h()\},\ \ldots\}$.
7. $MCU_{RF}(\langle rh(\overline{x}) \stackrel{?}{=}_{RF} rh(a)\rangle) = \{\{\overline{x} \leftarrow a\}, \overline{x} \leftarrow rh(a)\}$.
8. $MCU_{RF}(\langle rh(\overline{x}, y) \stackrel{?}{=}_{RF} rh(a, b, c)\rangle) = \{\{\overline{x} \leftarrow,\ y \leftarrow rh(a, b, c)\},\ \{\overline{x} \leftarrow a,\ y \leftarrow rh(b, c)\}, \{\overline{x} \leftarrow rh(a),\ y \leftarrow rh(b, c)\},\ \{\overline{x} \leftarrow a, b,\ y \leftarrow c\},\ \{\overline{x} \leftarrow rh(a), b,\ y \leftarrow c\},\ \{\overline{x} \leftarrow a, rh(b),\ y \leftarrow c\},\ \{\overline{x} \leftarrow rh(a), rh(b),\ y \leftarrow c\},\ \{\overline{x} \leftarrow rh(a, b),\ y \leftarrow c\},\ \{\overline{x} \leftarrow a, b,\ y \leftarrow rh(c)\},\ \{\overline{x} \leftarrow rh(a), b,\ y \leftarrow rh(c)\},\ \{\overline{x} \leftarrow a, rh(b),\ y \leftarrow rh(c)\},\ \{\overline{x} \leftarrow rh(a), rh(b),\ y \leftarrow rh(c)\},\ \{\overline{x} \leftarrow rh(a, b),\ y \leftarrow rh(c)\}\}$.

Below in this paper we consider only the $\emptyset$-theory, although the results valid for arbitrary E-theories with sequence variables and flexible arity symbols are formulated in a general setting. Note that in the case of $\emptyset$-theory it is enough to consider single equations instead of systems of equations in unification problems, because $\langle s_1 \stackrel{?}{=}_\emptyset t_1, \ldots, s_n \stackrel{?}{=}_\emptyset t_n\rangle$ has the same set of unifiers as $f(s_1, \ldots, s_n) \stackrel{?}{=}_\emptyset f(t_1, \ldots, t_n)$, where f is a free flexible arity symbol.

We answer the following two questions about $\emptyset$-unification:

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

4 General Unification in the Free Theory with Sequence Variables and Flexible Arity Symbols - Decidability

In this section we consider decidability of general unification problem in the free theory with sequence variables and flexible arity symbols. The problem has a form $t_1 \stackrel{?}{=}_{\emptyset} t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, free constants and free fixed arity function symbols. We denote it as $GUP_{\emptyset}$.

The problem is to prove that $GUP_{\emptyset}$ is decidable. To do so, first we try to reduce $GUP_{\emptyset}$ to a "simpler" unification problem RUP (reduced unification problem) such that $GUP_{\emptyset}$ is solvable if and only if RUP is solvable, and then try to prove that solvability of RUP is decidable.

4.1 Reduction

We reduce the problem of solvability of $GUP_{\emptyset}$ to the problem of solvability of a reduced unification problem RUP.

Let $fflex_1, \ldots, fflex_n$ $(n \geq 1)$ be all flexible arity function symbols occurring in $GUP_{\emptyset}$. First we reduce $GUP_{\emptyset}$ to an intermediate F-unification problem $IUP_F = t_1^{IUP} \stackrel{?}{=}_F t_2^{IUP}$ by performing the following steps:

1. Introduce
   - a new flat flexible arity symbol Seq,
   - a new unary function symbol $nfix_i$ for each $fflex_i$ $(1 \leq i \leq n)$.
2. Replace each term $fflex_i(t_1, \ldots, t_m)$ in $GUP_{\emptyset}$ by $nfix_i(Seq(t_1, \ldots, t_m))$ $(m \geq 0)$.

Note that in IUP_F sequence variables occur only as arguments of terms with the head Seq. We impose *individual variable restrictions on terms* on IUP_F demanding that, for a solution θ of IUP_F and for any individual variable x, $x\theta$ must not have Seq as a head.

Theorem 3. *$GUP_{\emptyset}$ is solvable iff IUP_F with individual variable restrictions on terms is solvable.*

Proof. ($\Rightarrow$) Let θ be a unifier of $GUP_{\emptyset}$. Then from θ we can get a unifier of IUP_F with individual variable restrictions on terms by repeating the step 2 of the reduction procedure above on each term with flexible arity head from $Cod(\theta)$.

($\Leftarrow$) Let IUP_F with individual variable restrictions on terms have a solution ϕ. Show that $GUP_{\emptyset}$ is solvable.

Let θ be a substitution obtained from ϕ by

- replacing each unary function symbol $nfix$ introduced at the first step of the reduction procedure above by the corresponding flexible arity symbol $fflex$,
- replacing each term of the form $Seq(t_1, \ldots, t_m)$ $(m \geq 0)$ by the sequence of terms $t_1, \ldots, t_m$.

Then θ is a unifier of $GUP_\emptyset$.

Remark. Note that solvability of IUP_F (without individual variable restrictions on terms) does not imply solvability of $GUP_\emptyset$. For instance, let $GUP_\emptyset = fflex(x) \stackrel{?}{=}_\emptyset fflex(a,b)$, then $IUP_F = nfix(Seq(x)) \stackrel{?}{=}_F nfix(Seq(a,b))$. It is clear that $GUP_\emptyset$ does not have a solution, while $\{x \leftarrow Seq(a,b)\}$ is a solution of IUP_F because the flatness of Seq implies $Seq(Seq(a,b)) = Seq(a,b)$.

Next, our goal is to reduce IUP_F with individual variable restrictions on terms to a finite set $SRUP$ of reduced F-unification problems with the property that IUP with individual variable restrictions on terms is solvable iff there exists a $RUP \in SRUP$ which is solvable.

First, we define two new sets:

1. Let $CONST_{IUP}$ be the set of constants of IUP_F and *const* be a new constant. Then
$$C_{IUP} = CONST_{IUP} \cup \{const\}.$$
2. Let $FFIX_{IUP} = \{ffix_1, \ldots, ffix_m\}$, $m \geq 1$ be the set of all fixed arity function symbols in IUP_F and $y_1^1, \ldots, y^1_{AR(ffix_1)}, \ldots, y_1^m, \ldots, y^m_{AR(ffix_m)}$ be new distinct individual variables. Then
$$T_{IUP} = \{f_1(y_1^1, \ldots, y^1_{AR(f_1)}), \ldots, f_m(y_1^m, \ldots, y^m_{AR(f_m)})\}.$$

Let $IV_{IUP} = \{x_1, \ldots, x_n\}$ be the set of all individual variables of IUP_F. Then $SRUP$ is defined as:

$$\begin{aligned} SRUP = \{ & gfix\ (t_1^{IUP}, x_1, \ldots, x_n) \stackrel{?}{=}_F gfix(t_2^{IUP}, s_1, \ldots, s_n) \mid \\ & gfix \in FFIX,\ AR(gfix) = n+1, \\ & \text{for all } 1 \leq i \leq n,\ s_i \in C_{IUP} \cup T_{IUP}\}. \end{aligned}$$

Theorem 4. *IUP_F with individual variable restrictions on terms is solvable iff there exists a $RUP \in SRUP$ which is solvable.*

Proof. ($\Rightarrow$) Let θ be a unifier of IUP_F with individual variable restrictions on terms. Then for each $x \in IV_{IUP}$, $x\theta$ does not have Seq as a head, i.e. $x\theta$ is either a individual variable, a constant from $CONST_{IUP}$ or has a form $ffix(s_1, \ldots, s_{AR(ffix)})$ for some function symbol $ffix \in FFIX_{IUP}$ and terms $s_1, \ldots, s_{AR(ffix)}$.

Let $\sigma = \{x \leftarrow c \,|\, x \in IV_{IUP} \cup (IV \cap Ran(\theta))\}$. Then $\theta \circ \sigma$ is a solution of IUP_F such that for each $x \in IV_{IUP}$, either $x\theta \circ \sigma \in C_{IUP}$ or there exists a substitution λ such that $x\theta \circ \sigma = t\lambda$ for some $t \in T_{IUP}$.

Let
$$X_C = \{x \,|\, x \in IV_{IUP} \text{ and } x\theta \circ \sigma \in C_{IUP}\},$$
$$\begin{aligned} X_F = \{x \,|\ & x \in IV_{IUP} \text{ and } x\theta \circ \sigma = t\lambda \\ & \text{for some } t \in T_{IUP} \text{ and substitution } \lambda\}. \end{aligned}$$

Then $X_C \cup X_F = IV_{IUP}$.

Let

– for each $x \in X_C$, c_x be the constant $c \in C_{IUP}$ such that $c = x\theta \circ \sigma$;
– for each $x \in X_F$, t_x be the term $t \in T_{IUP}$ such that $t\lambda = x\theta \circ \sigma$ for some substitution λ;
– for each $x \in X_F$, λ_x be the substitution λ such that
  - if $t_x = ffix(y_1, \ldots, y_{AR(ffix)})$ then $Dom(\theta) = \{y_1, \ldots, y_{AR(ffix)}\}$
  - $x\theta \circ \sigma = t_x\lambda$.

We choose $RUP \in SRUP$ such that

$$RUP = \begin{array}{l} gfix(t_1^{IUP}, x_1, \ldots, x_i, x_{i+1}, \ldots, x_n) \stackrel{?}{=}_F \\ gfix(t_2^{IUP}, c_{x_1}, \ldots, c_{x_i}, t_{x_{i+1}}, \ldots, t_{x_n}) \end{array}$$

where $\{x_1, \ldots, x_i\} = X_C$ and $\{x_{i+1}, \ldots, x_n\} = X_F$.

Let $\eta = \cup_{x \in X_F} \lambda_x$. Then

$$\begin{array}{l} gfix(t_1^{IUP}, x_1, \ldots, x_i, x_{i+1}, \ldots, x_n)\theta \circ \sigma \circ \eta = \\ gfix(t_1^{IUP}\theta\sigma, c_{x_1}, \ldots, c_{x_i}, x_{i+1}\theta\sigma, \ldots, x_n\theta\sigma) \end{array}$$

and

$$\begin{array}{l} gfix(t_2^{IUP}, c_{x_1}, \ldots, c_{x_i}, t_{x_{i+1}}, \ldots, t_{x_n})\theta \circ \sigma \circ \eta = \\ gfix(t_2^{IUP}\theta\sigma, c_{x_1}, \ldots, c_{x_i}, t_{x_{i+1}}\lambda, \ldots, t_{x_n}\lambda). \end{array}$$

Since $t_1^{IUP}\theta\sigma =_F t_2^{IUP}\theta\sigma$ and for all $i+1 \le j \le n$, $t_{x_j}\lambda = x_j\theta\sigma$, we get that $\theta \circ \sigma\eta$ is a solution of RUP. Thus, solvability of IUP_F with individual variable restrictions on terms implies solvability of some RUP from $SRUP$.

($\Leftarrow$) Let ϕ be a solution of some $RUP \in SRUP$. Then, obviously, ϕ is a solution of IUP_F. Moreover, ϕ binds every individual variable of IUP_F either with a constant or a term with fixed arity head. Hence, ϕ is a solution of IUP with individual variable restrictions on terms.

Thus, the problem of decidability of the unification problem $GUP_\emptyset$ is reduced to the problem of decidability of a reduced F-unification problem RUP.

4.2 Decidability of the Reduced Problem

We prove decidability of the reduced F-unification problem RUP using Baader-Schulz combination method [5]:

Theorem 5 (Baader and Schulz).
Let $E_1, \ldots, E_n$ be equational theories over disjoint signatures such that solvability of E_i-unification problems with linear constant restriction is decidable for $i = 1, \ldots, n$. Then unifiability is decidable for the combined theory $E_1 \cup \ldots \cup E_n$.

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

Let Ω_1 be the set $\{Seq\}$ and Ω_2 be the set of fixed arity function symbols and object constants which occur in RUP. Let E_1 be an equational theory over the signature Ω_1 where terms are built from individual variables, sequence variables and the symbol Seq from Ω_1. Since Seq is a flat symbol every non-variable term of E_1 has a form $Seq(t_1, \ldots, t_n)$ $(n \geq 0)$ where each t_i is either a sequence variable or an individual variable. Let E_2 be a free theory over Ω_2, where terms are built from individual variables and symbols from Ω_2. Then we can consider RUP as a unification problem in the combined theory $E_1 \cup E_2$. Since $\Omega_1 \cap \Omega_2 = \emptyset$, by Theorem 5, in order to show that solvability of RUP in $E_1 \cup E_2$ is decidable we need to show that solvability of E_1- and E_2-unification problems with linear constant restrictions is decidable.

Decidability of E_1-unification problem with linear constant restrictions The decidability problem for E_1-unification is equivalent to the decidability problem for word equations with an additional restriction on certain variables in the equation. The restriction demands that for each of those variables and a unifier θ of the equation, the length of an instance of the variable with respect to θ must be 1. We call these additional restrictions *individual variable restrictions on length*. Thus, The decidability problem for E_1-unification with linear constant restrictions is equivalent to the decidability problem for word equations with linear constant restrictions and individual variable restrictions on length. Therefore, we will show that solvability of word equations with linear constant restrictions and individual variable restrictions on length is decidable.

Note that we can consider ground unifiability case only. Let V be a set of (sequence or individual) variables, C be a set of constants and C‘ be the set $C \cup \{c\}$, where c is a constant not occurring in C. Suppose that a word equation WE is given with variables in V, constants in C, with linear constant restrictions induced by a linear ordering $<$ on $V \cup C$ and for each individual variable from V, with individual variable restrictions on length. WE has a unifier θ over $V \cup C$ which satisfies the linear constant restrictions induced by $<$ and the individual variable restrictions on length iff WE has a ground unifier θ‘ over C‘ which satisfies the linear constant restrictions induced by $<$ and the individual variable restrictions on length.

We need the following general result of [27]:

Theorem 6 (Schulz). *If WE is a word equation with variables $v_1, \ldots, v_n$ and constants in the alphabet C‘, and if $L_1, \ldots, L_n$ are regular languages over C‘, then it is decidable whether WE has a solution θ such that $v_i\theta \in L_i$ for $i = 1, \ldots, n$.*

Using this result we can prove a theorem which solves of word equations with linear constant restrictions and individual variable restrictions on length and thus, solves the decidability problem for E_1-unification with linear constant restrictions:

Theorem 7. *Solvability of a word equation with linear constant restrictions and individual variable restrictions on length is decidable.*

Proof. Let WE be a word equation with variables $x_1, \ldots, x_m, \overline{y}_1, \ldots, \overline{y}_k$ and constants in the alphabet C‘, where the variables $x_1, \ldots, x_m$ are subjects of individual variable restrictions on length. Let $L_1, \ldots, L_{m+k}$ be regular languages over C‘ such that

- for all $1 \leq i \leq m$, $L_i = \{a \in C \mid a < x_i\} \cup \{c\}$,
- for all $1 \leq i \leq k$, $L_{m+i} = (\{a \in C \mid a < \overline{y}_i\} \cup \{c\})^+$.

A solution θ of WE satisfies the linear constant restrictions and the individual variable restrictions on length iff $\theta(x_i) \in L_i$ for all $1 \leq i \leq m$ and $\theta(\overline{y}_i) \in L_{m+i}$ for all $1 \leq i \leq k$. Thus, Theorem 6 implies that solvability of a word equation with linear constant restrictions and individual variable restrictions on length is decidable.

Decidability of E_2-unification problem with linear constant restrictions Obviously, the E_2-unification problem is a Robinson unification problem. Solvability of Robinson unification problems with linear constant restrictions is decidable (see [4]).

Thus, solvability of unification problem in the $\emptyset$-theory with sequence variables and flexible arity symbols is decidable.

5 General Unification in the Free Theory with Sequence Variables and Flexible Arity Symbols - Unification Procedure

In this section we design a unification procedure to solve general unification problem in the free theory with sequence variables and flexible arity symbols. Recall that the problem has a form $t_1 \overset{?}{=}_{\emptyset} t_2$, built over the alphabet which consists of sequence and individual variables, free flexible arity function symbols, free constants and free fixed arity function symbols. We denote it as $GUP_{\emptyset}$.

We design the unification procedure as a tree generation process based on two basic steps: projection and transformation. Below we describe these steps in details.

For a unification problem $UP_{\emptyset}$, we denote by $SUC(UP_{\emptyset})$ the successors of UP under projection or transformation. $SUC(UP_{\emptyset})$ can be $SUCCESS$, $FAILURE$ or a tuple of unification problems. $SUB(UP_{\emptyset})$ denotes the tuple of substitutions which were applied on $UP_{\emptyset}$ to get the successors of $UP_{\emptyset}$.

5.1 Projection

The idea of projection is to eliminate some sequence variables from the given unification problem ([1]). Let $S_1, \ldots, S_n$ be all subsets of the set of all sequence variables of $GUP_{\emptyset}$ and Θ be the set of substitutions $\{\theta_1, \ldots, \theta_n\}$ such that for all $1 \leq i \leq n$, $\theta_i = \{\overline{x} \leftarrow \ \mid \overline{x} \in S_i\}$. Then under projection

$$SUB(GUP_{\emptyset}) = \langle \theta_1, \ldots, \theta_k \rangle$$

and

$$SUC(GUP_{\emptyset}) = \langle GUP_{\emptyset}\theta_1, \ldots, GUP_{\emptyset}\theta_k \rangle.$$

5.2 Transformation

To find $SUC(GUP_\emptyset)$ and $SUB(GUP_\emptyset)$ under transformation we distinguish the following three cases:

1. t_1 and t_2 are identical. Then
$$SUB(GUP_\emptyset) = \langle\varepsilon\rangle$$
and
$$SUC(GUP_\emptyset) = SUCCESS.$$
2. t_1 and t_2 are neither identical terms nor non-variable terms with the same head. Then $SUB(GUP_\emptyset)$ and $SUC(GUP_\emptyset)$ are defined in Table 1.
3. t_1 and t_2 are non-identical non-variable terms with the same head g, where g is a function symbol with either fixed or flexible arity. Then we have the following two cases:
   (a) Only one from t_1 and t_2 has the form $g()$. Then
   $$SUB(GUP_\emptyset) = \langle\rangle$$
   and
   $$SUC(GUP_\emptyset) = FAILURE.$$
   (b) None of t_1 and t_2 is a term of the form $g()$. Let t_1 be $g(r_1, \overline{r})$ and t_2 be $g(s_1, \overline{s})$, where $\overline{r}$ and $\overline{s}$ are (possibly empty) sequences of terms. Then we have the following cases:
      i. r_1 and s_1 are identical. Then
      $$SUB(GUP_\emptyset) = \langle\varepsilon\rangle$$
      and
      $$SUC(GUP_\emptyset) = \langle f(\overline{r}) \stackrel{?}{=}_\emptyset f(\overline{s})\rangle,$$
      where f is a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.
      ii. r_1 and s_1 are not identical. Then $SUB(GUP_\emptyset)$ and $SUC(GUP_\emptyset)$ are defined in Table 2.

5.3 Unification Procedure - Tree Generation

Projection and transformation can be seen as single steps in a tree generation process. Each node of the tree is labeled either with a unification problem, $SUCCESS$ or $FAILURE$. The edges of the tree are labeled by substitutions. The nodes labeled with $SUCCESS$ or $FAILURE$ are terminal nodes. The nodes labeled with unification problems are non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Given a nonterminal node, let UP be a unification problem attached to it. First, we decide whether UP is unifiable. If the answer is negative, we replace

Table 1. Transformation table for $GUP_\emptyset$ of the form $t_1 \stackrel{?}{=}_\emptyset t_2$ where t_1 and t_2 are neither identical nor non-variable terms with the same head.

t_1	t_2	$SUB(GUP_\emptyset)$	$SUC(GUP_\emptyset)$
Ind. variable	Ind. variable	$\langle\{t_1 \leftarrow t_2\}\rangle$	$SUCCESS$
Ind. variable	Non-variable term	$\langle\rangle$, if t_1 occurs in t_2 $\langle\{t_1 \leftarrow t_2\}\rangle$, otherwise	$FAILURE$ if t_1 occurs in t_2 $SUCCESS$ otherwise
Non-variable term	Ind.variable	Symmetric to the case t_1 - ind. variable, t_2 - non-variable term	Symmetric to the case t_1 - ind. variable, t_2 - non-variable term
Non-variable term	Non-variable term	$\langle\rangle$	$FAILURE$

UP with the new label $FAILURE$. If UP is unifiable, we apply projection or transformation on UP and get $SUB(UP)$ and $SUC(UP)$. If $SUC(UP)$ is $SUCCESS$, the node has a single child with the label $SUCCESS$ and the edge to that node is labeled with $SUB(UP)$. If $SUC(UP) = \langle P_1, \ldots, P_n\rangle$ and $SUB(UP) = \langle \sigma_1, \ldots, \sigma_n\rangle$, the node UP has n children, labeled respectively with $P_1, \ldots, P_n$ and the edge to the P_i node is labeled with σ_i $(1 \leq i \leq n)$.

We design the general unification procedure as a breadth first (level by level) tree generation process. Let $GUP_\emptyset$ be a unification problem. We label the root of the tree with $GUP_\emptyset$ (zero level). First level nodes (the children of the root) of the tree are obtained from the original problem by projection. Starting from the second level, we apply only a transformation step to a unification problem of each node, thus getting new successor nodes. The branch which ends with a node labeled by $SUCCESS$ is called a successful branch. The branch which ends with a node labeled by $FAILURE$ is a failed branch. For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node and attach the obtained substitution to the node together with the unification problem the node was labeled with. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N. Let $\Sigma(GUP_\emptyset)$ be the set of all substitutions associated with the $SUCCESS$ nodes. We call the tree a unification tree for $GUP_\emptyset$ and denote it $UT(GUP_\emptyset)$.

Table 2. Transformation table for $GUP_\emptyset$ of the form $g(r_1, \overline{r}) \stackrel{?}{=}_\emptyset g(s_1, \overline{s})$ where $\overline{r}$, and $\overline{s}$ are possibly empty sequences of terms. The function symbol f in the table is a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

r_1	s_1	$SUB(GUP_\emptyset)$	$SUC(GUP_\emptyset)$
Ind. var.	Ind. var.	$\sigma = SUB(r_1 \stackrel{?}{=}_\emptyset s_1)$	$\langle f(\overline{r}\sigma) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma)\rangle$
Ind. var.	Seq. var.	$\langle \sigma_1, \sigma_2 \rangle$, where $\sigma_1 = \{s_1 \leftarrow r_1\}$, $\sigma_2 = \{s_1 \leftarrow r_1, s_1\}$,	$\langle f(\overline{r}\sigma_1) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma_1)$, $f(\overline{r}\sigma_2) \stackrel{?}{=}_\emptyset f(s_1, \overline{s}\sigma_2)\rangle$
Ind. var.	Non-var. term	$\langle\rangle$, if $SUB(r_1 \stackrel{?}{=}_\emptyset s_1) = \langle\rangle$ $\langle\sigma\rangle = SUB(r_1 \stackrel{?}{=}_\emptyset s_1)$, otherwise	$FAILURE$ if $SUC(r_1 \stackrel{?}{=}_\emptyset s_1) = FAILURE$ $\langle f(\overline{r}\sigma) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma)\rangle$ otherwise
Seq. var.	Ind. var.	Symmetric to the case r_1 - ind. variable s_1 - seq. variable	Symmetric to the case r_1 - ind. variable s_1 - seq. variable
Seq. var.	Seq. var.	$\langle \sigma_1, \sigma_2, \sigma_3 \rangle$, where $\sigma_1 = \{r_1 \leftarrow s_1\}$, $\sigma_2 = \{r_1 \leftarrow s_1, r_1\}$, $\sigma_3 = \{s_1 \leftarrow r_1, s_1\}$	$\langle f(\overline{r}\sigma_1) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma_1)$, $f(r_1, \overline{r}\sigma_2) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma_2)$, $f(\overline{r}\sigma_3) \stackrel{?}{=}_\emptyset f(s_1, \overline{s}\sigma_2)\rangle$
Seq. var.	Non-var. term	$\langle\rangle$, if r_1 occurs in s_1 $\langle \sigma_1, \sigma_2 \rangle$, where $\sigma_1 = \{r_1 \leftarrow s_1\}$, $\sigma_2 = \{r_1 \leftarrow s_1, r_1\}$, otherwise	$FAILURE$ if r_1 occurs in s_1 $\langle\ f(\overline{r}\sigma_1) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma_1)$, $f(r_1, \overline{r}\sigma_2) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma_2)$, otherwise
Non-var. term	Ind.var.	Symmetric to the case r_1 - ind. variable s_1 - non-variable term	Symmetric to the case r_1 - ind. variable s_1 - non-variable term
Non-var. term	Seq. var.	Symmetric to the case r_1 - seq. variable s_1 - non-variable term	Symmetric to the case r_1 - seq. variable s_1 - non-variable term
Non-var. term	Non-var. term	$SUB(r_1 \stackrel{?}{=}_\emptyset s_1)$	$FAILURE$ if $SUC(r_1 \stackrel{?}{=}_\emptyset s_1) = FAILURE$ $\langle f(\overline{r}\sigma) \stackrel{?}{=}_\emptyset f(\overline{s}\sigma)\rangle$, if $SUC(r_1 \stackrel{?}{=}_\emptyset s_1) = SUCCESS$ and $SUB(r_1 \stackrel{?}{=}_\emptyset s_1) = \langle\sigma\rangle$ $\langle f(q_{11}, \overline{r}\sigma_1) \stackrel{?}{=}_\emptyset f(q_{12}, \overline{s}\sigma_1), \ldots,$ $f(q_{k1}, \overline{r}\sigma_k) \stackrel{?}{=}_\emptyset f(q_{k2}, \overline{s}\sigma_k)\rangle$, if $SUC(r_1 \stackrel{?}{=}_\emptyset s_1) =$ $\langle q_{11} \stackrel{?}{=}_\emptyset q_{12}, \ldots, q_{k1} \stackrel{?}{=}_\emptyset q_{k2}\rangle$ and $SUB(r_1 \stackrel{?}{=}_\emptyset s_1) = \langle \sigma_1, \ldots, \sigma_k\rangle$

Example 4. Figure 1 shows development of successful branches in the unification tree for $GUP_\emptyset = f(x, b, \overline{y}, f(\overline{x})) \stackrel{?}{=}_\emptyset f(a, \overline{x} f(b, \overline{y}))$. $\Sigma(GUP_\emptyset) = \{\{x \leftarrow a, \overline{x} \leftarrow b, \overline{x},\ \overline{y} \leftarrow \overline{x}\}, \{x \leftarrow a, \overline{x} \leftarrow b, \overline{y} \leftarrow\}\}$.

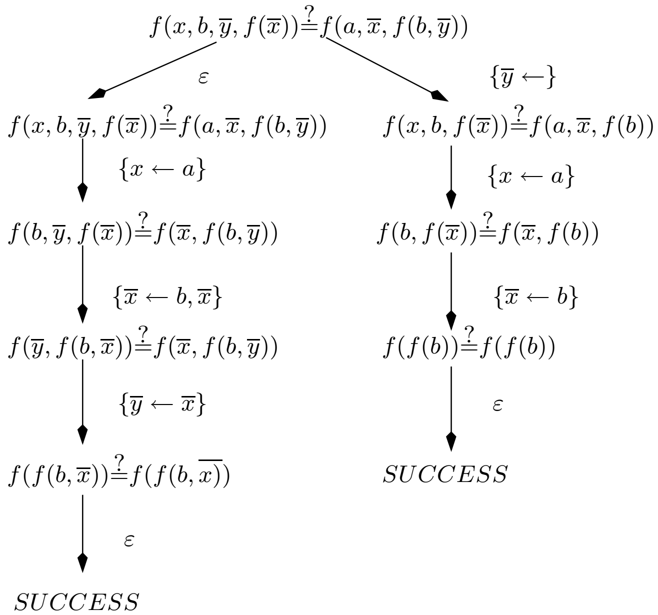

Fig. 1. *Successful branches of* $UT(f(x, b, \overline{y}, f(\overline{x})) \stackrel{?}{=}_\emptyset f(a, \overline{x}, f(b, \overline{y})))$.

Our goal is to prove that $\Sigma(GUP_\emptyset)$ is a minimal complete set of free unifiers for $GUP_\emptyset$. In fact, we will prove a stronger statement: $\Sigma(GUP_\emptyset)$ is a disjoint complete set of free unifiers for $GUP_\emptyset$, where disjointness is defined as follows:

Definition 14 (Disjoint Set of Substitutions). *A set of substitutions Σ is called disjoint modulo E with respect to a set of variables Var iff for all θ, $\sigma \in \Sigma$, if there exist substitutions λ_1, λ_2 such that*

- *for all sequence variables $\overline{x} \in Var$,*
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - *there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and*
- *for all individual variables $x \in Var$,*
  - $x\theta \circ \lambda_1 \doteq_E x\sigma \circ \lambda_2$,

then $\theta = \sigma$.

First we need to establish some preliminary results.

Lemma 1. *If a set of substitutions Σ is E-disjoint with respect to a set of variables Var, then Σ is E-minimal with respect to Var.*

Proof. Let Σ be disjoint with respect to a set of variables Var and θ and σ be two substitutions from Σ such that $\theta \ll_E^{Var} \sigma$. We will show that $\theta = \sigma$.

By Definition 11, from $\theta \ll_E^{Var} \sigma$ we have that there exists a substitution λ such that

- for all $\overline{x} \in Var$,
  - $\overline{x} \leftarrow\ \notin \lambda$;
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma = t_1, \ldots, t_n$, $\overline{x}\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$;
- for all $x \in Var$, $x\sigma \doteq_E x\theta \circ \lambda$.

On the other hand, for the empty substitution ε we have for all $\overline{x} \in Var$,

- $\overline{x} \leftarrow\ \notin \varepsilon$;
- $\overline{x}\sigma = \overline{x}\sigma \circ \varepsilon$.

Therefore, for the substitutions ε and λ we have that

- for all sequence variables $\overline{x} \in Var$,
  - $\overline{x} \leftarrow \notin \varepsilon$,
  - $\overline{x} \leftarrow \notin \lambda$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \varepsilon = t_1, \ldots, t_n$, $\overline{x}\sigma \circ \lambda = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and
- for all individual variables $x \in Var$,
  - $x\theta \circ \varepsilon \doteq_E x\sigma \circ \lambda$,

which, by Definition 14, implies $\theta = \sigma$.

First, we prove completeness.

Lemma 2. *For every free unifier ϕ of a general unification problem $GUP_\emptyset$ there exists a branch β in $UT(GUP_\emptyset)$ with the following property: if P is a unification problem occurring in β with the associated substitution θ, then $\theta \ll_\emptyset^{Var} \phi$, where Var is the set of variables of $GUP_\emptyset$.*

Proof. Let ϕ be an arbitrary free unifier of $GUP_\emptyset$ and Var be the set of variables of $GUP_\emptyset$. We should find a branch β in $UT(GUP_\emptyset)$ such that if P is a unification problem occurring in β and θ is the substitution associated with P in $UT(GUP_\emptyset)$, then $\theta \ll_\emptyset^{Var} \phi$.

We define β recursively.

First, let the root of the tree, labeled with $GUP_\emptyset$, be in β. The substitution associated with $GUP_\emptyset$ is ε. Obviously, $\varepsilon \ll_\emptyset^{Var} \phi$.

Next, let the first level node of β be that successor P of $GUP_\emptyset$, for which the associated substitution $\theta = \{\overline{x} \leftarrow\ \mid \overline{x} \leftarrow\ \in \phi \text{ and } \overline{x} \in Var\}$. We show that $\theta \ll_\emptyset^{Var} \phi$. Let $\lambda = \phi \backslash \theta$. Then we have that $\phi = \theta \circ \lambda$. Therefore

– for all $\overline{x} \in Var$,
  - $\overline{x} \leftarrow\ \notin \lambda$;
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\,\theta \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$;

– for all $x \in Var$, $x\phi \doteq_\emptyset x\theta \circ \lambda$,

which implies that $\theta \ll_\emptyset^{Var} \phi$.

Let now a node in $UT(GUP_\emptyset)$, labeled with P_n, be in β. It means that for the substitution θ_n associated with P_n, it is true that $\theta_n \ll_\emptyset^{Var} \phi$. It implies that there exists a substitution λ such that

– for all $\overline{x} \in Var$,
  - the binding
$$\overline{x} \leftarrow\ \notin \lambda; \tag{3}$$
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or
$$t_i \doteq_\emptyset s_i; \tag{4}$$

– for all $x \in Var$,
$$x\phi \doteq_\emptyset x\theta_n \circ \lambda. \tag{5}$$

If there exists a successor P_{n+1} of P_n in $UT(GUP_\emptyset)$ with the associated substitution θ_{n+1} such that $\theta_{n+1} \ll_\emptyset^{Var} \phi$, we can include P_{n+1} into β and, thus, we will have that for all $P \in \beta$, if θ is a substitution associated with P, then $\theta \ll_\emptyset^{Var} \phi$.

Thus, the problem of constructing β is reduced to finding the successor P_{n+1} of P_n such that for the substitution θ_{n+1}, associated with P_{n+1}, we have $\theta_{n+1} \ll_\emptyset^{Var} \phi$. We show how to find such a P_{n+1}.

The unification problem P_n can have one of the following four forms:

1. P_n is a pair of identical terms.
2. P_n is a pair of individual variables.
3. P_n is a pair of an individual variable and non-variable term.
4. P_n is a pair of non-variable terms.

We consider each of them separately:

1. There is only one possible choice: P_{n+1} is $SUCCESS$ with the associated substitution $\theta_{n+1} = \theta_n \circ \varepsilon$. Then $\theta_{n+1} = \theta_n \ll_\emptyset^{Var} \phi$.

2-3. Let P_n be $x \overset{?}{\doteq}_\emptyset t$, t being either an individual variable or a non-variable term. The substitution $\theta_n \circ \lambda$ is a unifier of $GUP_\emptyset$. The unification problems $GUP_\emptyset\theta_n$ and P_n have exactly the same set of unifiers, because P_n can be obtained from $GUP_\emptyset\theta_n$ by iterated deletion of identical first arguments in both sides of $GUP_\emptyset\theta_n$. Therefore, we have that λ is a unifier of P_n. To obtain P_{n+1} from P_n we choose the substitution $\mu = \{x \leftarrow t\}$. Let θ_{n+1} be the substitution $\theta_n \circ \mu$. Then from 3, 4, 5 and the fact that $\mu \circ \lambda = \lambda$ we get

– for all $\overline{x} \in Var$,
  - $\overline{x} \leftarrow \notin \lambda$;
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \lambda = \overline{x}\theta_n \circ \mu \circ \lambda = \overline{x}\theta_{n+1} \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$;

– for all $x \in Var$, $x\phi \doteq_\emptyset x\theta_n \circ \lambda \doteq_\emptyset x\theta_n \circ \mu \circ \lambda \doteq_\emptyset x\theta_{n+1} \circ \lambda$.

Thus, $\theta_{n+1} =\ll_\emptyset^{Var} \phi$.

4. Let P_n have the form $g(t_1, \overline{t}) \stackrel{?}{\doteq}_\emptyset g(s_1, \overline{s})$, where $\overline{t}$ and $\overline{s}$ are (possibly empty) sequences of terms. We can have the following 6 cases with respect to t_1 and s_1:

4.1. t_1 and s_1 are identical.
4.2. t_1 and s_1 are individual variables.
4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable term.
4.4. t_1 and s_1 are sequence variables.
4.5. One of t_1 and s_1 is a sequence variable, the other is not.
4.6. t_1 and s_1 are non-variable terms.

We consider each of these cases.

4.1-4.3. In these cases P_{n+1} can be chosen analogously to the cases 1-3 above.

4.4. Suppose t_1 is a sequence variable $\overline{x}$ and s_1 is a sequence variable $\overline{y}$. Let us define substitutions μ and ν as follows:

– $\mu = \{\overline{x} \leftarrow \overline{y}\}$, $\nu = \{v \leftarrow v\lambda \mid v \in Var, v \neq \overline{x}, v \neq v\lambda\}$, if $f(\overline{x})\lambda \doteq_\emptyset f(\overline{y})\lambda$;

– $\mu = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$, $\nu = \{\overline{x} \leftarrow \overline{r}\} \circ \{v \leftarrow v\lambda \mid v \in Var, v \neq \overline{x}, v \neq v\lambda\}$, if there exists a non-empty sequence of terms $\overline{r}$ such that $f(\overline{x})\lambda \doteq_\emptyset f(\overline{y}, \overline{r})\lambda$;

– $\mu = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$, $\nu = \{\overline{y} \leftarrow \overline{r}\} \circ \{v \leftarrow v\lambda \mid v \in Var, v \neq \overline{y}, v \neq v\lambda\}$, if there exists a non-empty sequence of terms $\overline{r}$ such that $f(\overline{y})\lambda \doteq_\emptyset f(\overline{x}, \overline{r})\lambda$,

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

We obtain P_{n+1} from P_n by the substitution μ (since λ is a unifier of P_n, these three cases for μ are the only possibilities to get P_{n+1} from P_n). Therefore, $\theta_{n+1} = \theta_n \circ \mu$.

On the other hand, for all $\overline{x} \in Var$,

$$\overline{x} \leftarrow \notin \mu. \tag{6}$$

From 6, 3 and definitions of μ and ν, by Definition 9 we get for all $\overline{x} \in Var$

$$\overline{x} \leftarrow \notin \nu. \tag{7}$$

From 4 and definitions of μ and ν we get that for all $\overline{x} \in Var$ there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \lambda = \overline{x}\theta_n \circ \mu \circ \nu = \overline{x}\theta_{n+1} \circ \nu = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or

$$t_i \doteq_\emptyset s_i; \tag{8}$$

From 5 and definitions of μ and ν we get that for all $x \in Var$

$$x\phi \doteq_{\emptyset} x\theta_n \circ \lambda \doteq_{\emptyset} x\theta_n \circ \mu \circ \nu \doteq_{\emptyset} x\theta_{n+1} \circ \nu. \tag{9}$$

From 7, 8 and 9, by Definition 11 we get $\theta_{n+1} \ll_{\emptyset}^{Var} \phi$.

4.5. This case can be proved similarly to the case 4.4, considering only two cases for μ.

4.6. This case recursively can be reduced to one of the cases 4.1-4.4 above.

Thus, for all possible forms of P_n we found its successor P_{n+1} such that for the substitution θ_{n+1}, associated with P_{n+1}, we have $\theta_{n+1} \ll_{\emptyset}^{Var} \phi$. It finishes the proof.

Theorem 8. *$\Sigma(GUP_{\emptyset})$ is a complete set of free unifiers for $GUP_{\emptyset}$.*

Proof. The theorem follows from Lemma 2 by the definition of Σ.

Next, we want to show that $\Sigma(GUP_{\emptyset})$ is a disjoint set of free unifiers for $GUP_{\emptyset}$. First, we prove that for any general $\emptyset$-unification problem P, the set $SUB(P)$ is disjoint with respect to the set of variables of UP. After that we show that the substitutions associated with distinct successful leaves in $UT(GUP_{\emptyset})$ are disjoint with respect to the set of variables of $GUP_{\emptyset}$.

Lemma 3. *Let P be a general $\emptyset$-unification problem. Then $SUB(P)$ is disjoint with respect to the set of variables of P.*

Proof. Let Var be the set of variables of P. We show that $SUB(P)$ is disjoint with respect to Var.

Let $SUB(P)$ be the set of projecting substitutions. Assume by contradiction that $SUB(P)$ is not disjoint with respect to Var. Then there exist two substitutions θ and σ from $SUB(P)$ and two substitutions λ_1 and λ_2 such that

- for all sequence variables $\overline{x} \in Var$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \dots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \dots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_{\emptyset} s_i$ and
- for all individual variables $x \in Var$,
  - $x\theta \circ \lambda_1 \doteq_{\emptyset} x\sigma \circ \lambda_2$.

Since $\theta \neq \sigma$, without loss of generality we can assume that there exists a sequence variable $\overline{x} \in Var$ such that $\overline{x} \leftarrow \in \theta$ and $\overline{x} \leftarrow \notin \sigma$. Then $\overline{x}\theta \circ \lambda_1$ is the empty sequence, while $\overline{x}\sigma = \overline{x}$. The only way to make $\overline{x}\sigma \circ \lambda_2$ the empty sequence is to have $\overline{x} \leftarrow \in \lambda_2$, but it contradicts the assumption that $\overline{x} \leftarrow \notin \lambda_2$. The obtained contradiction proves that the set $SUB(P)$ of projecting substitutions of P is disjoint with respect to Var.

Now let $SUB(P)$ be the set of transformation substitutions. Since, by Definition 14, the empty set and a singleton are trivially disjoint modulo any E

with respect to any set of variables, we consider only the cases when $SUB(P)$ is neither empty set nor a singleton. Let P have a form $g(r_1, \overline{r}) \stackrel{?}{\doteq}_{\emptyset} g(s_1, \overline{s})$ where $\overline{r}$, and $\overline{s}$ are possibly empty sequences of terms and $g \in FFIX \cup FFLEX$. The we have the following three cases with respect to the form of r_1 and s_1:

1. r_1 and s_1 are sequence variables.
2. One of r_1 and s_1 is a sequence variable, the other is not.
3. r_1 and s_1 are non-variable terms.

We prove in each case that $SUB(P)$ is disjoint with respect to Var.

1. Let r_1 be $\overline{x}$ and s_1 be $\overline{y}$. Then $SUB(P) = \{\eta_1, \eta_2, \eta_3\}$, where $\eta_1 = \{\overline{x} \leftarrow \overline{y}\}$, $\eta_2 = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$ and $\eta_3 = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$. Assume by contradiction that $SUB(P)$ is not disjoint. Then for some θ and σ from $SUB(P)$ there exist substitutions λ_1, λ_2 such that
   - for all sequence variables $\overline{x} \in Var$,
     - $\overline{x} \leftarrow \notin \lambda_1$,
     - $\overline{x} \leftarrow \notin \lambda_2$,
     - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and
   - for all individual variables $x \in Var$,
     - $x\theta \circ \lambda_1 \doteq_E x\sigma \circ \lambda_2$.

   Let θ be η_1 and σ be η_2. Then
   - for $\overline{x} \in Var$ we have $\overline{x}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{x}\sigma \circ \lambda_2 = \overline{y}\lambda_2, \overline{x}\lambda_2$;
   - for $\overline{y} \in Var$ we have $\overline{y}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{y}\sigma \circ \lambda_2 = \overline{y}\lambda_2$,

   which implies that $\overline{x}\lambda_2$ must be the empty sequence, but it is impossible because $\overline{x} \leftarrow \notin \lambda_2$.
   Now let θ be η_1 and σ be η_3. Then
   - for $\overline{x} \in Var$ we have $\overline{x}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{x}\sigma \circ \lambda_2 = \overline{x}\lambda_2$;
   - for $\overline{y} \in Var$ we have $\overline{y}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{y}\sigma \circ \lambda_2 = \overline{y}\lambda_2, \overline{x}\lambda_2$,

   which implies that $\overline{y}\lambda_2$ must be the empty sequence, but it is impossible because $\overline{y} \leftarrow \notin \lambda_2$.
   Now let θ be η_2 and σ be η_3. Then
   - for $\overline{x} \in Var$ we have $\overline{x}\theta \circ \lambda_1 = \overline{y}\lambda_1, \overline{x}\lambda_1$ and $\overline{x}\sigma \circ \lambda_2 = \overline{x}\lambda_2$;
   - for $\overline{y} \in Var$ we have $\overline{y}\theta \circ \lambda_1 = \overline{y}\lambda_1$ and $\overline{y}\sigma \circ \lambda_2 = \overline{y}\lambda_2, \overline{x}\lambda_2$,

   which implies that $\overline{x}\lambda_2$ and $\overline{y}\lambda_2$ must be the empty sequences, but it is impossible because $\overline{x} \leftarrow \notin \lambda_2$ and $\overline{y} \leftarrow \notin \lambda_2$.
   Thus, in all possible cases we got contradiction, which proves that $SUB(P)$ is disjoint with respect to Var.
2. This case can be proved similarly to the case 1.
3. This case can be reduced recursively to one of the previous cases.

Now we show that the substitutions associated with distinct successful leaves in $UT(GUP_{\emptyset})$ are disjoint with respect to the set of variables of $GUP_{\emptyset}$. For this, first we introduce the notion of D-preserving (disjointness-preserving) substitution due to Schulz [28].

Definition 15 (D-preserving substitution). *A substitution θ is D-preserving modulo E with respect to a set Var of variables iff for any two substitutions σ_1 and σ_2, if the set $\{\sigma_1, \sigma_2\}$ is disjoint modulo E with respect to the set of variables $\cup_{v \in Var} VarSet(v\theta)$, then the set of substitutions $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint modulo E with respect to Var.*

Lemma 4. *For a general unification problem $GUP_\emptyset$ any of the projecting or transformation substitutions from $SUB(GUP_\emptyset)$ is D-preserving modulo $\emptyset$-theory with respect to the set of variables of $GUP_\emptyset$.*

Proof. Let Var be the set of variables of $GUP_\emptyset$.

First we show that any projecting substitution preserves disjointness modulo $\emptyset$-theory with respect to Var. Let π be a projecting substitution for $GUP_\emptyset$. If $\pi = \varepsilon$, then, by Definition 15, π is trivially D-preserving modulo $\emptyset$-theory. Assume $\pi \neq \varepsilon$. Let σ_1 and σ_2 be two substitutions such that the set $\{\sigma_1, \sigma_2\}$ is disjoint modulo E with respect to the set of variables $\cup_{v \in Var} VarSet(v\theta) = Var \backslash \{\overline{x} \,|\, \overline{x} \leftarrow\, \in \pi\}$. We have to show that $\{\pi \circ \sigma_1, \pi \circ \sigma_2\}$ is disjoint modulo $\emptyset$-theory with respect to Var. To prove this, by Definition 14, we assume that there exist substitutions λ_1, λ_2 such that

- for all sequence variables $\overline{x} \in Var$,
  - $\overline{x} \leftarrow\, \notin \lambda_1$,
  - $\overline{x} \leftarrow\, \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\pi \circ \sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\pi \circ \sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
- for all individual variables $x \in Var$,
  - $x\pi \circ \sigma_1 \circ \lambda_1 \doteq_\emptyset x\pi \circ \sigma \circ \lambda_2$,

and prove $\pi \circ \sigma_1 = \pi \circ \sigma_2$.

Since $Var \backslash \{\overline{x} \,|\, \overline{x} \leftarrow\, \in \pi\} \subset Var$, we have for all (sequence or individual) variables $v \in Var \backslash \{\overline{x} \,|\, \overline{x} \leftarrow\, \in \pi\}$ that $v\pi = v$. Then from the assumption we have that

- for all sequence variables $\overline{x} \in Var \backslash \{\overline{x} \,|\, \overline{x} \leftarrow\, \in \pi\}$
  - $\overline{x} \leftarrow\, \notin \lambda_1$,
  - $\overline{x} \leftarrow\, \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$,
- for all individual variables $x \in Var$,
  - $x\sigma_1 \circ \lambda_1 \doteq_\emptyset x\sigma \circ \lambda_2$,

which, by disjointness of the set $\{\sigma_1, \sigma_2\}$, implies that $\sigma_1 = \sigma_2$. Therefore, $\pi \circ \sigma_1 = \pi \circ \sigma_2$.

Now we prove the lemma for the transformation substitutions. There are the following 4 possible forms of the unification problem $GUP_\emptyset$ the transformation step of the unification procedure deals with:

1. $GUP_\emptyset$ is a pair of identical terms.
2. $GUP_\emptyset$ is a pair of individual variables.
3. $GUP_\emptyset$ is a pair of an individual variable and non-variable term.
4. $GUP_\emptyset$ is a pair of non-variable terms.

We will prove in each case that transformation substitution is a D-preserving substitution with respect to the set of variables of $GUP_\emptyset$.

1. In this case the transformation substitution is ε. The fact that it preserves disjointness, directly follows from the definition of D-preserving substitution.

2. Let $GUP_\emptyset$ have a form $x \stackrel{?}{\doteq}_\emptyset y$ and the corresponding transformation substitution be $\theta = \{x \leftarrow y\}$. Then $Var = \{x, y\}$ and $\cup_{v \in Var} VarSet(v\theta) = \{y\}$. Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to $\{y\}$. We have to show that $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to $\{x, y\}$. Assume by contradiction that it is not disjoint. Then there exist substitutions λ_1 and λ_2 such that for all $v \in \{x, y\}$, $v\theta \circ \sigma_1 \circ \lambda_1 \doteq_\emptyset v\theta \circ \sigma_2 \circ \lambda_2$, in particular, $y\theta \circ \sigma_1 \circ \lambda_1 \doteq_\emptyset y\theta \circ \sigma_2 \circ \lambda_2$. But since $y\theta = y$, we get $y\sigma_1 \circ \lambda_1 \doteq_\emptyset y\sigma_2 \circ \lambda_2$, which contradicts to disjointness of $\{\sigma_1, \sigma_2\}$ with respect to $\{y\}$. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to Var, which implies that θ preserves disjointness on the set of variables of $GUP_\emptyset$.

3. Let $GUP_\emptyset$ have a form $x \stackrel{?}{\doteq}_\emptyset t$ and the corresponding transformation substitution be $\theta = \{x \leftarrow t\}$. Then we have that $Var = \{x\} \cup VarSet(t)$ and $\cup_{v \in Var} VarSet(v\theta) = VarSet(t)$. Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to Var. We have to show that $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to Var. Assume by contradiction that it is not disjoint. Then there exist substitutions λ_1 and λ_2 such that
   - for all sequence variables $\overline{x} \in Var$,
     - $\overline{x} \leftarrow \notin \lambda_1$,
     - $\overline{x} \leftarrow \notin \lambda_2$,
     - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\theta \circ \sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
   - for all individual variables $x \in Var$,
     - $x\theta \circ \sigma_1 \circ \lambda_1 \doteq_\emptyset x\theta \circ \sigma_2 \circ \lambda_2$.

   Then we have
   - for all sequence variables $\overline{x} \in VarSet(t)$,
     - $\overline{x} \leftarrow \notin \lambda_1$,
     - $\overline{x} \leftarrow \notin \lambda_2$,
     - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, $t_i \doteq_\emptyset s_i$ and
   - for all individual variables $x \in VarSet(t)$,
     - $x\sigma_1 \circ \lambda_1 \doteq_\emptyset x\sigma_2 \circ \lambda_2$.

   It implies that $\{\sigma_1, \sigma_2\}$ is not disjoint with respect to $VarSet(t)$, which is a contradiction. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect Var, which implies that θ preserves disjointness on Var.

4. Let $GUP_\emptyset$ have the form $f(t_1,\overline{t})\stackrel{?}{\doteq}_\emptyset f(s_1,\overline{s})$, where $\overline{t}$ and $\overline{s}$ are (possibly empty) sequences of terms. We have the following 6 cases with respect to t_1 and s_1:

4.1. t_1 and s_1 are identical.

4.2. t_1 and s_1 are individual variables.

4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable term.

4.4. t_1 and s_1 are sequence variables.

4.5. One of t_1 and s_1 is a sequence variable, the other is not.

4.6. t_1 and s_1 are non-variable terms.

We show in each subcase separately that the corresponding transformation substitution is D-preserving with respect to the set of variables of UP.

4.1-4.3. These cases can be proved analogously to the cases 1-3 above.

4.4 Let $GUP_\emptyset$ have a form $\overline{x}\stackrel{?}{\doteq}_\emptyset\overline{y}$. Then we have the following cases for the transformation substitution θ:

(a) $\theta = \{\overline{x} \leftarrow \overline{y}\}$;

(b) $\theta = \{\overline{x} \leftarrow \overline{y}, \overline{x}\}$;

(c) $\theta = \{\overline{y} \leftarrow \overline{x}, \overline{y}\}$.

Then $Var = \{\overline{x},\overline{y}\}\cup VarSet(\overline{t})\cup VarSet(\overline{s})$. Let σ_1 and σ_2 be two substitutions such that $\{\sigma_1,\, \sigma_2\}$ is disjoint with respect to $\cup_{v\in Var}VarSet(v\theta)$. We have to show that $\{\theta\circ\sigma_1,\, \theta\circ\sigma_2\}$ is disjoint with respect to Var. Assume by contradiction that it is not disjoint. Then there exist substitutions λ_1 and λ_2 such that

- for all sequence variables $\overline{x}\in Var$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1,\dots,t_n,s_1,\dots,s_n$, $n\geq 0$ such that $\overline{x}\theta\circ\sigma_1\circ\lambda_1 = t_1,\dots,t_n$, $\overline{x}\theta\circ\sigma_2\circ\lambda_2 = s_1,\dots,s_n$ and for all $1\leq i\leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
- for all individual variables $x\in Var$,
  - $x\theta\circ\sigma_1\circ\lambda_1 \doteq_\emptyset x\theta\circ\sigma_2\circ\lambda_2$.

Let us consider each case separately:

(a) In this case $\cup_{v\in Var}VarSet(v\theta) = Var\backslash\{\overline{x}\}$ and for all $\overline{z}\in Var\backslash\{\overline{x}\}$ we have $\overline{z}\theta = \overline{z}$. Then for λ_1 and λ_2 we have
- for all sequence variables $\overline{z}\in Var\backslash\{\overline{x}\}$,
  - $\overline{z} \leftarrow \notin \lambda_1$,
  - $\overline{z} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1,\dots,t_n,s_1,\dots,s_n$, $n\geq 0$ such that $\overline{z}\sigma_1\circ\lambda_1 = t_1,\dots,t_n$, $\overline{z}\sigma_2\circ\lambda_2 = s_1,\dots,s_n$ and for all $1\leq i\leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_\emptyset s_i$ and
- for all individual variables $x\in Var$,
  - $x\sigma_1\circ\lambda_1 \doteq_\emptyset x\sigma_2\circ\lambda_2$,

which implies that $\{\sigma_1,\, \sigma_2\}$ is disjoint with respect to Var. But this is a contradiction. Thus, $\{\theta\circ\sigma_1,\, \theta\circ\sigma_2\}$ is disjoint with respect to Var, which implies that θ preserves disjointness on the set of variables of $GUP_\emptyset$.

(b) In this case $\cup_{v \in Var} VarSet(v\theta) = Var$. For $\overline{x} \in Var$ there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \sigma_1 \circ \lambda_1 = (\overline{y}, \overline{x})\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\theta \circ \sigma_2 \circ \lambda_2 = (\overline{y}, \overline{x})\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_{\emptyset} s_i$. On the other hand for $\overline{y} \in Var$, we have that for some $m \leq n$, $\overline{y}\theta \circ \sigma_1 \circ \lambda_1 = \overline{y}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_m$, $\overline{y}\theta \circ \sigma_2 \circ \lambda_2 = \overline{y}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_m$. Therefore, $\overline{x}\sigma_1 \circ \lambda_1 = t_m, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_m, \ldots, s_n$. As for the individual variables, for all $x \in Var$ we have $x\theta = x$. Thus, we got that there exist substitutions λ_1 and λ_2 such that
   - for all sequence variables $\overline{x} \in Var$,
     - $\overline{x} \leftarrow \notin \lambda_1$,
     - $\overline{x} \leftarrow \notin \lambda_2$,
     - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_{\emptyset} s_i$ and
   - for all individual variables $x \in Var$,
     - $x\sigma_1 \circ \lambda_1 \doteq_{\emptyset} x\sigma_2 \circ \lambda_2$,

   which implies that $\{\sigma_1, \sigma_2\}$ is disjoint with respect to Var. But this is a contradiction. Thus, $\{\theta \circ \sigma_1, \theta \circ \sigma_2\}$ is disjoint with respect to Var, which implies that θ preserves disjointness on the set of variables of $GUP_{\emptyset}$.

(c) Proof in this case is symmetric to the case b).

4.5. This case can be proved similarly to the case 4.4.

4.6. This case recursively can be reduced to one of the cases 4.1-4.4 above.

Lemma 5. *Let θ_1 and θ_2 be two substitutions such that $\{\theta_1, \theta_2\}$ is disjoint modulo E with respect to a set of variables Var. Let σ_1 and σ_2 be two substitutions such that for all $\overline{x} \in Var$, $\overline{x} \leftarrow \notin \sigma_1$ and $\overline{x} \leftarrow \notin \sigma_2$. Then $\{\theta_1 \circ \sigma_1, \theta_2 \circ \sigma_2\}$ is disjoint modulo E with respect to Var.*

Proof. Assume by contradiction that $\{\theta_1 \circ \sigma_1, \theta_2 \circ \sigma_2\}$ is not disjoint modulo E with respect to Var. Then there exist substitutions λ_1, λ_2 such that

- for all sequence variables $\overline{x} \in Var$,
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta_1 \circ \sigma_1 \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\theta_2 \circ \sigma_2 \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and
- for all individual variables $x \in Var$,
  - $x\theta_1 \circ \sigma_1 \circ \lambda_1 \doteq_E x\theta_2 \circ \sigma_2 \circ \lambda_2$,

Let μ_1 be $\sigma_1 \circ \lambda_1$ and μ_2 be $\sigma_2 \circ \lambda_2$. Then we have

- for all sequence variables $\overline{x} \in Var$,
  - $\overline{x} \leftarrow \notin \mu_1$,
  - $\overline{x} \leftarrow \notin \mu_2$,

- there exist terms $t_1, \dots, t_n, s_1, \dots, s_n$, $n \geq 0$ such that $\overline{x}\theta_1 \circ \mu_1 = t_1, \dots, t_n$, $\overline{x}\theta_2 \circ \mu_2 = s_1, \dots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or $t_i \doteq_E s_i$ and

– for all individual variables $x \in Var$,
  - $x\theta_1 \circ \mu_1 \doteq_E x\theta_2 \circ \mu_2$,

which, by Definition 14 implies that $\{\theta_1, \theta_2\}$ is not disjoint modulo E with respect to Var. This is a contradiction which proves that $\{\theta_1 \circ \sigma_1, \theta_2 \circ \sigma_2\}$ is disjoint modulo E with respect to Var.

Theorem 9. *$\Sigma(GUP_\emptyset)$ is a disjoint set of $\emptyset$-unifiers for $GUP_\emptyset$ with respect to the set of variables of $GUP_\emptyset$.*

Proof. Let Var be the set of variables of $GUP_\emptyset$. Suppose σ_1 and σ_2 are two substitutions associated with distinct successful leaves of $UT(GUP_\emptyset)$. We prove that σ_1 and σ_2 are disjoint with respect to Var.

Let ρ be the common part (edges) of branches leading to the respective leaves σ_1 and σ_2 are associated with. Then the substitutions σ_1 and σ_2 can be represented respectively as $\gamma \circ \theta_1 \circ \cdots \circ \theta_k$ and $\gamma \circ \lambda_1 \circ \cdots \circ \lambda_n$, where γ is the substitution associated with the last node in ρ.

First, we show that γ is D-preserving modulo $\emptyset$-theory with respect to Var.

Let P_n denote the unification problem at the n-th node of ρ, Var_n be the set of variables of P_n and γ_n be the substitution associated with P_n. We show that for any n, the substitution γ_n in ρ is D-preserving modulo $\emptyset$-theory with respect to Var. We use induction on n.

Induction base. $n = 0$. Then $\gamma_n = \varepsilon$ which, by Definition 15, is D-preserving modulo $\emptyset$-theory with respect to Var.

Induction hypothesis. We assume that γ_n is D-preserving modulo $\emptyset$-theory with respect to Var for $n > 0$.

Induction step. We show that γ_{n+1} is D-preserving modulo $\emptyset$-theory with respect to Var. By Definition 15, it suffices to show that for any two substitutions η_1 and η_2, if the set $\{\eta_1, \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables $\cup_{v \in Var} VarSet(v\gamma_{n+1})$, then the set of substitutions $\{\gamma_{n+1} \circ \eta_1, \gamma_{n+1} \circ \eta_2\}$ is disjoint modulo $\emptyset$ with respect to Var.

Let η_1 and η_2 be two substitutions such that the set $\{\eta_1, \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables $\cup_{v \in Var} VarSet(v\gamma_{n+1})$. We have to show that $\{\gamma_{n+1} \circ \eta_1, \gamma_{n+1} \circ \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to Var.

Let μ be a substitution such that $\gamma_{n+1} = \gamma_n \circ \mu$. Since, By Lemma 4 μ is D-preserving modulo $\emptyset$-theory with respect to Var_n, $\{\mu \circ \eta_1, \mu \circ \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables $Var_n = \cup_{v \in Var} VarSet(v\gamma_n)$. By induction hypothesis, γ_n is D-preserving modulo $\emptyset$-theory with respect to Var. Then, by Definition 15, $\{\gamma_n \circ \mu \circ \eta_1, \gamma_n \circ \mu \circ \eta_2\} = \{\gamma_{n+1} \circ \eta_1, \gamma_{n+1} \circ \eta_2\}$ is disjoint modulo $\emptyset$-theory with respect to Var.

Thus, any substitution γ_n in ρ is D-preserving modulo $\emptyset$-theory with respect to Var. In particular, γ preserves disjointness modulo $\emptyset$-theory with respect to Var.

Now, we show that the set $\{\theta_1 \circ \cdots \circ \theta_k,\, \lambda_1 \circ \cdots \circ \lambda_n\}$ is disjoint modulo $\emptyset$-theory with respect to the set of variables Var_ρ of the last node of ρ.

By Lemma 3, the set $\{\theta_1,\, \lambda_1\}$ is disjoint modulo $\emptyset$-theory with respect to Var_ρ. The substitutions $\theta_2, \ldots, \theta_k, \lambda_2, \ldots, \lambda_n$ are transformations, therefore, for all $\overline{x}$ from Var_ρ, $\overline{x} \leftarrow \notin \theta_2 \circ \cdots \circ \theta_k$ and $\overline{x} \leftarrow \notin \lambda_2 \circ \cdots \circ \lambda_n$. Then, by Lemma 5, the set $\{\theta_1 \circ \cdots \circ \theta_k,\, \lambda_1 \circ \cdots \circ \lambda_n\}$ is disjoint modulo $\emptyset$-theory with respect to Var_ρ.

Note that $Var_\rho = \cup_{v \in Var} VarSet(v\gamma)$. Therefore, from the fact that γ is D-preserving with respect to Var and $\{\theta_1 \circ \cdots \circ \theta_k,\, \lambda_1 \circ \cdots \circ \lambda_n\}$ is disjoint with respect to Var_ρ, by Definition 15 we get that $\{\gamma \circ \theta_1 \circ \cdots \circ \theta_k,\, \gamma \circ \lambda_1 \circ \cdots \circ \lambda_n\} = \{\sigma_1,\, \sigma_2\}$ is disjoint modulo $\emptyset$-theory with respect to Var.

Theorem 10. *Let $GUP_\emptyset$ be a general $\emptyset$-unification problem. Then*

$$\Sigma(GUP_\emptyset) = MCU_\emptyset(GUP_\emptyset).$$

Proof. By Theorem 8, Theorem 9 and Lemma 1.

The last results in this section give sufficient conditions for termination of the procedure.

Theorem 11. *The unification procedure for $GUP_\emptyset$ terminates if $GUP_\emptyset$ contains no sequence variables.*

Proof. Assume $GUP_\emptyset$ contains no sequence variables. Then for each term with a flexible arity head $fflex$ and the number of arguments n, we replace each occurrence of $fflex$ with a new n-ary function symbols $ffix$. Then termination of the unification procedure for $GUP_\emptyset$ follows from termination of Robinson unification algorithm.

Another terminating case is when one of the terms to be unified is ground. It yields to the following result:

Theorem 12. *Matching in the free theory with individual and sequence variables, free constants, free fixed and flexible arity function symbols is finitary.*

Proof. Let $GUP_\emptyset = t_1 \stackrel{?}{=} t_2$ be a unification problem such that t_2 is ground. Then each transformation step in the unification procedure strictly reduces the size of t_2, which eventually leads to termination of the procedure.

We can add a loop-checking method to the procedure: stop with failure if a unification problem attached to a node of unification tree coincides with a unification problem in the same branch of the tree. Then the following theorem holds:

Theorem 13. *The unification procedure with loop-checking for $GUP_\emptyset$ terminates if no variable occurs more than twice in $GUP_\emptyset$.*

Proof. If no variable occurs more than twice in $GUP_\emptyset$, then no transformation rule increases the size of unification problems. Then we can have only finitely many unification problems in the tree.

If $GUP_\emptyset = t_1 \stackrel{?}{=} t_2$ has the property that sequence variables occur only as arguments of t_1 or t_2, then we can weaken the previous theorem:

Theorem 14. *The unification procedure with loop-checking for $GUP_\emptyset = t_1 \stackrel{?}{=} t_2$, where sequence variables occur only as arguments of t_1 or t_2, terminates if no sequence variable occurs more than twice in $GUP_\emptyset$.*

Proof. Using the same argument as in the proof of previous theorem.

The following termination condition does not require loop-checking and does not depend on the number of occurrences of sequence variables. Instead, it requires for a unification problem of the form $f(\overline{x}) \stackrel{?}{=}_\emptyset f(t_1, \ldots, t_n)$, $n > 1$, to check whether $\overline{x}$ occurs in $f(t_1, \ldots, t_n)$. We call it *the sequence variable occurrence checking.* We can tailor this checking into the unification tree generation process as follows: if in the tree a successor of the unification problem of the form $f(\overline{x}) \stackrel{?}{=}_\emptyset f(t_1, \ldots, t_n)$, $n > 1$, has to be generated, perform the sequence variable occurrence checking. If $\overline{x}$ occurs in $f(t_1, \ldots, t_n)$, label the node with $FAILURE$, otherwise proceed in the usual way (projection or transformation).

Theorem 15. *If $GUP_\emptyset$ is a unification problem such that all sequence variables occurring in $GUP_\emptyset$ are only the last arguments of the term they occur, then the unification procedure with the sequence variable occurrence checking terminates.*

Proof. In this case, every transformation step involving a sequence variable either immediately generates a terminal node in the tree, or produces a new unification problem with the size strictly smaller then the size of the unification problem in the predecessor node. Then we can have only a finite number of nodes in the tree.

The fact that in most of the applications sequence variables occur precisely only at the last position in terms, underlines the importance of Theorem 15. The theorem provides an efficient method to terminate unification procedure in many practical applications.

6 First Extension: Pattern $\emptyset$-Unification

In this section we discuss extension of unification with sequence variables and flexible arity symbols with patterns. First we extend the language by introducing a new set of variables XV - called index variables - and the set LP of linear polynomials with integer coefficients, whose variables are in XV.

Definition 16 (Patterns). *A set of patterns PAT (over (V, C, P, LP)) is the smallest set satisfying the following conditions:*

- If $c \in CONST$, $m, k \in LP$, then $c_{m,k}$ is a pattern.
- If $f \in FFIX$, $AR(f) = n$, $n \geq 0$ and $t_1, \ldots, t_n$ are terms such that for all $1 \leq i \leq n$, $t_i \notin SV$, $m, k \in LP$, then $f_{m,k}(t_1, \ldots, t_n)$ is a pattern.
- If $f \in FFLEX$, each of $t_1, \ldots, t_n$ $(n \geq 0)$ is a term or pattern, $m, k \in LP$, then $f_{m,k}(t_1, \ldots, t_n)$ is a pattern.

f is called the head of $f_{m,k}(t_1, \ldots, t_n)$. We say that a pattern $c_{m,k}$ or $f_{m,k}(t_1, \ldots, t_n)$ is explicit iff m and k are positive integers.

If not otherwise stated, the following symbols, with or without indices, are used as metavariables: vm, vn, vk - over index variables, cm, cn, ck - over integer constants, vcm, vcn, vck - over index variables and integer constants.

Definition 17 (Quasi Pattern-Term). *A set of quasi pattern-terms (over (V, C, P, LP)), or, shortly, QP-terms, is the smallest set satisfying the following conditions:*

- If $v \in IV \cup SV$ then v is a quasi pattern-term.
- If $c \in CONST$ then c is a quasi pattern-term.
- If $p \in PAT$ then p is a quasi pattern-term.
- If $f \in FFIX$, $AR(f) = n$, $n \geq 0$ and $t_1, \ldots, t_n$ are quasi pattern-terms such that for all $1 \leq i \leq n$, $t_i \notin SV \cup PAT$, then $f(t_1, \ldots, t_n)$ is a quasi pattern-term.
- If $f \in FFLEX$ and $t_1, \ldots, t_n$ are quasi pattern-terms, then $f(t_1, \ldots, t_n)$ is a quasi pattern-term $(n \geq 0)$.

f is called the head of $f(t_1, \ldots, t_n)$.

Definition 18 (Quasi Pattern-Equation). *A set of quasi pattern-equations (over (V, C, P, LP)), or, shortly, QP-equations, is the smallest set satisfying the following property: If t_1 and t_2 are QP-terms over (V, C, P, LP) such that $t_1 \notin SV \cup PAT$ and $t_2 \notin SV \cup PAT$, then $\doteq (t_1, t_2)$ is a quasi pattern-equation over (V, C, P, LP). The symbol $\doteq$ is called the head of the quasi pattern-equation $\doteq (t_1, t_2)$.*

We write QP-equations in infix notation.

Definition 19 (Quasi Pattern-Substitution). *A quasi pattern-substitution, or, shortly, QP-substitution, is a finite set*

$$\begin{array}{l} \{\ x_1 \leftarrow s_1, \ldots, x_n \leftarrow s_n, \\ \quad \overline{x}_1 \leftarrow t_1^1, \ldots, t_{k_1}^1, \ldots, \overline{x}_m \leftarrow t_1^m, \ldots, t_{k_m}^m, \\ \quad vn_1 \leftarrow p_1, \ldots, vn_l \leftarrow p_l \qquad\qquad\qquad\qquad \} \end{array}$$

where

- $n \geq 0$, $m \geq 0$ and for all $1 \leq i \leq m$, $k_i \geq 0$,
- $x_1, \ldots, x_n$ are distinct individual variables,
- $\overline{x}_1, \ldots, \overline{x}_m$ are distinct sequence variables,

– $nv_1, \ldots, nv_l$ *are distinct index variables,*
– *for all* $1 \leq i \leq n$, s_i *is a QP-term,* $s_i \notin SV \cup PAT$ *and* $s_i \neq x_i$,
– *for all* $1 \leq i \leq m$, $t^i_1, \ldots, t^i_{k_i}$ *is a sequence of QP-terms and if* k_i=1 *then* $t^i_{k_i} \neq \overline{x}_i$,
– *for all* $1 \leq i \leq l$, $p_i \in LP$ *and* $p_i \neq vn_i$.

Each $x_i \leftarrow s_i$, $\overline{x}_i \leftarrow t^i_1, \ldots, t^i_{k_i}$ *and* $vn_i \leftarrow p_i$ *is called a QP-binding respectively for* x_i, $\overline{x_i}$ *and* vn_i.

A QP-substitution is called ground iff all $s_1, \ldots, s_n, t^1_1, \ldots, t^1_{k_1}, t^m_1, \ldots, t^m_{k_m}$ are variable-free terms and all $p_1, \ldots, p_l$ are integers. A QP-substitution is called empty iff $n = 0$, $m = 0$ and $l = 0$. Greek letters are used to denote QP-substitutions. The letter ε denotes the empty substitution.

For a QP-substitution θ, we denote by $\theta^{=}$ the following system of linear Diophantine equations:

$$\{vn = p \mid vn \leftarrow p \in \theta\}.$$

We define a notion of instance for index variables and polynomials from LP:

Definition 20. *Let* θ *be a QP-substitution. Then:*

– *An instance of a index variable* vn *with respect to* θ, *denoted as* $vn\theta$, *is defined as*

$$vn\theta = \begin{cases} p & \text{if } vn \leftarrow p \in \theta, \\ vn & \text{otherwise} \end{cases}$$

– *An instance of a polynomial* $p = cn_1 vn_1 + \cdots + cn_k vn_k \in LP$ *with respect to* θ, *denoted as* $p\theta$, *is a polynomial from* LP *obtained from* $cn_1 vn_1\theta + \cdots + cn_k vn_k\theta$ *by arithmetic simplification.*

On the basis of this definition, we extend the notion of instance on QP-terms and QP-equations with respect to QP-substitutions. The notions of domain, codomain, range and composition are extended to QP-substitutions in a straightforward way.

Example 5. Let $\theta = \{x \leftarrow f(y),\ \overline{x} \leftarrow \overline{y}, \overline{x},\ \overline{y} \leftarrow \overline{y}, \overline{z}, vn \leftarrow 3vn + vm,\ vk \leftarrow vm - 2\}$ and $\lambda = \{y \leftarrow g(c),\ \overline{x} \leftarrow c,\ \overline{z} \leftarrow,\ vn \leftarrow 2vm + 1,\ vm \leftarrow vk + 2\}$. Then $\theta \circ \lambda = \{x \leftarrow f(g(c)),\ y \leftarrow g(c),\ \overline{x} \leftarrow \overline{y}, c,\ \overline{z} \leftarrow, vn \leftarrow 6vm + vk + 5, vm \leftarrow vk + 2\}$.

Definition 21 (Quasi Pattern E-unification Problem). *A general quasi pattern E-unification, or, shortly, QP-E-unification problem with sequence variables, flexible arity symbols and QP-terms is a finite system of QP-equations* $\langle s_1 \stackrel{?}{=}_E t_1, \ldots, s_n \stackrel{?}{=}_E t_n \rangle$.

Definition 22 (Explicit Pattern Expansion). *Let* Q *be one of the following: QP-term, QP-equation, QP-substitution or QP-E-unification problem. The explicit pattern expansion in* Q, *denoted as* $EPE(Q)$, *is respectively a QP-term, QP-equation, QP-substitution or QP-E-unification problem obtained from* Q *by replacing each occurrence of an explicit pattern in* Q *with the sequence of QP-terms as long as possible in the following way:*

- *each occurrence of an explicit pattern $c_{cm,cm}$ is replaced by the single QP-term c_{cm};*
- *each occurrence of an explicit pattern $c_{cm,ck}$, $cm < ck$, is replaced by the sequence $c_{cm}, c_{cm+1}, \ldots, c_{ck}$;*
- *each occurrence of an explicit pattern $f_{cm,cm}(t_1, \ldots, t_n)$ is replaced by the single QP-term $f_{cm}(t_1, \ldots, t_n)$.*
- *each occurrence of an explicit pattern $f_{cm,ck}(t_1, \ldots, t_n)$, $cm < ck$, is replaced by the sequence $f_{cm}(t_1, \ldots, t_n), f_{cm+1}(t_1, \ldots, t_n), \ldots, f_{ck}(t_1, \ldots, t_n)$.*

Example 6. Let qpt be a QP-term $f(a, g_{1,3}(c_{1,2}, \overline{y}), h_{1,vn}(x))$. Then

$$EPE(qpt) = f(x, g_1(c_1, c_2, \overline{y}), g_2(c_1, c_2, \overline{y}), g_3(c_1, c_2, \overline{y}), h_{1,vn}(x)).$$

Below by QP-expression we mean either QP-term, QP-equation, QP-substitution or QPE-unification problem.

Definition 23 (Pattern-Term, Pattern-Equation, Pattern-Substitution, Pattern-E-Unification Problem). *Let Q be a QP-expression. Q is called*

- *a pattern-term or, shortly, P-term;*
- *a pattern-equation or, shortly, P-equation;*
- *a pattern-substitution or, shortly, P-substitution or*
- *a pattern-E-unification problem or, shortly, P-E-unification problem*

over (V, C, P, LP) iff there exist a substitution σ such that $Dom(\sigma) \subset XV$ and $EPE(Q\sigma)$ is respectively

- *a term;*
- *an equation;*
- *a substitution or*
- *an E-unification problem*

over (V, C, P).

In order to decide whether a QP-expression is a corresponding P-expression, we associate a system of linear Diophantine constraints to a QP-expression and show that if the system has a positive integer solution, then the QP-expression is the corresponding P-expression.

Definition 24 (Linear Diophantine Constraints Associated with a QP-Expression). *For a QP-expression Q, the associated system of linear Diophantine constraints $LDC(Q)$ is defined in the following way:*

- *Q is a QP-term. Then*
  - *If $Q \in IV \cup SV \cup CONST$ then $LDC(Q)$ is empty.*
  - *If Q is $c_{vcm,vck}$ or $f_{vcm,vck}(t_1, \ldots, t_n)$, then $LDC(Q)$ is*

$$1 \leq vcm \;\&\; vcm \leq vck.$$

- *If Q is $f(t_1,\dots,t_n)$, where $f \in FFIX \cup FFLEX$, then $LDC(Q)$ is*

$$LDC(t_1)\&\dots\&LDC(t_n),$$

– *Q is either QP-equation, QP-substitution or a QP-E-unification problem. Then $LDC(Q)$ is*

$$LDC(qpt_1)\&\dots\&LDC(qpt_m),$$

where $qpt_1,\dots,qpt_m$ are all QP-terms occurring in Q.

Theorem 16. *A QP-expression Q is the corresponding P-expression if $LDC(Q)$ has a positive integer solution.*

Proof. We prove the theorem for a QP-term. For the other QP-expressions the proof is analogous. Let $vm_1,\dots,vm_s$ be all index variables occurring in Q and $vm_1 = cn_1,\dots,vm_s = cn_s$ be a positive integer solution of $LDC(Q)$. Then we assemble substitution σ as follows: $\sigma = \{vm_1 \leftarrow cn_1,\dots,vm_s \leftarrow cn_s\}$. By the construction of the system $LDC(Q)$, all the patterns occurring in $Q\sigma$ are explicit with the following property: for each pattern $c_{cm,ck}$ or $f_{cm,ck}(t_1,\dots,t_r)$ we have $1 \le cm$ and $cm \le ck$. Therefore, $EPE(Q\sigma)$ contains no patterns and thus, is a term over (V, C, P). Hence, by Definition 23, Q is a P-term. □

Definition 25 (Unifier of a P-E-unification problem). *Let*

$$\Gamma = \langle s_1 \stackrel{?}{\doteq}_E t_1,\dots,s_n \stackrel{?}{\doteq}_E t_n\rangle$$

be a P-E-unification problem. A P-substitution θ is called a unifier of Γ iff

- *$Dom(\theta)$ contains all the index variables which occur in Γ;*
- *for each $vn \in Dom(\theta)$, $vn\theta$ is a positive integer;*
- *each pattern which occurs in P-terms in $Cod(\theta)$ is explicit;*
- *for each $1 \le i \le n$, the P-equality $s_i\theta \doteq_E t_i\theta$ holds.*

Example 7. Let $\Gamma = f(\overline{x},\overline{y}) \stackrel{?}{\doteq}_\emptyset f(h_{vm,vk}(z))$. Then

$$\theta = \{\overline{x} \leftarrow h_{1,2}(z), \overline{y} \leftarrow h_{3,6}(z), vm \leftarrow 1, vk \leftarrow 6\}$$

is one of $\emptyset$-unifiers of Γ.

We have the following proposition:

Proposition 1. *If a P-substitution θ is a unifier of a P-E-unification problem Γ, then $\theta^{=}$ gives a (unique) positive integer solution to the system $LDC(\Gamma)\&LDC(\theta)$.*

Definition 26 (More General P-Substitution). *A P-substitution θ is more general than a P-substitution σ on a finite set of variables Var modulo a theory E (denoted $\theta \ll_E^{Var} \sigma$) iff there exists a P-substitution λ such that*

– *for all $\overline{x} \in Var$,*
  - *$\overline{x} \leftarrow \ \notin \lambda$;*

  - *there exist P-terms* $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ *such that* $\overline{x}\sigma = t_1, \ldots, t_n$, $\overline{x}\theta \circ \lambda = s_1, \ldots, s_n$ *and for each* $1 \leq i \leq n$, *either* t_i *and* s_i *are the same sequence variables or the P-equality* $t_i \doteq_E s_i$ *holds;*
- *for all* $x \in Var$, *the P-equality* $x\sigma \doteq_E x\theta \circ \lambda$ *holds;*
- *for all* $vn \in Var$, $vn\sigma = vn\theta \circ \lambda$.

Notions of minimal set of substitutions and minimal complete set of unifiers are extended for P-substitutions and P-E-unification problems in a straightforward way.

We represent a minimal complete set of unifiers of Γ - $MCU_E(\Gamma)$ as a set of P-substitution/constraint pairs. The constraints are linear Diophantine equations and/or inequalities. The representation must satisfy the following properties:

- for each pair $\{\theta, ldc\}$ in the representation and for each positive integer solution μ of ldc, the P-substitution $(\theta \circ \mu)|_{VarSet(\Gamma)}$ must be in $MCU_E(\Gamma)$;
- for each substitution $\sigma \in MCU_E(\Gamma)$ there must be a pair $\{\theta, ldc\}$ in the representation such that $\sigma = (\theta \circ \mu)|_{VarSet(\Gamma)}$ for a positive integer solution μ of ldc.

The following two example give a demonstration of a representation of a minimal complete set of unifiers as a set of P-substitution/constraint pairs:

Example 8. Let $\Gamma = f(\overline{x}, \overline{y}) \stackrel{?}{=}_\emptyset f(h_{vm,vk}(z))$. Then we can represent $MCU_\emptyset(\Gamma)$ as a finite set of P-substitution/constraint pairs:

$$\begin{aligned} S = \{\ & \{\{\overline{x} \leftarrow\ , \overline{y} \leftarrow h_{vm,vk}(z)\}, 1 \leq vm \,\&\, vm \leq vk\}, \\ & \{\{\overline{x} \leftarrow\ , h_{vm,vk}(z), \overline{y} \leftarrow\}, 1 \leq vm \,\&\, vm \leq vk\}, \\ & \{\{\overline{x} \leftarrow h_{vm,vn}(z), \overline{y} \leftarrow h_{vn+1,vk}(z)\}, \\ & \qquad 1 \leq vm \,\&\, vm \leq vn \,\&\, vn + 1 \leq vk\} \ \}. \end{aligned}$$

In fact,

$$\begin{aligned} MCU_E(\Gamma) = \{\sigma \mid\ & \text{there exists } \{\theta, ldc\} \in \mathrm{S} \text{ and } \mu \text{ such that} \\ & \mu \text{ is a positive integer solution of } ldc \\ & \text{and } \sigma = (\theta \circ \mu)|_{\mathrm{VarSet}(\Gamma)}\}. \end{aligned}$$

For instance, a solution $\{vm \leftarrow 1,\ vn \leftarrow 3,\ vk \leftarrow 4\}$ of the constraint $1 \leq vm \,\&\, vm \leq vn \,\&\, vn + 1 \leq vk$, applied on the substitution $\{\overline{x} \leftarrow h_{vm,vn}(z), \overline{y} \leftarrow h_{vn+1,vk}(z)\}$ gives $\{\overline{x} \leftarrow h_{1,3}(z), \overline{y} \leftarrow h_{4,4}(z), vm \leftarrow 1, vn \leftarrow 3, vk \leftarrow 4\}$. The restriction of the latter to $VarSet(\Gamma)$ is $\{\overline{x} \leftarrow h_{1,3}(z), \overline{y} \leftarrow h_{4,4}(z), vm \leftarrow 1, vk \leftarrow 4\}$, which belongs to $MCU_E(\Gamma)$. In the expanded form the substitution looks like $\{\overline{x} \leftarrow h_1(z), h_2(z), h_3(z), \overline{y} \leftarrow h_4(z), vm \leftarrow 1, vk \leftarrow 4\}$.

Example 9. Let $\Gamma = f(\overline{x}, h_{vm,vk}(z)) \stackrel{?}{=}_\emptyset f(h_{vm,vk}(z), \overline{x})$. Then the following set gives an infinite representation of $MCU_\emptyset(\Gamma)$ as a set of substitution/constraint pairs:

$$\begin{aligned} S = \{\ & \{\{\overline{x} \leftarrow\}, 1 \leq vm \,\&\, vm \leq vk\}, \\ & \{\{\overline{x} \leftarrow h_{vm,vk}(z)\}, 1 \leq vm \,\&\, vm \leq vk\}, \\ & \{\{\overline{x} \leftarrow h_{vm,vk}(z), h_{vm,vk}(z)\}, 1 \leq vm \,\&\, vm \leq vk\} \\ & \ldots \qquad\qquad\qquad\qquad\qquad\qquad \}. \end{aligned}$$

Again,

$$MCU_E(\Gamma) = \{\sigma \mid \begin{array}{l}\text{there exists } \{\theta, ldc\} \in \mathrm{S} \text{ and } \mu \text{ such that} \\ \mu \text{ is a positive integer solution of } ldc \\ \text{and } \sigma = (\theta \circ \mu)|_{\mathrm{VarSet}(\Gamma)}\}.\end{array}$$

Pattern-terms naturally appear in the proving context, when one wants to skolemize, for instance, the expression $\forall x \exists \overline{y}\, (g(x) \doteq g(\overline{y}))$. Here $\overline{y}$ should be replaced with a sequence of terms $f_1(x), \ldots, f_{n(x)}(x)$, where $f_1, \ldots, f_{n(x)}$ are Skolem functions. The problem is that we can not know in advance the length of such a sequence.

Note that we consider vn instead of $n(x)$. This is because, given an unification problem UP in which $n(x)$ occurs, we can do a variable abstraction on $n(x)$ with a fresh index variable vn and instead of UP consider UP' with the constraint $vn = n(x)$, where UP' is obtained from UP by replacing each occurrence of $n(x)$ with vn. One of the tasks for unification with patterns is to find a proper value for vn, if possible.

In the next subsection we design a unification procedure for a P-$\emptyset$-unification problem. Note that analogously to the case of $\emptyset$-unification, it is enough to consider single P-equations instead of systems of P-equations in P-$\emptyset$-unification problems.

6.1 Pattern $\emptyset$-Unification Procedure

The problem has a form of P-equation $t_1 \stackrel{?}{\doteq}_{\emptyset} t_2$. We denote it as $GPUP_{\emptyset}$ and refer to it as a general P-$\emptyset$-unification problem.

We design the unification procedure as a tree generation process based on two basic steps: projection and transformation. We describe them in terms of "quasi-patterns" instead of "patterns".

For a QP-$\emptyset$-unification problem $QPUP_{\emptyset}$, we denote by $SUC(QPUP_{\emptyset})$ the successors of $QPUP_{\emptyset}$ under projection or transformation. $SUC(QPUP_{\emptyset})$ can be $SUCCESS$, $FAILURE$ or a tuple of QP-$\emptyset$-unification problems. $SUB(QPUP_{\emptyset})$ denotes the tuple of QP-substitutions which are applied on $QPUP_{\emptyset}$ to get $SUC(QPUP_{\emptyset})$. $CON(QPUP_{\emptyset})$ denotes the tuple of linear Diophantine constraints which have to be satisfied to make a step from $QPUP_{\emptyset}$ to $SUC(QPUP_{\emptyset})$.

Projection. Projection for QP-$\emptyset$-unification problems is defined similarly to projection for $\emptyset$-unification problems: Let $S_1, \ldots, S_n$ be all subsets of the set of all sequence variables of a general QP-$\emptyset$-unification problem $GQPUP_{\emptyset}$ and Θ be the set of substitutions $\{\theta_1, \ldots, \theta_n\}$ such that for all $1 \leq i \leq n$, $\theta_i = \{\overline{x} \leftarrow \ \mid \overline{x} \in S_i\}$. Then under projection

- $SUB(GQPUP_{\emptyset}) = \langle \theta_1, \ldots, \theta_k \rangle$,
- $CON(GQPUP_{\emptyset}) = \langle \underbrace{LDC(GQPUP_{\emptyset}), \ldots, LDC(GQPUP_{\emptyset})}_{k\ times} \rangle$
- $SUC(GQPUP_{\emptyset}) = \langle GQPUP_{\emptyset}\theta_1, \ldots, GQPUP_{\emptyset}\theta_k \rangle$.

Transformation. To find $SUB(GQPUP_\emptyset)$, $CON(GQPUP_\emptyset)$ and $SUC(GQPUP_\emptyset)$ under transformation we distinguish the following three cases:

1. t_1 and t_2 are identical. Then
   - $SUB(GQPUP_\emptyset) = \langle \varepsilon \rangle$,
   - $CON(GQPUP_\emptyset) = \langle TRUE \rangle$
   - $SUC(GQPUP_\emptyset) = SUCCESS$.
2. t_1 and t_2 are neither identical QP-terms nor non-variable QP-terms with the same head. Then $SUB(GQPUP_\emptyset)$ and $SUC(GQPUP_\emptyset)$ are defined in Table 1. As for the constraints, $CON(GQPUP_\emptyset) = \langle TRUE \rangle$ in the cases when $SUC(GQPUP_\emptyset) = SUCCESS$ and $CON(GQPUP_\emptyset) = \langle FALSE \rangle$ in the cases when $SUC(GQPUP_\emptyset) = FAILURE$.
3. t_1 and t_2 are non-identical non-variable QP-terms with the same head g, where g is a function symbol with either fixed or flexible arity. Then we have the following two cases:
   (a) Only one from t_1 and t_2 has the form $g()$. Then
      - $SUB(GQPUP_\emptyset) = \langle \rangle$,
      - $CON(GQPUP_\emptyset) = \langle FALSE \rangle$
      - $SUC(GQPUP_\emptyset) = FAILURE$.

   (b) None of t_1 and t_2 is a QP-term of the form $g()$. Let t_1 be $g(r_1, \overline{r})$ and t_2 be $g(s_1, \overline{s})$, where $\overline{r}$ and $\overline{s}$ are (possibly empty) sequences of QP-terms. Then we have the following cases:
      i. r_1 and s_1 are identical. Then
         - $SUB(GQPUP_\emptyset) = \langle \varepsilon \rangle$,
         - $CON(GQPUP_\emptyset) = \langle TRUE \rangle$
         - $SUC(GQPUP_\emptyset) = \langle f(\overline{r}) \stackrel{?}{=}_\emptyset f(\overline{s}) \rangle$,

         where f is a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

      ii. r_1 and s_1 are not identical. Then
         - none of r_1 and s_1 is a pattern. Then we define $SUB(GQPUP_\emptyset)$ and $SUC(GQPUP_\emptyset)$ in Table 2. As for the constraints, we have $CON(GQPUP_\emptyset) = \langle FALSE \rangle$ when $SUC(GQPUP_\emptyset) = FAILURE$ and $CON(GQPUP_\emptyset) = \langle \underbrace{TRUE, \ldots, TRUE}_{k\ times} \rangle$ otherwise, where k is a length of $SUB(GQPUP_\emptyset)$.
         - Only one of r_1 and s_1 is a pattern. We can assume without loss of generality that r_1 is a pattern. Then we define $SUB(GQPUP_\emptyset)$, $CON(GQPUP_\emptyset)$ and $SUC(GQPUP_\emptyset)$ in Table 3.
         - Both r_1 and s_1 are patterns. If the heads of r_1 and s_1 are different, then $SUB(GQPUP_\emptyset) = \langle \rangle$, $CON(GQPUP_\emptyset) = \langle FALSE \rangle$ and $SUC(GQPUP_\emptyset) = FAILURE$. Otherwise we define the tuples $SUB(GQPUP_\emptyset)$, $CON(GQPUP_\emptyset)$ and $SUC(GQPUP_\emptyset)$ in Table 4.

Table 3. Transformation table for $GQPUP_\emptyset$ of the form $g(r_1, \overline{r}) \stackrel{?}{=}_\emptyset g(s_1, \overline{s})$ where r_1 is a pattern of the form $h_{m,k}(\overline{t})$, $m, k \in LP$, the sequences $\overline{r}$, $\overline{s}$ and $\overline{t}$ are possibly empty sequences of QP-terms. The function symbol f in the table is a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g. The index variable vn is a new variable, $l \in LP$, $\overline{q}$ is a possible empty sequence of QP-terms.

s_1	$SUB(GQPUP_\emptyset)$	$CON(GQPUP_\emptyset)$	$SUC(GQPUP_\emptyset)$
Ind. var.	$\langle\rangle$, if s_1 occurs in r_1 $\langle \sigma_1, \sigma_2 \rangle$, where $\sigma_1 = \{s_1 \leftarrow h_k(\overline{t})\}$, $\sigma_2 = \{s_1 \leftarrow h_m(\overline{t})\}$, otherwise	$\langle FALSE \rangle$, if s_1 occurs in r_1 $\langle C_1, C_2 \rangle$, where C_1 is $m = k$, C_2 is $m + 1 \leq k$, otherwise	$FAILURE$ if s_1 occurs in r_1 $\langle f(\overline{r})\sigma_1 \stackrel{?}{=}_\emptyset f(\overline{s})\sigma_1$, $f(h_{m+1,k}(\overline{t}), \overline{r})\sigma_2 \stackrel{?}{=}_\emptyset f(\overline{s})\sigma_2 \rangle$, otherwise
Seq. var.	$\langle\rangle$, if s_1 occurs in r_1 $\langle \sigma_1, \sigma_2, \sigma_3 \rangle$, where $\sigma_1 = \{s_1 \leftarrow h_{m,vn}(\overline{t})\}$, $\sigma_2 = \{s_1 \leftarrow h_{m,k}(\overline{t}), s_1\}$, $\sigma_3 = \{s_1 \leftarrow h_{m,k}(\overline{t})\}$, otherwise	$\langle FALSE \rangle$, if s_1 occurs in r_1 $\langle C_1, C_2, C_3 \rangle$, where C_1 is $m \leq vn \,\&\, vn + 1 \leq k$, C_2 is $TRUE$, C_3 is $TRUE$, otherwise	$FAILURE$ if s_1 occurs in r_1 $\langle f(h_{vn+1,k}(\overline{t}), \overline{r})\sigma_1 \stackrel{?}{=}_\emptyset$ $f(\overline{s})\sigma_1$, $f(\overline{r})\sigma_2 \stackrel{?}{=}_\emptyset f(s_1, \overline{s})\sigma_2$, $f(\overline{r})\sigma_3 \stackrel{?}{=}_\emptyset f(\overline{s})\sigma_3 \rangle$, otherwise
Non-var. term	$\langle \varepsilon \rangle$, if $s_1 = h_l(\overline{q})$ $\langle\rangle$, otherwise	$m = l \,\&\, m + 1 \leq k$, if $s_1 = h_l(\overline{q})$ $\langle FALSE \rangle$, otherwise	$\langle f(h_m(\overline{t}), h_{m+1,k}(\overline{t}), \overline{r}) \stackrel{?}{=}_\emptyset$ $f(h_m(\overline{q}), \overline{s}) \rangle$, if $s_1 = h_l(\overline{q})$ $FAILURE$, otherwise

Table 4. Transformation table for $GQPUP_\emptyset$ of the form $g(r_1, \overline{r}) \stackrel{?}{=}_\emptyset g(s_1, \overline{s})$ where $r_1 = h_{m_1,k_1}(\overline{t_1})$, $s_1 = h_{m_2,k_2}(\overline{t_1})$, $m_1, k_1, m_2, k_2 \in LP$ and $\overline{t_1}$, $\overline{r}$ and $\overline{s}$ are possibly empty sequences of QP-terms.

$SUB(GQPUP_\emptyset)$	$CON(GQPUP_\emptyset)$	$SUC(GQPUP_\emptyset)$
$\langle \varepsilon, \varepsilon, \varepsilon \rangle$	$\langle C_1, C_2, C_3 \rangle$, where C_1 is $m_1 = m_2 \,\&\, k_1 + 1 \leq k_2$, C_2 is $m_1 = m_2 \,\&\, k_2 + 1 \leq k_1$, C_3 is $m_1 = m_2 \,\&\, k_1 = k_2$	 $\langle g(h_{k_1}(\overline{t_1}), \overline{r}) \stackrel{?}{=}_\emptyset$ $g(h_{k_1}(\overline{t_2}), h_{k_1+1,k_2}(\overline{t_2}), \overline{s})$, $g(h_{k_2}(\overline{t_1}), h_{k_2+1,k_1}(\overline{t_1}), \overline{r}) \stackrel{?}{=}_\emptyset$ $g(h_{k_2}(\overline{t_2}), \overline{s})$, $g(h_{k_2}(\overline{t_1}), \overline{r}) \stackrel{?}{=}_\emptyset g(h_{k_2}(\overline{t_2}), \overline{s}) \rangle$

Unification Procedure - Tree Generation. Projection and transformation can be seen as single steps in a tree generation process. Each node of the tree is labeled either with a QP-$\emptyset$-unification problem, $SUCCESS$ or $FAILURE$. The edges of the tree are labeled by substitutions and linear Diophantine constraints. The nodes labeled with $SUCCESS$ or $FAILURE$ are terminal nodes. The nodes labeled with QP-$\emptyset$-unification problems are non-terminal nodes. The children of a non-terminal node are constructed in the following way:

Let $QPUP$ be a QP-$\emptyset$-unification problem attached to a non-terminal node and LDC be a conjunction of linear Diophantine constraints attached to the edges in the branch, from the root of the tree till the current node. First, we check whether LDC is satisfiable. If it is not, we replace $QPUP$ with the new label $FAILURE$. Otherwise we proceed as follows: If we can decide whether $QPUP$ is not unifiable, then we replace $QPUP$ with the new label $FAILURE$. Otherwise we apply projection or transformation on $QPUP$ and get $SUB(QPUP)$, $CON(QPUP)$ and $SUC(QPUP)$. If $SUC(QPUP)$ is $SUCCESS$, then the node has a single child with the label $SUCCESS$ and the edge to that node is labeled with $SUB(QPUP)$ and $CON(QPUP)$. If $SUC(QPUP) = \langle P_1, \ldots, P_n \rangle$, then $SUB(QPUP) = \langle \sigma_1, \ldots, \sigma_n \rangle$ and $CON(QPUP) = \langle C_1, \ldots, C_n \rangle$, the node $QPUP$ has n children, labeled respectively with $P_1, \ldots, P_n$ and the edge to the P_i node is labeled with σ_i and C_i $(1 \leq i \leq n)$.

Satisfiability of LDC can be checked by an algorithm for solving linear Diophantine equational and inequational systems [2].

We design the general P-$\emptyset$-unification procedure as a breadth first (level by level) tree generation process. Let $GPUP_\emptyset$ be a P-$\emptyset$-unification problem. We label the root of the tree with $GPUP_\emptyset$ (zero level). First level nodes (the children of the root) of the tree are obtained from the original problem by projection[1]. Starting from the second level, we apply only a transformation step to a QP-$\emptyset$-unification problem of each node, thus getting new successor nodes. The branch which ends with a node labeled by $SUCCESS$ is called a successful branch. The branch which ends with a node labeled by $FAILURE$ is a failed branch. All QP-$\emptyset$-unification problems attached to the nodes of a successful branch are in fact P-$\emptyset$-unification problems.

For each node in the tree, we compose substitutions (top-down) displayed on the edges of the branch which leads to this node and attach the obtained substitution to the node. The empty substitution is attached to the root. For a node N, the substitution attached to N in such a way is called the associated substitution of N.

Similarly, for each node in the tree, we take a conjunction of the linear Diophantine constraints displayed on the edges of the branch which leads to this node and attach the obtained constraint to the node. The linear Diophantine constraint $LDC(GPUP_\emptyset)$ is attached to the root. For a node N, the constraint attached to N in such a way is called the associated constraint of N.

[1] Starting from the first level, the unification problems attached to the nodes in the tree might not be P-$\emptyset$-unification problems, but they are, of course, QP-$\emptyset$-unification problems.

We call the tree a P-$\emptyset$-unification tree for the problem $GPUP_\emptyset$ and denote it $PUT(GPUP_\emptyset)$.

Let $\Delta(GPUP_\emptyset)$ be the set of all P-substitution/constraint pairs associated with the $SUCCESS$ nodes. Then we define the set $\Sigma(GPUP_\emptyset)$ as follows:

$$\Sigma(GPUP_\emptyset) = \{\sigma \mid \begin{array}{l} \text{there exists } \{\theta, ldc\} \in \Delta(\mathrm{GPUP}_\emptyset) \text{ and } \mu \text{ such that} \\ \mu \text{ is a positive integer solution of } ldc \\ \text{and } \sigma = (\theta \circ \mu)|_{\mathrm{VarSet}(\Gamma)} \}. \end{array}$$

Next, our goal is to show that $\Sigma(GPUP_\emptyset)$ is the minimal complete set of unifiers of $GPUP_\emptyset$.

First, we show correctness.

Lemma 6. *$\Sigma(GPUP_\emptyset)$ is a set of unifiers of $GPUP_\emptyset$.*

Proof. Let $\sigma \in \Sigma(GPUP_\emptyset)$. Then there exists $\{\theta, ldc\} \in \Delta(GPUP_\emptyset)$ and μ such that μ is a positive integer solution of ldc and $\sigma = (\theta \circ \mu)|_{VarSet(\Gamma)}$. We show that σ is a unifier of $GPUP_\emptyset$. Recall that $GPUP_\emptyset$ has a form of P-equation $t_1 \stackrel{?}{\doteq}_\emptyset t_2$.

Since μ is a positive integer solution of ldc and the variables in ldc are all the index variables from $GPUP_\emptyset$ and $Cod(\theta)$, we have:

- $Dom(\sigma)$ contains all the index variables which occur in $GPUP_\emptyset$;
- for each $vm \in Dom(\sigma)$, vm is a positive integer;
- each pattern which occurs in $Cod(\sigma)$ is explicit.

Moreover, since $t_1\theta \doteq_\emptyset t_2\theta$, we have $t_1\theta \circ \mu \doteq_\emptyset t_2\theta \circ \mu$ and, thus, $t_1\sigma \doteq_\emptyset t_2\sigma$. Now, by Definition 25 we get that σ is a unifier of $GPUP_\emptyset$. □

Second, we prove completeness.

Lemma 7. *For every unifier ϕ of a general P-$\emptyset$-unification problem $GPUP_\emptyset$ there exists a branch β in $PUT(GPUP_\emptyset)$ with the following property: if θ and ldc are respectively a substitution and a constraint attached to the same node in β, then there exists a positive integer solution τ of ldc such that $\theta \circ \tau \ll_\emptyset^{Var} \phi$, where Var is the set of variables of $GPUP_\emptyset$.*

Proof. Let ϕ be an arbitrary unifier of $GPUP_\emptyset$ and Var be the set of variables of $GPUP_\emptyset$. We should find a branch β in $PUT(GPUP_\emptyset)$ such that if θ and ldc are respectively a substitution and a constraint attached to the same node in β, then for some positive solution τ of ldc, $\theta \circ \tau \ll_\emptyset^{Var} \phi$.

We define β recursively. We search for β among successful branches of the tree $PUT(GPUP_\emptyset)$, therefore all the nodes in β will have P-$\emptyset$-unification problems attached.

First, let the root of the tree, labeled with $GPUP_\emptyset$, be in β. The P-substitution associated with the node $GPUP_\emptyset$ is ε. The constraint associated with $GPUP_\emptyset$ is $LDC(GPUP_\emptyset)$. By Proposition 1, $\phi^=$ contains a positive integer solution of $LDC(GPUP_\emptyset)$. Let τ be the positive integer solution of $LDC(GPUP_\emptyset)$

contained in $\phi^{=}$. Let $\lambda = \phi\backslash\tau$. Then $\phi = \tau \circ \lambda = \varepsilon \circ \tau \circ \lambda$, from which we can easily derive that $\varepsilon \circ \tau \ll_{\emptyset}^{Var} \phi$.

Next, let the first level node of β be that successor UP of $GPUP_{\emptyset}$, for which the associated P-substitution $\theta = \{\overline{x} \leftarrow \ \mid \overline{x} \leftarrow \ \in \phi$ and $\overline{x} \in Var\}$. The constraint associated with UP is $ldc = LDC(GPUP_{\emptyset})$. Let τ be the positive integer solution of $LDC(GPUP_{\emptyset})$ contained in $\phi^{=}$. Let $\lambda = \phi\backslash\theta$ and $\psi = \lambda\backslash\tau$. Then $\lambda = \tau \circ \psi$ and $\phi = \theta \circ \lambda = \theta \circ \tau \circ \psi$. Therefore

- for all $\overline{x} \in Var$,
  - $\overline{x} \leftarrow \ \notin \psi$;
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\,\theta \circ \tau \circ \psi = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or a P-equality $t_i \doteq_{\emptyset} s_i$ holds;
- for all $x \in Var$, the P-equality $x\phi \doteq_{\emptyset} x\theta \circ \tau \circ \psi$ holds,
- for all $vn \in Var$, $vn\phi = vn\theta \circ \tau \circ \psi$,

which implies that $\theta \circ \tau \ll_{\emptyset}^{Var} \phi$.

Let now a node in $PUT(GPUP_{\emptyset})$, labeled with a P-$\emptyset$-unification problem UP_n, be in β. It means that for a P-substitution θ_n and a constraint ldc_n, associated with UP_n, there exists a positive integer solution τ_n of ldc_n such that $\theta_n \circ \tau_n \ll_{\emptyset}^{Var} \phi$. Then we have that there exists a P-substitution λ such that

- for all $\overline{x} \in Var$,
  - the binding
$$\overline{x} \leftarrow \ \notin \lambda; \tag{10}$$
  - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \tau_n \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality
$$t_i \doteq_{\emptyset} s_i \tag{11}$$
    holds;
- for all $x \in Var$, the P-equality
$$x\phi \doteq_{\emptyset} x\theta_n \circ \tau_n \circ \lambda \tag{12}$$
  holds;
- for all $vn \in Var$,
$$vn\phi = vn\theta_n \circ \tau_n \circ \lambda. \tag{13}$$

We can assume without loss of generality that

$$Dom(\tau_n) \cap Dom(\lambda) = \emptyset. \tag{14}$$

If there exists a successor UP_{n+1} of UP_n in $PUT(GPUP_{\emptyset})$ with the associated P-substitution θ_{n+1} and the associated constraint ldc_{n+1} such that for a positive integer solution τ_{n+1} of ldc_{n+1} we have $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{Var} \phi$, then we can include UP_{n+1} into β and, thus, we will have that for all $UP \in \beta$, if θ is a

P-substitution and ldc is a constraint associated with UP, then there exists a positive integer solution τ of ldc such that $\theta \circ \tau \ll_{\emptyset}^{Var} \phi$.

Thus, the problem of constructing β is reduced to the problem of finding the successor UP_{n+1} of UP_n with the property that for the P-substitution θ_{n+1} and the constraint lpc_{n+1} associated with UP_{n+1}, there exists a positive integer solution τ_{n+1} of ldc_{n+1} such that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{Var} \phi$. We show how to find such a UP_{n+1}.

The P-$\emptyset$-unification problem UP_n can have one of the following four forms:

1. UP_n is a pair of identical P-terms.
2. UP_n is a pair of individual variables.
3. UP_n is a pair of an individual variable and non-variable term.
4. UP_n is a pair of non-variable terms.

We consider each of them separately:

1. There is only one possible choice: P_{n+1} is $SUCCESS$ with the associated substitution $\theta_{n+1} = \theta_n \circ \varepsilon = \theta_n$ and associated linear Diophantine constraint $ldc_{n+1} = ldc_n \,\&\, TRUE = ldc_n$. Let τ_{n+1} be τ_n. Then $\theta_{n+1} \circ \tau_{n+1} = \theta_n \circ \tau_n \ll_{\emptyset}^{Var} \phi$.

2-3. Let P_n be $x \overset{?}{\doteq}_{\emptyset} t$, t being either an individual variable or a non-variable non-pattern term. The substitution $\theta_n \circ \lambda$ is a unifier of $GUP_{\emptyset}$. The unification problems $GUP_{\emptyset}\theta_n$ and P_n have exactly the same set of unifiers, because P_n can be obtained from $GUP_{\emptyset}\theta_n$ by iterated deletion of identical first arguments in both sides of $GUP_{\emptyset}\theta_n$. Therefore, we have that λ is a unifier of P_n. To obtain P_{n+1} from P_n we choose the substitution $\mu = \{x \leftarrow t\}$ and the constraint $TRUE$. Let θ_{n+1} be the substitution $\theta_n \circ \mu$, ldc_{n+1} be the constraint $ldc_n \,\&\, TRUE = ldc_n$ and τ_{n+1} be τ_n. Then from (10), (11), (12), (13) and the facts that $\mu \circ \lambda = \lambda$ and $\tau_n \circ \lambda = \lambda \circ \tau_n$ we get
  - for all $\overline{x} \in Var$,
    - $\overline{x} \leftarrow \notin \lambda$;
    - there exist terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_n \circ \tau_n \circ \lambda = \overline{x}\theta_n \circ \lambda \circ \tau_n = \overline{x}\theta_n \circ \mu \circ \lambda \circ \tau_n = \overline{x}\theta_{n+1} \circ \lambda \circ \tau_n = \overline{x}\theta_{n+1} \circ \tau_{n+1} \circ \lambda = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality $t_i \doteq_{\emptyset} s_i$ holds;
  - for all $x \in Var$, the P-equality $x\phi \doteq_{\emptyset} x\theta_n \circ \tau_n \circ \lambda \doteq_{\emptyset} x\theta_n \circ \lambda \circ \tau_n \doteq_{\emptyset} x\theta_n \circ \mu \circ \lambda \circ \tau_n \doteq_{\emptyset} x\theta_{n+1} \circ \lambda \circ \tau_n \doteq_{\emptyset} x\theta_{n+1} \circ \tau_{n+1} \circ \lambda$ holds;
  - for all $vn \in Var$, we have $vn\phi = vn\theta_n \circ \tau_n \circ \lambda = vn\theta_n \circ \lambda \circ \tau_n = vn\theta_n \circ \mu \circ \lambda \circ \tau_n = vn\theta_{n+1} \circ \lambda \circ \tau_n = vn\theta_{n+1} \circ \tau_{n+1} \circ \lambda$ holds;

  Thus, $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{Var} \phi$.

4. Let UP_n have the form $g(t_1, \overline{t}) \overset{?}{\doteq}_{\emptyset} g(s_1, \overline{s})$, where $\overline{t}$ and $\overline{s}$ are (possibly empty) sequences of terms. We can have the following 9 cases with respect to t_1 and s_1:
  - 4.1. t_1 and s_1 are identical.
  - 4.2. t_1 and s_1 are individual variables.
  - 4.3. One of t_1 and s_1 is an individual variable, the other is a non-variable non-pattern P-term.

4.4. t_1 and s_1 are sequence variables.

4.5. One of t_1 and s_1 is a sequence variable, the other is neither a sequence variable nor a pattern.

4.6. t_1 and s_1 are non-variable non-pattern P-terms.

4.7. One of t_1 and s_1 is an individual variable, the other is a pattern.

4.8. One of t_1 and s_1 is a sequence variable, the other is a pattern.

4.9. One of t_1 and s_1 is a non-variable non-pattern P-term, the other is a pattern.

4.10. t_1 and s_1 are patterns.

We consider each of these cases.

4.1-4.6. These cases can be proved in the same way as the corresponding cases from the proof of Lemma 2, taking into account the fact that $ldc_{n+1} = ldc_n \& TRUE = ldc_n$.

4.7. Suppose t_1 is an individual variable x and s_1 is a pattern $h_{m,k}(\overline{t})$. Let us define P-substitutions μ, η and ν and a constraint CON as follows:
- $\mu = \{x \leftarrow h_k(\overline{t})\}$, η is a positive integer solution of $m = k$, $\nu = \{v \leftarrow v\lambda \mid v \neq x,\, v \notin Dom(\eta), v \neq v\lambda\}$ and CON is $m = k$, if λ is a unifier of a P-$\emptyset$-unification problem $f(x) \doteq_\emptyset f(h_k(\overline{t}))$ and $\lambda^=$ contains a positive integer solution of $m = k$;
- $\mu = \{x \leftarrow h_m(\overline{t})\}$, η is a positive integer solution of $m + 1 \leq k$, $\nu = \{v \leftarrow v\lambda \mid v \neq x,\, v \notin Dom(\eta), v \neq v\lambda\}$ and CON is $m + 1 \leq k$, if λ is a unifier of a P-$\emptyset$-unification problem $f(x)\lambda \doteq_\emptyset f(h_m(\overline{t}))\lambda$ and $\lambda^=$ contains a positive integer solution of $m + 1 \leq k$;

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

We obtain UP_{n+1} from UP_n by the P-substitution μ (since λ is a unifier of UP_n, these two cases for μ are the only possibilities to get UP_{n+1} from UP_n). Therefore, $\theta_{n+1} = \theta_n \circ \mu$. Since $ldc_{n+1} = ldc_n \,\&\, CON$, $\tau_n \circ \eta$ is a positive solution of ldc_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then, using the facts $\mu \circ \eta \circ \nu = \lambda$ and $\mu \circ \tau_n = \tau_n \circ \mu$ we get

$$\theta_n \circ \tau_n \circ \lambda = \theta_n \circ \tau_n \circ \mu \circ \eta \circ \nu = \theta_n \circ \mu \circ \tau_n \circ \eta \circ \nu = \theta_{n+1} \circ \tau_{n+1} \circ \nu. \quad (15)$$

On the other hand, for all $\overline{x} \in Var$,

$$\overline{x} \leftarrow \ \notin \mu \circ \eta. \quad (16)$$

From (16) and (10), by Definition 9 we get for all $\overline{x} \in Var$

$$\overline{x} \leftarrow \ \notin \nu. \quad (17)$$

From (11) and (15) we get that for all $\overline{x} \in Var$ there exist terms $t_1, \ldots, t_n$, $s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\phi = t_1, \ldots, t_n$, $\overline{x}\theta_{n+1} \circ \tau_{n+1} \circ \nu = s_1, \ldots, s_n$ and for each $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality

$$t_i \doteq_\emptyset s_i; \quad (18)$$

holds.

From 12 and 15 we get that for all $x \in Var$ the P-equality

$$x\phi \doteq_{\emptyset} x\theta_n \circ \tau_n \circ \lambda \doteq_{\emptyset} x\theta_{n+1} \circ \tau_{n+1} \circ \nu. \tag{19}$$

holds.
From 13 and 15 we get that for all $vn \in Var$

$$vn\phi = vn\theta_n \circ \tau_n \circ \lambda = vn\theta_{n+1} \circ \tau_{n+1} \circ \nu. \tag{20}$$

From 17, 18, 19 and 20, by Definition 26 we get $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{Var} \phi$.

4.8. Suppose t_1 is a sequence variable $\overline{x}$ and s_1 is a pattern $h_{m,k}(\overline{t})$. Let us define P-substitutions μ, η and ν and a constraint CON as follows:
 - $\mu = \{\overline{x} \leftarrow h_{m,i}(\overline{t})\}$, η is a positive integer solution of $m \leq i \,\&\, i + 1 \leq k$, $\nu = \{v \leftarrow v\lambda \,|\, v \neq \overline{x},\, v \notin Dom(\eta),\, v \neq v\lambda\}$ and CON is $m \leq i \,\&\, i+1 \leq k$, if λ is a unifier of P-$\emptyset$-unification problem $f(\overline{x}) \doteq_{\emptyset} f(h_{m,i}(\overline{t}))$ and $\lambda^{=}$ contains a positive integer solution of $m \leq i \,\&\, i+1 \leq k$;
 - $\mu = \{\overline{x} \leftarrow h_{m,k}(\overline{t})\}$, $\eta = \varepsilon$, $\nu = \{v \leftarrow v\lambda \,|\, v \neq \overline{x},\, v \neq v\lambda\}$ and CON is $TRUE$, if λ is a unifier of P-$\emptyset$-unification problem $f(\overline{x}) \doteq_{\emptyset} f(h_{m,k}(\overline{t}))$;
 - $\mu = \{\overline{x} \leftarrow f(h_{m,k}(\overline{t}), \overline{x}\}$, $\eta = \varepsilon$, $\nu = \{\overline{x} \leftarrow \overline{r}\} \circ \{v \leftarrow v\lambda \,|\, v \neq \overline{x},\, v \neq v\lambda\}$ and CON is $TRUE$, if there exists a non-empty sequence of P-terms $\overline{r}$ such that λ is a unifier of P-$\emptyset$-unification problem $f(\overline{x}) \doteq_{\emptyset} f(h_{m,k}(\overline{t}), \overline{r})$,

 with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.
 We obtain P_{n+1} from P_n by the P-substitution μ (since λ is a unifier of P_n, these three cases for μ are the only possibilities to get P_{n+1} from P_n). Therefore, $\theta_{n+1} = \theta_n \circ \mu$. Since $ldc_{n+1} = ldc_n \,\&\, CON$, $\tau_n \circ \eta$ is a positive solution of ldc_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then, with the same reasoning as in the case 4.7., we conclude that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{Var} \phi$.

4.9. Suppose t_1 is a P-term $h_l(\overline{t_1})$ and s_1 is a pattern $h_{m,k}(\overline{t_2})$. Let us define P-substitutions μ, η and ν and a constraint CON as follows:
 - $\mu = \varepsilon$, η is a positive integer solution of $m = l \,\&\, m + 1 \leq k$, $\nu = \{v \leftarrow v\lambda \,|\, v \notin Dom(\eta),\, v \neq v\lambda\}$ and CON is $m = l \,\&\, m+1 \leq k$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_m(\overline{t_1})) \doteq_{\emptyset} f(h_m(\overline{t_2}))$ and $\lambda^{=}$ contains a positive integer solution of $m = l \,\&\, m+1 \leq k$,

 with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.
 We obtain P_{n+1} from P_n by the P-substitution μ (since λ is a unifier of P_n, this is the only possibility to get P_{n+1} from P_n). Therefore, $\theta_{n+1} = \theta_n \circ \mu = \theta_n$. Since $ldc_{n+1} = ldc_n \,\&\, CON$, $\tau_n \circ \eta$ is a positive solution of ldc_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then, with the same reasoning as in the case 4.7., we conclude that $\theta_{n+1} \circ \tau_{n+1} \ll_{\emptyset}^{Var} \phi$.

4.10. Suppose t_1 is a pattern $h_{m_1,k_1}(\overline{t_1})$ and s_1 is a pattern $h_{m_2,k_2}(\overline{t_2})$. Let us define P-substitutions μ, η and ν and a constraint CON as follows:

- $\mu = \varepsilon$, η is a positive integer solution of $m_1 = m_2 \,\&\, k_1 + 1 \leq k_2$, $\nu = \lambda \backslash \eta$, CON is $m_1 = m_2 \,\&\, k_1 + 1 \leq k_2$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{k_1}(\overline{t_1})) \doteq_\emptyset f(h_{k_1}(\overline{t_2}))$ and $\lambda^=$ contains a positive integer solution of $m_1 = m_2 \,\&\, k_1 + 1 \leq k_2$;
- $\mu = \varepsilon$, η is a positive integer solution of $m_1 = m_2 \,\&\, k_2 + 1 \leq k_1$, $\nu = \lambda \backslash \eta$, CON is $m_1 = m_2 \,\&\, k_2 + 1 \leq k_1$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{k_2}(\overline{t_1})) \doteq_\emptyset f(h_{k_2}(\overline{t_2}))$ and $\lambda^=$ contains a positive integer solution of $m_1 = m_2 \,\&\, k_2 + 1 \leq k_1$;
- $\mu = \varepsilon$, η is a positive integer solution of $m_1 = m_2 \,\&\, k_1 = k_2$, $\nu = \lambda \backslash \eta$, CON is $m_1 = m_2 \,\&\, k_1 = k_2$, if λ is a unifier of P-$\emptyset$-unification problem $f(h_{k_1}(\overline{t_1})) \doteq_\emptyset f(h_{k_1}(\overline{t_2}))$ and $\lambda^=$ contains a positive integer solution of $m_1 = m_2 \,\&\, k_1 = k_2$;

with f being a new flexible arity function symbol, if g has a fixed arity. Otherwise f is g.

We obtain UP_{n+1} from UP_n by the P-substitution μ and constraint CON (since λ is a unifier of UP_n, these three cases are the only possibilities to get UP_{n+1} from UP_n). Since $ldc_{n+1} = ldc_n \,\&\, CON$, $\tau_n \circ \eta$ is a positive solution of ldc_{n+1}. Let τ_{n+1} be $\tau_n \circ \eta$. Then, with the same reasoning as in the case 4.7., we conclude that $\theta_{n+1} \circ \tau_{n+1} \ll_\emptyset^{Var} \phi$.

Thus, for all possible forms of UP_n we found its successor UP_{n+1} such that for the P-substitution θ_{n+1} and the constraint ldc_{n+1} associated with UP_{n+1}, there exists a positive integer solution τ_{n+1} of ldc_{n+1} such that $\theta_{n+1} \circ \tau_{n+1} \ll_\emptyset^{Var} \phi$. It finishes the proof. □

Theorem 17. *$\Sigma(GPUP_\emptyset)$ is a complete set of unifiers for $GPUP_\emptyset$.*

Proof. The theorem follows from Lemma 7 and Lemma 6 by the definition of Σ. □

Now, we prove minimality. As in the case of $\emptyset$-unification, we prove a stronger statement: $\Sigma(GPUP_\emptyset)$ is a disjoint set of unifiers of $GPUP_\emptyset$, where disjointness for a set of P-substitutions is defined as follows:

Definition 27 (Disjoint Set of P-Substitutions). *A set of P-substitutions Σ is called disjoint modulo E with respect to a set of variables Var iff for all θ, $\sigma \in \Sigma$, if there exist P-substitutions λ_1, λ_2 such that*

- *for all sequence variables $\overline{x} \in Var$,*
  - $\overline{x} \leftarrow \notin \lambda_1$,
  - $\overline{x} \leftarrow \notin \lambda_2$,
  - *there exist P-terms $t_1, \ldots, t_n, s_1, \ldots, s_n$, $n \geq 0$ such that $\overline{x}\theta \circ \lambda_1 = t_1, \ldots, t_n$, $\overline{x}\sigma \circ \lambda_2 = s_1, \ldots, s_n$ and for all $1 \leq i \leq n$, either t_i and s_i are the same sequence variables or the P-equality $t_i \doteq_E s_i$ holds;*
- *for all individual variables $x \in Var$, the P-equality*
  - *$x\theta \circ \lambda_1 \doteq_E x\sigma \circ \lambda_2$ holds,*
- *for all index variables $vn \in Var$,*
  - *$vn\theta \circ \lambda_1 \doteq_E vn\sigma \circ \lambda_2$.*

then $\theta = \sigma$*.*

The result analogous to Lemma 1 holds for P-substitutions: disjointness implies minimality.

Theorem 18. *$\Sigma(GPUP_\emptyset)$ is a disjoint set of unifiers for $GPUP_\emptyset$ with respect to the set of variables of $GPUP_\emptyset$.*

Proof. Let Var be the set of variables of $GPUP_\emptyset$ and σ_1 and σ_2 be two substitutions from $\Sigma(GPUP_\emptyset)$. We consider the following cases:

1. σ_1 and σ_2 are attached to the same $SUCCESS$ node and differ from each other by a positive integer solution of the constraint attached to the same node. Clearly, in this case the set $\{\sigma_1, \sigma_2\}$ is disjoint with respect to Var.
2. σ_1 and σ_2 are obtained from two different $SUCCESS$ nodes to which the same constraint is attached. If σ_1 and σ_2 contain different positive integer solutions of the constraint, then disjointness is clear. Otherwise, let σ_1 be $\theta_1 \circ \mu$ and σ_2 be $\theta_2 \circ \mu$, where θ_1 and θ_2 are substitutions associated with the $SUCCESS$ nodes σ_1 and σ_2 are obtained from, and μ is a positive integer solution of the constraint. Then disjointness of $\{\sigma_1, \sigma_2\}$ follows from disjointness $\{\theta_1, \theta_2\}$ with respect to Var, which itself follows from Theorem 9.
3. σ_1 and σ_2 are obtained from two different $SUCCESS$ nodes to which different constraints are attached. Then by the construction of the unification tree, these two constraints do not have a common solution, which implies disjointness of $\{\sigma_1, \sigma_2\}$ with respect to Var.

The fact that every pair of P-substitutions from $\Sigma(GPUP_\emptyset)$ is disjoint with respect to Var implies disjointness of $\Sigma(GPUP_\emptyset)$ with respect to Var. □

7 Second Extension - Sequence Variables in Expressions with Fixed Arity Heads

The results of this section are valid for arbitrary E-unification, therefore formulations are in general form, not restricted to the $\emptyset$-unification. We consider E-unification problems written in extended syntax - allowing sequence variables to appear as arguments in expressions with fixed arity heads. Expressions written in such an extended syntax can be considered as abbreviations for terms or equations written in the standard syntax (defined in Section **??**) and can be used to have more compact representation. For instance, for a ternary function symbols $ffix$, instead of referring to three terms $ffix(a, x, y)$, $ffix(x, a, y)$ and $ffix(x, y, a)$ explicitly, we could use an expression in the extended syntax $ffix(\overline{x}, a, \overline{y})$. But note that not all the expressions in the extended syntax abbreviate terms or equations written in the standard syntax. For instance, the expression $ffix(\overline{x}, \overline{x})$ for a ternary $ffix$ has no counterparts written in the standard syntax.

Intuitively, an E-unification problem written in the extended syntax represents a set of E-unification problems written in the standard syntax. Therefore, to solve an extended E-unification problem, first we need to reduce them to the corresponding set E-unification problems in standard syntax and then apply known unification procedures on each element of the set.

We discuss these issues more formally. We defined the alphabet (V, C, P) in Section **??**. Now we introduce two new notions[2]:

Definition 28 (Quasi-Term). *The set of quasi-terms (over (V, C, P)) is the smallest set of strings over (V, C, P) that satisfies the following conditions:*

- *If $v \in IV \cup SV$ then v is a quasi-term.*
- *If $c \in CONST$ then c is a quasi-term.*
- *If $f \in FFIX \cup FFLEX$ and $t_1, \ldots, t_n$ $(n \geq 0)$ are quasi-terms, then $f(t_1, \ldots, t_n)$ is a quasi-term.*

Definition 29 (Quasi-Equation). *The set of quasi-equations (over the alphabet (V, C, P)) is the smallest set of strings over (V, C, P) that satisfies the following condition:*

- *If $t_1, \ldots, t_n$ $(n \geq 0)$ are quasi-terms over (V, C, P), then $\doteq (t_1, \ldots, t_n)$ is a quasi-equation over (V, C, P).*

We extend the notion of instance on quasi-terms and quasi-equation. We are interested in quasi-terms and quasi-equation which have a counterpart written in the standard syntax in the sense that at least one instance of the quasi-term (quasi-equation) is a term (equation) over (V, C, P).

Definition 30 (Extended Terms and Equations). *A quasi-term qt (resp. a quasi-equation qe) is called an extended term (extended equation) over the alphabet (V, C, P) iff there exist a substitution σ such that $qt\sigma$ is a term over (V, C, P) ($qe\sigma$ is an equation over (V, C, P)).*

In order to decide whether a quasi-term (quasi-equation) is an extended term (extended equation), we associate a system of linear Diophantine equations to them and show that if the system has a natural solution, then quasi-term (quasi-equation) is an extended term (extended equation).

Definition 31 (System of Linear Diophantine Equations Associated with a Quasi-Term). *For a quasi-term qt, the associated system of linear Diophantine equations $DS(qt)$ is defined in the following way:*

- *If $qt \in IV \cup SV \cup CONST$ then $DS(qt)$ is empty.*
- *If qt is $f(t_1, \ldots, t_l)$, where $f \in FFIX$ and $AR(f) = n$, then $DS(qt)$ is*

$$k_1 X_1 + \ldots + k_s X_s = n - (l - (k_1 + \ldots + k_s)) \& DS(t_1) \& \ldots \& DS(t_l)$$

where $k_1, \ldots, k_s$ $(s \geq 0)$ are numbers of occurrences of all distinct sequence variables among $t_1, \ldots, t_l$, the variables $x_1, \ldots, x_s$ are metavariables corresponding to the sequence variables and & denotes conjunction on the meta level.

[2] The results below are valid for unification with patterns as well.

– *If qt is $f(t_1, \ldots, t_l)$, where $f \in FFLEX$, then $DS(qt)$ is*

$$DS(t_1)\& \ldots \& DS(t_l).$$

Definition 32 (System of Linear Diophantine Equations Associated with a Quasi-Equation). *For a quasi-equation qe, the associated system of linear Diophantine equations $DS(qe)$ is defined in the following way:*

– *If qe is $\doteq(t_1, \ldots, t_l)$, then $DS(qe)$ is*

$$k_1X_1 + \ldots + k_sX_s = 2 - (l - (k_1 + \ldots + k_s))\&DS(t_1)\&DS(t_2)$$

where $k_1, \ldots, k_s$ ($s \geq 0$) are numbers of occurrences of all distinct sequence variables among $t_1, \ldots, t_l$, the variables $X_1, \ldots, X_s$ are metavariables corresponding to the sequence variables and & denotes conjunction on the meta level.

Theorem 19. *A quasi-term qt (quasi-equation qe) is an extended term (extended equation) if $DS(qt)$ ($DS(qe)$) has a natural solution.*

Proof. We prove the theorem for a quasi-term qt. For a quasi-equation the proof is the same. Let $\overline{x_1}, \ldots, \overline{x_s}$ be all sequence variables of qt, occurring as arguments of subterms with fixed arity head and $X_1 = n_1, \ldots, X_s = n_s$ be a natural solution of $DS(qt)$. Then we assemble the substitution σ as follows: $\sigma = \{\overline{x_1} \leftarrow z_1^1 \ldots, z_{n_1}^1, \ldots, \overline{x_s} \leftarrow z_1^s \ldots, z_{n_s}^s\}$, where all z-s are new distinct individual variables. Then by the construction of the system $DS(qt)$, for all subterm no sequence variable occurs as an argument of a subterm of $qt\sigma$ with fixed arity head and for all such subterms, number of arguments coincide with the arity of its head. Therefore, by Definition30, qt is an extended term. □

It is clear that an extended E-unification problem is decidable iff E-unification problem is decidable.

We design a unification procedure for extended E-unification problem as a tree generation process with three basic steps: arity completion, projection and transformation. Projection and transformation are those of E-unification. As for the arity completion, we define it as follows:

Let $\overline{x_1}, \ldots, \overline{x_k}$ be all sequence variables occurring in the unification problem UP as arguments of terms with fixed arity head and let $DS(UP)$ be the system of linear Diophantine equations associated with UP, with the variables $X_1, \ldots, X_k$, corresponding to $\overline{x_1}, \ldots, \overline{x_k}$. Then:

– if $DS(UP)$ has no solution in natural numbers, then $SUB(UP) = \langle\rangle$ and $SUC(UP) = FAILURE$;
– otherwise, let

$$X_1 = n_1^1, \ldots, X_k = n_k^1,$$

$$\ldots$$

$$X_1 = n_1^s, \ldots, X_k = n_k^s$$

be all natural solutions of $DS(UP)$, then $SUB(UP) = \langle \sigma_1, \ldots, \sigma_s \rangle$ and $SUC(UP) = \langle\ t_1\sigma_1 \doteq t_2\sigma_1, \ldots, t_1\sigma_s \doteq t_2\sigma_s \rangle$, where

$$\sigma_1 = \{\overline{x_1} \leftarrow z_1^1, \ldots, z_{n_1^1}^1, \ldots, \overline{x_k} \leftarrow z_1^k, \ldots, z_{n_k^1}^k\}$$

$$\ldots$$

$$\sigma_s = \{\overline{x_1} \leftarrow z_1^1, \ldots, z_{n_1^s}^1, \ldots, \overline{x_k} \leftarrow z_1^k, \ldots, z_{n_k^s}^k\},$$

with all z-s being new distinct individual variables.

σ-s are called arity completing substitutions for UP. Now we define the extended E-unification procedure as a tree generation process. The root of the tree is labeled with UP, first level nodes of are successors of UP with arity completion, and the corresponding edges are labeled with arity completing substitutions. After completing arity, we proceed exactly as constructing the unification tree for E-unification.

8 Implementation

We have implemented the extended $\emptyset$-unification procedure in Mathematica. As a decision procedure for being a term and on the arity completion phase of the procedure we use the Omega package [3] to solve linear Diophantine equations over naturals. In the tree generation process for the efficiency reasons we did not implement the decision procedure for unifiability but put the bound on the depth of the tree which can be changed by the user - the idea which was used in the system for combining equational unification algorithms UNIMOK [19]. We added a loop-checking as an option, which makes the procedure terminate in case of infinite cyclic success. Another termination condition, based on Theorem 15, is also implemented.

The package is incorporated into the Theorema system [9] as a part of the Theorema Equational Prover. It makes Theorema probably the only system being able to handle solving and proving equations which involve sequence variables and flexible arity symbols. The package also enhances Mathematica solving capabilities, considering unification as a solving method. We used the package, for instance, to find matches for S-polynomials in non-commutative Gröbner basis algorithm [23].

9 Conclusion

We considered a unification problem for the equational theory with sequence and individual variables, free fixed arity function symbols and free flexible arity function symbols, showed that the problem is decidable and gave a unification procedure which enumerates the minimal complete set of unifiers for the given unification problem. Sufficient conditions for termination are established.

We extended the theory in two ways. First extension is with constructs called pattern-terms, used to abbreviate sequences with unknown length of terms that match certain pattern. We proved that the unification procedure for the extension enumerates substitution/constraint pairs which constitute the minimal complete set of solutions of the problem.

The question which remains open is decidability of unification problem for the theory extended with pattern-terms. Semi-decidability of the problem is obvious.

Second extension allows sequence variables to occur in terms with fixed arity heads. We showed how a unification problem in this extension can be reduced to a finite set of standard unification problems sequence and individual variables, free fixed and flexible arity function symbols (where sequence variables occur only in terms with flexible arity heads).

As a future work, we intend to study unification in various non-free (for instance, flat, restricted flat, orderless, flat-orderless, restricted flat-orderless) equational theories with sequence variables and flexible arity symbols. Extensions with pattern-terms which abbreviate more general forms of sequences of unknown length is another interesting topic.

10 Acknowledgments

I wish to thank Prof. Bruno Buchberger for supervising the work and for many helpful discussions.

References

1. H. Abdulrab and J.-P. Pécuchet. Solving word equations. *J. Symbolic Computation*, 8(5):499–522, 1990.
2. F. Ajili and E. Contejean. Complete solving of linear diophantine equations and inequations without adding variables. In *Principles and Practice of Constraint Programming*, pages 1–17, 1995.
3. G.E. Andrews, P. Paule, and A. Riese. Macmahon's partition analysis III: The Omega package. *European J. Combinatorics*, 22:887–904, 2001.
4. F. Baader and K. Schulz. General A- and AX-unification via optimized combination procedures. In *Proceedings of the Second International Workshop on Word Equations and Related Topics, IWWERT-91*, volume 677 of *LNCS*, pages 23–42, Rouen, France, 1992. Springer Verlag.
5. F. Baader and K. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction, CADE-92*, volume 607 of *LNCS*, pages 50–65, Saratoga Springs, USA, 1992. Springer Verlag.
6. M. Benedikt, L. Libkin, T. Schwentick, and L. Segoufin. String operations in query languages. In *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 2001.
7. B. Buchberger. Mathematica as a rewrite language. In T. Ida, A. Ohori, and M. Takeichi, editors, *Proceedings of the 2nd Fuji International Workshop on Functional and Logic Programming)*, pages 1–13, Shonan Village Center, Japan, 1–4 November 1996. World Scientific.
8. B. Buchberger. Personal communication, 2001.

9. B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. The Theorema project: A progress report. In M. Kerber and M. Kohlhase, editors, *Symbolic Computation and Automated Reasoning (Proceedings of CALCULEMUS 2000)*, pages 98–113, St.Andrews, 6–7 August 2000.
10. A. Colmerauer. An introduction to Prolog III. *CACM*, 33(7):69–91, 1990.
11. A. Farquhar, R. Fikes, and J. Rice. The Ontolingua Server: A tool for collaborative ontology construction. *Int. Journal of Human-Computer Studies*, 46(6):707–727, 1997.
12. M. R. Genesereth. Epilog for Lisp 2.0 Manual. Technical report, Epistemics Inc., Palo Alto, 1995.
13. M. R. Genesereth and R. E. Fikes. Knowledge Interchange Format, Version 3.0 Reference Manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, Stanford, June 1992.
14. S. Ginsburg and X. S. Wang. Pattern matching by Rs-operations: Toward a unified approach to querying sequenced data. In *Proceedings of the 11th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 293–300, San Diego, 2–4 June 1992.
15. G. Grahne, M. Nykänen, and E. Ukkonen. Reasoning about strings in databases. In *Proceedings of the Thirteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 303–312, Minneapolis, 24–26 May 1994.
16. G. Grahne and E. Waller. How to make SQL stand for string query language. *LNCS*, 1949:61–79, 2000.
17. M. Hamana. Term rewriting with sequences. In *Proceedings of the First International Theorema Workshop, RISC-Linz Technical Report 97-20*, Hagenberg, Austria, 9–10 June 1997.
18. J. Jaffar. Minimal and complete word unification. *J. of ACM*, 37(1):47–85, 1990.
19. S. Kepser and J. Richts. UniMoK: A system for combining equational unification algorithms. In P. Narendran and M. Rusinowitch, editors, *Proceedings of the 10th International Conference on Rewriting Techniques and Applications RTA-99*, volume 1631 of *LNCS*, pages 248–251. Springer Verlag, 1999.
20. D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–298, Oxford, 1967. Pergamon Press. Appeared 1970.
21. G. S. Makanin. The problem of solvability of equations on a free semigroup. *Math. USSR Sbornik*, 32(2), 1977.
22. G. Mecca and A. J. Bonner. Sequences, Datalog and transducers. In *Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 23–35, San Jose, 22–25 May 1995.
23. F. Mora. Gröbner bases for non-commutative polynomial rings. In J. Calmet, editor, *Proceedings of the 3rd International Conference on Algebraic Algorithms and Error-Correcting Codes (AAECC-3)*, volume 229 of *LNCS*, pages 353–362, Grenoble, July 1985. Springer Verlag.
24. P.Hayes and C. Menzel. A semantics for the knowledge interchange format. Available from http://reliant.teknowledge.com/IJCAI01/HayesMenzel-SKIF-IJCAI2001.pdf, 2001.
25. G. Plotkin. Building in equational theories. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 7, pages 73–90, Edinburgh, 1972. Edinburgh University Press.
26. A. Rubio. Theorem proving modulo associativity. In *Proceedings of the Conference of European Association for Computer Science Logic*, LNCS, Paderborn, Germany, 1995. Springer Verlag.

27. K. Schulz. Makanin's algorithm for word equations - two improvements and a generalization. In K. Schulz, editor, *Proceedings of Word Equations and Related Topics (IWWERT'90)*, number 572 in LNCS, pages 85–150. Springer Verlag, 1990.
28. K. U. Schulz. Word unification and transformation of generalized equations. *J. Automated Reasoning*, 11(2):149–184, 1993.
29. J. Siekmann. Unification and matching problems. Memo, Essex University, 1978.
30. M. Widera and C. Beierle. A term rewriting scheme for function symbols with variable arity. Technical Report 280, Praktische Informatik VIII, FernUniversitaet Hagen, Germany, 2001.
31. S. Wolfram. *The Mathematica Book*. Cambridge University Press and Wolfram Research, Inc., fourth edition, 1999.