# Unification in the Empty and Flat Theories with Sequence Variables and Flexible Arity Symbols*

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, Austria
teimuraz.kutsia@scch.at

The problem of studying equational theories with sequence variables and flexible arity symbols was posed by Bruno Buchberger and was inspired by a mathematical software system Mathematica [19], the programming language of which is a functional language [5] with an additional feature - sequence variables. In the usual variants of predicate logic sequence variables are not included. However, they are a convenient feature that often is used implicitly in every-day mathematics. In fact, sequence variables together with a natural induction principle for sequence variables (see [6]) expand the power of logic by an implicit formalization of natural numbers.

Sequence variables are used in the Theorema system [7] - an integrated environment for proving, solving and computing, built on the top of Mathematica. The proving context, where they occur in Theorema, makes it important to study unification problem with sequence variables. Sequence variables in the context of term rewriting were studied in [10]. In [8] they are used in inductive theorem proving over lists.

We use the term "flexible arity symbols" for symbols that can have an arbitrary arity. Such symbols occur in programming languages, term rewriting systems, etc. and sometimes are called "variable arity symbols". We build terms over two types of variables - individual and sequence variables, constants and two types of function symbols - of fixed and flexible arity. Sequence variables can be substituted by any (possibly empty) sequence of terms. We do not introduce an additional constructor for sequences in the theory. Presence of fixed arity symbols and sequence variables makes it necessary to solve a system of linear Diophantine equations over naturals to decide whether an expression is a term (or an equation).

We address the problem of solving equations (unification) in a theory with sequence variables and flexible arity symbols. This problem can be seen as a generalization of the problems like word equations [14–16], general A-unification [3],

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

equations over lists of atoms with concatenation [9] or Mathematica pattern matching.

We consider a general unification problem in empty and flat theories (a theory with a flat flexible arity symbol). A specific feature of the problem is that we require from unifiers to complete arity, which means that after applying the unifier on the terms to be unified no sequence variable occurs as an argument of a term with a fixed arity head. We answer two questions for these theories: a) is it decidable whether a unification problem is solvable and b) how to obtain a minimal unification procedure.

We show decidability of unification in empty and flat theories in two steps. First, with solvability preserving transformations we reduce the problem to a "simpler" unification problem. Second, we apply Baader-Schulz combination method [4] to the reduced problem and show its decidability. In fact, after these two steps the decidability problem for unification in the empty theory is reduced to the decidability problems of a word equation with linear constant restrictions, a word equation with individual variable restrictions on length (demanding that for a variable $x$ in the word equation, which corresponds to an individual variable in the original unification problem, and for a unifier $\theta$ of the word equation, the length of $x\theta$ must be 1) and Robinson unification with linear constant restrictions. For the flat theory, in addition, we have decidability of word equations with linear constant restrictions among reduced decidability problems.

Unification procedure both in the empty and flat theories is designed as a tree generation process. We design three kinds of single steps of the process, called projection, arity completion and transformation. The idea of projection (see e.g. [1]) is to substitute some sequence variables with the empty sequence. Arity completion replaces sequence variables, which occur in the unification problem as arguments of terms with fixed arity heads, by sequences of new individual variables such that the obtained expressions are unification problems. Transformation for the empty theory is based on "widening" techniques and generalizes the basic algorithm for word unification [13, 15, 17, 18]. Transformation for the flat theory, in addition, applies special rules based on the "widening" techniques as well, to unify terms with a flat head.

We build the unification tree both for the empty and flat theories in a breadth first (level by level) manner. Each node of the tree is labeled either with a unification problem, $SUCCESS$ or $FAILURE$. Only the nodes labeled with $SUCCESS$ or $FAILURE$ are terminal nodes. The tree is generated as follows: If $UP$ is a unification problem, we label the root of the tree with $UP$ (zero level). First level nodes (the children of the root) of the tree are obtained from $UP$ by projection if $UP$ is solvable, otherwise we stop with failure. Second level nodes are obtained by arity completion, completing arity for each solvable unification problem attached to the first level nodes. Starting from the second level, we apply only a transformation step to a solvable unification problem of each node, thus getting new successor nodes. Note that for each node in the tree, if the unification problem attached to the node is not solvable we replace the label on the node by $FAILURE$. The branch with the leaf labeled by $SUCCESS$ is called a successful branch. For each node in the tree, the edge to a successor of the node is labeled with the substitution which was used to derive the successor. We compose substitutions (top-down) displayed on the edges of the successful branch, restrict the composed substitution on the variables of the unification problem and associate the obtained substitution to the leaf of the branch, which is labeled with $SUCCESS$. If $\Sigma(UP)$ is the set of all substitutions associated with successful leaves of the unification tree for $UP$, then $\Sigma(UP)$ is the minimal complete set of the unifiers of $UP$. Tree generation process terminates if the set is finite.

*Example 1.* Compute the minimal complete set of unifiers in the empty and flat theories ($\overline{x}, \overline{y}, \overline{u}, \overline{v}$ are sequence variables, $z$ is an individual variable, $a$, $b$, $c$ are constants, $f$ and $g$ are flexible arity function symbols, $h$ is a flat flexible arity function symbol, $s$ is a binary function symbol):

1. $MCU_{\emptyset}(f(\overline{x}, \overline{y}, z), f(a, b, c)) = \{\{\overline{x} \leftarrow, \ \overline{y} \leftarrow a, b, z \leftarrow c\}, \ \{\overline{x} \leftarrow a, \overline{y} \leftarrow b, z \leftarrow c\}, \ \{\overline{x} \leftarrow a, b, \overline{y} \leftarrow, z \leftarrow c\}\}$.
2. $MCU_{\emptyset}(f(g(a, \overline{x}), \ g(\overline{y}, c)), \ f(\overline{u}, \ g(b, \overline{v}))) = \{\{\overline{u} \leftarrow g(a, \overline{x}), \overline{y} \leftarrow b, \overline{v} \leftarrow c\}, \ \{\overline{x} \leftarrow, \overline{u} \leftarrow g(a), \overline{y} \leftarrow b, \overline{v} \leftarrow c\}, \ \{\overline{u} \leftarrow g(a, \overline{x}), \overline{y} \leftarrow b, \overline{y}, \ \overline{v} \leftarrow \overline{y}, c\}, \ \{\overline{x} \leftarrow, \overline{u} \leftarrow g(a), \overline{y} \leftarrow b, \overline{y}, \ \overline{v} \leftarrow \overline{y}, c\}\}$.
3. $MCU_{\emptyset}(f(s(a, \overline{x}), s(\overline{y}, \ c)), f(\overline{u}, s(b, \overline{v}))) = \{\{\overline{x} \leftarrow z, \ \overline{u} \leftarrow s(a, z), \ \overline{y} \leftarrow b, \ \overline{v} \leftarrow c\}\}$ (here $z$ appears to complete arity of $s(a, \overline{x})$).
4. $MCU_{\emptyset}(f(\overline{x}), f(\overline{y})) = \{\{\overline{x} \leftarrow, \overline{y} \leftarrow\}, \{\overline{x} \leftarrow \overline{y}\}\}$. Note that the substitution $\{\overline{x} \leftarrow \overline{y}\}$ is not more general than $\{\overline{x} \leftarrow, \overline{y} \leftarrow\}$ because any substitution $\sigma$, such that $\{\overline{x} \leftarrow \overline{y}\} \circ \sigma = \{\overline{x} \leftarrow, \overline{y} \leftarrow\}$, replaces $\overline{y}$ with the empty sequence, while, by the definition of more general substitutions, it should not replace any sequence variables with the empty sequence.
5. $MCU_{\emptyset}(\overline{x}, \overline{y}) = \{\{\overline{x} \leftarrow \overline{y}\}\}$.
6. $MCU_{Flat}(h(\overline{x}, z), \ h(a, \ b, \ c)) = \{\{\overline{x} \leftarrow, \ z \leftarrow h(a, \ b, \ c)\}, \ \{\overline{x} \leftarrow a, \ z \leftarrow h(b, \ c)\}, \ \{\overline{x} \leftarrow h(a), \ z \leftarrow h(b, \ c)\}, \ \{\overline{x} \leftarrow a, b, \ \ z \leftarrow c\}, \ \{\overline{x} \leftarrow h(a), \ b, \ \ z \leftarrow c\}, \ \{\overline{x} \leftarrow a, h(b), \ z \leftarrow c\}, \ \{\overline{x} \leftarrow h(a), h(b), \ z \leftarrow c\}, \ \{\overline{x} \leftarrow h(a, b), \ z \leftarrow c\}, \ \{\overline{x} \leftarrow a, b, \ z \leftarrow h(c)\}, \ \{\overline{x} \leftarrow h(a), b, \ z \leftarrow h(c)\}, \ \{\overline{x} \leftarrow a, h(b), \ z \leftarrow h(c)\}, \ \{\overline{x} \leftarrow h(a), h(b), \ z \leftarrow h(c)\}, \ \{\overline{x} \leftarrow h(a, b), \ z \leftarrow h(c)\}\}$.

We have implemented the unification procedure for the empty theory in Mathematica 4.0. As a decision procedure for being a term as well as on the arity completion phase of the procedure we use the Omega package [2] 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 unification algorithms UniMoK [11].

As a future work, we intend to study unification in orderless and flat-orderless theories with sequence variables and flexible arity symbols. Orderless flexible arity symbols are counterparts of commutative fixed arity symbols.

The full version of this paper ( [12]) is available from `http://www.sfb013.risc.uni-linz.ac.at/`.

## References


1. H. Abdulrab and J.-P. Pécuchet. Solving word equations. *Journal of Symbolic Computation*, 8(5):499–522, 1990.
2. G.E. Andrews, P. Paule, and A. Riese. Macmahon's partition analysis III: the Omega package. SFB-Report 99-24, Johannes Kepler University Linz, 1999.
3. 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*, volume 677 of *Lecture Notes in Artificial Intelligence*, pages 23–42. Springer-Verlag, 1993.
4. F. Baader and K. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. *Journal of Symbolic Computation*, 21(2):211–244, 1996.
5. B. Buchberger. Mathematica as a rewrite language. In *Proceedings of Second Fuji International Workshop on Functional and Logic programming*, pages 1–13. Word Scientific, 1997.
6. B. Buchberger. The concept of sequence variables in logic languages. SFB-report, Johannes Kepler University Linz, April 2001.
7. B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. Theorema: A progress report. In M. Kerber and M. Kohlhase, editors, *Proceedings of the 8th Symposium on the Integration of Symbolic Computation and Mechanized Reasoning*, pages 100–115, St. Andrews, Scotland, August 6-7 2000.
8. B. Buchberger and D. Vasaru. Theorema: the induction prover over lists. In B. Buchberger, T. Ida, and D. Vasaru, editors, *Proceedings of First International Theorema Workshop*, number 97-20 in RISC Technical Report Series. Research Institute for Symbolic Computation, Johannes Kepler University Linz, 1997.
9. A. Colmerauer. An introduction to Prolog III. *Communications of the ACM*, 33(7):69–91, 1990.
10. M. Hamana. Term rewriting with sequences. In B. Buchberger, T. Ida, and D. Vasaru, editors, *Proceedings of First International Theorema Workshop*, number 97-20 in RISC Technical Report Series. Research Institute for Symbolic Computation, Johannes Kepler University Linz, 1997.
11. 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*, volume 1631 of *Lecture Notes in Computer Science*, pages 248–251. Springer-Verlag, 1999.
12. T. Kutsia. Unification in the empty and flat theories with sequence variables and flexible arity symbols. SFB-Report 01-13, Johannes Kepler University Linz, 2001.
13. A. Lentin. Equations in free monoids. In M. Nivat, editor, *Automata, Languages and Programming*, pages 67–85. North-Holland, Amsterdam, 1972.
14. G. Makanin. The problem of solvability of equations on a free semi-group. *Math. USSR Sbornik*, 32(2), 1977.
15. G. Plotkin. Building-in equational theories. *Machine Intelligence*, 7:73–90, 1972.
16. K. Schulz. Word unification and transformation of generalized equations. *Journal of Automated Reasoning*, 11(2):149–184, 1993.
17. J. Siekmann. String unification. Technical report, Essex University, 1975.
18. J. Siekmann. *Unification and matching problems*. PhD thesis, Essex University, 1978.
19. S. Wolfram. *The Mathematica book*. Cambridge University Press and Wolfram Research, Inc., fourth edition, 1999.