Parametric Terms and Stratified Second-Order Terms

Manfred Schmidt-Schauß,
Fachbereich Informatik,
Johann Wolfgang Goethe-Universität Frankfurt
60054 Frankfurt
E-mail: schauss@informatik.uni-frankfurt.de

For CADE-96 Workshop on term representation

Abstract.

We consider terms that are constructed from function symbols, constant symbols, variables, and monadic second-order variables that may stand for a term with one hole. The problem of unifying a set of equations between these second-order terms requires an extension to include also parametric terms.

The unification problem of stratified second order systems, where for every first-order and second-order variable, the string of second-order variables on the path from the root of a term to every occurrence of this variable is always the same, and where the input system does not contain parametric terms is decidable.

The non-deterministic decision algorithm usues eventually Makanin's algorithm for deciding the unifiability of word equations.

As a generalization, we also permit second order terms with parametric terms in the input, and describe conditions for decidablity. A slight extension of the unification rules can be used for a terminating unification algorithm in the general case. However, the answer may in certain cases be “unknown”.

1 Introduction.

Finding solutions for equations between terms containing variables is a basic mechanism for several algorithms in automated deduction, term rewriting, and logic programming. This operation is commonly called unification of terms.

Most work has been devoted to unification of first order terms, though unification of higher order terms has always attracted some interest. This interest is now increasing, which is due to the popularisation of higher order programming languages and higher-order extension of logic programming and automated deduction systems. A further source of higher order problems is the schematisation of terms in term rewriting and automated deduction .

The general second order unification problem is undecidable [Gol81], even if only one binary function symbol is available (besides constants and unary function symbols), (cf. [Fa91]). However, the semantics used there to show undecidability is different from ours. It appears to be an open problem whether the second order unification problem under the restricted “term with one hole” semantics is decidable.

The “one-hole-semantics” is an interesting specialisation of the general semantics, and also appears in a natural way in schematising infinite sets of terms. For example, investigating unification of first-order terms under two-sided distributivity (cf. [Sz82, Si89, BS93, Con93, Sch94] reduces to a unification problem for second-order terms with the “one-hole-semantics”.

The formulation of the unification problem which we will consider in this paper is as follows. We have function symbols of arbitrary arity in the signature. Terms may contain second order variables in function positions. Such a second order variable always stands for a term with exactly one hole. We consider the restriction that for every first or second-order variable x (resp. X), the second-order context is the same for every occurrence of x. I.e., for every first-order and second-order variable, the string of second-order variables on the path from the root of a term to every occurrence of this variable is always the same. For example f(x, X(h(y))) = f(X(Y(z)), x) is an admissable equation, but f(X(x),x) = f(y,y) is not, since in the last equation, there are two paths from the root to x, and the first contains a second order variable, whereas the second path doesn't.

2. Notation

We use a fixed signature that contains an infinite number of free constants and some functions symbols with fixed arity, where there is at least one function symbol with arity greater than 1 (i.e., arity(f) $\geq$ 2). We also have an infinite supply of first order variables, usually denoted x,y,z, an infinite supply of second-order monadic variables, which we denote by upper-case letters like X, Y, Z, and an infinite supply of integer variables, usually denoted n.
We define second order terms and parametric terms in a mutual recursive way:

2.1 Definition.
A **parametric term** can be:
- i) Ω: the trivial parametric term
- ii) a **basic parametric term** $f(t_1,\ldots t_{i-1},\Omega,t_{i+1},\ldots, t_n)$, where t_i are second order terms (the **parameters**),
- iii) a concatenation of two parametric terms p•q.
- iv A power p^n of a parametric term p, where n is an integer variable (representing a positive natural number).

A **term** can be:
- i) a first-order variable,
- ii) $f(t_1,\ldots, t_n)$ for terms t_i,
- iii) X(t), where X is a second-order variable and t a term,
- iv) p(t), where p is a parametric term and t a term.

At some places it is necessary to consider Ω as a trivial parametric term. In general we will assume that parametric terms are not trivial. ♦

An example for a syntactically permitted term is $g(a,\Omega)\ (f(X(x),\ f(\Omega,Y(X(y)))\ (Y(g(a)))\ )\ )$, where X,Y are SO-variables, x,y are FO-variables, g is a unary function symbol, f is a binary function symbol, and a is a constant. Note that $f(X(\Omega),x)$ is not a parametric term, whereas $f(X(x), \Omega)$ and $f(X(g(\Omega)(a)), x)$ are permitted.

We call a term **ground**, iff it does not contain variables. A term is called a **first order** term, iff it does not contain parametric terms nor second order variables. Every ground term can be reduced to a first order ground term using the following reductions:

i) $f(t_1,\ldots t_{i-1},\Omega,t_{i+1},\ldots, t_n)(s) \rightarrow f(t_1,\ldots t_{i-1},s,t_{i+1},\ldots, t_n)$
ii) $\Omega(t) \rightarrow t$
iii) $p \bullet q(t) \rightarrow p(q(t))$
iv) $p^1(t) \rightarrow p(t)$
v) $p^N(t) \rightarrow p(p^{N-1}(t))$, where N is an integer greater than 1.

We need extended substitutions as solutions to unification problems. Therefore we need ground parametric terms, which can be seen as lambda-abstracted terms, and are also called "terms with one hole". Then a term with a hole is a first-order term t (up to an occurrence of Ω), and Ω occurs exactly once in t. A (second-order ground-) substitution is a mapping that can be represented by a finite set of pairs (x_i, t_i), (X_i, p_i), (n,N) where t_i is a first-order ground term, p_i is a first-order ground parametric term, and N is a positive integer.
The application of a ground substitution σ to a second-order term s is as follows: If $\sigma x = t$, then all x's in s are replaced by t; if $\sigma X = p$, then every occurrence $X(u)$ in s is replaced by $p[\Omega \rightarrow u]$, i.e., the usual β-reduction, and all integer variables are replaced by the corresponding integer.

A ground parametric term can be considered as a string of basic ground parametric terms, where composition is viewed as concatenation. The "letters" are the basic ground parametric terms and concatentation of string corresponds to composition. Using this point of view, we can also speak of powers, i.e. strings of the form s^n, where power means n-fold concatenation.

For the purposes of the algorithm, we simplify parametric terms and applications of parametric terms to terms as follows:
Using associativity, we can flatten concatenations and applications. Thus every application of a parametric term is of the form $(p_1 \bullet \ldots \bullet p_n)(t)$, where p_i are basic parametric terms, or proper powers p^m, and t does not start with a parametric term.

The depth of a term that is adapted to our unification algorithm and denoted as p-depth. The differences to the usual depth are:
$\text{p-depth}((p_1 \bullet \ldots \bullet p_n) = 1 + \max\{\text{p-depth}(p_i), i=1,\ldots,n\}$,
$\text{p-depth}(X(t)) = 1+\text{p-depth}(t)$.
Note that the reduction to a flattened representation makes the p-depth smaller.

In the following we will work with systems of equations (and also disequations), i.e. a multi-set of pairs of terms, where the pairs of terms are interpreted as equations (or disequations, respectively). We will use $\mathrm{Term}(\Gamma)$ for the set of terms occurring in Γ and $\mathrm{Var}(\Gamma)$ for the set of variables occurring in Γ. A substitution σ is a (usual) **unifier** of Γ, if for every equation $s = t$, the terms s and t become equal after applying the substitution σ (and for every disequation $s \neq t$, we have $\sigma s \neq \sigma t$). We consider equations and disequations as symmetric. This makes the denotation of some unification rules simpler, but also implies some non-determinism, for example the rule Replace-Variable is non-deterministic, if the equation is $x = y$.

2.2 Definition.
For every position π of some term in Γ, the **second-order-prefix** of π is the string of second-order variables on a path from the root to this position. To be more rigorous, let SOP be a function computing this second-order prefix as follows:
$SOP(t,\pi) = \varepsilon$, if t is a variable or constant or Ω, and ε denotes the empty word.
$SOP(f(t_1,\ldots,t_n),k\bullet\pi) = SOP(t_k,\pi)$ for terms or basic parametric terms.
$SOP(X(t),\ 1\bullet\pi) = X\bullet SOP(t,\pi)$
$SOP(p(t),\ 0\bullet\pi) = SOP(p,\pi)$
$SOP((p_1\bullet\ldots\bullet p_n),\ k\bullet\pi) = SOP(p_k,\pi)$
A set of terms S is called **stratified**, iff
i) for every first and second-order variable x, the second-order-prefixes of every position of x in some term in S are the same.
ii) the second order-prefix of every Ω in some term from S is the empty string.
The system Γ is called **stratified**, iff the set of terms $Term(\Gamma)$ is stratified.
♦

Note that this property is invariant w.r.t. the flattening operation.

3. Stratified Systems of Second Order Terms

Unification Problem.
Input: a stratified system Γ of equations
no occurrences of parametric terms.
The issue is to decide, whether there exists a unifier of Γ.

This does not mean that we can forget about parametric terms. The parametric terms are an important ingredient for the intermediate systems of equations and they are immediately introduced by the elimination rules for SO-cycles.

It is shown in [Sch95], that these stratified unification problems can be decided by Makanin's decision algorithm for unification [Ma71, Sch93]. In the following we try to sketch the ideas of the algorithm.

A ground parametric term p can be interpreted as a description of a set of words, where the alphabet is the set of basic ground parametric terms occurring in p. We say a word w is a **power**, if there is another word u, such that $w = u^k$ for some integer $k \geq 2$, (where exponentiation means k-fold concatenation of words), otherwise, we say w is a **non-power**.

Furthermore, we say a string u is a **cyclic permutation** of a string w, if there are two strings u_1 and u_2, such that $u = u_1u_2$ and $w = u_2u_1$. The empty word is denoted as ε.

We will also use a restricted variant of second-order variables, so-called **F-variables** that can be viewed as abstracting a function symbol. An F-variable has a built-in arity n. A substitution also has to instantiate F-variables as follows: A substitution σ for a F-variable F can be represented as a lambda-expression as follows: $\sigma F = \lambda x_1,\ldots,x_n$: $f(s_1, \ldots s_m)$, where $\{s_1,\ldots,s_m\} = \{x_1,\ldots,x_n\} \cup M$, as multi-sets, where M is a multi-set of ground terms. The application of σ to $F(t_1,\ldots,t_n)$ gives the term $f(s_1, \ldots s_m)$, where the x_i ´s are replaced by σt_i. Note that these F-variables do not appear in term equations, but only in an extra set of equations between parametric terms

Eliminating SO-cycles is a central method for the unification algorithm:
Definition. SO-cycles.
Let Γ be a system of equations.
An **SO-cycle** in Γ is a sequence of equations in Γ_T of the following form.

$$X_1(s_1) = t_1,\ X_2(s_2) = t_2, \ldots ,X_n(s_n) = t_n,$$

For $i = 2,\ldots,n$, X_i occurs in t_{i-1} and X_1 occurs in t_n, such that at least one such occurrence of some X_i is not at the top.♦

Note that for SO-cycles in a stratified system of equations the occurrences of the X_i in t_{i-1} and X_1 in t_n are not under a further second-order variable.

Overview on the main ideas of the unification algorithm:

There are the following steps:

- Decomposition Rules for the cases:
  term = term
  $p(s) = f(t_1,\ldots,t_n)$
  $p(s) = q(t)$.

 These rules are sufficient to transform all equations into the form $X(s) = t$.

- Eliminating SO-Cycles
  a) rules to standardise the cycle, make it shorter ...
  b) rules to eliminate cycles
  introduces parametric terms
  removes a second order variable

- Operating on SO-cycle-free Stratified SO-systems.
  rules to guess the instantiation of SO-variables top-down.

- If there are no more term equations, there may be some remaining equations between parametric terms. The transformations are made carefully, such that associative unification can be applied.

  ♦

We have a closer look at the decomposing rules that transform system of equations into **decomposed normal form** (all equations are of the form X(s)□=□t)
As a shorthand for replacing a second order variable X by Ω, and immediate simplification, we will say that X is removed from Γ. We will do the same for integer variables n, if they are instantiated with 1.

Definition. Basic transformation rules
Rule *Replace-Variable*. If x = t is an equation in Γ, and x does not occur in t, then remove the equation x = t, and replace x by t everywhere in Γ.
Rule *Trivial-Equations*. Remove equations of the form x = x, where x is a variable.
Rule *Occur-Check***.** If there is an equation x = t, such that t is not a variable, and x occurs in t, then FAIL: the system is not unifiable.
Rule *Decomposition*
If $f(s_1,\ldots,s_n) = f(t_1,\ldots,t_n)$ is an equation in Γ, then
remove the equation and add n equations $s_i = t_i$, $i = 1,\ldots,n$ to Γ.
Rule *Clash*. If $f(s_1,\ldots,s_n) = g(t_1,\ldots,t_m)$ is an equation in Γ, and $f \neq g$, then FAIL.
the system is not unifiable.
Rule *Constants*. If a =X(t) is in the system, then remove X from Γ.
If a = p(t) is in the system, where p is a (non-trivial) parametric term, then FAIL: the system is not unifiable.

Note that the case x = X(x) cannot occur, since then the system would not be stratified.
The decomposition rule that replaces X(s) = X(t) by s = t is obviously correct, however, it cannot be used as a stand-alone rule, since the stratifiedness of Γ could be destroyed.

Example
As an example for a run of the non-deterministic algorithm on a stratified system consider the second-order unification problem {X(f(x,y)) = f(a, X(z)), X(f(a,b)) = f(u,v)}. This problem satisfies the restrictions above. There is one second-order cycle for the second order variable X. Now it is possible to solve the cycle w.r.t. X in the first equation, which means to remove the X in the first equation and to replace X by the parametric term $(f(a, \Omega))^n$ in the second, where n is an (positive) integer variable. This gives the equations $\{f(x,y) = f(a,z),\ (f(a, \Omega))^n (f(a,b)) = f(u,v)\}$. Decomposing the first equation gives $\{x = a,\ y = z,\ (f(a, \Omega))^n (f(a,b)) = f(u,v)\}$. Splitting n into the cases n= 1 and n= 1+m gives two cases: i) $\{x = a,\ y = z,\ (f(a, \Omega))\ (f(a,b)) = f(u,v)\}$ and thus $\{x = a, y = z, a = u, f(a,b) = v\}$. ii) $\{x = a,\ y = z,\ f(a, (f(a, \Omega))^m (f(a,b))) = f(u,v)\}$. This gives $\{x = a, y = z,\ a = u, v = (f(a, \Omega))^m (f(a,b))\}$.

4 Unification of Parametric Stratified Second Order Systems

We want to apply the unification rules also to a generalised input, such that parametric terms are permitted in the input. Therefore, we need some notions from [Fa88].

A parametric word w is **simple**, if for all p^n that occur in w, the term p has only instances that are non-powers. For example, $(a^n b)^m$ is simple, but $(abb(ab^n))^m$ is not simple, since for n = 2, the basis of the exponent is abbabb, which is a non-power. This property of (general) parametric words is decidable, as shown in [Fa88]

The following restrictions must hold in order to permit parametric terms in the input:

i) there are no powers of the form $(p^n)^m$,
ii) for two occurrences p^n and q^n, the parametric terms p and q are identical
iii) the system is stratified

We call such systems **parametric stratified second order systems**.

It is possible with the methods developed in [Sch95] to give a simple proof of two Corollaries in [Fa88], however, we use the decidability of associative unification, whereas Farmer describes a different algorithm, which may have a better complexity.

Proposition.

i) It is decidable, whether parametric words are simple.
ii) The unification problem for simple parametric words is decidable.
(only the exponents can be instantiated)

The following first non-deterministic transformations checks, whether the input is acceptable, and if the check succeeds, makes the input acceptable for our algorithm. The intuition behind this check is that the algorithm can not handle parametric terms that have instances of the form $(p^m)^n$.

Definition. Input transformation for parametric stratified second order systems Γ.

i) Make a non-deterministic identification on all the basic parametric terms in Γ, i.e., select an equivalence relation $\approx$ on all basic parametric terms, such that $p \approx q$ implies that the top function symbol and the index of Ω in p and q is the same.
For all basic parametric terms p, q with $p \approx q$, replace p by q, and then add the equation resulting from decomposition. I.e. If $p = f(t_1,\ldots,\Omega,\ldots t_n)$ and $q = f(s_1,\ldots,\Omega,\ldots,s_n)$, and Ω occurs at index j, then add $s_i = t_i$ for all $i \neq j$.
ii) Check the following:
Either for all parametric terms of the form p^n, the base p is a simple word, if different parametric terms are considered different;
Or every possible decomposition to a decomposed normal form terminates with FAIL:
not unifiable.

If for all possible non-deterministic choices in i), the check in ii) succeeds, then we call the system **admissable**. ♦

Proposition. Admissability of a parametric stratified second order system is decidable.
Proof. This follows from results in [Fa88] and from Lemma above and since the number of choices for the equivalence relations is bounded by the input size of Γ, and the decomposition algorithm terminates. ♦

Theorem. Unifiability of admissable parametric stratified second-order systems is decidable.

Note that permitting nesting powers $(p^m)^n$ would lead to an unresolved problem during encoding the problem as associative unification problem. This would require an encoding of a set of solutions "$x = a^n$ and $y = x^m$" using a word equation. I have found no way of encoding such a solution, and conjecture that such an encoding doesn't exist.

Open Problem. A related problem that appears to be open is the decidability of the unification problem for parametric words with unique basis of exponents, i.e. systems of equations that satisfy the condition that for all p^n and q^n occurring somewhere, p and q are syntactically equal

For input systems satisfying these restrictions, we will also show that unifiability is decidable. A further generalization is to consider systems that are not stratified, but dynamically stratified, i.e., the algorithm can always use a rule that makes the system smaller. We will show that the unification problem for these systems is decidable.

5 Dynamically Stratified Second Order Systems

In the following we will talk about a generalization to dynamically stratified systems.

The idea for the generalisation is to use the rules without taking care of stratifiedness until we get stuck, or the system is solved. If there is no non-deterministic execution that never gets stuck, then the system can be tested for unifiability using the methods in this paper.

We have to adapt some definitions to meet the purposes of this generalization.
A **flat SO-cycle** is a cycle $X_1(s_1) = t_1$, $X_2(s_2) = t_2$, … ,$X_n(s_n) = t_n$, such that for i = 2,…,n, X_i occurs in t_{i-1} and X_1 occurs in t_n, such that for every i at least one such occurrence has empty second order prefix.
A **deep SO-cycle** is such a cycle that does not satisfy the condition on the second order prefix.
An **SOV-cluster** C is a $\geq_{SO}$-maximal equivalence class, such that every SO-variable in C has an empty second-order prefix.

Now we adapt the rules of section 3 to the new situation:
The following non-deterministic rules may be used in addition
Rule: Remove X from Γ.

Rule Occur-Check**.** If there is an equation x = t, such that t is not a variable, and x occurs in t, and there is some first order function symbol on a path from the root of t to some occurrence of x, then FAIL: the system is not unifiable.

Rule X(s) = X(t) can be replaced by s = t.

Removing SO-cycles, the rules are the same, however they have to concentrate on a flat SO-cycle. It is easy to verify that the rules produce always a new flat and smaller SO-cycle.
In the case of an SO-cycle of length 1, we have the additional rule:

Rule If there is a flat SO-cycle X(s) = t of length 1, and the occurrence of X in t is not unique, then remove X from Γ.

Definition dynamically stratified systems.
We permit second order systems of equations that obey the following restrictions:

i) there are no directly nested powers in parametric terms, i.e. $(p^m)^n$ is forbidden
ii) for two occurrences p^n and q^n, the parametric terms p and q are identical.

Furthermore, the admissablitiy check above should accept the system.
The (dynamic) restriction is that there is some non-deterministic execution of the algorithm, such that no specific execution generates a system with the following property:

i) It is in decomposed normal form
ii) There is no flat SO-cycle.
iii) There is no SOV-cluster.

We will call such systems **dynamically stratified systems.** ♦

This is a decidable condition, simply adding an appropriate check to the algorithm, and then trying all possibilities of branching until it stops with “no rule applicable” or with an empty set Γ_T or with “not unifiable”.

An example of dynamically stratified systems are second-order matching problems (without parametric terms), i.e., unification problems, where in every equation, one term is a ground first -order term.
Another example is a system that becomes stratified after some applications of rules, for example x = X(y), z = Y(x) is not stratified, but elimination of x makes it stratified.

Definition. Structure of the generalized algorithm for dynamically stratified systems.
The main operations of the global algorithm are:

1) If the input contains parametric terms, check whether the system is admissable.
2) Transform all equations into decomposed normal form.
3) If there is a flat SO-cycle, eliminate it.
4) If all equations are in the form X(s) = t, and there is some SOV-cluster,
   Use elimination of SOV-clusters
5) If Γ_T is not empty, but there is no SO-cycle nor a SOV- cluster,
   then return “algorithm is not applicable”.
6) If Γ_T is empty, use the method of section 7) to decide unifiability.

Theorem. The unification problem for dynamically stratified second-order systems is decidable.

In order to extend the algorithm to the case, where the algorithm answers “not applicable”, it appears to be necessary to permit second order variables in Γ_P. This, however, would require new methods and is beyond the scope of this paper.

6 Application to “infinite” term rewriting systems

Using second order variables together with constraints, it is possible to represent several infinite term rewriting systems in a finite way and to show local confluence by overlapping and computing second order unifiers. In order to be general enough, we have to use dynamically stratified systems, or equivalently, we use the unification algorithm until it signals “not applicable”.
We do not formally treat this topic since this is beyond the scope of this paper. In order to apply the results, the unification algorithm in this paper must be modified to be a complete unification algorithm. This however, does not help since the final step of associative unification may generate an infinite set of unifiers. Nevertheless, we want to give some hints on the consequences, under the premise that the only finitely many unifiers are necessary.

An example of a term rewriting system that produces an infinite set of rewrite rules on Knuth Bendix completion is the system
R: $fgfx \to gfx$
On completion, the infinitely many rules $fg^nf\,x \to g^nfx$ are generated
These rules can be represented by the second order rule
R_2: $fXf(x) \to X(f(x)$ together with the constraint $X(z) = g(X(a))$.
Now we make the computation that check for local confluency in the generalized case.

We have several kinds of overlap:
1) Overlap at top level.
$fXf(x) = fYf(x'), X(z) = g(X(a)), Y(z') = g(Y(a))$.
Using the second order unification algorithm for dynamically stratified systems, we get $Xf(x) = Yf(x')$, $X(z) = g(X(a))$ and $Y(z') = g(Y(a))$
We replace X by $g(\Omega)^n$ and Y by $g(\Omega)^m$
$g^n(\Omega)(f(x)) = g^m(\Omega)(f(x')$, $z = g(a)$ and $z' = g(a)$
Since g is unary, the only alternative that will not fail is that $g^n(\Omega) = g^m(\Omega)$ is added to the associative subsystem, which has as only solution $n = m$, and that $x = x'$. In this case the right hand sides are also identical.
2) The overlap is at the inner position of f.
The unification problem is now: $f(x) = fYf(x')$, $X(z) = g(X(a))$, $Y(z') = g(Y(a))$.
We can ignore the second equation, since it is solvable, but does not influence the other parts. The next step is decomposition
$x = Yf(x')$, $Y(z') = g(Y(a))$.

Now we can check for confluence of the critical pair:

$fXfYf\ x' \to XfYf\ x' \ \to\ XYf\ x'$ with constraint $X(z) = g(X(a))$, $Y(z') = g(Y(a))$.

$fXfYf\ x' \to fXYf(x') \to XYfx'$ with the same constraints.

The terms on the right hand side are identical, hence all the critical pairs that are represented can be joined.

3) The overlap is within the second order variable.

This gives $X_2(f(x)) = fYf(x')$, $X_1X_2(z) = g(X_1X_2(a))$, $Y(z') = g(Y(a))$.

We will get $X_1(\Omega) = g^n(\Omega)$, $X_2(\Omega) = g^m(\Omega)$ and $Y(\Omega) = g^k(\Omega)$. This will lead to FAIL.

The right alternative is to remove X_2, giving $f(x) = fYf(x')$, $X_1(z) = g(X_1(a))$, $Y(z') = g(Y(a))$, which is exactly case 2) of the overlapping.

This shows that the system R_2 is locally confluent.

This application to term rewriting systems has its problems, since we have to define reduction. The computation of a reduction step may require to solve second order unification problems, which may be undecidable in the most general case.

Comparison

It is not easy to compare the expressivity of the different versions of stratified systems with existing approaches. If compared with respect to the possibilitiy to express infinite sets of terms in a schematic way, it appears to be impossible to encode terms like $(f^ng)^m$ with usual stratified second order systems. However, this is possible in parametric stratified systems, as well as in dynamically stratified systems.
Most papers on the subject have an explicit representation mechanism for infinite sets of terms ([Gra88], [CH91], [CHK90], [Her92], [KiH91], [Sa92], [So94], [Her92], [Com95]), and permit no second order variables. B. Gramlich ([Gra88]) permits second-order variables and describes restrictions such that a first order unification algorithm can be used. In the work of H. Kirchner [KiH91]), there are meta-variables permitted that are similar to second-order variables, but the meta-variables come in general with a domain of terms that are permitted for instantiation. H. Comon [Com94a] considers completion methods of rewrite systems with membership constraints and has to solve unification problems containing second order variables. The occurrences of these second order variables are restricted to be always of the same shape, i.e. if X(s) and X(t) occur in a system, then s and t must be syntactically equal. Since the constraints in [Com94a] can also express terms like a regular expression (f*g)*, the approach is not directly comparable with the approach in this paper.
W. Farmer [Fa88] has worked on second order unification problems, and has shown that unification of monadic second order terms is decidable. These terms are built from second order variables (arity may be arbitrary) and monadic function symbols. The difference to our problem is that in [Fa88], function symbols of arity ≥ 2 are not permitted, nested applications of second order variables are permitted, and that the semantics of second-order variables is different from ours: a second order variable of arity n may be instantiated with a term function of smaller arity.

The method we will use to solve the second order unification problem has its roots in the paper of W. Farmer [Fa88] in that we make extensive use of parametric terms and parametric words, and in work on unification, where the technique of non-deterministic transformation rules is elaborated (cf. [Sch89, BS93]), which means that the algorithm makes a lot of guessing steps.

References.

BS93 Baader, F, Siekmann, J., Unification Theory, in D.M. Gabbay, C.J. Hogger, J.A: Robinson (eds.), Handbook of Logic in Artifiial Intelligence and Logic Programming, Oxford University Press, (1994)

BHS89 Bürckert, H.-J., Herold, A., Schmidt-Schauß, M., On Equational Theories, Unification and (Un)Decidability, J. Symbolic Computation 8, pp. 3-49, 1989

CH91 Chen, H., Hsiang, J., Logic programming with recurrence domains, ICALP 91, LNCS 510, pp. 20-34, (1991)

CHK90 Chen, H., Hisang, J., Kong, H.C., On finite representation of infinite sequences of terms. Proc. of 2nd CTRS, Springer LNCS 516, pp. 100-114, (1990)

Con93 Contejean, E., Solving *-problems modulo Distributivity by a reduction to AC1-Unification, J. of Symbolic Computation 16, pp. 493-521. (1993)

Com91 Comon. H., Completion of rewrite systems with membership constraints, technical report 699, LRI, Orsay, France, (1991).

Com92 Comon, H., On unification of terms with exponents, Technical report 770, LRI, Orsay, France, (1992). to appear in Mathematical System Theory

Com94a Comon. H., Completion of rewrite systems with membership constraints, part I: Deduction rules and Part II: Constraint Solving, (to appear in J. Symbolic Computation), (1994)

Com94b Comon, H., presentation on the unification workshop , 1994, (Nancy), (1994)

Com95 Comon, H., On unification of terms with integer exponents, Mathematical Systems Theory, 28(1), pp 67-88, (1995)

Fa88 Farmer, W., A unification algorithm for second order monadic terms, Annals of Pure and Applied Logic, 39, pp. 131-174, (1988)

Fa91 Farmer, W., A., Simple second-order languages for which unification is undecidable. Theoretical Computer Science 87, pp. 173-214, (1991)

Gol81 Goldfarb, W. The undecidability of the second-order unification problem, J. TCS 13, pp. 225-230, (1981)

Gra88 Gramlich, B., Unification of term schemes - theory and applications, SEKI report SR-88-18, Universität Kaiserslautern, Germany, (1988)

Her92 Hermann, M. On the relation between primitive recursion, schematization, and divergence, Proc. 3rd Conf. on Algebraic and Logic programming, Volterra (Italy), LNCS 632, pp. 115-127, (1992)

Jaf90 Jaffar, Minimal and complete word unification , J. of the ACM, 37, 47-85, (1990)

KiC89 Kirchner, C. (ed.), Unification, reprint from J. of Symbolic Computation, Academic Press, (1989)

KiH89 Kirchner, H., Schematization of infinite sets of rewrite rules generated by divergent completion processes, J. TCS 67, pp. 303-332, (1989) Ma77 Makanin G.S:, The problem of solvability od equations in a free semigroup. Math. Sbornik, 103, 147-236, (1977), english translation in Math USSR Sbornik 32, (1977)

Sa92 Salzer, G., The unification of infinite sets of terms and its applications, Proc. Logic Programming and Automated Reasoning (LPAR '92) LNAI 624, pp. 409-420, (1992)

Sch89 Schmidt-Schauß, M., Unification in a combination of arbitrary disjoint equational theories, J. Symbolic computation 8, 51-99, (1989)

Sch93 Schulz, K.U., Word unification and transformation of generalized equations, J. Automated Reasoning 11, pp. 149-184, (1993)

Sch94 Schmidt-Schauß, M., An algorithm for distributive unification, internal report 13/94, J.W.Goethe-Universität Frankfurt, (1994)

Sch95 Schmidt-Schauß, M., unification of stratified second-order terms, J.W.Goethe-Universität Frankfurt, (1994), available at http://kiste-5.ki.informatik.uni-frankfurt.de/

Si89 Siekmann, J., Unification theory: a survey, in C. Kirchner (ed.), Special issue on unification, Journal of Symbolic Computation 7, (1989)

So94 Socher-Ambrosius, R., Unification of terms with exponents, internal report, MPI-I-93-217, Max Planck Institut, Saarbrücken, (1994),

Sz82 Szabó, P., Unifikationstheorie erster Ordnung, Dissertation, Karlsruhe, (1982)