Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

Idempotent most general unifiers for infinite sets

W.J. Fokkink

Computer Science/Department of Software Technology

Report CS-R9442 July 1994

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

Idempotent Most General Unifiers for Infinite Sets

WAN FOKKINK
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
e-mail: wan@cwi.nl

Abstract

A standard result from unification theory says that if a finite set E of equations between terms is unifiable, then there exists an idempotent most general unifier for E. In this paper, the theorem is generalized, in first-order logic, to the case where E may be infinite.

AMS Subject Classification (1991): 68N17.
CR Subject Classification (1991): F.4.1.
Key words & Phrases: Unification, most general unifier, infinite sets.

1 Introduction

The unification problem is to determine, given an equation $s = t$ in some logic, whether there exists a substitution σ such that $(s)\sigma = (t)\sigma$. The substitution σ is a 'unifier', and $s = t$ is called 'unifiable'. For an introduction into the field of unification theory, see [1, 6].

A first algorithm, which solves the unification problem in first-order logic, stems from Herbrand's thesis [3] (see [12]). This algorithm was rediscovered by Prawitz [10], and its full significance was recognized only after Robinson [11] had employed it in his resolution principle for automatic theorem-proving. Robinson was the first to define the basic concepts for unification. His algorithm decides whether an equation is unifiable or not, and if so, then it produces a unifier which is idempotent and 'most general', which means that all other unifiers for the equation can be derived from it. More efficient unification algorithms, for finite sets of equations, were proposed by Paterson and Wegman [8] and by Martelli and Montanari [7].

Pietrzykowski [9] defined an algorithm to detect whether or not an equation is unifiable in second-order logic. Huet [4] showed that such an algorithm does not exist in third-order logic. In [5] however, he introduced an algorithm in ω-order logic which, given a unifiable equation, computes a unifier for this equation. For non-unifiable equations, this algorithm may not terminate.

In this paper, we prove that each *infinite* unifiable collection of equations, in first-order logic, allows an idempotent most general unifier. Since the collection

Report CS-R9442
ISSN 0169-118X
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

of equations is infinite, our construction involves limit procedures, which take infinitely many steps.

For an application of this result, in the setting of operational semantics à la Plotkin, see [2]. of this thesis.

Acknowledgements. Special thanks go to Catuscia Palamidessi, who provided fine comments.

2 Preliminaries

In the sequel we assume an alphabet, which consists of the disjoint union of an infinite set of variables and a set of function symbols. Each function symbol f is provided with an arity $ar(f)$, being a natural number ≥ 0. The collection of *terms* over the alphabet is defined inductively as follows:

- each variable is a term,
- if f is a function symbol, and $t_1, ..., t_{ar(f)}$ are terms, then $f(t_1, ..., t_{ar(f)})$ is a term.

The number of function symbols in a term is called the *size* of the term. Syntactic equivalence between terms is denoted by $\_ = \_$.

A *substitution* is a mapping from variables to terms. The notation $\sigma = \tau$ means that $(x)\sigma = (x)\tau$ for all variables x. Each substitution is extended to a mapping from terms to terms in the standard way.

Definition 1 *The* domain *of a substitution σ is the collection of variables x for which $(x)\sigma \neq x$.*

Note that we did not require substitutions to have a finite domain.

Definition 2 *A substitution σ is* idempotent *if $\sigma\sigma = \sigma$.*

In the sequel, E denotes a set of equations $s = t$ between terms.

Definition 3 *A substitution σ is a* unifier *for E if for all $s = t \in E$ we have $(s)\sigma = (t)\sigma$. The set E is called* unifiable *if it allows a unifier.*

A substitution σ is a unifier *for a substitution τ if $\tau\sigma = \sigma$.*

Definition 4 *A unifier Θ for E is called* most general *if for each unifier σ for E there exists a substitution σ' such that $\Theta\sigma' = \sigma$.*

An equation $s = t$ will be called a *proper sub-equation* of equations $C[s] = C[t]$ for non-trivial contexts $C[]$.

3 The main theorem

A standard result from unification theory says that if a *finite* set E of equations between terms is unifiable, then there exists an idempotent most general unifier for E. In this paper, the theorem is generalized to the case where E may be infinite. First, we rephrase the theorem.

Proposition 5 *The following two statements for a substitution Θ are equivalent.*

1. *Θ is an idempotent most general unifier for E.*

2. *Θ is a unifier for E, and each unifier for E is a unifier for Θ.*

Proof. ($\Rightarrow$) Let σ unify E. Since Θ is most general, there is a substitution σ' such that $\Theta\sigma' = \sigma$. Furthermore, Θ is idempotent, so

$$\Theta\sigma = \Theta\Theta\sigma' = \Theta\sigma' = \sigma.$$

($\Leftarrow$) Each unifier σ for E unifies Θ, which means that $\Theta\sigma = \sigma$. So Θ is most general. Furthermore, Θ unifies each unifier for E, so in particular it unifies itself. Hence, $\Theta\Theta = \Theta$. □

Theorem 6 *If E is unifiable, then there exists a unifier Θ for E such that each unifier for E is also a unifier for Θ.*

Proof. Let τ_0 denote the identity substitution, and define $E_0^e = \{e\}$ for each $e \in E$. We define a construction which produces from a substitution τ_{n-1} and unifiable sets of equations E_{n-1}^e, a substitution τ_n and unifiable sets of equations E_n^e.

- If E_{n-1}^e contains an equality $f(s_1, ..., s_{ar(f)}) = g(t_1, ..., t_{ar(g)})$, then it must be the case that $f \equiv g$, because E_{n-1}^e is unifiable. Replace each such equation in E_{n-1}^e by its proper sub-equations $s_i = t_i$ for $i = 1, ..., ar(f)$. Denote the resulting set by F_n^e. Note that a substitution unifies E_{n-1}^e if and only if it unifies F_n^e.

- Suppose that a variable x is not in the domain of τ_{n-1}, and that $\cup_{e \in E} F_n^e$ contains (one or more) equations of the form $x = t$ or $t = x$, where t is not a single variable. Then choose one of these equations $x = t$ or $t = x$, and put $(x)\tau_n = t$. Put $(y)\tau_n = (y)\tau_{n-1}$ for all other variables y. In particular, τ_n equals τ_{n-1} on the domain of τ_{n-1}.

- Put $E_n^e = (F_n^e)\tau_n$.

From the following Property 1, and from the fact that τ_0 and E are unifiable, it follows immediately that all E_n^e are unifiable.

1. Each unifier σ for τ_{n-1} and $\bigcup_{e\in E} E^e_{n-1}$, is also a unifier for τ_n and $\bigcup_{e\in E} E^e_n$.

   Proof. Since σ unifies E^e_{n-1}, it also unifies F^e_n, for each $e \in E$.

   If $(x)\tau_n = (x)\tau_{n-1}$ for a variable x, then $(x)\tau_n\sigma = (x)\tau_{n-1}\sigma = (x)\sigma$, because σ unifies τ_{n-1}. Otherwise, if $(x)\tau_n \neq (x)\tau_{n-1}$, then it follows from the construction of τ_n that $(x)\tau_n = x$ (or its reverse) is in $\bigcup_{e\in E} F^e_n$. Since σ unifies all F^e_n, it follows that $(x)\tau_n\sigma = (x)\sigma$. Hence, σ unifies τ_n.

   $(E^e_n)\sigma = (F^e_n)\tau_n\sigma = (F^e_n)\sigma$, because σ unifies τ_n. Since σ unifies F^e_n, it follows that σ unifies E^e_n, for each $e \in E$.

2. Each unifier σ for τ_n and E^e_n, is also a unifier for τ_{n-1} and E^e_{n-1}.

   Proof. τ_{n-1} equals τ_n on its domain, and σ unifies τ_n, so σ also unifies τ_{n-1}.

   $(F^e_n)\sigma = (F^e_n)\tau_n\sigma = (E^e_n)\sigma$, and σ unifies E^e_n, so σ unifies F^e_n. Then σ also unifies E^e_{n-1}.

Since τ_n equals τ_{n-1} on the domain of τ_{n-1}, we can define the 'union' τ of the substitutions τ_n:

$$(x)\tau = \begin{cases} (x)\tau_n & \text{if } (x)\tau_n \neq x \text{ for some } n, \\ x & \text{otherwise.} \end{cases}$$

3. For each variable x, either $(x)\tau = x$, or $(x)\tau$ is not a variable.

   Proof. If $(x)\tau \neq x$, then $(x)\tau = (x)\tau_n$ for some $n > 0$. Let n be the smallest natural number for which this equality holds, so that $(x)\tau_n \neq (x)\tau_{n-1}$. Then it follows from the construction of τ_n that there is an equation $x = t$ or $t = x$ in $\bigcup_{e\in E} F^e_n$, where t is not a variable, and $(x)\tau_n = t$. Hence, $(x)\tau = t$ is not a variable.

4. For each variable x, there is a natural $M(x)$ such that $(x)\tau^{M(x)+1} = (x)\tau^{M(x)}$.

   Proof. Fix a unifier σ for E. Since σ also unifies the identity τ_0, Property 1 implies that σ is a unifier for all τ_n. So σ is a unifier for their union τ, which means that $(x)\tau^m\sigma = (x)\sigma$ for all m. Thus, the size of the terms $(x)\tau^m$ cannot grow beyond the size of $(x)\sigma$. The term $(x)\tau^{m+1}$ is obtained from $(x)\tau^m$ by application of τ, so the size of $(x)\tau^{m+1}$ is at least the size of $(x)\tau^m$. Hence, there is a natural $M(x)$ such that for $m \geq M(x)$, all terms $(x)\tau^m$ have the same size. Then Property 3 of τ implies $(x)\tau^{m+1} = (x)\tau^m$ for $m \geq M(x)$.

We define the 'limit' $\bar{\tau}$ of τ by

$$(x)\bar{\tau} = (x)\tau^{M(x)}.$$

Property 4 implies that $\tau\bar{\tau} = \bar{\tau}$. So, since τ is the union of all τ_n, we have $\tau_n\bar{\tau} = \bar{\tau}$ for all n.

5. For each $e \in E$, there is a natural $N(e)$ such that $E^e_{N(e)}$ contains only equations of the form $x = y$, where x and y are not in the domain of $\bar{\tau}$.

   Proof. Fix an $e \in E$, and consider the sequence $\{(E^e_n)\bar{\tau}\}_{n=0}^{\infty}$.

   Each equation in E^e_{n-1} is either maintained, or replaced by proper sub-equations in F^e_n. Hence, each equation in $(E^e_{n-1})\bar{\tau}$ is either maintained, or replaced by proper sub-equations in $(F^e_n)\bar{\tau} = (F^e_n)\tau_n\bar{\tau} = (E^e_n)\bar{\tau}$. Since proper sub-equations always have a size smaller than the original equation, each chain of subsequent proper sub-equations in the subsequent $(E^e_n)\bar{\tau}$ is finite. So, by König's Lemma, there is some $N(e)$ such that all the equations in $(E^e_{n-1})\bar{\tau}$ are maintained in $(E^e_n)\bar{\tau}$ for each $n > N(e)$.

   Consider an equation $s = t$ in E^e_{n-1} for some $n > N(e)$. Since $(s = t)\bar{\tau} \in (E^e_{n-1})\bar{\tau}$ is maintained in $(E^e_n)\bar{\tau}$, it follows that $s = t \in E^e_{n-1}$ is maintained in F^e_n. So $s = t$ cannot have any proper sub-equations, or in other words, s or t must be a variable, say, $s = x$.

   Now suppose that t is *not* a variable. We deduce a contradiction. First, we show that then $(x)\tau_n$ is not a variable. Distinguish two cases.

   - $(x)\tau_{n-1} \neq x$. Since τ_n and τ_{n-1} coincide on the domain of τ_{n-1}, we have $(x)\tau_n = (x)\tau_{n-1} \neq x$. Then Property 3 yields that $(x)\tau_n$ is not a variable.
   - $(x)\tau_{n-1} = x$. Then x is not in the domain of τ_{n-1}. Furthermore, $x = t \in F^e_n$ where t is not a variable. So from the construction of τ_n we see that $(x)\tau_n$ is not a variable.

   The equation $x = t \in F^e_n$ takes the form $(x = t)\tau_n$ in E^e_n. Since $(x)\tau_n$ and $(t)\tau_n$ are not variables, this equation is replaced by proper sub-equations in F^e_{n+1}. But this contradicts the fact that equations in $(E^e_n)\bar{\tau}$ are maintained in $(E^e_{n+1})\bar{\tau}$. So apparently, t must be a variable.

   Thus, each equation in E^e_{n-1} for $n > N(e)$ is of the form $x = y$. In E^e_n, such an equation takes the form $(x = y)\tau_n$, so $(x)\tau_n$ and $(y)\tau_n$ are variables too. Then Property 3 yields $(x)\tau_n = x$ and $(y)\tau_n = y$. Hence, x and y are not in the domain of τ_n for any $n > N(e)$, so they are not in the domain of their union τ. Then x and y are not in the domain of $\bar{\tau}$.

We define the 'limit' $\bar{E}$ of E by

$$\bar{E} = \bigcup_{e \in E} E^e_{N(e)}.$$

Construct a unifier ρ for $\bar{E}$ as follows. Two variables are said to be 'equivalent' if they can be equated by equations in $\bar{E}$. We define ρ to contract the elements of each equivalence class C to one variable in this class. That is, just pick some $x_0 \in C$, and put $(x)\rho = x_0$ for $x \in C$.

Finally, we define the desired unifier Θ for E such that each unifier for E is also a unifier for Θ:

$$\Theta = \bar{\tau}\rho.$$

6. Θ is a unifier for E.

   Proof. Since $\tau_n\bar{\tau} = \bar{\tau}$, also $\tau_n\Theta = \tau_n\bar{\tau}\rho = \bar{\tau}\rho = \Theta$ for all n. In other words, Θ unifies τ_n for all n.

   Consider an equation $x = y \in \bar{E}$. Property 5 ensures that x and y are not in the domain of $\bar{\tau}$, so $(x = y)\Theta = (x = y)\bar{\tau}\rho = (x = y)\rho$. Since x and y can be equated in $\bar{E}$, ρ maps x and y to the same variable. So indeed this last equality holds, and thus $(x = y)\Theta$ holds. So Θ unifies $\bar{E}$.

   Since Θ unifies all τ_n and $\bar{E}$, in particular it unifies $\tau_{N(e)}$ and $E^{e}_{N(e)}$ for each $e \in E$. Then Property 2 yields that Θ unifies $E^{e}_{0} = \{e\}$, for each $e \in E$.

7. Each unifier for E is a unifier for Θ.

   Proof. Let σ unify E. Then according to Property 1, σ unifies τ_n and E^{e}_{n} for all naturals n and $e \in E$.

   Since σ unifies each τ_n, it unifies their union τ. And $(x)\bar{\tau} = (x)\tau^{M(x)}$, so then σ unifies $\bar{\tau}$.

   Since σ unifies all $E^{e}_{N(e)}$, it unifies their union $\bar{E}$. By definition of ρ, $(x)\rho$ and x can be equated in $\bar{E}$ for each x. Hence, $(x)\rho\sigma = (x)\sigma$ for each x.

   So, $\Theta\sigma = \bar{\tau}\rho\sigma = \bar{\tau}\sigma = \sigma$. □

References

[1] K.R. Apt. Logic programming. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science,* Volume B, *Formal Methods and Semantics*, pages 493–574. Elsevier, 1990.

[2] W.J. Fokkink. The tyft/tyxt format reduces to tree rules. In M. Hagiya and J.C. Mitchell, editors, *Proceedings TACS'94,* Sendai, Japan, *LNCS 789*, pages 440–453. Springer-Verlag, 1994.

[3] J. Herbrand. *Recherches sur la théorie de la démonstration.* PhD thesis, Université de Paris, 1930.

[4] G.P. Huet. The undecidability of unification in third order logic. *Information and Control,* 22(3):257–267, 1973.

[5] G.P. Huet. A unification algorithm for typed λ-calculus. *Theoretical Computer Science,* 1:27–57, 1975.

[6] Z. Manna and R. Waldinger. *The Logical Basis for Computer Programming. Volume II: Deductive Systems.* Addison-Wesley, 1990.

[7] A. Martelli and U. Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, 4(2):258–282, 1982.

[8] M. Paterson and M. Wegman. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167, 1978.

[9] T. Pietrzykowski. A complete mechanization of second-order type theory. *Journal of the ACM*, 20(2):333–364, 1973.

[10] D. Prawitz. An improved proof procedure. *Theoria*, 26:102–139, 1960.

[11] J.A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[12] J. van Heijenoort, editor. *Jacques Herbrand: Écrits Logiques.* Presses Universitaires de France, 1968. English translation: W.D. Goldfarb, editor. *Jacques Herbrand: Logical Writings.* Reidel, 1971.