On the Unification Problem for Cartesian Closed Categories

(Extended Abstract)

Paliath Narendran
Institute of Programming and Logics
Department of Computer Science
State University of NY at Albany
Albany, NY 12222
dran@cs.albany.edu

Frank Pfenning
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
fp@cs.cmu.edu

Richard Statman
Department of Mathematics
Carnegie Mellon University
Pittsburgh, PA 15213
rstatman@cs.cmu.edu

Abstract

Cartesian closed categories (CCC's) have played and continue to play an important role in the study of the semantics of programming languages. An axiomatization of the isomorphisms which hold in all Cartesian closed categories discovered independently by Soloviev and Bruce and Longo leads to seven equalities. We show that the unification problem for this theory is undecidable, thus settling an open question. We also show that an important subcase, namely unification modulo the linear isomorphisms*, is NP-complete. Furthermore, the problem of matching in CCC's is NP-complete when the subject term is irreducible.*

CCC-matching and unification form the basis for an elegant and practical solution to the problem of retrieving functions from a library indexed by types investigated by Rittri. It also has potential applications to the problem of polymorphic higher-order unification, which in turn is relevant to theorem proving, logic programming, and type reconstruction in higher-order languages.

1 Introduction

Cartesian closed categories (CCC's) have played and continue to play an important role in the study of the semantics of programming languages. Much work has been done in the study of solutions to particular domain equations such as Scott's construction of a CCC and a domain D to satisfy the isomorphism $D \cong D \Rightarrow D$ to provide a model for the untyped lambda calculus. Surprisingly little is known about which domain equations have solutions in *all* CCC's. An axiomatization of the isomorphisms due to Soloviev [Sol83] and independently discovered by Bruce and Longo [BL85] leads to a first-order equational unification problem under seven equalities. We show that the unification problem for this theory is undecidable by a reduction from Hilbert's tenth problem, thus settling an open question. The corresponding *CCC-matching* problem is NP-complete, when the subject is irreducible. The isomorphisms valid in all closed categories (CC's) are characterized by five of the seven equations as shown by Soloviev [Sol93] (exactly those which can be realized by linear functions). We show that the unification problem thus restricted is NP-complete.

Matching and unification in closed and Cartesian closed categories have applications to the retrieveal of functions in a library indexed by the function's type (see [Rit90, Rit91, Rit92, RT91]). The equational theory of the types comes into play here, since library entries of essentially the same functionality may have many different, but isomorphic types. Retrieval which is indexed by types should abstract away from whether a given function is curried or not, or in which order it accepts its arguments. Rittri further argues convincingly that such retrieval is more precise and intuitive if one considers only linear isomorphisms, which leads to the second problem we consider.

An early version of the NP-completeness of CCC-matching was reported in [NPS89], and a variant of our algorithm for CC-unification has been implemented by Rittri and turned out to be practical in library retrieval applications. The undecidability result is new, and none of the results have previously appeared in print, as far as we know.

Our original motivation for the study of CCC-unification was a potential application to the problem of *polymorphic unification*, that is, unification in the simply typed λ-calculus with products and polymorphic types (in the style of ML). Intuitively the connection can be seen when considering the solutions to a

polymorphic unification problem with a free type variable D: for a solution at type A, there are guaranteed to be solutions at all types isomorphic to A. Since various occurrences of D may place different constraints on its instances, these constraints must be unified using CCC-unification. Polymorphic matching and unification have applications in theorem proving, logic programming, and program transformation in higher-order logic.

2 Definitions

2.1 Equational Unification

The reader is referred to [DJ90, HO80] for extensive surveys of the important concepts of term rewriting systems such as *reduction, confluence,* and *canonicity* (or *convergence*).

Two terms s and t are said to be *unifiable modulo an equational theory* E if and only if there exists a substitution θ such that $\theta(s) =_E \theta(t)$. If E has a canonical term rewriting system R, i.e., there exists a canonical system R whose associated equational theory is E, then this is equivalent to having a θ such that $\theta(s)$ and $\theta(t)$ are joinable modulo R. In this case we sometimes say *s and t are unifiable modulo R.* The *unification problem* modulo an equational theory E is to determine, given a list of pairs $\langle s_1, t_1\rangle, \ldots, \langle s_n, t_n\rangle$, whether there exists σ such that $\sigma(s_1) =_E \sigma(t_1), \ldots, \sigma(s_n) =_E \sigma(t_n)$, where the terms ($s_i$'s and t_i's) consist of function symbols occurring in the signature of E and free constants[1]. Often we distinguish this problem from the *single-pair unification problem* where only one pair is input.

A term t is *E-reducible* modulo R if and only if there is a rule $l \to r$ in R, a subterm t' at occurrence a of t, and a substitution σ such that $\sigma(l) =_E t'$. The term $t[a \leftarrow \sigma(r)]$ is the result of *E-reducing* t by $l \to r$ at a. Many of the notions such as *noetherian, confluent* etc. can be extended to *E-noetherian, E-confluent* etc.; we do not include those definitions here, instead referring the reader to [JK86, PS81].

A substitution σ is said to be *irreducible modulo a term rewriting system* R if and only if $\sigma(x)$ is irreducible for every x in the support of σ. Clearly one needs to look only for irreducible substitutions if the unification problem is modulo a canonical system.

A term rewriting system R is said to be *confluent over a set of terms* S if and only if every term in S has a unique normal form modulo R. It is said to be *canonical over* S if it is noetherian as well.

[1] This is often called *elementary unification* to distinguish it from the *general* case where we allow, in the input, non-constant function symbols that are not in the signature of E.

2.2 Cartesian Closed Categories

A Cartesian closed category (CCC) is a category that has products ($\times$), exponentiation ($\Rightarrow$) and a terminal object (1). (See [LS86] for a detailed treatment.) The isomorphism relation between objects in a CCC ($\cong$) can be axiomatized equationally (as shown by Bruce and Longo [BL85] and later sharpened by Bruce, Di Cosmo, and Longo [BCL92]) and this results in an algebra consisting of binary operators $\times$ and $\Rightarrow$ and nullary (constant) operator 1, and the following equational axioms:

- $\times$ is associative and commutative, i.e.,

$$\begin{aligned} x \times (y \times z) &= (x \times y) \times z \\ x \times y &= y \times x \end{aligned}$$

- there is an (identity) element 1 for $\times$ with the properties

$$\begin{aligned} x \times 1 &= x \\ 1 \Rightarrow x &= x \\ x \Rightarrow 1 &= 1 \end{aligned}$$

- $\Rightarrow$ left-distributes over $\times$ (Dist-2):

$$x \Rightarrow (y \times z) = (x \Rightarrow y) \times (x \Rightarrow z)$$

- Currying of $\Rightarrow$ (Cur-2):

$$x \Rightarrow (y \Rightarrow z) = (x \times y) \Rightarrow z$$

Let $\mathcal{E}$ denote this set of equations. An isomorphism holds between expressions e_1 and e_2 (which denote objects) in every CCC if and only if $e_1 =_{\mathcal{E}} e_2$.

3 Results on Term Rewriting Systems

A rule $l \to r$ in a term rewriting system $\mathcal{T}$ is *optimally reducing* if and only if it satisfies the following property:

> For every substitution θ, if $\theta(s)$ is irreducible for every proper subterm s of l, then $\theta(r)$ is irreducible too.

A term rewriting system $\mathcal{T}$ is *optimally reducing* if and only if every rule in it is optimally reducing. The significance of optimal reduction lies in the following three lemmas. We use $|t|$ to denote the size of a term t.

Lemma 1 *It is decidable whether a term rewriting system is optimally reducing.*

Lemma 2 *Let $\mathcal{T}$ be an optimally reducing term rewriting system, s be an irreducible term, and θ be an irreducible substitution. Then $\theta(s)$ can be reduced to its normal form in less than or equal to $|s|$ steps.*

Proof-sketch: It can be shown that doing reduction innermost (i.e., from the leaves upward — see [Klo87]) would involve at the most $|s|$ steps. □

Lemma 3 *Every confluent, optimally reducing term rewriting system has a decidable unification problem.*

Proof-sketch: Let $\mathcal{T}$ be an optimally reducing canonical term rewriting system and s and t be terms to be unified modulo $\mathcal{T}$. By Lemma 2, if θ is an irreducible substitution that unifies these two terms, then $\theta(s)$ and $\theta(t)$ can be reduced to their normal forms in at most $|s|$ and $|t|$ steps respectively using innermost reduction. Non-deterministically choose the sequence of positions where the reductions occur and the corresponding rules that are applied. Mimic the reductions without actually performing unification by suitably renaming the variables in the rules. For instance, if $\theta(s)$ is to be reduced at $p \in Occ(s)$ by $l \rightarrow r$, then form the (standard) unification problem $\langle s/p = \eta(l)\rangle$ for a suitable renaming η of l, and continue in the same way with the new term $s[p \leftarrow \eta(r)]$. This procedure gives rise to at most $|s| + |t| + 1$ equations to be unified. Finally, once all the simultaneous unification problems are found, we can easily check, using standard unification, whether our sequence of choices is valid. □

Note that an application of Lemma 3 is in *idempotence unification*, i.e., unification where one or more function symbols f can have the idempotence property $f(x, x) = x$.

Example: Consider the term rewriting system $\{f(x, x) \rightarrow g(x)\}$ which is optimally reducing and canonical. The terms $s = f(f(x, h(y)), g(x))$ and $f(g(h(v)), f(u, z)) = t$ are unifiable by the irreducible substitution $\theta = \{x \leftarrow h(y), v \leftarrow y, u \leftarrow h(y), z \leftarrow h(y)\}$. Now $\theta(s)$ is first reduced at the position 1 and then at Λ, to get $g(g(h(y)))$, and $\theta(t)$ is reduced at 2 and then at Λ to get $g(g(h(y)))$. Mimicking these, we get the pairs $\langle f(x, h(y)) = f(x_1, x_1)\rangle$, $\langle f(g(x_1), g(x)) = f(x_2, x_2)\rangle$, $\langle f(u, z) = f(x_3, x_3)\rangle$, $\langle f(g(h(v)), g(x_3)) = f(x_4, x_4)\rangle$ and finally $\langle g(x_2) = g(x_4)\rangle$. Simultaneously unifying these pairs yields the equational unifier. □

A function f is said to be an *associative-commutative operator* (*AC-operator*) if and only if it satisfies the associative and commutative laws. An operator that is neither associative nor commutative will be referred to in this paper as a *non-AC operator*. Unification modulo associativity and commutativity (generally known as AC-unification) has been investigated widely in the term rewriting systems literature — see [Fag87, KN99, Sti81], for instance. In particular, it has been proved in [KN99] that unifiability modulo associativity and commutativity can be checked in nondeterministic polynomial time (**NP**); we will make use of this result later in the paper.

For systems involving associative-commutative operators, Lemma 2 can be extended somewhat.

Lemma 4 *Let $\mathcal{T}$ be an optimally AC-reducing term rewriting system involving AC-operators such that the root symbols of its left-hand sides are non-AC operators. Let s be an AC-irreducible term, and θ be an AC-irreducible substitution. Then $\theta(s)$ can be AC-reduced to its normal form modulo $\mathcal{T}$ in less than or equal to $|s|$ steps.*

We believe Lemma 4 can be improved upon; we do not attempt to do so here, since Lemma 4 covers the cases that we are interested in. We obtain as a corollary to this lemma that every confluent, optimally AC-reducing term rewriting system has a decidable unification problem.

4 Unification in Cartesian Closed Categories

Let $\mathcal{E}'$ be the equational theory consisting of all the equations in $\mathcal{E}$ except the left-distributivity axiom (Dist-2) and the equation $(x \Rightarrow 1) = 1$. As shown by Soloviev [Sol93], $\mathcal{E}'$ is the set of equations that characterizes the isomorphisms valid in all *closed categories* (CC's). Our main results are as follows:

1. Both theories have canonical rewriting systems modulo associativity and commutativity (of $\times$) which can be obtained by orienting the remaining equations left to right.
2. Unification modulo $\mathcal{E}'$ is NP-complete.
3. Matching modulo $\mathcal{E}$ is NP-complete.
4. Unification modulo $\mathcal{E}$ is undecidable.

Let $\mathcal{R}$ and $\mathcal{R}'$ stand for the canonical systems for $\mathcal{E}$ and $\mathcal{E}'$ respectively. It can be easily shown that the unification problem, for both $\mathcal{E}$ and $\mathcal{E}'$, can be reduced to the single-pair unification problem for these theories. Hence, throughout the rest of the paper, by "unification" we mean single-pair unification. Also, since the rest of the paper is concerned with theories involving AC-operators, we often do not prefix all relevant concepts with "AC-" as in AC-reduction, AC-confluent etc.

4.1 Unification modulo $\mathcal{E}'$

As mentioned earlier, obtaining a canonical system, modulo associativity and commutativity, for $\mathcal{E}'$ is not difficult at all. We merely orient the equations left-to-right. Thus the rules in $\mathcal{R}'$ are

$$\begin{array}{rcl} x \times 1 & \rightarrow & x \\ (1 \Rightarrow x) & \rightarrow & x \\ (x \Rightarrow (y \Rightarrow z)) & \rightarrow & ((x \times y) \Rightarrow z) \end{array}$$

Let $\mathcal{R}''$ refer to the set consisting of the last two rules, i.e.,

$$\mathcal{R}'' = \mathcal{R}' \setminus \{ (x \times 1) \rightarrow x \}$$

Lemma 5 *$\mathcal{R}'$ is canonical.*

Let $\mathcal{S}$ stand for the set of terms that are irreducible wrt the rules $x \times 1 \rightarrow x$ and $(1 \Rightarrow x) \rightarrow x$.

Lemma 6 *$\mathcal{R}''$ is optimally reducing and canonical over $\mathcal{S}$.*

We are now in a position to outline our **NP**-algorithm for unification modulo $\mathcal{R}'$. Let s and t be two terms to be unified. We can non-deterministically choose the variables that are to be substituted for by 1; let s' and t' be the normal forms of the terms thus obtained. Assume neither s' nor t' is 1. (The case where one of them is 1 is very easily handled.) Thus if θ is the "remaining" part of our target substitution, then $1 \notin \theta(x)$ for all x in its support (domain). Hence $\theta(s')$ and $\theta(t')$ must also be in normal form wrt the rule $x \times 1 \rightarrow x$. By Lemma 6 and Lemma 4, $\theta(s')$ and $\theta(t')$ can be reduced to their respective normal forms using innermost reduction in at the most $|s'|$ and $|t'|$ AC-reduction steps respectively. Therefore all that we have to do is to "mimic" those reductions non-deterministically, as in the proof of Lemma 3, and verify that the reduction sequence is indeed feasible. Since AC-unifiability can be checked in **NP**, the whole algorithm can be done in **NP**. (We would also like to mention here that this technique also leads to an algorithm to compute a finite, complete set of $\mathcal{E}'$-unifiers.)

NP-hardness of the problem can be shown by reduction from one-in-three 3SAT with positive literals. (Since the construction is the same as that given in the next section for $\mathcal{E}$, we omit it here.) Thus,

Theorem 1 *Unification modulo $\mathcal{E}'$ is NP-complete.*

It can also be shown, in a similar way, that the unification problem modulo $\{\text{Cur-}2 + A + C\}$ is in **NP** since $\{\text{Cur-}2\}$ is also optimally AC-reducing. (Lemma 4 applies here as well.)

4.2 Matching modulo $\mathcal{E}$

We show that given a term s and an $\mathcal{R}$-irreducible term t, the problem of checking whether s matches t modulo $\mathcal{E}$ is NP-complete. The NP-hardness proof uses a reduction from one-in-three 3SAT with positive literals. Let $C_1, \ldots, C_m$ be clauses consisting only of positive literals over the propositional variables $x_1, \ldots, x_n$. The task is to find a satisfying assignment such that one and only one literal from each clause becomes true. Let $c_1, \ldots, c_m$ be distinct free constants, one for each clause. For each clause C_j we form the term $\tau(C_j) = (y_1 \times y_2 \times y_3) \Rightarrow c_j$ where y_1, y_2, y_3 are the variables in the clause. Now the satisfiability problem is equivalent to the problem of matching $\tau(C_1) \times \ldots \times \tau(C_m)$ with $(a \Rightarrow c_1) \times \ldots \times (a \Rightarrow c_m)$, where a is a free constant. Here we interpret substitution of a as 'true' and substitution of 1 as 'false.'

To show that the matching problem is in **NP**, we only have to show that the matching problem modulo the last two axioms, namely

$$\begin{array}{rcl} x \Rightarrow (y \Rightarrow z) & = & (x \times y) \Rightarrow z \qquad \text{and} \\ x \Rightarrow (y \times z) & = & (x \Rightarrow y) \times (x \Rightarrow z), \end{array}$$

and associativity and commutativity of $\times$, is in **NP**. To begin with, it can be shown that these rules form an AC-convergent term rewriting system for the equational theory. Let us denote this rewrite system by $\mathcal{R}'''$. Note that the latter rule (Dist-2) is the only rule that causes an increase in the size of a term; if t is rewritten to t' using this rule (modulo AC) then the size of t' must be at least $|t| + 2$. Therefore if $s \rightarrow^*_{\mathcal{R}''', AC} s'$ and s' is in normal form, then the number of rewrites using (Dist-2) cannot be greater than $\lfloor \frac{|s'|-|s|}{2} \rfloor$.

Thus if s is any term and t is an irreducible ground term, then s is matchable with t by θ if and only if there exist terms $t_1, t'_1, \ldots, t_K, t'_K$, $K \leq$

$\lfloor \frac{|s'|-|s|}{2} \rfloor$, such that $\theta(s) \rightarrow^*_{\text{Cur-2},AC} t_1$, $t_1 \rightarrow_{\text{Dist-2},AC} t'_1$, $t'_1 \rightarrow^*_{\text{Cur-2},AC} t_2$, $\ldots$, $t'_K \rightarrow^*_{\text{Cur-2},AC} t$. Since each of these terms can be guessed in **NP** time, and the matching problem modulo $\{\text{Cur-2} + A + C\}$ is in **NP**, the result follows.

Theorem 2 *Matching a term with an irreducible term modulo $\mathcal{E}$ is* **NP***-complete.*

4.3 Unification modulo $\mathcal{E}$

In this section, we outline our undecidability proof of CCC-unification. The reduction is from Hilbert's 10th problem which is on checking whether a polynomial Diophantine equation with integer coefficients has a solution in integers. For the sake of clarity, we look at the equational theory $\mathcal{E}$ a little differently — namely, $\times$ as $*$ (multiplication) and $\Rightarrow$ as $\uparrow$ (exponentiation). We also use $+$ on natural numbers as an abbreviation; e.g., a^{x+y} is an abbreviation for $a^x * a^y$, and $2x$ is a (further) abbreviation of $x + x$. Thus for any polynomial p with positive integral coefficients, x^p and a^p are valid terms in our algebra, where x is a variable and a a free constant. Thus a term of the form b^i, where i is a natural number, is an abbreviation for $\underbrace{b \times b \times \ldots \times b}_{i}$ and b^0 stands for 1.

To begin with, we note two cancellation properties of $\mathcal{E}$:

(C1) $a^s =_{\mathcal{E}} a^t$ implies $s =_{\mathcal{E}} t$ for any free constant a.

(C2) $r * s =_{\mathcal{E}} r * t$ implies $s =_{\mathcal{E}} t$.

Lemma 7 *Every solution to the unification problem*

$$x * a^y \quad = \quad x^b * a$$

where a and b are free constants, is of the form $y \leftarrow b^n, x \leftarrow a^{b^{n-1}+b^{n-2}+\cdots+b+1}$ where n is a natural number.

Proof: It is easier to prove a slightly more generalized version of this lemma, namely,

Claim: For all i, the unification problem

$$x * a^y \quad = \quad x^b * a^{b^i}$$

is solvable iff $y = b^m$ for some $m \geq i$. It is clear that for every $n \geq i$, $\{y \leftarrow b^n, x \leftarrow a^{b^{n-1}+b^{n-2}+\cdots+b^{i+1}+b^i}\}$ is indeed a solution. We prove the converse by contradiction. Let θ be a unifier in which $\theta(y) \neq b^n$ for any n. Let $\theta(y) = t_y$, where t_y is the smallest irreducible ground term (by the subterm ordering) for which the counterexample will work. Then $\theta(x)$ cannot be 1. Let $\theta(x) = a^{t_1} * \cdots * a^{t_k}$. Thus $a^{t_1} * \cdots * a^{t_k} * a^{t_y} =_{\mathcal{E}} a^{b*t_1} * \cdots * a^{b*t_k} * a^{b^i}$. Note that the lhs and the rhs of this equality are already in normal form, unless one of the t_i's, $1 \leq i \leq k$, is 1. Since t_y cannot be equal to b^i, we can assume, without loss of generality, that $t_y = b * t_k$. Cancelling on both sides, we get

$$a^{t_1} * \cdots * a^{t_{k-1}} * a^{t_k} = a^{b*t_1} * \cdots * a^{b*t_{k-1}} * a^{b^i}.$$

Note that t_k cannot be of the form b^j for any j. By examining the identity above, it can be seen that the substitution θ', where $\theta'(y) = t_k$ and $\theta'(x) = a^{t_1} * \cdots * a^{t_{k-1}}$, unifies the terms too, and t_k is strictly smaller than t_y. □

Thus the above equation can be used to force a variable to have a solution of the form c^j, where c is any free constant and j is a natural number.

Lemma 8 *Let b and c be free constants and j be a natural number. Then the equations*

$$\begin{aligned} x^c * a^{b^j} &= x^b * a^u \\ z * a^u &= z^c * a \end{aligned}$$

force u to be equal to c^j. (In other words, $x^c * a^{b^j} = x^b * a^{c^k}$ is unifiable if and only if $j = k$.)

Lemma 9 *Let b and c be free constants and j and k be natural numbers. Then the equations*

$$\begin{aligned} x^{c^k} * a^{b^j} &= x^b * a^u \\ z * a^u &= z^c * a \end{aligned}$$

force u to be equal to c^{j*k}.

This shows that multiplication of natural numbers can be simulated. Consider, for instance, the equation $z = x * y$. If $x = b^i$ and $y = b^j$, then we can force z to be b^{i*j} in the following way:

(i) Copy x to x' (its 'alter ego') changing b's to c's; i.e., $x' = c^i$. This can be done using equations as given in the statement of Lemma 8.

(ii) Multiply x' and y to get $z' = c^{i*j}$.

(iii) Copy z' to z changing c's to b's.

Thus the equations we get are

$$\begin{aligned} w_1 * a^{x'} &= {w_1}^c * a \\ {w_2}^c * a^x &= {w_2}^b * a^{x'} \\ {w_3}^{x'} * a^y &= {w_3}^b * a^{z'} \\ w_4 * a^{z'} &= {w_4}^c * a \\ {w_5}^c * a^z &= {w_5}^b * a^{z'} \end{aligned}$$

Simulating addition is easy, since if $x = b^i$ and $y = b^j$, then $(a^x)^y = a^{b^{i+j}}$.

Theorem 3 *Unification modulo $\mathcal{E}$ is undecidable.*

Acknowledgements: We wish to thank Mikael Rittri and Michael Rusinowitch for their comments and suggestions.

References

[BCL92] K. B. Bruce, R. Di Cosmo, and G. Longo. Provable isomorphisms of types. *Mathematical Structures in Computer Science*, 2(2):231–247, 1992.

[BL85] Kim B. Bruce and Giuseppe Longo. Provable isomorphisms and domain equations in models of typed languages. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 263–272. ACM Press, 1985.

[DJ90] N. Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, pages 243–320. Elsevier, Amsterdam, 1990.

[Fag87] François Fages. Associative-commutative unification. *Journal of Symbolic Computation*, 3:257–275, 1987.

[HO80] Gérard Huet and Derek C. Oppen. Equations and rewrite rules: A survey. In R. Book, editor, *Formal Languages: Perspectives and Open Problems*, pages 349–405, New York, 1980. Academic Press.

[JK86] Jean-Pierre Jouannaud and Claude Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, 15(4):1155–1194, November 1986.

[Klo87] J. W. Klop. Term rewriting systems: A tutorial. *Bulletin of the EATCS*, 32:143–182, June 1987.

[KN99] Deepak Kapur and Paliath Narendran. Complexity of unification problems with associative-commutative operators. *Journal of Automated Reasoning*, To appear, 199?

[LS86] J. Lambek and P. J. Scott. *Introduction to Higher Order Categorical Logic*. Cambridge University Press, Cambridge, England, 1986.

[NPS89] Paliath Narendran, Frank Pfenning, and Richard Statman. On the unification problem for Cartesian closed categories. Talk presented at the Workshop on Higher-Order Logic, Banff, Alberta, September 1989.

[PS81] G. E. Peterson and M. E. Stickel. Complete sets of reductions for some equational theories. *Journal of the ACM*, 28(3):233–264, 1981.

[Rit90] Mikael Rittri. Retrieving library identifiers via equational matching of types. In M. E. Stickel, editor, *Proceedings of the 10th International Conference on Automated Deduction*, pages 603–617, Kaiserslautern, Germany, July 1990. Springer-Verlag LNAI 449.

[Rit91] Mikael Rittri. Using types as search keys in function libraries. *Journal of Functional Programming*, 1(1):71–89, 1991. A preliminary version appeared in the proceedings of FPCA'89, pages 174–183.

[Rit92] Mikael Rittri. Retrieving library functions by unifying types modulo linear isomorphism. Programming Methodology Group Report 66, Chalmers University of Technology and University of Göteborg, Göteborg, Sweden, May 1992.

[RT91] Colin Runciman and Ian Toyn. Retrieving re-usable software components by polymorphic type. *Journal of Functional Programming*, 1(2):191–211, 1991. A preliminary version appeared in the proceedings of FPCA'89, pages 166–173.

[Sol83] Sergei V. Soloviev. The category of finite sets and Cartesian closed categories. *Soviet Mathematics*, 22(3):1387–1400, 1983.

[Sol93] Sergei V. Soloviev. The ordinary identities form a complete axiom system for isomorphism of types in closed categories. In

A. Voronkov, editor, *Proceedings of the International Conference on Logic Programming and Automated Reasoning, LPAR'93*, St. Petersburg, Russia, July 1993. Springer-Verlag LNAI. To appear.

[Sti81] Mark Stickel. A unification algorithm for associative-commutative functions. *Journal of the ACM*, 28(3):423–434, July 1981.