E-Unification for Subsystems of $S4$

Renate A. Schmidt

MPI–I–98–2-003 April 1998

Author's Address

Department of Computing and Mathematics,
Manchester Metropolitan University,
Chester Street, Manchester M1 5GD, United Kingdom.
URL: `http://www.doc.mmu.ac.uk/STAFF/R.Schmidt/`
E-mail: `R.A.Schmidt@doc.mmu.ac.uk`

Publication Notes

The short version of this paper is published in Nipkow, T. (ed.) (1998), *Rewriting Techniques and Applications: 9th International Conference, RTA'98, Tsukuba, Japan, Proceedings*, Vol. 1379 of *Lecture Notes in Computer Science*, Springer, 106–120 (1998).

Acknowledgements

I thank H. Ganzinger, A. Herzig, U. Hustadt, A. Nonnengart, H. J. Ohlbach, M. Veanes and C. Weidenbach as well as the five anonymous referees of the RTA'98 paper for their most helpful comments. This research was conducted while I was employed at the Max-Planck-Institut für Informatik, and was funded by the MPG as well as the DFG through the TraLos-Project.

Abstract

This paper is concerned with the unification problem in the path logics associated by the optimised functional translation method with the propositional modal logics *K*, *KD*, *KT*, *KD4*, *S4* and *S5*. It presents improved unification algorithms for certain forms of the right identity and associativity laws. The algorithms employ mutation rules, which have the advantage that terms are worked off from the outside inward, making paramodulating into terms superfluous.

Keywords

Unification, modal logic, theorem proving.

1 Introduction

An area of application for unification theory which has not been explored much is modal logic. Modal inference can be facilitated by theory resolution via the so-called functional translation or its variation for propositional modal logics, the optimised functional translation approach. The functional translation method was proposed independently in the late eighties by a number of groups. Fariñas del Cerro and Herzig (1989, 1995) describe a transformation of quantified modal logics into so-called deterministic logics and use a modal resolution calculus. Ohlbach (1988, 1991) and Auffray and Enjalbert (1992) embed quantified modal logics into fragments of first-order logic and employ first-order resolution theorem proving. Zamov (1989) describes a lock decision procedure for the translation of *S4*. All procedures involve theory unification. The *optimised functional translation* method (Herzig 1989, Ohlbach and Schmidt 1997) applies to propositional normal modal logics and gives rise to a class of *path logics*, which this paper considers. Very much like modal logics, path logics form a lattice with the weakest being the *basic path logic* associated with the basic modal logic K and also KD. Different path logics are distinguished by different theories involving equations. This paper focuses on a subclass of path logics with theories consisting exclusively of equations. Path logics with equational theories are associated with serial modal logics, which are modal logics stronger than KD. Clauses in path logics satisfy two important properties. One, they satisfy *prefix stability* which determines a certain ordering of the variables, and two, all Skolem terms in input clauses are constants.

The purpose of this paper is to give a formal treatment of unification and normalisation for equational path theories explaining the core issues exemplified for the equations corresponding to the modal schemas T and 4. Due to the characteristic properties of clauses the unification problems are easier than in free semi-groups or monoids, for example. Related unification algorithms and resolution calculi found in the literature are designed either for more specific or more general unification problems. The unification algorithm of Otten and Kreitz (1996) is unsuitable for our purposes, as it is designed for terms satisfying the T-string property, and although any set of terms which satisfies this property is prefix stable, the converse does not hold. On the other hand, the algorithms designed for the non-optimised translations are too general. The non-optimised functional translations require extended (strong) forms of Skolemisation in order that a particular ordering within terms is preserved. The resulting clauses satisfy the unique prefix property (Auffray and Enjalbert 1992), which is a weaker property than prefix stability. Although it is possible to use the more general algorithms for our purposes, much of the computations are unnecessarily wasted. Another problem with some of the existing systems is their incompleteness. The unification algorithms we present are sound and complete (and terminating), they have fewer rules, the search spaces are smaller, and no unifiers are computed repeatedly. Moreover, our proofs are considerably simpler, though remaining technical, and special consideration is given to normalisation.

The paper is organised as follows. Section 2 defines the class of path logics and the equational schemas we consider. Section 3 considers unification for the basic path logic, recalling some essential definitions and facts of syntactic unification. In Section 4 we discuss E-unification for the schemas T and 4 reviewing what is known from the literature. The main parts are Sections 5 and 6. Section 5 presents a mutation algorithm for the combination of T and 4 illustrating the computational gain and presenting new proofs of termination, soundness and completeness. Section 6 proves prefix stability is an invariance property under binary E-resolution. The conclusion mentions some open problems.

2 Path logics

Path logics arise from quantified modal logics by the functional translation method (Ohlbach 1991, Auffray and Enjalbert 1992) and are related to deterministic logics (Fariñas del Cerro and Herzig 1995). We focus on a subclass of path logics associated with propositional modal logics by the optimised functional translation method (Ohlbach and Schmidt 1997). More precisely, we consider the basic path logic and its extensions associated with popular serial modal logics *KD*, *KT*, *S4* (which coincides with *KT4* or *KDT4*) and *S5* (which coincides with *KDB4*, *KTB4* and *KT5*).

The functional translation is based on the functional semantics of modal logic. In the functional semantics the central concept is that of a functional frame (W, AF) consisting of a set W of worlds and a set AF of accessibility functions (for serial modal logics). A modal formula $\Box\varphi$ is true in a world x iff φ is true in the world $\alpha(x)$ for any function α in AF. The functional translation mapping Π mimics this semantics. For serial modal logics it is defined by

$$\Pi(\varphi) = \forall x\ \pi(\varphi, x)$$

with π being an auxiliary function given by

$$\begin{aligned} \pi(p, s) &= Ps \\ \pi(\bot, s) &= \bot \\ \pi(\varphi \to \psi, s) &= \pi(\varphi, s) \to \pi(\psi, s) \\ \pi(\Box\varphi, s) &= \forall\alpha\ \pi(\varphi, [s\alpha]). \end{aligned}$$

P is a unary predicate symbol uniquely associated with the propositional variable p. The variables x and α belong to disjoint sorts, the sorts W and AF. $[\cdot,\cdot]$ is the application function $W \times AF \longrightarrow W$ representing terms of the form $\alpha(s)$ by $[s\alpha]$. Instead of $[[s\alpha]\beta]$ we write $[s\alpha\beta]$. The reason for this notation is that it reflects paths in the underlying semantics. For example, $[s\alpha\beta]$ denotes the world(s) reached from the world s via the function α and the function β. For non-serial modal logics the operation Π is slightly more complex (Schmidt 1997).

Applying the quantifier exchange operator Υ yields the optimised functional translation. Υ swaps functional quantifiers according the rule

$$\exists\alpha\forall\beta\ \varphi \quad \text{becomes} \quad \forall\beta\exists\alpha\ \varphi.$$

Although $\Upsilon\Pi(\varphi)$ is not logically equivalent to $\Pi(\varphi)$, we have:

Theorem 2.1 (Ohlbach and Schmidt 1997) For any complete propositional modal logic $K\Sigma$, a formula φ is a theorem of $K\Sigma$ iff $\neg\Upsilon\Pi(\varphi)$ is refutable modulo the theory $\Pi(\Sigma)$ or $\Upsilon\Pi(\Sigma)$.

Σ denotes a (possibly empty) set of additional modal schemas. For the purposes of this paper we assume any non-empty Σ includes the schema $D = \Box p \to \Diamond p$, which ensures seriality. Important for this paper is that it ensures the theories are equational.

The net effect of the exchange operation followed by negation is that no non-constant Skolem terms occur in the clausal form of $\neg\Upsilon\Pi(\varphi)$. As $\neg\Upsilon\Pi(\varphi)$ is of the form $\exists x \ldots$, every term in the clausal form starts with the same constant, the initial world, which we choose to denote by the empty list $[]$.

Figure 1 illustrates the steps of the conversion of a modal formula φ into clausal form. The functional translation Π associates the boxes in (the

$$
\begin{array}{rl}
\varphi: & \Diamond\Box(\Box\Diamond p \vee \Box p) \\
\Pi(\varphi): & \forall x \exists \alpha \forall \beta\ (\forall \gamma \exists \delta\ \neg P[x\alpha\beta\gamma\delta] \vee \forall \epsilon\ P[x\alpha\beta\epsilon]) \\
\Upsilon\Pi(\varphi): & \forall x \forall \beta \forall \gamma \forall \epsilon \exists \alpha \exists \delta\ (\neg P[x\alpha\beta\gamma\delta] \vee P[x\alpha\beta\epsilon]) \\
\neg\Upsilon\Pi_f(\varphi): & \exists x \exists \beta \exists \gamma \exists \epsilon \forall \alpha \forall \delta\ (P[x\alpha\beta\gamma\delta] \vee \neg P[x\alpha\beta\epsilon]) \\
S_\varphi: & 1.\quad P[\alpha\underline{\beta}\underline{\gamma}\delta] \\
 & 2.\ \neg P[\alpha'\underline{\beta}\underline{\epsilon}].
\end{array}
$$

Figure 1: A sample translation

disjunctive normal form of) φ with different universal quantifiers and the diamonds with different existential quantifiers. It also transforms the propositional symbol p into a monadic predicate P. The next conversion by the quantifier exchange operator Υ moves existential quantifiers inward. It does so in a way that the Skolemised clause form of the negation of $\Upsilon\Pi(\varphi)$ does not contain any complex Skolem terms. This optimisation is always possible. In the example the order of the quantifiers in the prefix of the first part of the disjunction is reversed. Now, the formula is negated and transformed into clausal form S_φ. By convention the underlined symbols denote Skolem constants.

Formally, the target logic of the translation of *KD* (and also *K* and *S5*) is the *basic path logic*. It is a clausal logic defined over a language with two principal sorts: the sort W and the sort AF. The vocabulary includes finitely many unary predicates $P, Q, \ldots$, variables $\alpha, \beta, \ldots$ of sort AF, finitely many constants $\underline{\alpha}, \underline{\beta}, \ldots$ of sort AF, a special constant $[]$ of sort W, and the function $[\cdot, \cdot]$. Terms in the basic path logic have the form

$$t = [[[[[]u_1]u_2] \ldots]u_m], \quad \text{or in shorthand notation} \quad t = [u_1 u_2 \ldots u_m],$$

with any u_i denoting a functional term (by which we mean a term of sort AF). We also refer to such terms as *paths*.

By definition, the *prefix* of a variable or constant u_i in t is $[]$ if $i = 1$ and $[u_1 \ldots u_{i-1}]$, otherwise. A set T of terms is said to be *prefix stable* if for any variable α all its occurrences in T have one common prefix. A clause is said to be prefix stable if the set of its terms has this property. By definition, a formula ψ is a *path formula* iff it is a conjunction of prefix stable clauses.

Stronger path logics which we consider are obtained by extending the basic path logic with (finite equational) presentations given by a subset of the following two schemas.

$$\textbf{Right identity:} \quad [x\underline{e}] = x \tag{1}$$

$$\textbf{Associativity:} \quad [x(\alpha \circ \beta)] = [x\alpha\beta] \tag{A}$$

The symbol x denotes a variable of sort world, $\underline{e}$ (the identity constant) is a functional constant and $\circ$ (composition) is an operation of the kind $AF \times AF \longrightarrow AF$. The universal closures of the two equations are the global versions of the functional correspondence properties of the modal schemas $T = \Box p \rightarrow p$ and $4 = \Box p \rightarrow \Box\Box p$, respectively. The two equations can be seen to be reformulations of the the relational correspondence properties, namely, reflexivity and transitivity. Right identity says that $\underline{e}$ is the identity function in AF which maps any world x to itself. Associativity says that for any world x and any two functions α and β applied consecutively, there is a function, namely the composition of α and β, which maps x to the same world.

The inferences rules of path logics with equational theories are those of standard E-resolution. For efficiency reasons, as general E-unification is an expensive operation, we adopt a calculus comprising of at least binary E-resolution, syntactic factoring and normalisation under E. As we do not do semantic factoring there is no need for considering E-unification (with $E \neq \emptyset$) of non-singleton problems sets.

More formally, let L, L', L_i denote literals, C, C' clauses, and S a set of clauses. The rules are the following.

Binary E-resolution: $\dfrac{C \vee L \qquad C' \vee \neg L'}{(C \vee C')\sigma}$

where σ is a minimal (most general) E-unifier of $L \vee L'$. Implicit renaming of variables ensures that the premises are variable disjoint.

Syntactic factoring: $\dfrac{C \vee L_1 \vee \ldots \vee L_n}{(C \vee L_1)\sigma}$

where σ is the (syntactic) most general unifier of $L_1 \vee \ldots \vee L_n$.

Normalisation under E: $\quad S \cup \{C\} \triangleright S \cup \{N_E(C)\}$

where $N_E(C)$ is an E-normal form of C. The symbol $\triangleright$ denotes the derivability relation (for example, $S \triangleright S \cup \{C\}$ if C is an E-resolvent of two clauses in S). The normalisation rule will be applied eagerly. The following rule is optional.

Syntactic condensing: $S \cup \{C\} \triangleright S \cup \{\textsc{cond}(C)\}$,

where $\textsc{cond}(C)$ is a minimal subclause of C which is also a factor of it. Here, minimality is with respect to size. Condensations are unique modulo variable renaming (Joyner Jr. 1976).

We use the notation $\mathcal{R}^E_{N_E}$ and $\mathcal{R}^E_{\textsc{cond}\,\circ\, N_E}$ to denote complete E-resolution calculi employing the above rules. The index $\textsc{cond} \circ N_E$ indicates that normalisation is applied before condensing. The calculi $\mathcal{R}^E_{N_E}$ and $\mathcal{R}^E_{\textsc{cond}\,\circ\, N_E}$ are refutationally sound and complete.

This completes the syntactic definition of basic path logic and some of its extensions. Their semantics is defined as usual by Herbrand models.

Later we will refer to the following alternative characterisation of prefix stability (inspired by the definition of tree-likeness found in Zamov 1989). A set T of terms is prefix stable iff for any two terms $[u_1 \ldots u_m]$ and $[v_1 \ldots v_n]$ in T these conditions hold for variables:

T1 If some variable u_i and some variable v_j are identical then $i = j$, and

T2 the terms of each pair u_k and v_k preceding u_i and v_i, respectively, are also identical.

T1 implies paths are linear terms, and it also implies every variable that occurs at position i in some term of the set T occurs at position i in every term, when it does occur in that term.

A note on our notation is in order. The symbols $u, u_1, u_2, \ldots, v, v_1, v_2, \ldots$ are reserved for terms of sort AF. The symbols $s, t, \ldots$ are reserved for any world terms (of sort W). A term $[s\,t]$ is strictly speaking malformed since the second argument of $[\cdot,\cdot]$ is meant to have the sort AF, but when we write $[s\,t]$ we mean the term $[s\,u_1 \ldots u_i]$ given that $t = [u_1 \ldots u_i]$. For any term $s = [u_1 \ldots u_m]$, define $s|_i$ by

$$\begin{aligned} s|_0 &= [], \quad \text{and} \\ s|_i &= u_i \quad \text{for any } 0 < i \leq m. \end{aligned}$$

If each subterm u_i of s is either a variable or a constant then s is called a *basic path*.

Delete	$P \cup \{s =^? s\}$	$\rightsquigarrow$	P	(for world terms)
	$P \cup \{u =^? u\}$	$\rightsquigarrow$	P	(for terms of sort AF)
Decompose	$P \cup \{[su] =^? [tv]\}$	$\rightsquigarrow$	$P \cup \{s =^? t,\, u =^? v\}$	
Conflict	$P \cup \{[su] =^? []\}$	$\rightsquigarrow$	$\perp$	
	$P \cup \{\underline{\alpha} =^? \underline{\beta}\}$	$\rightsquigarrow$	$\perp$	when $\underline{\alpha} \neq \underline{\beta}$
Coalesce	$P \cup \{\alpha =^? \beta\}$	$\rightsquigarrow$	$P\{\alpha \mapsto \beta\} \cup \{\alpha =^? \beta\}$	
	when $\alpha \neq \beta$ are variables both occurring in P.			
Eliminate	$P \cup \{\alpha =^? \underline{\beta}\}$	$\rightsquigarrow$	$P\{\alpha \mapsto \underline{\beta}\} \cup \{\alpha =^? \underline{\beta}\}$	
	when α is a variable occurring in P and $\underline{\beta}$ is constant.			

Figure 2: Syntactic unification rules for the basic path logic

3 Unification for the basic path logic

Because the basic path language has no compound functional terms, any non-empty substitution σ defined over sets of basic paths consists of bindings that have one of two forms, namely $\alpha \mapsto \beta$ or $\alpha \mapsto \underline{\gamma}$. A substitution is said to be *admissible* for the basic path logic iff its bindings have this form. It is immediate that admissible substitutions or unifiers do not change the depths (or lengths) of paths, and only terms of equal depth are unifiable.

The general transformation rules of syntactic tree based unification (from Jouannaud and Kirchner (1991), for example) adapt to those of Figure 2 for the basic path logic. P denotes a problem set of pairs $s =^? t$, of world terms, or $u =^? v$, of functional terms. The equality relation $=^?$ is assumed to be symmetric. The symbol $\rightsquigarrow$ denotes the derivability relation in a unification calculus. $\perp$ indicates failure of the unification problem. All other symbols have the usual interpretation, and s and t may be empty paths. It is important that we keep in mind any term $[u_1 \ldots u_m]$ is an abbreviation for a nested term $[[[[[]u_1]u_2] \ldots]u_m]$. This determines how paths are decomposed, namely from right to left, as this example illustrates.

$$\begin{aligned} \{[\underline{\alpha}\beta\underline{\gamma}] =^? [\underline{\alpha}\delta\epsilon]\} \quad &\overset{\text{Dec}}{\rightsquigarrow} \quad \{\underline{\gamma} =^? \epsilon,\, [\underline{\alpha}\beta] =^? [\underline{\alpha}\delta]\} \\ \overset{\text{Dec}}{\rightsquigarrow} \quad \{\underline{\gamma} =^? \epsilon,\, \beta =^? \delta,\, [\underline{\alpha}] =^? [\underline{\alpha}])\} \quad &\overset{\text{Del.}}{\rightsquigarrow} \quad \{\underline{\gamma} =^? \epsilon,\, \beta =^? \delta\}. \end{aligned}$$

Evidently, any most general unifier of a unification problem over basic paths obtained by the rules of Figure 2 is an admissible substitution. As no world variables occur in basic path clauses, the occurs check rule is superfluous. Soundness and completeness is immediate by soundness and completeness of the general rules for syntactic unification.

In the example we did not use the rules Coalesce and Eliminate. In fact, they are redundant for singleton problem sets:

Theorem 3.1 Let P be a singleton set $\{s =^? t\}$ with s and t terms for which T1 for variables holds. Then the rules Coalesce and Eliminate are redundant.

Proof. The unification rules are assumed to be applied don't care non-deterministically. This allows us to apply Decompose repeatedly until we get $\{s' =^? [], u_1 =^? v_1, \ldots, u_m =^? v_m\}$, where s' is not empty when the length of s is greater than that of t. Now it is easy to see that since s and t are linear terms the conditions of Coalesce and Eliminate cannot be satisfied. $\square$

The situation is pleasantly simple for the logic *S5*. In *S5* any sequence of modal operators can be replaced by the first one in the sequence, and *S5* corresponds to the fragment of monadic first-order logic in one variable (via the relational translation). This is reflected in the corresponding path logic by the fact that any singleton unification problem $[u_1 \ldots u_m] =^? [v_1 \ldots v_n]$ can be seen to reduce to the unification problem of $[u_1] =^? [v_1]$. Such problems can be solved modulo (a subset of) the rules of Figure 2.

4 Unification for (1) and (A)

We turn to unification of paths under the right identity law (1) and the associativity law (A).

Unification under (1) is finitary and decidable. This can be seen easily by considering a unification problem in n variables and forming 2^n syntactic unification problems by replacing some of the variables by $\underline{e}$. Each of the problems is decidable by syntactic unification in linear time. Therefore, the decision problem of unification under (1) is in NP, and by a result of Arnborg and Tidén (1985) for standard right identity it is at least NP-complete.

Unification under (A) is related to unification under standard associativity. Plotkin (1972) shows unification in free semi-groups is infinitary and he gives a unification algorithm that is sound and complete, but it is not

Delete	$P \cup \{s =^? s\}$	$\rightsquigarrow$	P
Decompose	$P \cup \{s * s' =^? t * t'\}$	$\rightsquigarrow$	$P \cup \{s =^? t,\, s' =^? t'\}$
Check	$P \cup \{\alpha =^? s\}$	$\rightsquigarrow$	$\bot$
	when s is not a variable and α occurs in s		
Coalesce	$P \cup \{\alpha =^? \beta\}$	$\rightsquigarrow$	$P\{\alpha \mapsto \beta\} \cup \{\alpha =^? \beta\}$
	when $\alpha \neq \beta$ are variables both occurring in P		
Eliminate	$P \cup \{\alpha =^? s\}$	$\rightsquigarrow$	$P\{\alpha \mapsto s\} \cup \{\alpha =^? \underline{\beta}\}$
	when α is a variable occurring in P and s is not a variable		
Identity	$P \cup \{s * \alpha * s' =^? t\}$	$\rightsquigarrow$	$P \cup \{\alpha =^? \underline{e},\, s * s' =^? t\}$
Path-separat.	$P \cup \{\alpha * s =^? t * t'\}$	$\rightsquigarrow$	$P \cup \{\alpha =^? t,\, s =^? t'\}$
Splitting	$P \cup \{\alpha * s'' * s =^? t * t'' * \beta * t'\}$		
	$\rightsquigarrow P \cup \{\alpha =^? t * t'' * \beta_1,\, \beta =^? \beta_1 * \beta_2,\, s'' * s =^? \beta_2 * t'\}$		
	when s'' and t'' are non-empty and β_1, β_2 are new variables.		

Figure 3: Ohlbach's unification rules for (1) and (A)

guaranteed to terminate. Fortunately, though Plotkin's algorithm is non-terminating in the general case, it decides unification problems of one linear equation, or one equation in which no variable occurs more than twice (Schulz 1992). This implies, unification of one pair of paths under the form of associativity we consider is also finitary and decidable. There are decision procedures for testing satisfiability by Makanin (1977) and Jaffar (1990), for example, but these are far too complex for our purposes. In work still in manuscript form we showed that unifiability under (A) and/or (1) can be achieved in the worst case by a quadratic time algorithm. This algorithm exploits a correspondence to regular expressions.

Unification algorithms are described in Ohlbach (1988, 1991), Fariñas del Cerro and Herzig (1995) and Auffray and Enjalbert (1992) for the non-optimised translation of quantified modal logics and in Zamov (1989) for the non-optimised translation of propositional *S4*. The first three algorithms are not complete. Problems of the form $\{s =^? s * \underline{e}\}$, $\{s =^? s\,;\mathrm{ID}\}$ and $\{s\,!\,\alpha\,!\,f(s\,!\,\alpha) =^? s\,!\,f(s)\}$ (using in essence the notation of the respective authors) are not treated properly, which can be rectified by adding a rule for deleting the identity constant. The standard deletion rule suffices for solving singleton problems of basic paths though, so that under this condition the system from Ohlbach (1991) relevant for propositional *S4*, presented in

$$
\begin{array}{l}
\alpha * \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\alpha} * \beta * \underline{\gamma} \\
\overset{\text{Dec}}{\leadsto} \quad \alpha =^? \underline{\alpha},\ \underline{\beta} * \underline{\gamma} * \delta =^? \beta * \underline{\gamma} \\
\qquad\qquad \left.\begin{array}{l}
\overset{\text{Dec}}{\leadsto} \quad \beta =^? \underline{\beta},\ \underline{\gamma} * \delta =^? *\underline{\gamma} \\
\qquad\qquad \overset{\text{Id}}{\leadsto} \quad \delta =^? \underline{e} \ \leadsto\ 1. \\
\qquad\qquad \overset{\text{Dec}}{\leadsto} \quad \delta =^? \underline{e} \quad \text{redundant} \\
\overset{\text{Id}}{\leadsto} \quad \beta =^? \underline{e},\ \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\gamma} \ \leadsto\ \ldots \quad \text{not solved} \\
\overset{\text{Sep}}{\leadsto} \quad \beta =^? \underline{\beta} * \underline{\gamma},\ \delta =^? \underline{\gamma} \ \leadsto\ 2. \\
\overset{\text{Sep}}{\leadsto} \quad \beta =^? \underline{\beta},\ \underline{\gamma} * \delta =^? \underline{\gamma} \ \leadsto\ \ldots \quad \text{not solved} \\
\overset{\text{Sep}}{\leadsto} \quad \beta =^? \underline{e},\ \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\gamma} \quad \text{redundant} \\
\overset{\text{Id}}{\leadsto} \quad \delta =^? \underline{e},\ \underline{\beta} * \underline{\gamma} =^? \beta * \underline{\gamma} \\
\qquad\qquad \overset{\text{Dec,Del}}{\leadsto} \quad \beta =^? \underline{\beta} \ \leadsto\ 3. \\
\qquad\qquad \overset{\text{Id}}{\leadsto} \quad \beta =^? \underline{e},\ \underline{\beta} * \underline{\gamma} =^? \underline{\gamma} \quad \text{not solved} \\
\qquad\qquad \overset{\text{Sep}}{\leadsto} \quad \beta =^? \underline{e},\ \underline{\beta} * \underline{\gamma} =^? \underline{\gamma} \quad \text{redundant} \\
\overset{\text{Split}}{\leadsto} \quad \beta =^? \underline{\beta} * \underline{\gamma} * \delta_1,\ \delta =^? \delta_1 * \delta_2,\ \delta_2 =^? \underline{\gamma} \ \leadsto\ 4.
\end{array}\right\} (') \\
\overset{\text{Split}}{\leadsto} \quad \alpha =^? \underline{\alpha} * \beta_1,\ \beta =^? \beta_1 * \beta_2,\ \underline{\beta} * \underline{\gamma} * \delta =^? \beta_2 * \underline{\gamma} \\
\qquad\qquad\qquad \vdots \ \text{like } (') \\
\overset{\text{Id}}{\leadsto} \quad \alpha =^? \underline{e},\ \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\alpha} * \beta * \underline{\gamma} \ \leadsto\ \ldots \\
\overset{\text{Id}}{\leadsto} \quad \delta =^? \underline{e},\ \alpha * \underline{\beta} * \underline{\gamma} =^? \underline{\alpha} * \beta * \underline{\gamma} \ \leadsto\ \ldots \\
\overset{\text{Id}}{\leadsto} \quad \beta =^? \underline{e},\ \alpha * \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\alpha} * \underline{\gamma} \ \leadsto\ \ldots
\end{array}
$$

Figure 4: A sample derivation of A1-unifiers according to Ohlbach's algorithm

Figure 3, is complete.[1] The language is a variation from ours. Terms are lists built from variables and constants of the sort AF with an associative operation $*$, making the additional operation $\circ$ superfluous. (The correspondence to world terms of basic path logic is given by $h([[]u]) = u$ and $h([su]) = h(s) * u$ when $s \neq []$.) The symbols s, s', t and t' in the figure denote (possibly empty) lists.

By way of an example we will demonstrate the system can be improved. Figure 4 sketches a derivation of A1-unifiers for $\{\alpha * \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\alpha} * \beta * \underline{\gamma}\}$.[2]

[1] Strictly, Figure 3 is a reformulation in terms of rules. It corrects an omission in the RTA'98 paper.

[2] An *A1-unifier* is a unifier modulo the equations (1) and (A).

Terms are decomposed from left to right. Failure branches are those that do not produce solved forms, that is, sets of the form $\{\alpha_1 =^? u_1, \ldots, \alpha_n =^? u_n\}$ with each α_i occurring exactly once in the set. The successful branches in the derivation tree are those marked with numbers, whose solved forms yield the following unifiers:

1. $\{\alpha \mapsto \underline{\alpha},\ \beta \mapsto \underline{\beta},\ \delta \mapsto \underline{e}\}$
2. $\{\alpha \mapsto \underline{\alpha},\ \beta \mapsto \underline{\beta} * \underline{\gamma},\ \delta \mapsto \underline{\gamma}\}$
3. $\{\alpha \mapsto \underline{\alpha},\ \beta \mapsto \underline{\beta},\ \delta \mapsto \underline{e}\}$ (a duplicate of 1.)
4. $\{\alpha \mapsto \underline{\alpha},\ \beta \mapsto \underline{\beta} * \underline{\gamma} * \delta_1,\ \delta \mapsto \delta_1 * \underline{\gamma}\}$

etcetera.

In the next section we will give a set of rules applying paramodulation only at the top symbols of the terms of an equation $s =^? t$ bearing a more efficient unification algorithm. These restricted forms of paramodulation rules are known as *mutation rules* and are sound and complete only for very particular E. For instance, they may be applied where E (is a finite resolvent set of equations and) defines a syntactic theory. Two results from the literature are of relevant. One, Kirchner and Klay (1990) prove mutation rules are complete for syntactic collapse-free theories. Two, Comon, Haberstrau and Jouannaud (1994) consider mutation with (and without) collapsing equations for shallow theories and prove any shallow theory is syntactic. A *collapsing equation* has the form $x = t$ with x a variable of t and $x \neq t$.[3] This is relevant for the identity law which is collapsing and shallow. The result of Kirchner and Klay is relevant for our associativity law which can be shown to be syntactic by an analogous argument as in Klay (1991) for ordinary associativity.

There are a number of negative results concerning termination for algorithms with mutation rules. Klay (1991) showed collapse-free syntactic theories exist with undecidable unification problems. It is also undecidable whether a given finite set E of collapse-free identities is resolvent or whether the theory defined by E is syntactic. The standard A-unification algorithm by mutation coincides with the algorithm of Plotkin (1972). Mutation together with decomposition and merging need not terminate, but since we will

[3]A theory E is *collapse-free* if no presentation of E contains a collapsing equation. A *shallow* theory has a finite presentation of shallow equations, defined to be equations $s = t$ with all variables of s or t occurring at depth one.

consider only singleton unification problems and our terms are linear, we do not need merging.[4]

It is not clear from the literature whether the combination of mutation rules for shallow and collapsing axioms and those for syntactic axioms automatically bear a complete procedure. The next section outlines a proof for the completeness of the combination of right identity and associativity, without relying on the notion of syntactic-ness.

5 Mutation and normalisation for (1) and (A)

The normalising functions N_1 and N_{A} rearrange terms according to the rewrite rules

$$[x\underline{e}] \to x \quad \text{and} \quad [x(\alpha \circ \beta)] \to [x\alpha\beta].$$

Inductive specifications of N_1 and N_{A} are: $N_1([]) = []$ and $N_1([s\underline{e}]) = N_1(s)$, and $N_{\mathrm{A}}([]) = []$,

$$\begin{aligned} N_{\mathrm{A}}([su]) &= [N_{\mathrm{A}}(s)u] \quad \text{provided } u \text{ is a variable or constant, and} \\ N_{\mathrm{A}}([s(v \circ v')] &= N_{\mathrm{A}}([N_{\mathrm{A}}([sv])v']). \end{aligned}$$

Normalisation under both (1) and (A) is by $N_{\mathrm{A}1}(s) = N_1(N_{\mathrm{A}}(s))$, which first eliminates the operation $\circ$ and then the identity constant $\underline{e}$. Clearly, all three functions are recursive.

As we employ a resolution calculus requiring unification under a non-empty theory E in the resolution rule only, and not the factoring rule, we make the following assumption.

Assumption: Any unification problem has the form $P = \{s =^? t\}$ where s and t are variable disjoint basic paths, (i) $\{s, t\}$ is prefix stable, (ii) s and t do not contain world variables, and (iii) are normalised by $N_{\mathrm{A}1}$.

(ii) is ensured for the negation of the translation of any modal formula and it is preserved since no world variables will be introduced during unification. Thus, *admissible* substitutions have the form $\alpha \mapsto u$ with u a functional term.

[4] I do not know whether there are syntactic theories with undecidable unification problems even without merging.

Delete	$P \cup \{s =^? s\}$	$\rightsquigarrow$	P
	$P \cup \{u =^? u\}$	$\rightsquigarrow$	P
Variable Elim.	$P \cup \{\alpha =^? u\}$	$\rightsquigarrow$	$P\{\alpha \mapsto u\}$
	when α is an introduced variable and does not occur in u.		
Decompose	$P \cup \{[su] =^? [tv]\}$	$\rightsquigarrow$	$P \cup \{s =^? t,\, u =^? v\}$
Mutate-1	$P \cup \{[s\alpha] =^? t\}$	$\rightsquigarrow$	$P \cup \{s =^? t,\, \alpha =^? \underline{e}\}$
Mutate-A	$P \cup \{[s\alpha] =^? [tv]\}$	$\rightsquigarrow$	$P \cup \{[s\alpha'] =^? t,\, \alpha =^? \alpha' \circ v\}$
	when α' is a new variable and not both s and t are empty.		

Figure 5: Unification rules for the path logics closed under (1) and (A)

Our unification rules for path logics closed under (1) or (A), or both, are those listed in Figure 5.[5] Here, s and t may denote empty paths, except where stated otherwise. Observe, the variable elimination rule applies only to variables introduced by applications of the Mutate-A rule. The system does not decompose or mutate functional terms involving $\circ$, and no normalisation is done in the unification algorithm. The rules are (in essence) instances of a subset of the rules from Comon et al. (1994).[6]

There is a subtle difference of Mutate-1 and Mutate-A to the respective instances of the general mutation rules, which are

$$\begin{aligned} P \cup \{[su] =^? t\} &\rightsquigarrow P \cup \{s =^? t,\, u =^? \underline{e}\} \quad \text{and} \\ P \cup \{[su] =^? [tv]\} &\rightsquigarrow P \cup \{[s\alpha'] =^? t,\, u =^? \alpha' \circ v\}. \end{aligned}$$

It is clear that the case, when u is a constant different from $\underline{e}$ in the first rule, leads to an unsatisfiable situation $\underline{\alpha} =^? \underline{e}$. In the second rule when u is a constant this leads to the situation $\underline{\alpha} =^? \alpha' \circ v$ which is unsatisfiable when v is not logically equivalent to $\underline{e}$, and in this case $\underline{\alpha} =^? \alpha'$ is redundant.

Mutate-1 binds a variable in a right-most position with the identity constant $\underline{e}$ and deletes the variable from the original term. For example, the only minimal (most general) 1-unifier for $\{[\alpha\underline{\beta}] =^? [\gamma]\}$ is $\{\alpha \mapsto \underline{e}, \gamma \mapsto \underline{\beta}\}$. The unification problem $\{[\alpha\beta] =^? [\gamma]\}$ has two minimal 1-unifiers: $\{\alpha \mapsto \gamma,\, \beta \mapsto$

[5]In the RTA'98 paper one of the deletion rules was forgotten.

[6]For readers familiar with this paper we note, in our context their cycle breaking rule can be seen to be superfluous, since there are no world variables in the original problem, and for the functional variables Cycle applies to equations of the form $\alpha \circ u =^? \alpha$ or $u \circ \alpha =^? \alpha$, which our algorithm does not produce as we will see.

$\underline{e}\}$ and $\{\alpha \mapsto \underline{e},\, \beta \mapsto \gamma\}$. The algorithm computes a third unifier, namely $\{\alpha \mapsto \underline{e},\, \beta \mapsto \underline{e},\, \gamma \mapsto \underline{e}\}$, which is not most general.

Mutate-A applies to terms $s = [u_1 \dots u_{m+1}] =^? [v_1 \dots v_{n+1}] = t$ with the pair (u_{m+1}, v_{n+1}) being either a variable-constant pair, a constant-variable pair or a variable-variable pair. For the first two constellations there is one transformation by Mutate-A and for the last constellation there are two.

$$\begin{array}{rcl} \{[u_1 \dots u_m\alpha] =^? [v_1 \dots v_n\underline{\beta}]\} & \overset{\mathrm{A}}{\rightsquigarrow} & \{\alpha =^? \alpha' \circ \underline{\beta},\, [u_1 \dots u_m\alpha'] =^? [v_1 \dots v_n]\} \\ \{[u_1 \dots u_m\alpha] =^? [v_1 \dots v_n\gamma]\} & \overset{\mathrm{A}}{\rightsquigarrow} & \{\alpha =^? \alpha' \circ \gamma,\, [u_1 \dots u_m\alpha'] =^? [v_1 \dots v_n]\} \\ & \text{or} & \{\gamma =^? \gamma' \circ \gamma,\, [u_1 \dots u_m] =^? [v_1 \dots v_n\gamma']\}. \end{array}$$

This illustrates that the search tree for transformations with Mutate-A can be seen to be an instance of the search tree of Plotkin's (1972) algorithm for semi-groups (applied to paths and employing right-to-left as opposed to left-to-right decomposition).

Compare the derivation in Figure 4 according to Ohlbach's system with the derivation in Figure 6 according to the mutation system. The successful branches in the derivation tree yield the following unifiers:

1. $\{\delta \mapsto \underline{\gamma},\, \beta \mapsto \underline{\beta} \circ \underline{\gamma},\, \alpha \mapsto \underline{\alpha}\}$
2. $\{\delta \mapsto \underline{\gamma},\, \beta \mapsto (\beta'' \circ \underline{\beta}) \circ \underline{\gamma},\, \alpha \mapsto \underline{\alpha} \circ \beta''\}$
3. $\{\delta \mapsto \underline{e},\, \beta \mapsto \underline{\beta},\, \alpha \mapsto \underline{\alpha}\}$
4. $\{\delta \mapsto \underline{e},\, \beta \mapsto \beta' \circ \underline{\beta},\, \alpha \mapsto \underline{\alpha} \circ \beta'\}$
5. $\{\delta \mapsto \delta' \circ \underline{\gamma},\, \beta \mapsto (\underline{\beta} \circ \underline{\gamma}) \circ \delta',\, \alpha \mapsto \underline{\alpha}\}$
6. $\{\delta \mapsto \delta' \circ \underline{\gamma},\, \beta \mapsto ((\beta'' \circ \underline{\beta}) \circ \underline{\gamma}) \circ \delta',\, \alpha \mapsto \underline{\alpha} \circ \beta''\}$.

Clearly, the search tree is considerably smaller and there are no repetitions in the solution set. The solution set is not minimal though.

Now, we prove our system is sound and complete. By definition, a set of transformation rules is *sound and complete* in a theory E if the following two conditions hold:

Soundness: If P transforms to P' by the application of any of the transformation rules, written $P \overset{*}{\rightsquigarrow} P'$, then every E-unifier of P' is an E-unifier of P.

Completeness: For any E-unifier θ of P, there is some P' in solved form such that $P \overset{*}{\rightsquigarrow} P'$ and the idempotent unifier σ associated with P'

$$[\alpha\underline{\beta}\underline{\gamma}\delta] =^? [\underline{\alpha}\beta\underline{\gamma}]$$

$$\overset{\text{Dec}}{\rightsquigarrow} \quad \delta =^? \underline{\gamma},\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}\beta]$$

$$\left.\begin{array}{lll}
\overset{\text{Dec}}{\rightsquigarrow} & \beta =^? \underline{\gamma},\ [\alpha\underline{\beta}] =^? [\underline{\alpha}] \quad \text{not solved} & \\
\overset{1}{\rightsquigarrow} & \beta =^? \underline{e},\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}] \quad \text{not solved} & \\
\overset{\text{A}}{\rightsquigarrow} & \beta =^? \beta' \circ \underline{\gamma},\ [\alpha\underline{\beta}] =^? [\underline{\alpha}\beta'] & \\
 & \overset{2\times\text{Dec,Del}}{\rightsquigarrow} & \beta' =^? \underline{\beta},\ \alpha =^? \underline{\alpha} \ \rightsquigarrow\ 1. \\
 & \overset{1}{\rightsquigarrow} & \beta' =^? \underline{e},\ [\alpha\underline{\beta}] =^? [\underline{\alpha}] \\
 & & \quad \text{not solved} \\
 & \overset{2\times A,\text{Dec,Elim}}{\rightsquigarrow} & \beta' =^? \beta'' \circ \underline{\beta},\ \alpha =^? \underline{\alpha} \circ \beta'' \\
 & & \quad \rightsquigarrow\ 2.
\end{array}\right\} ('')$$

$$\overset{1}{\rightsquigarrow} \quad \delta =^? \underline{e},\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}\beta\underline{\gamma}]$$

$$\begin{array}{lll}
\overset{\text{Dec,Del}}{\rightsquigarrow} & [\alpha\underline{\beta}] =^? [\underline{\alpha}\beta] & \\
 & \overset{2\times\text{Dec}}{\rightsquigarrow} & \beta =^? \underline{\beta},\ \alpha =^? \underline{\alpha} \ \rightsquigarrow\ 3. \\
 & \overset{1}{\rightsquigarrow} & \beta =^? \underline{e},\ [\alpha\underline{\beta}] =^? [\underline{\alpha}] \quad \text{not solved} \\
 & \overset{2\times A,\text{Dec,Elim}}{\rightsquigarrow} & \beta =^? \beta' \circ \underline{\beta},\ \alpha =^? \underline{\alpha} \circ \beta' \ \rightsquigarrow\ 4.
\end{array}$$

$$\overset{\text{A}}{\rightsquigarrow} \quad \delta =^? \delta' \circ \underline{\gamma},\ [\alpha\underline{\beta}\underline{\gamma}\delta'] =^? [\underline{\alpha}\beta]$$

$$\begin{array}{ll}
\overset{\text{Dec}}{\rightsquigarrow} & \beta =^? \delta',\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}] \quad \text{not solved} \\
\overset{1}{\rightsquigarrow} & \delta' =^? \underline{e},\ [\alpha\underline{\beta}\underline{\gamma} =^? [\underline{\alpha}\beta] \ \rightsquigarrow\ \ldots \quad \text{not solved} \\
\overset{1}{\rightsquigarrow} & \beta =^? \underline{e},\ \ldots \quad \text{not solved} \\
\overset{\text{A}}{\rightsquigarrow} & \beta =^? \beta' \circ \delta',\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}\beta] \\
 & \qquad \vdots \ \text{like } ('') \ \rightsquigarrow\ 5. \text{ and } 6.
\end{array}$$

Figure 6: A sample derivation of A1-unifiers

is more general than θ with respect to the variables occurring in P, written $\sigma \leq_E \theta[\mathit{Var}(P)]$.

Formally, a *solved form* is either the empty set or a finite set of the form $\{\alpha_1 =^? u_1, \ldots, \alpha_n =^? u_n\}$ and $\alpha_1, \ldots, \alpha_n$ are distinct variables occurring in no u_i. A variable α is *solved* in a set P if P includes a pair $\alpha =^? u$ (or $u =^? \alpha$) and α occurs exactly once in P. A variable that is not solved is an *unsolved variable*. By definition, $\sigma =_E \theta[V]$ iff for any variable x in V, $x\sigma$ and $x\theta$ are E-equivalent, and $\sigma \leq_E \theta[V]$ iff there is a substitution such that $\sigma\sigma' =_E \theta[V]$.

Equivalence (inequivalence and inclusion) modulo right identity and associativity will be denoted by $=_{\mathrm{A1}}$ ($\neq_{\mathrm{A1}}$ and $\leq_{\mathrm{A1}}$).

Theorem 5.1 The system of Figure 5 is sound.

Proof. By inspecting the rules. $\square$

In the remainder of this section, P denotes a singleton unification problem of variable disjoint basic paths satisfying (i), (ii) and (iii) of the assumption, and P' denotes a set obtained from P by any sequence of transformations in Figure 5. For the next lemma (and also Lemma 5.7) it is important that the initial pair in P is variable disjoint.

Lemma 5.2 For any identity $\alpha =^? u$ in P', the variable α does not occur in u.

Proof. By inspecting the rules. $\square$

Lemma 5.3 Each P' irreducible by the rules of Figure 5 is in solved form or it is unsatisfiable in $=_{\mathrm{A1}}$.

Proof. The only irreducible equations not in the form $\alpha =^? u$ with α a solved variable have the form: $[s\underline{\alpha}] =^? []$, $\underline{\alpha} =^? \underline{\beta}$ and $\underline{\alpha} =^? \underline{e}$. If P' includes any one of these it is not solvable. Observe, as $\underline{e}$ does not occur in the starting P, situations like $[s\underline{e}t] =^? []$ do not arise. $\square$

We now prove completeness.

Theorem 5.4 The system of Figure 5 is complete.

The core structure of the proof is standard. We let

$$P = \{[su] =^? [tv]\}$$

with $s = [s|_1 \dots s|_m]$ and $t = [t|_1 \dots t|_n]$, each non-empty, that is, $1 \leq m, n$. We let θ be any A1-unifier of P, that is,

$$[su]\theta =_{\mathrm{A1}} [tv]\theta.$$

The aim is to show there is a sequence of transformations of P to such that the associated unifier σ is more general than θ. In parallel to transforming P we extend the unifier θ by adding bindings of new variables to θ obtaining θ'. Below, in Lemmas 5.8 and 5.9, we will define θ' in such a way that if θ unifies P and $P \rightsquigarrow P'$, that is, if P transforms to P' in one step, then θ' unifies P'. The resulting procedure starts with the pair (P, θ) and computes at least one pair (P', θ'), such that

1. $P \overset{*}{\rightsquigarrow} P'$,
2. P' is in solved form,
3. $\theta \subseteq \theta'$ and $\theta'|_{Var(\theta)} = \theta$ (the restriction of θ' to the variables of θ coincides with θ).

By assumption θ is a unifier of P, and consequently, by induction on the proof length, θ' of the final pair is a unifier of P'. This establishes completeness, when every derivation is finite. The next lemmas supply the technical details.

Lemma 5.5 The unifier σ associated with the solved P' is more general than θ with respect to the variables of P' and P.

Proof. Every equation in P' has the form $\alpha =^? u$. Since θ' unifies P', $\alpha\theta' = u\theta'$. σ is also a unifier of P', hence $\alpha\sigma = u\sigma = u$. Therefore, for any variable α in P', $\alpha\theta' = \alpha\sigma\theta'$ and this means σ is more general than θ' with respect to $Var(P')$. □

Since θ and θ' restricted to the variables of P are equivalent, it follows that:

Lemma 5.6 The unifier σ associated with P' is more general than θ with respect to the variables of P' and P.

The procedure of transforming (P, θ) to a suitable (P', θ') terminates:

Lemma 5.7 Any fair implementation of a unification algorithm for the transformation system of Figure 5 terminates for any P satisfying the assumption.

Proof. Let $\tau(s)$ denote the functional depth of a term s. Define a measure μ of any unification problem P by $\mu(P) = (d, v)$, where v denotes the number of unsolved variables in P, and d is determined by the depths of the pairs of world terms in P (of which there are at most one). More specifically, $d = \tau(s) + \tau(t)$ if $s =^? t \in P$ and both s and t are of type world, and $d = 0$ if no such pair exists.

Examine each transformation rule in turn to see that $\mu(P')$ is smaller than $\mu(P)$ under the lexicographical ordering. Except for Variable Eliminate each rule decreases the value of d. Variable Eliminate leaves d unchanged but it decreases v. The rules do not convert the status of any variable from solved to unsolved. □

The following two lemmas are concerned with the one step conversions of any pair (P, θ) to a suitable pair (P', θ'). In order to avoid cluttering in the proofs we write just '$=$' in place of '$=_{\mathrm{A1}}$'.

Lemma 5.8 Consider $P \cup \{[su] =^? [tv]\}$ with $s = [s|_1 \ldots s|_m]$ and $t = [t|_1 \ldots t|_n]$ for $1 \leq m, n$. The terms $[su]$ and $[tv]$ are assumed to be in A1-normal form. Let θ be any A1-unifier of $P \cup \{[su] =^? [tv]\}$, in particular,

$$[su]\theta =_{\mathrm{A1}} [tv]\theta. \qquad (**)$$

θ' as defined in the following is in each case an A1-unifier of P'.

1. If $u\theta =_{\mathrm{A1}} v\theta$, then let $\theta' = \theta$ and apply Decompose to P, yielding

$$P' = P \cup \{s =^? t,\ u =^? v\}.$$

2. If $u\theta =_{\mathrm{A1}} \underline{e}$, then let $\theta' = \theta$ and apply Mutate-1 to P, yielding

$$P' = P \cup \{s =^? [tv],\ u =^? \underline{e}\}.$$

3. If u is a variable α, say, and $u\theta =_{\mathrm{A1}} [t|_k \ldots t|_n v]\theta$ for some $1 \leq k \leq n$, then let

$$\theta' = \theta\theta_0 \quad \text{with } \theta_0 = \{\alpha' \mapsto [t|_k \ldots t|_n]\}$$

$\theta' = \theta\theta_0$ with $\theta_0 = \{\alpha' \mapsto [t|_k \ldots t|_n]\}$ and apply Mutate-A to P, yielding

$$P' = P \cup \{[s\alpha'] =^? t,\ \alpha =^? \alpha' \circ v\},$$

for α' a new variable not occurring in P or θ.

Proof. 1. By assumption $(**)$, $[su]\theta = [tv]\theta$, which implies $[(s\theta)(u\theta)] = [(t\theta)(v\theta)]$. Then, since $u\theta = v\theta$, $[(s\theta)(u\theta)] = [(t\theta)(u\theta)]$. Consequently, $s\theta = t\theta$. Therefore, $\theta = \theta'$ is an A1-unifier of P'.

2. $\theta = \theta'$ is an A1-unifier of P', since $u\theta = \underline{e}$ and

$$\begin{aligned}
[tv]\theta &= [su]\theta \qquad \text{by } (**) \\
&= [(s\theta)(u\theta)] = [(s\theta)\underline{e}] \qquad \text{since } u\theta = \underline{e} \\
&= s\theta \qquad \text{by the right identity law.}
\end{aligned}$$

3. θ' is well-defined since α' is a new variable that does not occur in α.

(a) θ' is an A1-unifier of $\alpha =^? \alpha' \circ v$:

$$\begin{aligned}
(\alpha' \circ v)\theta' &= (\alpha' \circ v)\theta\theta_0 \\
&= ([t|_k \ldots t|_n] \circ v)\theta \qquad \text{since } \theta_0 = \{\alpha' \mapsto [t|_k \ldots t|_n]\} \\
&= \alpha\theta \qquad \text{since } \alpha\theta = [t|_k \ldots t|_n v]\theta = ([t|_k \ldots t|_n] \circ v)\theta \\
&= \alpha\theta\theta_0 = \alpha\theta' \qquad \text{since } \alpha\theta_0 = \alpha.
\end{aligned}$$

(b) θ' is an A1-unifier of $[s\alpha'] =^? t$:

$$\begin{aligned}
[s\alpha]\theta' = [s\alpha]\theta\theta_0 = [s\alpha]\theta &= [tv]\theta \qquad \text{by } (**) \\
&= [tv]\theta\theta_0 = [tv]\theta'.
\end{aligned}$$

That is, $[s\alpha]\theta' = [tv]\theta'$, which implies

$$[(s\theta')(\alpha\theta')] = [([t|_1 \ldots t|_{k-1}]\theta')([t|_k \ldots t|_n v]\theta')].$$

Since $[t|_k \ldots t|_n v]\theta' = [t|_k \ldots t|_n v]\theta\theta_0 = [t|_k \ldots t|_n v]\theta = \alpha\theta$ by (a),

$$[(s\theta')(\alpha\theta')] = [([t|_1 \ldots t|_{k-1}]\theta')(\alpha\theta')].$$

Therefore, $s\theta' = [t|_1 \dots t|_{k-1}]\theta'$. Then

$$[s\alpha']\theta' = [(s\theta')(\alpha'\theta')] = [([t|_1 \dots t|_{k-1}]\theta')(\alpha'\theta')].$$

$\alpha'\theta' = \alpha'\theta\theta_0 = [t|_k \dots t|_n]\theta = [t|_k \dots t|_n]\theta\theta_0 = [t|_k \dots t|_n]\theta'$. Hence

$$\begin{aligned}[s\alpha']\theta' &= [([t|_1 \dots t|_{k-1}]\theta')(\alpha'\theta')] \\ &= [([t|_1 \dots t|_{k-1}]\theta')([t|_k \dots t|_n]\theta')] \\ &= [t|_1 \dots t|_{k-1} t|_k \dots t|_n]\theta' \\ &= t\theta'.\end{aligned}$$

This means, θ' is an A1-unifier of $[s\alpha'] =^? t$. □

The lemma also covers the cases that $v\theta =_{\mathrm{A1}} \underline{e}$ and $v\theta =_{\mathrm{A1}} [s|_k \dots s|_n]\theta$ for some $1 \le k \le m$ and v a variable. Observe that when $u\theta =_{\mathrm{A1}} [t|_k \dots t|_n v]\theta$ but both u and v are constants, the conditions of either 1. or 2. hold. If u and v are both constants then either (a) $u = v$ or (b) $u = \underline{\alpha}$, say, and $v = \underline{e}$. (a) implies $u\theta = v\theta$, and (b) implies $v\theta = \underline{e}$.

It remains to clarify whether there are cases which the lemma does not cover. Are there cases such that neither of the following hold?

1. $u\theta =_{\mathrm{A1}} v\theta$,
2. $u\theta =_{\mathrm{A1}} \underline{e}$ (or $v\theta =_{\mathrm{A1}} \underline{e}$),
3. u is a variable and $u\theta =_{\mathrm{A1}} [t|_k \dots t|_n v]\theta$ for some $1 \le k \le n$ (or v is a variable and $v\theta =_{\mathrm{A1}} [s|_k \dots s|_n]\theta$ for some $1 \le k \le m$).

The answer is, yes, as in this example

$$P = \{[\underline{\alpha}\underline{\alpha}'\beta] =^? [\underline{\alpha}\delta\underline{\gamma}]\} \quad \text{and} \quad \theta = \{\beta \mapsto \underline{\epsilon} \circ \underline{\gamma},\ \delta \mapsto \underline{\alpha}' \circ \underline{\epsilon}\} \qquad (***)$$

when in the general case $[s|_k \dots s|_m u]\theta =_{\mathrm{A1}} [t|_l \dots t|_n v]\theta$ is true, for some $1 \le k \le m$ and $1 \le l \le n$. If u and v are both constants then, as above, either $u\theta = v\theta$ or $u\theta = \underline{e}$ or $v\theta = \underline{e}$. The following result deals with the case that one of u or v is a variable. (It implies 3. of the previous lemma.)

Lemma 5.9 Let θ be an A1-unifier of $P \cup \{[s\alpha] =^? [tv]\}$ with $s = [s|_1 \dots s|_m]$ and $t = [t|_1 \dots t|_n]$ for $1 \le m, n$, and both $[s\alpha]$ and $[tv]$ are in A1-normal form. Let

$$[s|_k \dots s|_m \alpha]\theta =_{\mathrm{A1}} [t|_l \dots t|_n v]\theta$$

for some $1 \leq k \leq m$ and $1 \leq l \leq n$. If θ includes a binding of α to u, that is, $\alpha \mapsto u \in \theta$, then let

$$\theta' = \theta\theta_0 \quad \text{with } \theta_0 = \{\alpha' \mapsto u'\}$$

where u' is given by $u =_{\mathrm{A}} u' \circ v'$ and $v' = v\theta$, and apply Mutate-A to P, yielding

$$P' = P \cup \{[s\alpha'] =^? t,\, \alpha =^? \alpha' \circ v\},$$

for α' a new variable not occurring in P or θ. Then θ' unifies P'.

Proof. (a) θ' is an A1-unifier of $\alpha =^? \alpha' \circ v$, since

$$(\alpha' \circ v)\theta' = (\alpha'\theta_0) \circ (v'\theta_0) = u' \circ v' = u = \alpha\theta = \alpha\theta'.$$

(b) θ' is an A1-unifier of $[s\alpha'] =^? t$: By assumption $[s\alpha]\theta = [tv]\theta$, hence $[(s\theta)(\alpha\theta)] = [(t\theta)(v\theta)]$. As $\alpha\theta = u = u' \circ v'$ and $v\theta = v'$ we have

$$[(s\theta)(u' \circ v') = [(s\theta)a'v'] = [(t\theta)v'].$$

It follows that $[(s\theta)a'] = t\theta$. Then, $[s\alpha]\theta' = [s\alpha]\theta\theta_0 = [s\alpha]\theta = t\theta = t\theta'$, as required. $\square$

For example, the pair $(***)$ is converted to

$$\begin{aligned} P' &= \{[\underline{\alpha}\underline{\alpha}'\beta'] =^? [\underline{\alpha}\delta],\, \beta =^? \beta' \circ \underline{\gamma}\} \quad \text{and} \\ \theta' &= \{\beta \mapsto \underline{\epsilon} \circ \underline{\gamma},\, \delta \mapsto \underline{\alpha}' \circ \underline{\epsilon},\, \beta' \mapsto \underline{\epsilon}\}. \end{aligned}$$

The lemma makes assumptions, which are not met in the following two situations. First, if no u' exists such that $u =_{\mathrm{A}} u' \circ (v\theta)$ then $v\theta$ is equivalent to $\underline{e}$. This case is dealt with in 2. of the previous lemma. Second, the situation that neither α nor v are in the domain of θ and $\alpha\theta \neq_{\mathrm{A1}} v\theta$ is impossible (for otherwise $[s\alpha]$ and $[tv]$ are not unifiable).

6 Preservation of prefix stability

Now, we verify that the application of A1-unifiers followed immediately by normalisation under N_{A1} preserves prefix stability. This justifies the assumptions made in the previous section, namely, that the terms in the initial problem set are basic paths and the world terms on the left hand sides of the transformation rules of Figure 5 are also basic paths. We also prove a preservation result for forming $\mathcal{R}^{\mathrm{A1}}_{N_{\mathrm{A1}}}$ and $\mathcal{R}^{\mathrm{A1}}_{\mathrm{COND}\,\circ\, N_{\mathrm{A1}}}$-resolvents. The proofs are very similar to those for applying syntactic unifiers and forming standard resolvents. For the reader inclined to verify the proof of Theorem 6.4 we now prove the corresponding theorem for the basic path logic from which it is adapted. It establishes the preservation of prefix stability under syntactic bindings.

Theorem 6.1 Let T be a set of terms in the vocabulary of the basic path logic. Let $s = [u_1 \ldots u_m]$ and $t = [v_1 \ldots v_n]$ be two terms in T such that for some $k > 0$,

$$u_1 = v_1, \ldots, u_{k-1} = v_{k-1} \quad \text{and} \quad u_k \neq v_k$$

and u_k is a variable. Let σ be the substitution $\{u_k \mapsto v_k\}$. Then $T\sigma$ satisfies T1 and T2, provided T does.

Proof. We consider two arbitrary terms in $T\sigma$. They are of the form $s\sigma$ and $t\sigma$ with s and t some terms in T. For s and t conditions T1 and T2 hold. We want to show they hold for the terms $s\sigma$ and $t\sigma$, too.

By definition, two paths s and t (of equal length) are *k-equal* if s and t are equal except possibly at position k, that is, for every position $i \neq k$, $s|_i = t|_i$.

Lemma 6.2 The terms $s\sigma$ and s are k-equal and differ only when $s|_k = u_k$.

Proof. σ affects only the variable u_k and in any term of T, u_k occurs only at position k else condition T1 is violated. Hence, if u_k occurs in s then $s|_k = u_k$ and for any $l \neq k$, $s|_l \neq u_k$. In this case $s\sigma|_k = v_k \neq s|_k$. $\square$

We continue the proof of Theorem 6.1. The lemma is true for $t\sigma$ and t, as well. If neither s nor t contain the variable u_k then the substitution σ does not effect s and t. Then $s\sigma = s$ and $t\sigma = t$. In this case $s\sigma$ and $t\sigma$ trivially satisfy T1 and T2 (since s and t do).

Therefore, we assume without loss of generality that $s|_k = u_k$. Then $s|_k\sigma = v_k$. Distinguish two cases:

1. $t|_k \neq u_k$ and $t|_k \neq v_k$. σ leaves t unchanged so that $t\sigma = t$. Suppose $s\sigma|_i = t\sigma|_j$ is a variable. Then $s\sigma|_i = t\sigma|_j = t|_j$. Also, $j \neq k$ and $i \neq k$, since otherwise $t|_j = v_k$ which contradicts our assumption. This implies $s|_i = s\sigma|_i = t\sigma|_j = t|_j$. By T1 which holds for s and t we get $i = j$. By T2 for any $l < i = j$ we have $s|_l = t|_l$. Hence $i = j < k$ since otherwise, if $i = j = k$ then $s|_i = u_k \neq t|_i$ which is a contradiction, or if $i = j > k$ then since $s|_k \neq t|_k$ by assumption, s and t contradict T2. Consequently by the Lemma $s\sigma|_l = s|_l = t|_l = t\sigma|_l$. Therefore, conditions T1 and T2 are true for case 1.

2. Now we consider the case that $t|_k = u_k$ or $t|_k = v_k$. Then $t\sigma|_k = v_k = s\sigma|_k$. Suppose $s\sigma|_i = t\sigma|_j$ is a variable.

   (a) If $i = k$ then $s\sigma|_i = v_k = t\sigma|_j$. Then, either $t|_j = u_k$ or $t|_j = v_k$. In either case, it follows that $j = k$ and hence $i = j$.

   (b) If $j = k$ then by a similar argument $i = j$.

   (c) If $i \neq k$ and $j \neq k$ then the Lemma implies $s\sigma|_i = s|_i$ and $t\sigma|_j = t|_j$. Since $s\sigma|_i = t\sigma|_j$ we have $s|_i = t|_j$ and it follows by T1 that $i = j$.

   Therefore, $s\sigma$ and $t\sigma$ satisfy T1.

   Let $l < i = j$ be arbitrary. By T2 we have $s|_l = t|_l$.

   (a) Consider the case that $l \neq k$. By the Lemma $s\sigma|_l = s|_l$ and $t\sigma|_l = t|_l$. Since $s|_l$ and $t|_l$ coincide, we conclude $s\sigma|_l = t\sigma|_l$. (Note that if $t|_k = v_k$ then $s|_k = u_k \neq t|_k$ and consequently $i = j < k$.)

   (b) For $l = k$: $s\sigma|_l = v_k = t\sigma|_l$ by assumption.

This completes the proof. $\square$

Based on this result it is not difficult to prove that prefix stability is preserved under syntactic factoring and ordinary resolution.[7] Also, as prefix stability remains invariant under the formation of subsets, it is immediate that prefix stability is preserved by subsumption deletion and condensing. Thus, the basic path logic is closed under ordinary resolution, syntactic factoring, subsumption deletion and condensing.

[7]Proofs can be obtained by following Ohlbach (1991) or Zamov (1989), or they can be found in Schmidt (1997).

Now, we address closure of the extensions of the basic path logic under the fundamental operations in our resolution calculus for $E = \{\mathrm{A}, 1\}$. We let T be a set of terms in the vocabulary of basic path logic, because remember, every theory resolvent is immediately normalised by N_{A1}. The analogue of Theorem 6.1 is not true in its full generality for bindings of A1-unifiers. It is true when suffixes are variable disjoint, and when more restrictions (to be made precise below) hold for instantiations with $\circ$ terms. For bindings of the form $\alpha \mapsto \underline{e}$ the following is immediate by Theorem 6.1.

Corollary 6.3 Let T be a set of terms in the vocabulary of basic path logic. Let $s = [u_1 \dots u_m]$ and $t = [v_1 \dots v_n]$ be two terms in T such that for some $k > 0$,

$$u_1 = v_1, \dots, u_{k-1} = v_{k-1} \quad \text{and} \quad u_k \neq v_k,$$

u_k is a variable, and the suffixes $[u_{k+1} \dots u_m]$ and $[v_{k+1} \dots v_n]$ are variable disjoint. Let σ be a substitution $\{u_k \mapsto \underline{e}\}$. Then $N_1(T\sigma)$ satisfies T1 and T2, provided T does.

For bindings of the form $u_k \mapsto v \circ v'$, which cause the term depth to increase, the concept of k-equality needs to be generalised to the concept of (k, l)-equality. Two basic paths s and t are *(k, l)-equal* if t is like s except possibly the term $s|_k$ in the k-th position is replaced by a list $w_1 \dots w_l$ of length l. In other words, s and t are (k, l)-equal provided $s = t$, or when $s = [s|_1 \dots s|_m]$ then $t = [s|_1 \dots s|_{k-1} w_1 \dots w_l s|_{k+1} \dots s|_m]$, or the other way around.

Theorem 6.4 Let s and t be two terms in T defined as in the previous result. Let σ be a substitution $\{u_k \mapsto w\}$ where

$$N_{\mathrm{A1}}([w]) = [w_1 \dots w_l] \quad \text{and} \quad w_1 = v_k,$$

and s and w are variable disjoint.[8] Then $N_{\mathrm{A1}}(T\sigma)$ satisfies T1 and T2, provided T, $N_{\mathrm{A1}}([w])$ and the set $\{[w], [v_k \dots v_n]\}$ do.

Proof. Proceed as in the proof of Theorem 6.1 with the obvious modifications. Let s and t be any terms in T that satisfy the conditions T1 and T2 and consider $N_{\mathrm{A1}}(s\sigma)$ and $N_{\mathrm{A1}}(t\sigma)$ in $N_{\mathrm{A1}}(T\sigma)$. It is not difficult to verify that

[8] More accurately, $N_{\mathrm{A1}}([w])$ coincides with $N_{\mathrm{A1}}([[], w]) = [[] w_1 \dots w_l]$.

the pairs $N_{\mathrm{A1}}(s)$ and $N_{\mathrm{A1}}(s\sigma)$, and also $N_{\mathrm{A1}}(t)$ and $N_{\mathrm{A1}}(t\sigma)$, are (k,l)-equal. We assume without loss of generality that $s|_k = u_k$ (for otherwise if u_k does not occur in either of s or t then the result is trivially true). Then

$$N_{\mathrm{A1}}(s\sigma) = [s|_1 \dots s|_{k-1} w_1 \dots w_l s|_{k+1} \dots s|_m].$$

Since s and $[w_1 \dots w_l]$ have no common variables and w satisfies T1 and T2, so does $s\sigma$.

Now, consider two cases: 1. σ leaves t unchanged so that $t\sigma = t$ and 2. it does not. In the either case we need to prove T1 and T2 hold for i and j strictly below $k+l$. This is tedious and as the arguments are similar to those of Theorem 6.1, we omit the details. □

This theorem and the previous corollary will be used in an induction argument over the decomposition into bindings of idempotent unifiers proving preservation of T1 and T2 (Theorem 6.6).

Lemma 6.5 Let s and t be two variable disjoint basic paths. Let σ be any A1-unifier computed by the system of Figure 5. Then

1. σ is an idempotent unifier.

2. $\sigma = \sigma_1 \dots \sigma_l$, where the σ_i are of the form $\{\alpha_i \mapsto w\}$ such that for any pair σ_i and σ_j with $1 \leq i < j \leq n$, if α_i and α_j occur at positions k_i and k_j in s or t, (that is, $s|_{k_i} = \alpha_i$ or $t|_{k_i} = \alpha_i$, and $s|_{k_j} = \alpha_j$ or $t|_{k_j} = \alpha_j$) then $k_i \leq k_j$.

3. If α_1 of σ_1 is a variable occurring in s then the following are equivalent.

   (a) $s = [u_1 \dots u_m]$ and $t = [v_1 \dots v_n]$ have a common prefix $[u_1 \dots u_{k+1}]$, $u_k \neq v_k$ and u_k is a variable.

   (b) Either $\sigma_1 = \{u_k \mapsto \underline{e}\}$ or $\sigma_1 = \{u_k \mapsto w\}$ where $N_{\mathrm{A}}([w]) = [w_1 \dots w_l]$ and $w_1 = v_k$.

4. $N_{\mathrm{A}}([w])$ satisfies T1 and T2 provided s and t do.

5. $\{N_{\mathrm{A}}([w]), [v_k \dots v_n]\}$ satisfies T1 and T2 provided s and t do.

Proof. 1. is evident by the definition of solved forms.

Consequently, σ coincides with a composition of bindings σ_i. Compose the bindings as determined by the positions in s and t of the variables α_i. This verifies 2.

3. is true for otherwise s and t are not unifiable.

4. No unification rule duplicates variables, hence $N_{\mathrm{A}}([w])$ is a linear term and satisfies T1 and T2.

5. A1-bindings of the form $\alpha_i \mapsto w$ are such that: $N_{\mathrm{A}}([w])$ has length smaller or equal to $[v_k \dots v_n]$, and $w_1 = v_k$, $w_2 = v_{k+1}$, ... , $w_{l-1} = v_{k+l-2}$. If $w_l = v_{k+l-1}$ then $N_{\mathrm{A}}([w])$ is a prefix of $[v_k \dots v_n]$, which means that $\{N_{\mathrm{A}}([w]), [v_k \dots v_n]\}$ satisfies T1 and T2. If $w_l \neq v_{k+l-1}$ then w_l is a variable introduced by an application of Mutate-A, v_{k+l-1} is a variable and $\sigma_2 = \{v_{k+l-1} \mapsto w'\}$. Also, in this case $\{N_{\mathrm{A}}([w]), [v_k \dots v_n]\}$ satisfies T1 and T2. □

Theorem 6.6 Let σ be an A1-unifier of two variable disjoint terms s and t in T. If T satisfies properties T1 and T2 then so does $N_{\mathrm{A1}}(T\sigma)$.

Proof. The proof is by an induction argument over the decomposition into bindings of idempotent unifiers. Let σ be $\sigma_1 \dots \sigma_l$ as in 2. of the previous lemma. Iteratively, consider the triples s, t and σ_1, then $N_{\mathrm{A1}}(s\sigma_1)$, $N_{\mathrm{A1}}(t\sigma_1)$ and σ_2, etcetera, and apply Corollary 6.3 and Theorem 6.4. By 3., 4. and 5. of the previous lemma, in any iteration the conditions T1 and T2 are satisfied by any $N_{\mathrm{A1}}(s\sigma_1 \dots \sigma_i)$, $N_{\mathrm{A1}}(t\sigma_1 \dots \sigma_i)$ and σ_{i+1}. □

Consequently, as the union of two variable disjoint sets of prefix stable terms is prefix stable, the preservation result for binary $\mathcal{R}^{\mathrm{A1}}_{N_{\mathrm{A1}}}$-resolvents follows. More generally, specialisation to just (1) or (A) renders:

Theorem 6.7 For $E \subseteq \{\mathrm{A}, 1\}$, the E-normal form of an E-resolvent of two variable disjoint clauses satisfying T1 and T2 also satisfies T1 and T2.

The main preservation theorem follows:

Theorem 6.8 Let S be a finite set of basic path clauses. Then $(\mathcal{R}^{E}_{N_E})^n(S)$ and $(\mathcal{R}^{E}_{\mathrm{COND} \,\circ\, N_E})^n(S)$, for any n, are well-formed in the basic path logic, when $E \subseteq \{\mathrm{A}, 1\}$.

Proof. By the previous theorem and by preservation of prefix stability under syntactic factoring and condensing. □

7 Conclusion

In summary, we have discussed issues concerning unification and normalisation of E-resolution for certain path logics, namely, those closed under right identity and associativity, or both. We have defined complete (and terminating) unification algorithms employing mutation rules. We have shown the search spaces are considerably smaller than those of Ohlbach's procedure. And, we have proved syntactic unification can be simplified for singleton problems.

We conclude with some remarks concerning further work.

Due to the assumption we make in Section 5, in particular, that the input set consists of one pair of terms, our resolution calculi are defined by binary E-resolution and syntactic factoring. For semantic factoring we need general E-unification for which our algorithm is not sufficient (this would require a deletion rule of the identity constant and a more general form of the variable elimination rule). Given a set of terms (literals), computing the syntactic most general unifier (when it exists) is easier than computing the set of minimal E-unifiers. Semantic factoring can produce an exponential number of factors causing a significant overhead. The price we pay for using syntactic factoring is incompatibility with strategies like tautology deletion. So, evidently there is a tradeoff which should be kept in mind and deserves further investigation.

The unification algorithm presented in this paper is not optimal. It does not compute a minimal complete set of E-unifiers. The redundant unifiers will need to be filtered out by post processing. Possibly this can be avoided by additional unification rules similar to those of Otten and Kreitz (1996) who present a system consisting of ten rules for terms satisfying the stronger T-string property.

Unification for other path theories has not been examined. Ohlbach (1988, 1991) considers unification for the modal schema B in the non-optimised context. In our context using the global form of the correspondence property of B is not sound and we are forced to use the local form, namely $[x\alpha\, i(x,\alpha)] = x$. Unification by mutation rules will not do in this case. For example, the solution $\{\gamma =^? i([s\alpha],\beta),\, \delta =^? i([s],\alpha)\}$ of the problem $\{[s\alpha\beta\gamma\delta] =^? s\}$ can only be derived by paramodulating into the left term, at a position not at the top.

As many other path theories (not considered here) are collapse-free, the results of Kirchner and Klay (1990) and also Doggaz and Kirchner (1991), which are about collapse-free syntactic theories, may be of value for developing terminating (mutation) unification algorithms. The latter paper presents a completion algorithm for automatically converting a presentation of linear and collapse-free equations to a finite resolvent set of equations.

References

Arnborg, S. and Tidén, E. (1985), Unification problems with one-sided distributivity, *in* J.-P. Jouannaud (ed.), *Proceedings of the 1st International Conference on Rewriting Techniques and Applications*, Vol. 202 of *Lecture Notes in Computer Science*, Springer, pp. 398–406.

Auffray, Y. and Enjalbert, P. (1992), Modal theorem proving: An equational viewpoint, *Journal of Logic and Computation* **2**(3), 247–297.

Comon, H., Haberstrau, M. and Jouannaud, J.-P. (1994), Syntacticness, cycle-syntacticness and shallow theories, *Information and Computation* **111**(1), 154–191.

Doggaz, N. and Kirchner, C. (1991), Completion for unification, *Theoretical Computer Science* **85**, 231–251.

Fari nas del Cerro, L. and Herzig, A. (1989), Automated quantified modal logic, *in* P. B. Brazdil and K. Konolige (eds), *Machine Learning, Meta-Reasoning and Logics*, Kluwer, pp. 301–317.

Fariñas del Cerro, L. and Herzig, A. (1995), Modal deduction with applications in epistemic and temporal logics, *in* D. M. Gabbay, C. J. Hogger and J. A. Robinson (eds), *Handbook of Logic in Artificial Intelligence and Logic Programming: Epistemic and Temporal Reasoning*, Vol. 4, Clarendon Press, Oxford, pp. 499–594.

Herzig, A. (1989), *Raisonnement automatique en logique modale et algorithmes d'unification.*, PhD thesis, Univ. Paul-Sabatier, Toulouse.

Jaffar, J. (1990), Minimal and complete word unification, *Journal of the ACM* **37**(1), 47–85.

Jouannaud, J.-P. and Kirchner, C. (1991), Solving equations in abstract algebras: A rule-based survey of unification, *in* J.-L. Lassez and G. Plotkin (eds), *Computational Logic: Essays in Honor of Alan Robinson*, MIT-Press, pp. 257–321.

Joyner Jr., W. H. (1976), Resolution strategies as decision procedures, *Journal of the ACM* **23**(3), 398–417.

Kirchner, C. and Klay, F. (1990), Syntactic theories and unification, *in* J. C. Mitchell (ed.), *Proceedings of the 5th Annual IEEE Symposium on Logic in Computer Science*, IEEE Computer Society Press, Philadelphia, pp. 270–277.

Klay, F. (1991), Undecidability properties of syntactic theories, *in* R. V. Book (ed.), *Rewriting Techniques and Applications: 4th International Conference, RTA'91, Proceedings*, Vol. 488 of *Lecture Notes in Computer Science*, Springer, Como, Italy, pp. 136–149.

Makanin, G. S. (1977), The problem of solvability of equations in a free semigroup, *Math. USSR Sbornik* **32**(2), 129–198.

Ohlbach, H. J. (1988), *A Resolution Calculus for Modal Logics*, PhD thesis, Univ. Kaiserslautern, Germany.

Ohlbach, H. J. (1991), Semantics based translation methods for modal logics, *Journal of Logic and Computation* **1**(5), 691–746.

Ohlbach, H. J. and Schmidt, R. A. (1997), Functional translation and second-order frame properties of modal logics, *Journal of Logic and Computation* **7**(5), 581–603.

Otten, J. and Kreitz, C. (1996), T-string unification: Unifying prefixes in non-classical proof methods, *in* P. Miglioli, U. Moscato, D. Mundici and M. Ornaghi (eds), *Proceedings of the 5th Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, Vol. 1071 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 244–260.

Plotkin, G. (1972), Building-in equational theories, *in* B. Meltzer and D. Michie (eds), *Machine Intelligence 7*, American Elsevier, New York, pp. 73–90.

Schmidt, R. A. (1997), *Optimised Modal Translation and Resolution*, PhD thesis, Universität des Saarlandes, Saarbrücken, Germany.

Schmidt, R. A. (1998), E-unification for subsystems of S4, *in* T. Nipkow (ed.), *Rewriting Techniques and Applications: 9th International Conference, RTA'98, Proceedings*, Vol. 1379 of *Lecture Notes in Computer Science*, Tsukuba, Japan, pp. 106–120.

Schulz, K. U. (1992), Makanin's algorithm for word equations: Two improvements and a generalization, *in* K. U. Schulz (ed.), *Word Equations and Related Topics (Proceedings of IWWERT'90)*, Vol. 572 of *Lecture Notes in Computer Science*, Springer, pp. 85–150.

Zamov, N. K. (1989), Modal resolutions, *Soviet Mathematics* **33**(9), 22–29. Translated from *Izv. Vyssh. Uchebn. Zaved. Mat.* **9** (328) (1989) 22–29.

I N F O R M A T I K

Below you find a list of the most recent technical reports of the research group *Logic of Programming* at the Max-Planck-Institut für Informatik. They are available by anonymous ftp from our ftp server `ftp.mpi-sb.mpg.de` under the directory `pub/papers/reports`. Most of the reports are also accessible via WWW using the URL `http://www.mpi-sb.mpg.de`. If you have any questions concerning ftp or WWW access, please contact `reports@mpi-sb.mpg.de`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik
Library
attn. Birgit Hofmann
Im Stadtwald
D-66123 Saarbrücken
GERMANY
e-mail: `library@mpi-sb.mpg.de`

MPI-I-98-2-007	S. Vorobyov	The Most Nonelementary Theory (A Direct Lower Bound Proof)
MPI-I-98-2-006	P. Blackburn, M. Tzakova	Hybrid Languages and Temporal Logic
MPI-I-98-2-005	M. Veanes	The Relation Between Second-Order Unification and Simultaneous Rigid *E*-Unification
MPI-I-98-2-004	S. Vorobyov	Satisfiability of Functional+Record Subtype Constraints is NP-Hard
MPI-I-97-2-012	L. Bachmair, H. Ganzinger, A. Voronkov	Elimination of Equality via Transformation with Ordering Constraints
MPI-I-97-2-011	L. Bachmair, H. Ganzinger	Strict Basic Superposition and Chaining
MPI-I-97-2-010	S. Vorobyov, A. Voronkov	Complexity of Nonrecursive Logic Programs with Complex Values
MPI-I-97-2-009	A. Bockmayr, F. Eisenbrand	On the Chvátal Rank of Polytopes in the 0/1 Cube
MPI-I-97-2-008	A. Bockmayr, T. Kasper	A Unifying Framework for Integer and Finite Domain Constraint Programming
MPI-I-97-2-007	P. Blackburn, M. Tzakova	Two Hybrid Logics
MPI-I-97-2-006	S. Vorobyov	Third-order matching in $\lambda \rightarrow$-Curry is undecidable
MPI-I-97-2-005	L. Bachmair, H. Ganzinger	A Theory of Resolution
MPI-I-97-2-004	W. Charatonik, A. Podelski	Solving set constraints for greatest models
MPI-I-97-2-003	U. Hustadt, R.A. Schmidt	On evaluating decision procedures for modal logic
MPI-I-97-2-002	R.A. Schmidt	Resolution is a decision procedure for many propositional modal logics
MPI-I-97-2-001	D.A. Basin, S. Matthews, L. Viganò	Labelled modal logics: quantifiers
MPI-I-96-2-010	A. Nonnengart	Strong Skolemization
MPI-I-96-2-009	D.A. Basin, N. Klarlund	Beyond the Finite in Automatic Hardware Verification
MPI-I-96-2-008	S. Vorobyov	On the decision complexity of the bounded theories of trees