# *E*-Unification for Subsystems of *S4*

Renate A. Schmidt*

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

**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.

## 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 functions in input clauses are nullary.

* I thank H. Ganzinger, A. Herzig, U. Hustadt, A. Nonnengart, H. J. Ohlbach and especially the anonymous referees for their comments. This research was conducted while I was employed at the Max-Planck-Institut für Informatik in Saarbrücken, Germany, and was funded by the MPG and the DFG through the TRALOS-Project.

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 semi-groups or monoids, for example. Related unification algorithms and resolution calculi found in the literature differ from ours in at least three respects. One, they are all designed for the non-optimised translations which require extended (strong) forms of Skolemisation in order that a particular ordering within terms is preserved. Accordingly, our unification algorithms are more elegant and the proofs are considerably simpler, though remaining technical. Two, most of the systems are incomplete. Three, our exposition pays special attention 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 Sect. 4 we discuss $E$-unification for the schemas $T$ and 4 reviewing what is known from the literature. The main parts are Sects. 5 and 6. Section 5 presents a mutation algorithm for the combination of $T$ and 4 illustrating the computational gain and outlining 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. Due to space restrictions most proofs are omitted, but can be found in Schmidt (1998).

## 2 Path logics

The *basic path logic* is a clausal logic defined over a language with two principal sorts: the sort $W$ and the sort $AF$. Expressions of the sort $W$ and $AF$ are called 'world' and 'functional expressions'. 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 one binary function $[\cdot, \cdot] : W \times AF \longrightarrow W$. Terms in the basic path logic have the form $t = [[[[[]u_1]u_2] \ldots]u_m]$, or in shorthand notation $t = [u_1 u_2 \ldots u_m]$, with any $u_i$ denoting a functional term (by which we mean a term of sort $AF$). A term defines a set of paths in the functional semantics of modal logics. $[]$ is the initial world and the term $[\alpha\underline{\beta}]$, for example, defines the set of worlds reached from $[]$ via any $\alpha$-step followed by some $\underline{\beta}$-step. 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 $\alpha$ 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 $\psi$ 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.

$$\begin{aligned} &\textit{right identity}: \quad [x\underline{e}] = x && (1)\\ &\textit{associativity}: \quad [x(\alpha \circ \beta)] = [x\alpha\beta] && (A)\end{aligned}$$

The symbol $x$ denotes a variable of sort world and $[xu_1u_2]$ is an abbreviation for $[[xu_1]u_2]$. The symbol $\underline{e}$ (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 $T$ and 4, respectively. Recall, the relational correspondence properties of $T$ and 4 are reflexivity and transitivity, respectively. *S4* is the logic *KT4* (or *KDT4*).

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. 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 $u, u_1, u_2, \ldots$ and $v, v_1, v_2, \ldots$ are reserved for terms of sort $AF$. The symbols $s, t, \ldots$ are reserved for any world terms. Strictly, the term $[s\,t]$ is malformed since $[\cdot,\cdot]$ is assumed to be left-associative, but when we write $[s\,t]$ we mean the term $[s\,u_1 \ldots u_i]$ provided $t = [u_1 \ldots u_i]$. Given a term $s = [u_1 \ldots u_m]$, define $s|_i$ by $s|_0 = []$ and $s|_i = u_i$ for any $0 < i \leq m$. If each subterm $u_i$ of $s$ is either a variable or a constant then $s$ is called a *basic path*.

## 3 Unification for the basic path logic

Because the basic path language has no compound functional terms, any non-empty substitution $\sigma$ 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 Fig. 1 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.

$$
\begin{array}{llll}
\textbf{Delete} & P \cup \{s =^? s\} & \rightsquigarrow P & \text{(for world terms)} \\
& P \cup \{u =^? u\} & \rightsquigarrow P & \text{(for terms of sort } AF) \\
\textbf{Decompose} & P \cup \{[su] =^? [tv]\} & \rightsquigarrow P \cup \{u =^? v,\ s =^? t\} & \\
\textbf{Conflict} & P \cup \{[su] =^? []\} & \rightsquigarrow \bot & \\
& P \cup \{\underline{\alpha} =^? \underline{\beta}\} & \rightsquigarrow \bot & \text{provided } \underline{\alpha} \neq \underline{\beta} \\
\textbf{Coalesce} & P \cup \{\alpha =^? \beta\} & \rightsquigarrow P\{\alpha \mapsto \beta\} \cup \{\alpha =^? \beta\} & \\
& \multicolumn{3}{l}{\text{provided } \alpha \neq \beta \text{ are variables both occurring in } P.} \\
\textbf{Eliminate} & P \cup \{\alpha =^? \underline{\beta}\} & \rightsquigarrow P\{\alpha \mapsto \underline{\beta}\} \cup \{\alpha =^? \underline{\beta}\} & \\
& \multicolumn{3}{l}{\text{provided } \alpha \text{ is a variable occurring in } P \text{ and } \underline{\beta} \text{ is constant.}}
\end{array}
$$

**Fig. 1.** Syntactic unification rules for the basic path logic

$\bot$ 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 $[u_1 \dots u_m]$ is an abbreviation for a nested term $[[[[[]u_1]u_2] \dots]u_m]$. This determines how paths are decomposed, namely from right to left. Evidently, any most general unifier of a unification problem over basic paths obtained by the rules of Fig. 1 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.

For singleton problem sets two rules are redundant:

**Theorem 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.*

The situation is pleasantly simple for the logic *S5* (which coincides with *KDB4*, *KTB4* and *KT5*). 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 $[u_1 \dots u_m] =^? [v_1 \dots v_n]$ 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 Fig. 1.

## 4 Unification for (1) and (A)

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

Unification of paths 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 guaranteed to terminate. There are decision procedures by Makanin (1977) and Jaffar (1990), for example, but these are far too complex for our purposes. 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. (By exploiting the correspondence to regular expressions and using methods from automata theory, we expect that unifiability under (1) and/or (A) can be decided in polynomial time.)

| | | |
|---|---|---|
| **Delete** | $P \cup \{s =^? s\}$ | $\rightsquigarrow P$ |
| **Decompose** | $P \cup \{s * s' =^? t * t'\}$ | $\rightsquigarrow P \cup \{s =^? t,\ s' =^? t'\}$ |
| | when $s$ and $t$ are non-empty strings | |
| **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 $\beta_1$, $\beta_2$ are new variables. | |

**Fig. 2.** Ohlbach's unification rules for (1) and (A)

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 rules from Ohlbach (1991) relevant for propositional *S4*, listed in Fig. 2, form a complete system. The language is a variation from ours. Terms are strings 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) strings.

By way of an example we will demonstrate the system can be improved. Fig. 3 sketches a derivation of A1-unifiers for $\{\alpha * \underline{\beta} * \underline{\gamma} * \delta =^? \underline{\alpha} * \beta * \underline{\gamma}\}$.[1] Terms are decomposed from left to right. Failure branches are those that do not produce

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

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

**Fig. 3.** A sample derivation of A1-unifiers according to Ohlbach's algorithm

solved forms, that is, sets of the form $\{\alpha_1 =^? u_1, \ldots, \alpha_n =^? u_n\}$ with each $\alpha_i$ 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. $\{\alpha \mapsto \underline{\alpha},\ \beta \mapsto \underline{\beta} * \underline{\gamma} * \delta_1,\ \delta \mapsto \delta_1 * \underline{\gamma}\}$

etc.

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. Kirchner and Klay (1990) prove mutation rules are complete for syntactic collapse-free theories. Comon, Haberstrau and Jouannaud (1994) consider mutation with (and without) collapsing equations for shallow theories and prove any shallow theory is syntactic. 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 for ordinary associativity. However, 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)

In this section we present a unification system with mutation rules for (1) and (A). This system and its appropriate subsystems are to be used in resolution calculi, denoted by $\mathcal{R}^E_{N_E}$ and $\mathcal{R}^E_{\mathrm{COND} \circ N_E}$, defined by binary $E$-resolution, syntactic factoring and normalisation modulo $E$ and possibly condensing. Normalisation and condensing are applied eagerly.

The normalising functions $N_1$ and $N_{\mathrm{A}}$ rearrange terms according to the rewrite rules $[x\underline{e}] \to x$ and $[x(\alpha \circ \beta)] \to [x\alpha\beta]$. Inductive specifications of $N_1$ and $N_{\mathrm{A}}$ 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. Any term $N_E(s)$ is said to be in *$E$-normal form.*

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.

| | | | |
|---|---|---|---|
| **Delete** | $P \cup \{s =^? s\}$ | $\rightsquigarrow$ | $P$ |
| **Variable Eliminate** | $P \cup \{\alpha =^? u\}$ | $\rightsquigarrow$ | $P\{\alpha \mapsto u\}$ |
| | when $\alpha$ 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 $\alpha'$ is a new variable and not both $s$ and $t$ are empty. | | |

**Fig. 4.** 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 Fig. 4. $s$ and $t$ may denote empty paths, except where stated otherwise.

Observe, the variable elimination rule applies only to introduced variables, by which we mean the variables not present in the original problem set. 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 the mutation rules of Comon et al. (1994).[2]

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: $\{\alpha \mapsto \gamma,\ \beta \mapsto \underline{e}\}$ 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 $s = [u_1 \ldots u_{m+1}] =^? [v_1 \ldots v_{n+1}] = t$ 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.

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

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 semigroups (applied to paths and employing right-to-left as opposed to left-to-right decomposition).

Compare the derivation in Fig. 3 according to Ohlbach's method with the derivation in Fig. 5 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:

[2] For readers familiar with this paper we note, in our context their cycle breaking rule can be easily 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.

$$
\begin{array}{l}
[\alpha\underline{\beta}\underline{\gamma}\delta] =^? [\underline{\alpha}\beta\underline{\gamma}] \\
\overset{\text{Dec}}{\rightsquigarrow} \delta =^? \underline{\gamma},\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}\beta] \\
\qquad\qquad \left.\begin{array}{l}
\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'] \\
\qquad\qquad \overset{2\times\text{Dec,Del}}{\rightsquigarrow} \quad \beta' =^? \underline{\beta},\ \alpha =^? \underline{\alpha} \ \rightsquigarrow\ 1. \\
\qquad\qquad \overset{1}{\rightsquigarrow} \quad \beta' =^? \underline{e},\ [\alpha\underline{\beta}] =^? [\underline{\alpha}] \quad \text{not solved} \\
\qquad\qquad \overset{2\times A,\text{Dec,Elim}}{\rightsquigarrow} \beta' =^? \beta'' \circ \underline{\beta},\ \alpha =^? \underline{\alpha} \circ \beta'' \ \rightsquigarrow\ 2.
\end{array}\right\} ('') \\
\overset{1}{\rightsquigarrow} \delta =^? \underline{e},\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}\beta\underline{\gamma}] \ \overset{\text{Dec,Del}}{\rightsquigarrow}\ [\alpha\underline{\beta}] =^? [\underline{\alpha}\beta] \\
\qquad\qquad\qquad\qquad \overset{2\times\text{Dec}}{\rightsquigarrow} \quad \beta =^? \underline{\beta},\ \alpha =^? \underline{\alpha} \ \rightsquigarrow\ 3. \\
\qquad\qquad\qquad\qquad \overset{1}{\rightsquigarrow} \quad \beta =^? \underline{e},\ [\alpha\underline{\beta}] =^? [\underline{\alpha}] \quad \text{not solved} \\
\qquad\qquad\qquad\qquad \overset{2\times A,\text{Dec,Elim}}{\rightsquigarrow} \beta =^? \beta' \circ \underline{\beta},\ \alpha =^? \underline{\alpha} \circ \beta' \ \rightsquigarrow\ 4. \\
\overset{\text{A}}{\rightsquigarrow} \delta =^? \delta' \circ \underline{\gamma},\ [\alpha\underline{\beta}\underline{\gamma}\delta'] =^? [\underline{\alpha}\beta] \\
\qquad\qquad \overset{\text{Dec}}{\rightsquigarrow} \beta =^? \delta',\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}] \quad \text{not solved} \\
\qquad\qquad \overset{1}{\rightsquigarrow} \delta' =^? \underline{e},\ [\alpha\underline{\beta}\underline{\gamma} =^? [\underline{\alpha}\beta] \ \rightsquigarrow\ \ldots \quad \text{not solved} \\
\qquad\qquad \overset{1}{\rightsquigarrow} \beta =^? \underline{e},\ \ldots \quad \text{not solved} \\
\qquad\qquad \overset{\text{A}}{\rightsquigarrow} \beta =^? \beta' \circ \delta',\ [\alpha\underline{\beta}\underline{\gamma}] =^? [\underline{\alpha}\beta] \ \rightsquigarrow\ \ldots \text{ like } ('') \ \rightsquigarrow\ 5. \text{ and } 6.
\end{array}
$$

**Fig. 5.** A sample derivation of A1-unifiers

*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 $\theta$ of $P$, there is some $P'$ in solved form such that $P \overset{*}{\rightsquigarrow} P'$ and the idempotent unifier $\sigma$ associated with $P'$ is more general than $\theta$ 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 $\alpha$ is *solved* in a set $P$ if $P$ includes a pair $\alpha =^? u$ (or $u =^? \alpha$) and $\alpha$ 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 $=_{\text{A1}}$ ($\neq_{\text{A1}}$ and $\leq_{\text{A1}}$). The following is also easy to verify by inspecting the rules.

**Theorem 2.** *The system of Fig. 4 is sound.*

In the remainder of the 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 Fig. 4. For the next lemmas it is important that the initial pair in $P$ is variable disjoint.

**Lemma 1.** *For any identity $\alpha =^? u$ in $P'$, the variable $\alpha$ does not occur in $u$.*

**Lemma 2.** *Each $P'$ irreducible by the rules of Fig. 4 is in solved form or it is unsatisfiable in $=_{A1}$.*

We now sketch the proof of completeness.

**Theorem 3.** *The system of Fig. 4 is complete.*

The core structure of the proof is standard. We let

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

with $s = [s|_1 \ldots s|_m]$ and $t = [t|_1 \ldots t|_n]$, each non-empty, that is, $1 \leq m, n$. We let $\theta$ 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 a solved $P'$ such that the associated unifier $\sigma$ is more general than $\theta$. In parallel to transforming $P$ we extend the unifier $\theta$ by adding bindings of new variables to $\theta$ obtaining $\theta'$. Below, in Lemmas 6 and 7, we will define $\theta'$ in such a way that if $\theta$ unifies $P$ and $P \rightsquigarrow P'$, that is, if $P$ transforms to $P'$ in one step, then $\theta'$ unifies $P'$. The resulting procedure starts with the pair $(P, \theta)$ and computes at least one pair $(P', \theta')$, such that 1. $P \overset{*}{\rightsquigarrow} P'$, 2. $P'$ is in solved form, 3. $\theta \subseteq \theta'$ and the restriction of $\theta'$ to the variables of $\theta$ coincides with $\theta$. By assumption $\theta$ is a unifier of $P$, and consequently, by induction on the proof length, $\theta'$ 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 3.** *The unifier $\sigma$ associated with the solved $P'$ is more general than $\theta'$ with respect to the variables of $P'$.*

**Lemma 4.** *The unifier $\sigma$ associated with the solved $P'$ is more general than $\theta$ with respect to the variables of $P'$ and $P$.*

**Lemma 5.** *Any fair implementation of a unification algorithm for the transformation system of Fig. 4 terminates for paths.*

*Proof.* Let $\tau(s)$ denote the functional depth of a term $s$. Define a measure $\mu$ of any unification problem $P$ by $\mu(P) = (d, v)$, where $v$ denotes the number of unsolved variables in $P$, and $d = \tau(s) + \tau(t)$ for $s =^? t \in P$ and both $s$ and $t$ are of type world. Examine each transformation rule in turn to see that $\mu(P')$ is smaller than $\mu(P)$ under the lexicographical ordering. The rules do not convert the status of any variable from solved to unsolved. $\square$

The following two lemmas are concerned with the one step conversions of any pair $(P, \theta)$ to a suitable pair $(P', \theta')$.

**Lemma 6.** *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 $\theta$ be any A1-unifier of $P \cup \{[su] =^? [tv]\}$, in particular, $[su]\theta =_{A1} [tv]\theta$. $\theta'$, as defined in the following, is in each case an A1-unifier of $P'$.*

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

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

2. *If $u\theta =_{A1} \underline{e}$, 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 $\alpha$, say, and $u\theta =_{A1} [t|_k \dots t|_n v]\theta$ for some $1 \le k \le n$, then let $\theta' = \theta\theta_0$ with $\theta_0 = \{\alpha' \mapsto [t|_k \dots t|_n]\}$ and apply Mutate-A to $P$, yielding*

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

*for $\alpha'$ a new variable not occurring in $P$ or $\theta$.*

The lemma also covers the cases that $v\theta =_{A1} \underline{e}$ and $v\theta =_{A1} [s|_k \dots s|_n]\theta$ for some $1 \le k \le m$ and $v$ a variable. Observe that when $u\theta =_{A1} [t|_k \dots t|_n v]\theta$ 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 that the lemma does not cover. The answer is, yes, as in this example

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

when in the general case $[s|_k \dots s|_m u]\theta =_{A1} [t|_l \dots t|_n v]\theta$ 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 7.** *Let $\theta$ 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 $[s|_k \dots s|_m \alpha]\theta =_{A1} [t|_l \dots t|_n v]\theta$ for some $1 \le k \le m$ and $1 \le l \le n$. If $\theta$ includes a binding of $\alpha$ to $u$, that is, $\alpha \mapsto u \in \theta$, then let*

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

*where $u'$ is given by $u =_A u' \circ v'$ and $v' = v\theta$, and apply Mutate-A to $P$, yielding*

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

*for $\alpha'$ a new variable not occurring in $P$ or $\theta$. Then $\theta'$ unifies $P'$.*

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

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

The lemma makes assumptions, which are not met in the following two situations. First, if no $u'$ exists such that $u =_A u' \circ (v\theta)$ then $v\theta$ is equivalent to $\underline{e}$. This case is dealt with in 2. of the previous lemma. Second, the situation that neither $\alpha$ nor $v$ are in the domain of $\theta$ and $\alpha\theta \ne_{A1} v\theta$ 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 $N_{\mathrm{A1}}$ 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 Fig. 4 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. We start by considering the preservation of prefix stability under syntactic bindings.

**Theorem 4.** *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 $\sigma$ be the substitution $\{u_k \mapsto v_k\}$. Then $T\sigma$ satisfies T1 and T2, provided $T$ does.*

Based on this result it is not difficult to prove that prefix stability is preserved under syntactic factoring and ordinary resolution. 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.

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 $N_{\mathrm{A1}}$. The analogue of Theorem 4 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 4.

**Corollary 1.** *Let $T$ be a set of terms in the vocabulary of 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 $k > 0$,*

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

*$u_k$ is a variable, and the suffixes $[u_{k+1} \ldots u_m]$ and $[v_{k+1} \ldots v_n]$ are variable disjoint. Let $\sigma$ 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 we need 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 string $w_1 \ldots w_l$ of length $l$. In other words, $s$ and $t$ are $(k, l)$-equal provided $s = t$, or when $s = [s|_1 \ldots s|_m]$ then $t = [s|_1 \ldots s|_{k-1} w_1 \ldots w_l s|_{k+1} \ldots s|_m]$, or the other way around.

**Theorem 5.** *Let $s$ and $t$ be two terms in $T$ defined as in the previous result. Let $\sigma$ be a substitution $\{u_k \mapsto w\}$ where*

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

*and $s$ and $w$ are variable disjoint.*[3] *Then $N_{A1}(T\sigma)$ satisfies T1 and T2, provided $T$, $N_{A1}([w])$ and the set $\{[w], [v_k \dots v_n]\}$ do.*

*Proof.* 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 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. $\sigma$ 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 $k + l$. As this is tedious we omit the details. □

**Lemma 8.** *Let $s$ and $t$ be two variable disjoint basic paths. Let $\sigma$ be any A1-unifier computed by the system of Fig. 4. Then*

1. *$\sigma$ is an idempotent unifier.*
2. *$\sigma = \sigma_1 \dots \sigma_l$, where the $\sigma_i$ are of the form $\{\alpha_i \mapsto w\}$ such that for any pair $\sigma_i$ and $\sigma_j$ with $1 \le i < j \le n$, if $\alpha_i$ and $\alpha_j$ occur at positions $k_i$ and $k_j$ in $s$ or $t$, then $k_i \le k_j$.*
3. *If $\alpha_1$ of $\sigma_1$ 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 $[u_1 \dots u_{k+1}]$, $u_k \ne v_k$ and $u_k$ is a variable.*
   (b) *Either $\sigma_1 = \{u_k \mapsto \underline{e}\}$ or $\sigma_1 = \{u_k \mapsto w\}$ where $N_A([w]) = [w_1 \dots w_l]$ and $w_1 = v_k$.*
4. *$N_A([w])$ satisfies T1 and T2 provided $s$ and $t$ do.*
5. *$\{N_A([w]), [v_k \dots v_n]\}$ satisfies T1 and T2 provided $s$ and $t$ do.*

**Theorem 6.** *Let $\sigma$ be an A1-unifier of two variable disjoint terms $s$ and $t$ in $T$. If $T$ satisfies properties T1 and T2 then so does $N_{A1}(T\sigma)$.*

*Proof.* The proof is by an induction argument over the decomposition into bindings of idempotent unifiers. Let $\sigma$ be $\sigma_1 \dots \sigma_l$ as in 2. of the previous lemma. Iteratively, consider the triples $s$, $t$ and $\sigma_1$, then $N_{\mathrm{A1}}(s\sigma_1)$, $N_{\mathrm{A1}}(t\sigma_1)$ and $\sigma_2$, etcetera, and apply Corollary 1 and Theorem 5. 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 $\sigma_{i+1}$. □

---

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

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 7.** *For $E \subseteq \{A, 1\}$, the binary $\mathcal{R}^{E}_{N_E}$-resolvent of two variable disjoint clauses satisfying T1 and T2 also satisfies T1 and T2.*

The main preservation theorem follows:

**Theorem 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 $E \subseteq \{A, 1\}$.*

## 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 Sect. 5, in particular, that the input set consists of one variable disjoint 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.

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.), *Proc. Intern. Conf. on Rewriting Techniques and Applications*, Vol. 202 of *LNCS*, 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.

Fariñas 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*, 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, *J. 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.

Kirchner, C. and Klay, F. (1990), Syntactic theories and unification, *in* J. C. Mitchell (ed.), *Proc. LICS'90*, IEEE Computer Society Press, Philadelphia, pp. 270–277.

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.

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. (1998), $E$-unification for subsystems of S4, *Research Report MPI-I-98-2-003*, Max-Planck-Institut für Informatik, Saarbrücken, Germany.

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*, Vol. 572 of *LNCS*, 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.