

A Constructor-Based Reachability Logic for Rewrite Theories

Stephen Skeirik, Andei Stefanescu and José Meseguer

Department of Computer Science
University of Illinois at Urbana-Champaign, USA

Abstract. Reachability logic is a *language-generic* logic associated to a \mathbb{K} rewrite-rule-based language definition. It has been proved successful in verifying a wide range of sophisticated programs in conventional languages. Since \mathbb{K} language definitions are a special class of rewrite theories, studying how reachability logic can be best made not just language-generic, but *rewrite-theory-generic* can widen its applicability not just to conventional languages, but to verify rewriting-logic-based programs and distributed system designs. A simple theory-generic reachability logic is presented and proved sound for a wide class of rewrite theories. Particular attention is given to increasing the logic’s automation by means of *constructor-based* semantic unification, matching, and satisfiability procedures. Besides supporting a much wider range of applications than \mathbb{K} ’s reachability logic, new methods for proving invariants of possibly never terminating distributed systems are developed. Experiments with a prototype implementation illustrating the new methods are also presented.

Keywords: reachability and rewriting logics, program verification.

1 Introduction

Reachability Logic is a *language-generic* and *semantics-based* program verification logic that has been shown effective in proving a wide range of programs in real programming languages within the \mathbb{K} Framework [41, 46, 36]. It assumes a \mathbb{K} specification of a language’s operational semantics by means of rewrite rules as the language’s “golden semantic standard,” and then automatically obtains a correct-by-construction program verification logic for such a language.

Could a similar, semantics-based reachability logic be developed to verify not just programs, but also *distributed system designs and algorithms*? And if so, what would be the most natural way to conceive such a logic? Since: (i) rewriting logic [32, 33] is a simple, general framework to specify distributed systems; and (ii) the foundations of the \mathbb{K} framework are based on rewriting logic [34] and \mathbb{K} is itself one of the most important achievements of the “rewriting logic semantics project” [31, 34], \mathbb{K} specifications are a special class of rewrite theories specifying *language designs*. Therefore, “all” we need to do is to generalize reachability logic from a *language-generic* logic to a *rewrite-theory-generic* logic capable of reasoning about reachability properties of both distributed systems *and* programs specified by rewrite theories satisfying simple requirements.

A first step in this direction has been taken in [26], but, as we explain in Section 7 and below, that first step still leaves several important questions open. The most burning one is: how can we prove *invariants* of a distributed system in this general reachability logic? Since invariants are the most basic safety properties, being able to prove invariants is a *sine qua non* requirement. A second, important open question is how to best take advantage of the wealth of equational reasoning techniques such as matching, unification, and narrowing modulo an equational theory (Σ, E) , e.g., [44, 25, 3, 4, 23, 24, 16, 47], as well as recent results on decidable satisfiability (and validity) of quantifier-free formulas in initial algebras, e.g., [28, 9, 10, 2, 11, 35, 19, 20, 17, 1, 30] to *automate* as much as possible reachability logic deduction. A third important issue is *simplicity*. \mathbb{K} 's reachability logic has eight inference rules [46]. Could the new logic be not only more general but also simpler? The main goal of this work is to tackle head on these three open questions and to provide a proof of concept for a general reachability logic suitable for reasoning about properties of *both* distributed systems and programs based on their rewriting logic semantics. What this makes possible is using a very effective logic to eliminate *coding* errors to also eliminate much more expensive distributed system *design* errors earlier in the software development process. What all this really means requires some further explanations.

Rewriting Logic in a Nutshell. A distributed system can be modeled as a *rewrite theory* $\mathcal{R} = (\Sigma, E, R)$ [32, 33] in the following way: (i) the distributed system's *states* are modeled as elements of the initial algebra $T_{\Sigma/E}$ associated to the equational theory (Σ, E) with function symbols Σ and equations E ; and (ii) the system's *concurrent transitions* are modeled by the rewrite rules R , which are applied *modulo* E . Let us consider the QLOCK [18] mutual exclusion protocol, explained in detail in Section 2 and used later as a running example. QLOCK allows an unbounded number of processes, which can be identified by numbers. Such processes can be in one of three states: “normal” (doing their own thing), “waiting” for a resource, and “critical,” i.e., using the resource. Waiting processes enqueue their identifier at the end of a waiting queue (a list), and can become critical when their name appears at the head of the queue. A QLOCK state can be represented as a tuple $\langle n \mid w \mid c \mid q \rangle$ where n , resp. w , resp. c denotes the set of identifiers for normal, resp. waiting, resp. critical processes, and q is the waiting queue. QLOCK can be naturally modeled as a rewrite theory $\mathcal{R} = (\Sigma, E, R)$ where Σ contains operators to build natural numbers, multisets of natural numbers, like n , w , and c , and lists of natural numbers like q , plus the above tupling operator. The equations E include axioms such as the associativity-commutativity of multiset union, and the associativity of list concatenation, and identity axioms for \emptyset and *nil*. QLOCK's behavior is specified by a set R of five rewrite rules. For example, the rule $w2c$ below specifies how a waiting process i can pass from waiting to critical

$$w2c : \langle n \mid w i \mid c \mid i; q \rangle \rightarrow \langle n \mid w \mid c i \mid i; q \rangle .$$

Reachability Logic in a Nutshell. A reachability logic formula has the form $A \rightarrow^{\otimes} B$, where A and B are state predicates. Assume, for simplicity, that the

formula has no parameters, that is, that $\text{vars}(A) \cap \text{vars}(B) = \emptyset$. We interpret such formulas in the initial model $\mathcal{T}_{\mathcal{R}}$ of a rewrite theory $\mathcal{R} = (\Sigma, E, R)$, whose states are E -equivalence classes $[u]$ of ground Σ -terms, and where a state transition $[u] \rightarrow_{\mathcal{R}} [v]$ holds iff $\mathcal{R} \vdash u \rightarrow v$ according to the rewriting logic inference system [32, 33] (computation = deduction). As a first approximation, $A \rightarrow^{\otimes} B$ is a Hoare logic *partial correctness* assertion of the form $\{A\}\mathcal{R}\{B\}$, but with the slight twist that B need not hold of a terminating state, but just *somewhere along the way*. To be fully precise, $A \rightarrow^{\otimes} B$ holds in $\mathcal{T}_{\mathcal{R}}$ iff for each state $[u_0]$ satisfying A and each finite terminating sequence $[u_0] \rightarrow_{\mathcal{R}} [u_1] \dots \rightarrow_{\mathcal{R}} [u_{n-1}] \rightarrow_{\mathcal{R}} [u_n]$ there is a j , $0 \leq j \leq n$ such that $[u_j]$ satisfies B . A key question is how to choose a good language of state predicates like A and B . Here is where the potential for increasing the logic's automation resides. We call our proposed logic *constructor-based*, because our choice is to make A and B positive (only \vee and \wedge) combinations of what we call *constructor patterns* of the form $u \mid \varphi$, where u is a constructor term and φ a quantifier-free (QF) Σ -formula. The state predicate $u \mid \varphi$ holds for a state $[u']$ iff there is a ground substitution ρ such that $[u'] = [u\rho]$ and $E \models \varphi\rho$. But this leaves unexplained what a constructor is. Recall that the set of states of $\mathcal{T}_{\mathcal{R}}$ is the initial algebra $T_{\Sigma/E}$. Assume that the equations E decompose as a union $E = G \uplus B$, where B are axioms such as associativity and/or commutativity and/or identity, and the equations G can be oriented as rewrite rules \vec{G} that are confluent and terminating *modulo* B . A subsignature $\Omega \subseteq \Sigma$ is called a *constructor* subsignature iff the unique (up to B -equality) canonical form of each ground term t when rewritten with \vec{G} modulo B is an Ω -term. This is important because the algebra of Ω -terms in \vec{G}/B -canonical form is typically *much simpler* than $T_{\Sigma/E}$, and this can be systematically exploited for matching, unification, narrowing, and satisfiability purposes to automate large portions of reachability logic.

The Invariant Paradox. This is all very well, but how can we *prove invariants* in such a reachability logic? For example, we would like to prove that for QLOCK a mutual exclusion invariant holds. But, paradoxically, we cannot! The simple reason is that QLOCK, like many other protocols, *never terminates*, that is, has no terminating sequences whatsoever. But this has the ludicrous trivial consequence that QLOCK's initial model $\mathcal{T}_{\mathcal{R}}$ vacuously satisfies *all* reachability formulas $A \rightarrow^{\otimes} B$. This of course means that it is in fact *impossible* to prove any invariants using reachability logic in the initial model $\mathcal{T}_{\mathcal{R}}$. But it does *not* mean that it is impossible using some other initial model. In Section 4.1 we give a systematic solution to this paradox by means of a *simple theory transformation* allowing us to prove any invariant in the original initial model $\mathcal{T}_{\mathcal{R}}$ by proving an equivalent reachability formula in the initial model of the transformed theory.

Our Contributions. Section 2 gathers preliminaries. The main theoretical contributions of a *simple* semantics and inference system for a rewrite-theory-generic reachability logic with just *two* inference rules and its soundness are developed in Sections 4 and 5. A systematic methodology to prove *invariants* by means of reachability formulas is developed in Section 4.1. The goal of increasing as much as possible the logic's potential for automation by making it constructor-

based is advanced in Sections 3–5. A proof of concept of the entire approach is given by means of a Maude-based prototype tool and a suite of experiments involving distributed system designs and programs in Section 6. Related work and conclusions are discussed in Section 7. Proofs are relegated to Appendix A.

2 Order-Sorted Algebra and Rewriting Logic

We present some preliminaries on order-sorted algebra and rewriting logic. The material is adapted from [29, 30, 33]. The presentation is self-contained: we only assume the notions of many-sorted signature and many-sorted algebra, e.g., [15].

Definition 1. An order-sorted (OS) signature is a triple $\Sigma = (S, \leq, \Sigma)$ with (S, \leq) a poset and (S, Σ) a many-sorted signature. $\hat{S} = S/\equiv_{\leq}$, the quotient of S under the equivalence relation $\equiv_{\leq} = (\leq \cup \geq)^+$, is called the set of connected components of (S, \leq) . The order \leq and equivalence \equiv_{\leq} are extended to sequences of same length in the usual way, e.g., $s'_1 \dots s'_n \leq s_1 \dots s_n$ iff $s'_i \leq s_i$, $1 \leq i \leq n$. Σ is called sensible if for any two $f : w \rightarrow s, f : w' \rightarrow s' \in \Sigma$, with w and w' of same length, we have $w \equiv_{\leq} w' \Rightarrow s \equiv_{\leq} s'$. A many-sorted signature Σ is the special case where the poset (S, \leq) is discrete, i.e., $s \leq s'$ iff $s = s'$.

For connected components $[s_1], \dots, [s_n], [s] \in \hat{S}$

$$f_{[s]}^{[s_1] \dots [s_n]} = \{f : s'_1 \dots s'_n \rightarrow s' \in \Sigma \mid s'_i \in [s_i], 1 \leq i \leq n, s' \in [s]\}$$

denotes the family of “subsort polymorphic” operators f . \square

Definition 2. For $\Sigma = (S, \leq, \Sigma)$ an OS signature, an order-sorted Σ -algebra A is a many-sorted (S, Σ) -algebra A such that:

- whenever $s \leq s'$, then we have $A_s \subseteq A_{s'}$, and
- whenever $f : w \rightarrow s, f : w' \rightarrow s' \in f_{[s]}^{[s_1] \dots [s_n]}$ and $\bar{a} \in A^w \cap A^{w'}$, then we have $A_{f:w \rightarrow s}(\bar{a}) = A_{f:w' \rightarrow s'}(\bar{a})$, where $A^{s_1 \dots s_n} = A_{s_1} \times \dots \times A_{s_n}$.

An order-sorted Σ -homomorphism $h : A \rightarrow B$ is a many-sorted (S, Σ) -homomorphism such that whenever $[s] = [s']$ and $a \in A_s \cap A_{s'}$, then we have $h_s(a) = h_{s'}(a)$. This defines a category \mathbf{OSAlg}_{Σ} . \square

Theorem 1. [29] The category \mathbf{OSAlg}_{Σ} has an initial algebra. Furthermore, if Σ is sensible, then the term algebra T_{Σ} with:

- if $a : \epsilon \rightarrow s$ then $a \in T_{\Sigma, s}$ (ϵ denotes the empty string),
- if $t \in T_{\Sigma, s}$ and $s \leq s'$ then $t \in T_{\Sigma, s'}$,
- if $f : s_1 \dots s_n \rightarrow s$ and $t_i \in T_{\Sigma, s_i}$, $1 \leq i \leq n$, then $f(t_1, \dots, t_n) \in T_{\Sigma, s}$,

is initial, i.e., there is a unique Σ -homomorphism to each Σ -algebra.

For $[s] \in \hat{S}$, $T_{\Sigma, [s]}$ denotes the set $T_{\Sigma, [s]} = \bigcup_{s' \in [s]} T_{\Sigma, s'}$. T_{Σ} will (ambiguously) denote: (i) the term algebra; (ii) its underlying S -sorted set; and (iii) the set $T_{\Sigma} = \bigcup_{s \in S} T_{\Sigma, s}$. For $[s] \in \hat{S}$, $T_{\Sigma, [s]} = \bigcup_{s' \in [s]} T_{\Sigma, s'}$. An OS signature Σ

is said to *have non-empty sorts* iff for each $s \in S$, $T_{\Sigma,s} \neq \emptyset$. We will assume throughout that Σ has non-empty sorts. An OS signature Σ is called *preregular* [21] iff for each $t \in T_{\Sigma}$ the set $\{s \in S \mid t \in T_{\Sigma,s}\}$ has a least element, denoted $ls(t)$. We will assume throughout that Σ is prerregular.

An S -sorted set $X = \{X_s\}_{s \in S}$ of *variables*, satisfies $s \neq s' \Rightarrow X_s \cap X_{s'} = \emptyset$, and the variables in X are always assumed disjoint from all constants in Σ . The Σ -term algebra on variables X , $T_{\Sigma}(X)$, is the *initial algebra* for the signature $\Sigma(X)$ obtained by adding to Σ the variables X as *extra constants*. Since a $\Sigma(X)$ -algebra is just a pair (A, α) , with A a Σ -algebra, and α an *interpretation of the constants* in X , i.e., an S -sorted function $\alpha \in [X \rightarrow A]$, the $\Sigma(X)$ -initiality of $T_{\Sigma}(X)$ can be expressed as the following theorem:

Theorem 2. (*Freeness Theorem*). *If Σ is sensible, for each $A \in \mathbf{OSAlg}_{\Sigma}$ and $\alpha \in [X \rightarrow A]$, there exists a unique Σ -homomorphism, $_ \alpha : T_{\Sigma}(X) \rightarrow A$ extending α , i.e., such that for each $s \in S$ and $x \in X_s$ we have $x\alpha_s = \alpha_s(x)$.*

In particular, when $A = T_{\Sigma}(Y)$, an interpretation of the constants in X , i.e., an S -sorted function $\sigma \in [X \rightarrow T_{\Sigma}(Y)]$ is called a *substitution*, and its unique homomorphic extension $_ \sigma : T_{\Sigma}(X) \rightarrow T_{\Sigma}(Y)$ is also called a substitution. Define $dom(\sigma) = \{x \in X \mid x \neq x\sigma\}$, and $ran(\sigma) = \bigcup_{x \in dom(\sigma)} vars(x\sigma)$.

The first-order language of *equational Σ -formulas* is defined in the usual way: its atoms are Σ -equations $t = t'$, where $t, t' \in T_{\Sigma}(X)_{[s]}$ for some $[s] \in \hat{S}$ and each X_s is assumed countably infinite. The set $Form(\Sigma)$ of *equational Σ -formulas* is then inductively built from atoms by: conjunction (\wedge), disjunction (\vee), negation (\neg), and universal ($\forall x:s$) and existential ($\exists x:s$) quantification with sorted variables $x:s \in X_s$ for some $s \in S$. The literal $\neg(t = t')$ is denoted $t \neq t'$. Given a Σ -algebra A , a formula $\varphi \in Form(\Sigma)$, and an assignment $\alpha \in [Y \rightarrow A]$, with $Y = fvars(\varphi)$ the free variables of φ , the *satisfaction relation* $A, \alpha \models \varphi$ is defined inductively as usual: for atoms, $A, \alpha \models t = t'$ iff $t\alpha = t'\alpha$; for Boolean connectives it is the corresponding Boolean combination of the satisfaction relations for subformulas; and for quantifiers: $A, \alpha \models (\forall x:s) \varphi$ (resp. $A, \alpha \models (\exists x:s) \varphi$) holds iff for all $a \in A_s$ (resp. some $a \in A_s$) we have $A, \alpha \uplus \{(x:s, a)\} \models \varphi$, where the assignment $\alpha \uplus \{(x:s, a)\}$ extends α by mapping $x:s$ to a . Finally, $A \models \varphi$ holds iff $A, \alpha \models \varphi$ holds for each $\alpha \in [Y \rightarrow A]$, where $Y = fvars(\varphi)$. We say that φ is *valid* (or *true*) in A iff $A \models \varphi$. We say that φ is *satisfiable* in A iff $\exists \alpha \in [Y \rightarrow A]$ such that $A, \alpha \models \varphi$, where $Y = fvars(\varphi)$. For a subsignature $\Omega \subseteq \Sigma$ and $A \in \mathbf{OSAlg}_{\Sigma}$, the *reduct* $A|_{\Omega} \in \mathbf{OSAlg}_{\Omega}$ agrees with A in the interpretation of all sorts and operations in Ω and discards everything in $\Sigma - \Omega$. If $\varphi \in Form(\Omega)$ we have the equivalence $A \models \varphi \Leftrightarrow A|_{\Omega} \models \varphi$.

An OS *equational theory* is a pair $T = (\Sigma, E)$, with E a set of (possibly conditional) Σ -equations. $\mathbf{OSAlg}_{(\Sigma, E)}$ denotes the full subcategory of \mathbf{OSAlg}_{Σ} with objects those $A \in \mathbf{OSAlg}_{\Sigma}$ such that $A \models E$, called the *(Σ, E) -algebras*. $\mathbf{OSAlg}_{(\Sigma, E)}$ has an *initial algebra* $T_{\Sigma/E}$ [29]. Given $T = (\Sigma, E)$ and $\varphi \in Form(\Sigma)$, we call φ *T -valid*, written $E \models \varphi$, iff $A \models \varphi$ for each $A \in \mathbf{OSAlg}_{(\Sigma, E)}$. We call φ *T -satisfiable* iff there exists $A \in \mathbf{OSAlg}_{(\Sigma, E)}$ with φ satisfiable in A . Note that φ is *T -valid* iff $\neg\varphi$ is *T -unsatisfiable*. The inference system in

[29] is *sound and complete* for OS equational deduction, i.e., for any OS equational theory (Σ, E) , and Σ -equation $u = v$ we have an equivalence $E \vdash u = v \iff E \models u = v$. Deducibility $E \vdash u = v$ is abbreviated as $u =_E v$, called *E-equality*. An *E-unifier* of a system of Σ -equations, i.e., a conjunction $\phi = u_1 = v_1 \wedge \dots \wedge u_n = v_n$ of Σ -equations is a substitution σ such that $u_i\sigma =_E v_i\sigma$, $1 \leq i \leq n$. An *E-unification algorithm* for (Σ, E) is an algorithm generating a *complete set* of *E-unifiers* $Unif_E(\phi)$ for any system of Σ equations ϕ , where “complete” means that for any *E-unifier* σ of ϕ there is a $\tau \in Unif_E(\phi)$ and a substitution ρ such that $\sigma =_E \tau\rho$, where $=_E$ here means that for any variable x we have $x\sigma =_E x\tau\rho$. The algorithm is *finitary* if it always terminates with a *finite set* $Unif_E(\phi)$ for any ϕ .

Given a set of equations B used for deduction modulo B , a preregular OS signature Σ is called *B-preregular*¹ iff for each $u = v \in B$ and substitutions ρ , $ls(u\rho) = ls(v\rho)$.

We now recall some basic concepts about *rewriting logic*. The survey in [33] gives a fuller account. The key purpose of a rewrite theory \mathcal{R} is to axiomatize a *distributed system*, so that concurrent computation is modeled as concurrent rewriting with the rules of \mathcal{R} modulo the equations of \mathcal{R} .

Recall the notation for term positions, subterms, and term replacement from [13]: (i) positions in a term viewed as a tree are marked by strings $p \in \mathbb{N}^*$ specifying a path from the root, (ii) $t|_p$ denotes the subterm of term t at position p , and (iii) $t[u]_p$ denotes the result of *replacing* subterm $t|_p$ at position p by u .

Definition 3. A rewrite theory is a 3-tuple $\mathcal{R} = (\Sigma, E \cup B, R)$ with $(\Sigma, E \cup B)$ an OS equational theory and R a set of (possibly conditional) Σ -rewrite rules, i.e., sequents $l \rightarrow r$ if ϕ , with $l, r \in T_\Sigma(X)_{[s]}$ for some $[s] \in \widehat{S}$, and ϕ a quantifier-free Σ -formula.²

We further assume that:

1. Each equation $u = v \in B$ is regular, i.e., $vars(u) = vars(v)$, and linear, i.e., there are no repeated variables in u , and no repeated variables in v . Furthermore, Σ is *B-preregular* (in the broader sense of Footnote 1).
2. The (possibly conditional) equations E , when oriented as rewrite rules $\vec{E} = \{u \rightarrow v \mid \psi \mid u = v \text{ if } \psi \in E\}$, are convergent modulo B , that is,

¹ If $B = B_0 \uplus U$, with B_0 associativity and/or commutativity axioms, and U identity axioms, the *B-preregularity* notion can be *broadened* by requiring only that: (i) Σ is preregular; (ii) Σ is B_0 -preregular in the standard sense that $ls(u\rho) = ls(v\rho)$ for all $u = v \in B_0$ and substitutions ρ ; and (iii) the axioms U oriented as rules \vec{U} are *sort-decreasing* in the sense of Definition 3 below. Maude automatically checks *B-preregularity* of an OS signature Σ in this broader sense [8].

² Usually, ϕ is assumed to be a *conjunction* of Σ -equations. We give here this more general definition for two reasons: (i) often, using *equationally-defined equality predicates* [22], a quantifier-free formula can be transformed into a conjunction of equalities; and (ii) the more general notion is particularly useful for symbolic methods such as rewriting modulo SMT [39], which is similar to both constrained conditional narrowing [6] and to the symbolic methods developed in this paper.

sort-decreasing, strictly coherent, confluent, and operationally terminating as rewrite rules modulo B [27].

3. The rules R are ground coherent with the equations E modulo B [14].

We refer to [33, 27, 14] for more details, but give here an intuitive high-level explanation of what the above conditions mean in practice. Conditions (1)–(2) ensure that the initial algebra $T_{\Sigma/E \cup B}$ is isomorphic to the *canonical term algebra* $C_{\Sigma/E, B}$, whose elements are B -equivalence classes of \vec{E}, B -irreducible ground Σ -terms.

Define the *one-step R, B -rewrite relation* $t \rightarrow_{R, B} t'$ between ground terms as follows. For $t, t' \in T_{\Sigma[s]}$, $[s] \in \hat{S}$, $t \rightarrow_{R, B} t'$ holds iff there is a rewrite rule $l \rightarrow r$ if $\phi \in R$, a ground substitution $\sigma \in [Y \rightarrow T_{\Sigma}]$ with Y the rule's variables, and a term position p in t such that $t|_p =_B l\sigma$, $t' = t[r\sigma]_p$, and $E \cup B \models \phi\sigma$. What, in the context of (1)–(2), condition (3) then ensures is “computing \vec{E}, B -canonical forms before performing R, B -rewriting” is a *complete* strategy. That is, if $t \rightarrow_{R, B} t'$ and $u = t!_{E, B}$, i.e., $t \rightarrow_{\vec{E}, B}^* u$ with u in \vec{E}, B -canonical form (abbreviated in what follows to $u = t!$), then there exists a u' such that $u \rightarrow_{R, B} u'$ and $t' =_B u'$. Note that $\text{vars}(r) \subseteq \text{vars}(l)$ is nowhere assumed for rules $l \rightarrow r$ if $\phi \in R$. This means that \mathcal{R} can specify an *open system*, in the sense of [39], that interacts with an external, non-deterministic environment.

Conditions (1)–(3) allow a simple and intuitive description of the *initial reachability model* $\mathcal{T}_{\mathcal{R}}$ [5] of \mathcal{R} as the *canonical reachability model* $\mathcal{C}_{\mathcal{R}}$ whose states are the elements of the *canonical term algebra* $C_{\Sigma/E, B}$, and where the one-step transition relation $[u] \rightarrow_{\mathcal{R}} [v]$ holds iff $u \rightarrow_{R, B} u'$ and $[u'] = [v]$. Furthermore, if $u \rightarrow_{R, B} u'$ has been performed with a rewrite rule $l \rightarrow r$ if $\phi \in R$ and a ground substitution $\sigma \in [Y \rightarrow T_{\Sigma}]$, then, assuming B -equality is decidable, checking whether condition $E \cup B \models \phi\sigma$ holds is *decidable* by reducing the terms in $\phi\sigma$ to \vec{E}, B -canonical form.

A Running Example. Consider the following rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ modeling a dynamic version of the QLOCK mutual exclusion protocol [18], where (Σ, B) defines the protocol's states, involving natural numbers, lists, and multisets over natural numbers. Σ has sorts $S = \{\text{Nat}, \text{List}, \text{MSet}, \text{Conf}, \text{State}, \text{Pred}\}$ with subsorts $\text{Nat} < \text{List}$ and $\text{Nat} < \text{MSet}$ and operators $F = \{0 : \rightarrow \text{Nat}, s_- : \text{Nat} \rightarrow \text{Nat}, mt : \rightarrow \text{Set}, nil : \rightarrow \text{List}, _ _ : \text{MSet MSet} \rightarrow \text{MSet}, _ ; _ : \text{List List} \rightarrow \text{List}, dupl : \text{MSet} \rightarrow \text{Pred}, tt : \rightarrow \text{Pred}, _ _ _ : \text{MSet MSet MSet List} \rightarrow \text{Conf}, < _ > : \text{Conf} \rightarrow \text{State}\}$, where underscores denote operator argument placement and the last two operators are state-building constructors. The axioms B are the associativity-commutativity of the multiset union $_ _$ with identity mt , and the associativity of list concatenation $_ ; _$ with identity nil . The only equation in E is $dupl(s i i) = tt$. It defines the *dupl* predicate by detecting a duplicated element i in the multiset $s i i$ (where s could be empty).

QLOCK [18] is a mutual exclusion protocol where the number of processes is unbounded. Furthermore, in the *dynamic* version of QLOCK presented below, such a number can grow or shrink. Each process is identified by a number. The system configuration has three sets of processes (normal, waiting, and critical)

plus a waiting queue. To ensure mutual exclusion, a normal process must first register its name at the end of the waiting queue. When its name appears at the front of the queue, it is allowed to enter the critical section. The first three rewrite rules in R below specify how a *normal* process i first transitions to a *waiting* process, then to a *critical* process, and back to normal. The last two rules in R specify how a process can dynamically join or exit the system.

$$\begin{aligned}
n2w &: \langle n \ i \mid w \mid c \mid q \rangle \rightarrow \langle n \mid w \ i \mid c \mid q ; i \rangle \\
w2c &: \langle n \mid w \ i \mid c \mid i ; q \rangle \rightarrow \langle n \mid w \mid c \ i \mid i ; q \rangle \\
c2n &: \langle n \mid w \mid c \ i \mid i ; q \rangle \rightarrow \langle n \ i \mid w \mid c \mid q \rangle \\
join &: \langle n \mid w \mid c \mid q \rangle \rightarrow \langle n \ i \mid w \mid c \mid q \rangle \text{ if } \phi \\
exit &: \langle n \ i \mid w \mid c \mid q \rangle \rightarrow \langle n \mid w \mid c \mid q \rangle
\end{aligned}$$

where $\phi = \text{dupl}(n \ i \ w \ c) \neq tt$, i is a number, n , w , and c are, respectively, normal, waiting, and critical process identifier sets, and q is a queue of process identifiers. Using [7] it is easy to check that $(\Sigma, E \cup B)$ satisfies the finite variant property [12], and that $\mathcal{R} = (\Sigma, E \cup B, R)$ satisfies requirements (1)–(3). Note that *join* makes QLOCK an *open* system in the sense explained above.

3 Constrained Constructor Pattern Predicates

Given an OS equational theory $(\Sigma, E \cup B)$, the *atomic state predicates* appearing in the constructor-based reachability logic formulas of Section 4 will be pairs $u \mid \varphi$, called *constrained constructor patterns*, with u a term in a subsignature $\Omega \subseteq \Sigma$ of constructors, and φ a quantifier-free Σ -formula. Intuitively, $u \mid \varphi$ is a pattern describing the set of states that are $E_\Omega \cup B_\Omega$ -equal to ground terms of the form $u\rho$ for ρ a ground constructor substitution such that $E \cup B \models \varphi\rho$. Therefore, $u \mid \varphi$ can be used as a *symbolic description* of a, typically infinite, *set of states* in the canonical reachability model $\mathcal{C}_\mathcal{R}$ of a rewrite theory \mathcal{R} .

Often, the signature Σ on which $T_{\Sigma/E \cup B}$ is defined has a natural decomposition as a disjoint union $\Sigma = \Omega \uplus \Delta$, where the elements of the canonical term algebra $C_{\Sigma/E, B}$ are Ω -terms, whereas the function symbols $f \in \Delta$ are viewed as *defined functions* which are *evaluated away* by \vec{E}, B -simplification. Ω (with some poset of sorts as Σ) is then called a *constructor subsignature* of Σ .

A *decomposition* of an order-sorted equational theory $(\Sigma, E \cup B)$ is a triple (Σ, B, \vec{E}) such that the rules \vec{E} are convergent modulo B . (Σ, B, \vec{E}) is called *sufficiently complete* with respect to the *constructor subsignature* Ω iff for each $t \in T_\Sigma$ we have: (i) $t! \in T_\Omega$, and (ii) if $u \in T_\Omega$ and $u =_B v$, then $v \in T_\Omega$. This ensures that for each $[u]_B \in C_{\Sigma/E, B}$ we have $[u]_B \subseteq T_\Omega$. Sufficient completeness is closely related to the notion of a *protecting* inclusion of decompositions.

Definition 4. Let $(\Sigma_0, E_0 \cup B_0) \subseteq (\Sigma, E \cup B)$ be a theory inclusion such that $(\Sigma_0, B_0, \vec{E}_0)$ and (Σ, B, \vec{E}) are respective decompositions of $(\Sigma_0, E_0 \cup B_0)$ and $(\Sigma, E \cup B)$. We then say that the decomposition (Σ, B, \vec{E}) protects $(\Sigma_0, B_0, \vec{E}_0)$ iff (i) for all $t, t' \in T_{\Sigma_0}(X)$ we have: (i) $t =_{B_0} t' \Leftrightarrow t =_B t'$, (ii) $t = t!_{\vec{E}_0, B_0} \Leftrightarrow t = t!_{\vec{E}, B}$, and (iii) $C_{\Sigma_0/E_0, B_0} = C_{\Sigma/E, B}|_{\Sigma_0}$.

$(\Omega, B_\Omega, \vec{E}_\Omega)$ is a constructor decomposition of (Σ, B, \vec{E}) iff (i) (Σ, B, \vec{E}) protects $(\Omega, B_\Omega, \vec{E}_\Omega)$, and (ii) (Σ, B, \vec{E}) is sufficiently complete with respect to the constructor subsignature Ω . Furthermore, Ω is called a subsignature of free constructors modulo B_Ω iff $E_\Omega = \emptyset$, so that $C_{\Omega/E_\Omega, B_\Omega} = T_{\Omega, B_\Omega}$.

We are now ready to define constrained constructor pattern predicates and their semantics. In what follows, X will always denote the countably infinite S -sorted set of variables used in the language of Σ -formulas.

Definition 5. Let $(\Omega, B_\Omega, \vec{E}_\Omega)$ be a constructor decomposition of (Σ, B, \vec{E}) . A constrained constructor pattern is an expression $u \mid \varphi$ with $u \in T_\Omega(X)$ and φ a QF Σ -formula. The set $PatPred(\Omega, \Sigma)$ of constrained constructor pattern predicates contains \perp and the set of constrained constructor patterns, and is closed under disjunction (\vee) and conjunction (\wedge). Capital letters A, B, \dots, P, Q, \dots range over $PatPred(\Omega, \Sigma)$. The semantics of a constrained constructor pattern predicate A is a subset $\llbracket A \rrbracket \subseteq C_{\Sigma/E, B}$ defined inductively as follows:

1. $\llbracket \perp \rrbracket = \emptyset$
2. $\llbracket u \mid \varphi \rrbracket = \{[(u\rho)!]_{B_\Omega} \in C_{\Sigma/E, B} \mid \rho \in [X \rightarrow T_\Omega] \wedge E \cup B \models \varphi\rho\}$.
3. $\llbracket A \vee B \rrbracket = \llbracket A \rrbracket \cup \llbracket B \rrbracket$
4. $\llbracket A \wedge B \rrbracket = \llbracket A \rrbracket \cap \llbracket B \rrbracket$.

Note that for any constructor pattern predicate A , if σ is is (sort-preserving) bijective renaming of variables we always have $\llbracket A \rrbracket = \llbracket A\sigma \rrbracket$. Given constructor patterns $u \mid \varphi$ and $v \mid \psi$ with $vars(u \mid \varphi) \cap vars(v \mid \psi) = \emptyset$, we say that $u \mid \varphi$ subsumes $v \mid \psi$ iff there is a substitution α such that: (i) $v =_{E_\Omega \cup B_\Omega} u\alpha$, and (ii) $\mathcal{T}_{E \cup B} \models \psi \Rightarrow (\varphi\alpha)$. It then follows easily from the above definition of $\llbracket u \mid \varphi \rrbracket$ that if $u \mid \varphi$ subsumes $v \mid \psi$, then $\llbracket v \mid \psi \rrbracket \subseteq \llbracket u \mid \varphi \rrbracket$. Likewise, $\bigvee_{i \in I} u_i \mid \varphi_i$ subsumes $v \mid \psi$ iff there is a $k \in I$ such that $u_k \mid \varphi_k$ subsumes $v \mid \psi$.

Pattern Predicate Example. Recall that QLOCK states have the general form $\langle n \mid w \mid c \mid q \rangle$ with n, w, c multisets of process identifiers and q an associative list of process identifiers. From the five rewrite rules defining QLOCK, it is easy to prove that if $\langle n \mid w \mid c \mid q \rangle \rightarrow^* \langle n' \mid w' \mid c' \mid q' \rangle$ and $n w c$ is a set (has no repeated elements), then $n' w' c'$ is also a set. Of course, it seems very reasonable to assume that these process identifier multisets are, in fact, sets, since otherwise we could, for example, have a process i which is *both* waiting and critical at the *same* time. We can rule out such ambiguous states by means of the pattern predicate $\langle n \mid w \mid c \mid q \rangle \mid \text{dupl}(n w c) \neq tt$.

Note that, assuming that $E_\Omega \cup B_\Omega$ has a finitary unification algorithm, any constrained constructor pattern predicate A is semantically equivalent to a finite disjunction $\bigvee_i u_i \mid \varphi_i$ of constrained constructor patterns. This is because: (i) by (2)–(3) above, we may assume φ in disjunctive normal form; and (ii) it is easy to check that $\llbracket (u \mid \varphi) \wedge (v \mid \phi) \rrbracket = \bigcup_{\alpha \in DUnif_{E_\Omega \cup B_\Omega}(u, v)} \llbracket u\alpha \mid (\varphi \wedge \phi)\alpha \rrbracket$, with $DUnif_{E_\Omega \cup B_\Omega}(u, v)$ the *disjoint* $E_\Omega \cup B_\Omega$ -unifiers of u and v , i.e., $E_\Omega \cup B_\Omega$ -unifiers of u with a variable-renaming v' of v not sharing variables with u .

In the above discussion of intersections we may assume without loss of generality that variables are disjoint. But this may not always be what we want. Consider constrained patterns $u \mid \varphi$ and $v \mid \phi$ with $Y = \text{vars}(u \mid \varphi) \cap \text{vars}(v \mid \phi) = \text{vars}(u) \cap \text{vars}(v)$. The sharing of variables Y may be intentional as *parameters* common to both $u \mid \varphi$ and $v \mid \phi$. This can be illustrated with an example of two patterns describing triples of natural numbers, namely, $\langle 0, y, z \rangle \mid \top$ and $\langle x, s(y), 1 \rangle \mid \top$ with shared parameter y . We can understand these patterns *parametrically* as describing the \mathbb{N} -indexed families of sets: $\{\langle 0, n, z \rangle \mid z \in \mathbb{N}\}_{n \in \mathbb{N}}$ and $\{\langle x, s(n), 1 \rangle \mid x \in \mathbb{N}\}_{n \in \mathbb{N}}$. Their \mathbb{N} -indexed intersection $\{\langle 0, n, z \rangle \mid z \in \mathbb{N}\} \cap \{\langle x, s(n), 1 \rangle \mid z \in \mathbb{N}\}_{n \in \mathbb{N}} = \{\emptyset\}_{n \in \mathbb{N}}$ can then be symbolically described by \perp , because the terms $\langle 0, y, z \rangle$ and $\langle x, s(y), 1 \rangle$ have *no unifier*, although they can be *disjointly unified* into the term $\langle 0, s(y'), 1 \rangle$, so that $\llbracket \langle 0, y, z \rangle \mid \top \rrbracket \cap \llbracket \langle x, s(y), 1 \rangle \mid \top \rrbracket = \llbracket \langle 0, s(y'), 1 \rangle \mid \top \rrbracket$. This suggests the notion of a *Y-parameterized intersection* defined as: $\llbracket u \mid \varphi \rrbracket \cap_Y \llbracket v \mid \phi \rrbracket = \bigcup_{\alpha \in \text{Unif}_{E_{\Omega} \cup B_{\Omega}}(u, v)} \llbracket u\alpha \mid (\varphi \wedge \phi)\alpha \rrbracket$, assuming $Y = \text{vars}(u \mid \varphi) \cap \text{vars}(v \mid \phi) = \text{vars}(u) \cap \text{vars}(v)$. Under the same assumption there is also a natural notion of *Y-parameterized subsumption* of $v \mid \phi$ by $u \mid \varphi$, namely, such subsumption holds iff there is a substitution α such that: (i) $(\forall y \in Y) \alpha(y) = y$, (ii) $v =_{E_{\Omega} \cup B_{\Omega}} u\alpha$, and (iii) $\mathcal{T}_{E \cup B} \models \psi \Rightarrow (\varphi\alpha)$. This ensures that for all ground substitutions ρ of the variables Y we have $\llbracket (v \mid \phi)\rho \rrbracket \subseteq \llbracket (u \mid \varphi)\rho \rrbracket$. These notions will be used in Section 4.1 to reason about *parameterized invariants* and *co-invariants*.

4 Constructor-Based Reachability Logic

The constructor-based reachability logic we shall define is a logic to reason about reachability properties of the canonical reachability model $\mathcal{C}_{\mathcal{R}}$ of a topmost rewrite theory \mathcal{R} , where “topmost” captures the intuitive idea that all rewrites with the rules R in \mathcal{R} happen at the top of the term. Many rewrite theories of interest, including theories specifying distributed object-oriented systems and rewriting logic specifications of (possibly concurrent) programming languages, can be easily specified as topmost rewrite theories by a simple theory transformation (see, e.g., [47]). The rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$, besides satisfying the requirements in Definition 3, satisfies two additional requirements:

1. $(\Sigma, E \cup B)$ has a sort *State*, the top sort of a connected component $[State]$, a decomposition (Σ, B, \vec{E}) , and a constructor decomposition $(\Omega, B_{\Omega}, \vec{E}_{\Omega})$ such that: (i) $\forall u \in T_{\Omega}(X)_{State}$, $\text{vars}(u) = \text{vars}(u!)$; and (ii) the axioms B_{Ω} are linear and regular and have a finitary $E_{\Omega} \cup B_{\Omega}$ -unification algorithm.
2. Rules in R have the form $l \rightarrow r$ if φ with $l \in T_{\Omega}(X)$. Furthermore, they are *topmost* in the sense that: (i) for all such rules, l and r have sort *State*, and (ii) for any $u \in T_{\Omega}(X)_{State}$ and any non-empty position p in u , $u|_p \notin T_{\Omega}(X)_{State}$.

Using the theory transformation in [47], requirement (2) is satisfied by many practical applications, including programming language definitions and distributed object-based system specifications. Requirements (1)–(2) ensure that in the

canonical reachability model $\mathcal{C}_{\mathcal{R}}$ if $[u] \rightarrow_{\mathcal{R}} [v]$ holds, then the R, B -rewrite $u \rightarrow_{R, B} u'$ such that $[u'] = [v]$ must have been performed with a rewrite rule $l \rightarrow r$ if $\varphi \in R$ and a ground substitution $\sigma \in [Y \rightarrow T_{\Omega}]$, with Y the rule's variables, such that $u =_{B, \Omega} l\sigma$, i.e., the rewrite must happen at the top of u .

We are now ready to define the formulas of our constructor-based version of reachability logic for \mathcal{R} satisfying above requirements (1)–(2). Such formulas have the form: $A \rightarrow^{\otimes} B$, with A and B constrained constructor pattern predicates in $PatPred(\Omega, \Sigma)$ for $(\Omega, B_{\Omega}, \vec{E}_{\Omega})$ the constructor decomposition of (Σ, B, \vec{E}) . Furthermore, we require all atomic constrained constructor predicates $u \mid \varphi$ appearing in either A or B to be such that u has sort *State*. By definition, the *parameters* Y of $A \rightarrow^{\otimes} B$ are the variables in the set $Y = vars(A) \cap vars(B)$, and $A \rightarrow^{\otimes} B$ is called *unparameterized* iff $Y = \emptyset$.

The presentation of reachability logic in [46] considers *two* different semantics: (i) a *one-path* semantics, which we denote $\mathcal{R} \models^1 A \rightarrow^{\otimes} B$, and (ii) an *all-paths* semantics, which we denote $\mathcal{R} \models^{\forall} A \rightarrow^{\otimes} B$. Since the all-paths semantics is the most general and expressive, and the one-path semantics applies mostly to sequential systems, in this work we focus on the all-paths semantics and leave the one-path semantics for future work.

The reachability logic in [46] is based on *terminating* sequences of state transitions and is such that all reachability formulas are *vacuously true* when there are no terminating states. Our purpose is to extend reachability logic so as to be able to verify properties of general distributed systems specified as rewrite theories \mathcal{R} which *may never terminate*. For this, as further explained in Section 4.1, we need to *generalize* the satisfaction relation $\mathcal{R} \models^{\forall} A \rightarrow^{\otimes} B$ to a *relativized* satisfaction relation $\mathcal{R} \models_T^{\forall} A \rightarrow^{\otimes} B$, where T is a constrained pattern predicate such that $\llbracket T \rrbracket$ is a set of terminating states. That is, let $Term_{\mathcal{R}} = \{[u] \in \mathcal{C}_{\mathcal{R}, State} \mid (\exists [v]) [u] \rightarrow_{\mathcal{R}} [v]\}$. We then require $\llbracket T \rrbracket \subseteq Term_{\mathcal{R}}$. The standard relation $\mathcal{R} \models^{\forall} A \rightarrow^{\otimes} B$ is then recovered as the special case where $\llbracket T \rrbracket = Term_{\mathcal{R}}$. Call $[u] \rightarrow_{\mathcal{R}}^* [v]$ a *T-terminating sequence* iff $[v] \in \llbracket T \rrbracket$.

Definition 6. *Given T with $\llbracket T \rrbracket \subseteq Term_{\mathcal{R}}$, the all-paths satisfaction relation $\mathcal{R} \models_T^{\forall} u \mid \varphi \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ asserts the satisfaction of the formula $u \mid \varphi \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ in the canonical reachability model $\mathcal{C}_{\mathcal{R}}$ of a rewrite theory \mathcal{R} satisfying requirements (1)–(2). It is defined as follows:*

For $u \mid \varphi \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ unparameterized, $\mathcal{R} \models_T^{\forall} u \mid \varphi \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ holds iff for each T -terminating sequence $[u_0] \rightarrow_{\mathcal{R}} [u_1] \dots [u_{n-1}] \rightarrow_{\mathcal{R}} [u_n]$ with $[u_0] \in \llbracket u \mid \varphi \rrbracket$ there exist k , $0 \leq k \leq n$ and $j \in J$ such that $[u_k] \in \llbracket v_j \mid \phi_j \rrbracket$. For $u \mid \varphi \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ with parameters Y , $\mathcal{R} \models_T^{\forall} u \mid \varphi \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ holds if $\mathcal{R} \models_T^{\forall} (u \mid \varphi)\rho \rightarrow^{\otimes} (\bigvee_{j \in J} v_j \mid \phi_j)\rho$ holds for each $\rho \in [Y \rightarrow T_{\Omega}]$.

Since any constrained pattern predicate is semantically equivalent to a disjunction of atomic ones, we then define the satisfaction relation on general reachability logic formulas as follows: $\mathcal{R} \models_T^{\forall} \bigvee_{i \in I} u_i \mid \varphi_i \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$ iff $\bigwedge_{i \in I} \mathcal{R} \models_T^{\forall} u_i \mid \varphi_i \rightarrow^{\otimes} \bigvee_{j \in J} v_j \mid \phi_j$.

$\mathcal{R} \models_T^\forall A \rightarrow^\otimes B$ is a *path-universal partial correctness assertion*: If state $[u]$ satisfies “precondition” A , then “postcondition” B is satisfied *somewhere* along *each* T -terminating sequences from $[u]$, generalizing a Hoare formula $\{A\}\mathcal{R}\{B\}$.

Recall that in requirement (2) for our rewrite theory \mathcal{R} we assumed topmost rewrite rules of the form $l \rightarrow r$ if ϕ with $l \in T_\Omega(X)$. For symbolic reasoning purposes it will be very useful to also require that $r \in T_\Omega(X)$. This can be done without any real loss of generality by means of a theory transformation $\mathcal{R} \mapsto \hat{\mathcal{R}}$ defined as follows. If $\mathcal{R} = (\Sigma, E \cup B, R)$, then $\hat{\mathcal{R}} = (\Sigma, E \cup B, \hat{R})$, where the rules \hat{R} are obtained from the rules R by transforming each $l \rightarrow r$ if ϕ in R into the rule $l \rightarrow r'$ if $\phi \wedge \hat{\theta}$, where: (i) r' is the Ω -abstraction of r obtained by replacing each length-minimal position p of r such that $t|_p \notin T_\Omega(X)$ by a fresh variable x_p whose sort is the least sort of $t|_p$, (ii) $\hat{\theta} = \bigwedge_{p \in P} x_p = t_p$, where P is the set of all length-minimal positions in r such that $t|_p \notin T_\Omega(X)$. The key semantic property about this transformation can be expressed as follows:

Lemma 1. *The canonical reachability models $\mathcal{C}_\mathcal{R}$ and $\mathcal{C}_{\hat{\mathcal{R}}}$ are identical.*

4.1 Invariants, Co-Invariants, and Never-Terminating Systems

The notion of an *invariant* makes sense for any transition system \mathcal{S} , that is, for any pair $\mathcal{S} = (S, \rightarrow_\mathcal{S})$ with S its set of *states* and $\rightarrow_\mathcal{S} \subseteq S \times S$ its *transition relation*. Given a set of “initial states” $S_0 \subseteq S$, the set *Reach*(S_0) of states *reachable* from S_0 is defined as $\text{Reach}(S_0) = \{s \in S \mid (\exists s_0 \in S_0) s_0 \rightarrow_\mathcal{S}^* s\}$, where $\rightarrow_\mathcal{S}^*$ denotes the reflexive-transitive closure of $\rightarrow_\mathcal{S}$. An invariant is a safety property about \mathcal{S} with initial states S_0 and can be specified in two ways: (i) by a “good” property $P \subseteq S$, the *invariant*, that *always holds* from S_0 , i.e., such that $\text{Reach}(S_0) \subseteq P$, or (ii) as a “bad” property $Q \subseteq S$, the *co-invariant*, that *never holds* from S_0 , i.e., such that $\text{Reach}(S_0) \cap Q = \emptyset$. Obviously, P is an invariant iff $S - P$ is a co-invariant. Sometimes it is easier to specify an invariant *positively*, as P , and sometimes *negatively*, as its co-invariant $S - P$.

All this is particularly relevant for the transitions system $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_\mathcal{R})$ associated to the canonical model $\mathcal{C}_\mathcal{R}$ of a rewrite theory \mathcal{R} . Here is an obvious question with a non-obvious answer. Suppose we have specified a distributed system as the canonical model $\mathcal{C}_\mathcal{R}$ of a rewrite theory \mathcal{R} satisfying conditions (1)–(2). Suppose further that we have specified constrained pattern predicates S_0 and P (resp. and Q) and we want to prove that $\llbracket P \rrbracket$ (resp. $\llbracket Q \rrbracket$) is an *invariant* (resp. *co-invariant*) of the system $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_\mathcal{R})$ from $\llbracket S_0 \rrbracket$. Can we specify such invariant or co-invariant by means of *reachability formulas* and use the inference system of Section 5 to try to prove such formulas?

The answer to the above question is not obvious. Suppose \mathcal{R} specifies a *never-terminating system*, i.e., a system such that $\text{Term}_\mathcal{R} = \emptyset$. Many distributed systems are never-terminating. For example, QLOCK and other mutual exclusion protocols are never-terminating. Then, no reachability formula can characterize and invariant (resp. co-invariant) holding by means of the satisfaction relation $\mathcal{R} \models_T^\forall A \rightarrow^\otimes B$. The reason for this impossibility is that, since $\text{Term}_\mathcal{R} = \emptyset$, $\mathcal{R} \models_T^\forall A \rightarrow^\otimes B$ holds vacuously for *all* reachability formulas $A \rightarrow^\otimes B$.

Is then reachability logic useless to prove invariants? Definitely *not*. It can indeed be very useful for proving invariants and co-invariants of distributed systems, regardless of whether they are: (a) terminating, (b) sometimes terminating, or (c) never terminating: we just need to first perform a simple *theory transformation*. Let us first focus on specifying invariants. We can distinguish two cases: (i) an invariant is *specifiable by constrained pattern predicates* S_0 and P if $\llbracket P \rrbracket$ is an *invariant* of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$; and (ii) an invariant is *conditionally specifiable by constrained pattern predicates* S_0 , A , and P if $\llbracket A \rrbracket \Rightarrow \llbracket P \rrbracket$ is an *invariant* of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$, where, by definition, $\llbracket A \rrbracket \Rightarrow \llbracket P \rrbracket = (\mathcal{C}_{\mathcal{R}, State} - \llbracket A \rrbracket) \cup \llbracket P \rrbracket$. To ease the exposition, we explain the transformation for the case where Ω has a single state constructor operator, say, $\langle -, \dots, - \rangle : s_1, \dots, s_n \rightarrow State$. The extension to several such operators is straightforward. The theory transformation in case (i) is of the form $\mathcal{R} \mapsto \mathcal{R}_{stop}$, where \mathcal{R}_{stop} is obtained from \mathcal{R} by just adding: (1) a new state constructor operator $[-, \dots, -] : s_1, \dots, s_n \rightarrow State$ to Ω , and (2) a new rewrite rule $stop : \langle x_1 : s_1, \dots, x_n : s_n \rangle \rightarrow [x_1 : s_1, \dots, x_n : s_n]$ to R . Also, let $[\]$ denote the pattern predicate $[x_1 : s_1, \dots, x_n : s_n] \mid \top$. Likewise, for any atomic constrained pattern predicate $B = \langle u_1, \dots, u_n \rangle \mid \varphi$ we define the pattern predicate $[B] = [u_1, \dots, u_n] \mid \varphi$ and extend this notation to any union Q of atomic predicates. Since $\langle -, \dots, - \rangle : s_1, \dots, s_n \rightarrow State$ is the only state constructor, we can assume without loss of generality that any atomic constrained pattern predicate in \mathcal{R} is semantically equivalent to one of the form $\langle u_1, \dots, u_n \rangle \mid \varphi$. Likewise, any constrained pattern predicate will be semantically equivalent to a union of atomic predicates of such form, called in *standard form*. In a similar manner, the theory transformation in case (ii) is of the form $\mathcal{R} \mapsto \mathcal{R}_{stop(A)}$, where $\mathcal{R}_{stop(A)}$ is obtained from \mathcal{R} by just adding: (1) a new state constructor operator $[-, \dots, -] : s_1, \dots, s_n \rightarrow State$ to Ω , and (2) new rewrite rules $stop(A) : \langle u_1^i, \dots, u_n^i \rangle \rightarrow [u_1^i, \dots, u_n^i]$ if φ_i , $1 \leq i \leq m$, to R , where $A = \bigvee_{1 \leq i \leq m} \langle u_1^i, \dots, u_n^i \rangle \mid \varphi_i$. Here is now the main theorem:

Theorem 3. *For $S_0, A, P \in PatPred(\Omega, \Sigma)$ constrained pattern predicates in standard form such that $vars(S_0) \cap vars(P) = \emptyset$ and $vars(S_0) \cap vars(A) = \emptyset$,*

1. $\llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ iff $\mathcal{R}_{stop} \models_{\llbracket \]}^{\forall} S_0 \rightarrow^{\otimes} \llbracket P \rrbracket$.
2. $\llbracket A \rrbracket \Rightarrow \llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ iff $\mathcal{R}_{stop(A)} \models_{\llbracket \]}^{\forall} S_0 \rightarrow^{\otimes} \llbracket P \rrbracket$.

Specifying Invariants for QLOCK. We illustrate how to specify invariants as reachability formulas using the QLOCK specification from Sections 2 and 3. Note that not only is QLOCK nonterminating: it is also *never* terminating. Thus, specifying any invariants as reachability formulas in the original theory is impossible. However, we can apply the theory transformation suggested by Theorem 3 by adding a fresh operator $[-] : Conf \rightarrow State$ and a rule $stop : \langle t \rangle \rightarrow [t]$ for $t : Conf$. Define the set of initial states having only normal processes by the pattern predicate $S_0 = \langle n' \mid mt \mid mt \mid nil \rangle \mid dupl(n') \neq tt$. Since QLOCK states have the form $\langle n \mid w \mid c \mid q \rangle$, mutual exclusion means $|c| \leq 1$, which is expressible by the pattern predicate $\langle n \mid w \mid i \mid i ; q \rangle \vee$

$\langle n \mid w \mid mt \mid q \rangle$. But we need also to ensure our multisets are actually *sets*. Thus, the pattern predicate $P = (\langle n \mid w \mid i \mid i ; q \rangle \mid \text{dupl}(n \ w \ i) \neq tt) \vee (\langle n \mid w \mid mt \mid q \rangle \mid \text{dupl}(n \ w) \neq tt)$ specifies mutual exclusion. By Theorem 3, QLOCK ensures mutual exclusion from $\llbracket S_0 \rrbracket$ iff $\mathcal{R}_{stop} \models_{\perp}^{\forall} S_0 \rightarrow^{\otimes} [P]$.

The following easy corollary can be very helpful in proving invariants. It can, in particular, be applied to prove the mutual exclusion of QLOCK.

Corollary 1. *For $S_0, P \in \text{PatPred}(\Omega, \Sigma)$ constrained pattern predicates in standard form such that $\text{vars}(S_0) \cap \text{vars}(P) = \emptyset$. Then $\llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ if: (i) $\llbracket S_0 \rrbracket \subseteq \llbracket P \rrbracket$, and (ii) $\mathcal{R}_{stop} \models_{\perp}^{\forall} P\sigma \rightarrow^{\otimes} [P]$, where σ is a sort-preserving bijective renaming of variables such that $\text{vars}(P) \cap \text{vars}(P\sigma) = \emptyset$.*

The notion of a *parametric* invariant can be reduced to the unparameterized one: if $Y = \text{vars}(S_0) \cap \text{vars}(P)$, then $\llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ with parameters Y , iff $\mathcal{R}_{stop} \models_{\perp}^{\forall} S_0 \rightarrow^{\otimes} [P]$. That is, iff $\llbracket P\rho \rrbracket$ is an (unparameterized) invariant of $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0\rho \rrbracket$ for each $\rho \in [Y \rightarrow T_{\Omega}]$.

Let us now turn to the case of co-invariants. Suppose we have specified constrained pattern predicates S_0 and Q and we want to prove that $\llbracket Q \rrbracket$ is a *co-invariant* of the system $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$. Can this be expressed by some reachability formula or formulas? Let $Q = \bigvee_{1 \leq i \leq m} u_i \mid \varphi_i$, and let $\mathcal{R} = (\Sigma, E \cup B, R)$. We can answer this question in most practical cases by means of a simple theory transformation $\mathcal{R} \mapsto \mathcal{R}^Q$, where $\mathcal{R}^Q = (\Sigma_Q, E \cup E_Q \cup B, R)$ with: (i) Σ_Q extending Σ by adding a sort *Pred* with a constant *tt*, and a “predicate” operator $p_Q : \text{State} \rightarrow \text{Pred}$, and E_Q the conditional equations: $p_Q(u_i) = tt$ if φ_i , $1 \leq i \leq m$. Since the φ_i can be general QF Σ -formulas, something should be said about such general “conditional equations.” Specifically: (i) if the φ_i are conjunctions of equalities, they are conditional equations in the standard sense; (ii) if the constructor decomposition of $(\Sigma, E \cup B)$ is of the form (Ω, B_{Ω}) , with B_{Ω} a combination of associativity and/or commutativity axioms, we can use the theory transformation $(\Sigma, E \cup B) \mapsto (\Sigma^{\equiv}, E^{\equiv} \cup B)$ in [22] that adds equationally-defined equality predicates to $(\Sigma, E \cup B)$ to turn each φ_i into a Boolean equation $\overline{\varphi}_i = \top$; and (iii) otherwise, if the formulas φ_i belong to a set of QF formulas whose satisfiability in $C_{\Sigma/E, B}$ is decidable, we can view the equations E_Q as conditional equations with decidable constraint conditions and use the symbolic methods in, e.g., [17, 39] to execute such equations by rewriting. Using the techniques for cases (i)–(iii), we can obtain a decomposition $(\Sigma_Q, B, \vec{E} \cup \vec{E}_Q)$ of $(\Sigma_Q, E \cup E_Q \cup B)$ making p_Q executable by rewriting in most practical cases. In particular, if we are either in cases (i)–(ii) and $\text{vars}(\varphi_i) \subseteq \text{vars}(u_i)$, or in case (iii), this can always be done. Also, the constructors of $(\Sigma_Q, E \cup E_Q \cup B)$ are those for $(\Sigma, E \cup B)$ plus the symbol p_Q .

The key point is that in all the above cases, after extending \mathcal{R} to \mathcal{R}^Q , we are able to express the *invariant* $\mathcal{C}_{\mathcal{R}, \text{State}} - \llbracket Q \rrbracket$ we want to prove as the (constrained) pattern predicate $\overline{Q} = \langle x_1:s_1, \dots, x_n:s_n \rangle \mid p_Q(\langle x_1:s_1, \dots, x_n:s_n \rangle) \neq tt$. Therefore, as an immediate corollary of both Theorem 3 and Corollary 1 we get:

Corollary 2. *Let $S_0, Q \in \text{PatPred}(\Omega, \Sigma)$ be pattern predicates in standard form such that $\text{vars}(S_0) \cap \text{vars}(Q) = \emptyset$. Under the above p_Q executability conditions $\llbracket Q \rrbracket$ is a co-invariant of $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ iff $\mathcal{R}_{\text{stop}}^Q \models_{\square}^{\forall} S_0 \rightarrow^{\otimes} \llbracket \overline{Q} \rrbracket$. Furthermore, $\mathcal{R}_{\text{stop}}^Q \models_{\square}^{\forall} \overline{Q}\sigma \rightarrow^{\otimes} \llbracket \overline{Q} \rrbracket$ is a sufficient condition, where σ is a sort-preserving bijective renaming of variables such that $\text{vars}(\overline{Q}) \cap \text{vars}(\overline{Q}\sigma) = \emptyset$.*

Of course, the techniques for specifying and verifying invariants and co-invariants with constrained pattern predicates can be combined. Quite often an invariant is strengthened as a conjunction of invariants $P_1 \wedge P_2$. But it may be easier to express, say, P_2 by means of its co-invariant Q , so that the strengthened invariant can be specified as the (constrained) pattern predicate $P_1 \wedge \overline{Q}$.

A drawback of the above coinvariant reasoning technique is that it requires defining the invariant \overline{Q} associated to the co-invariant Q . Is there a simpler method of proving a co-invariant by proving a reachability formula? Yes! By using the rules of \mathcal{R} backwards. Assume without loss of generality that $\mathcal{R} = \hat{\mathcal{R}}$. Then, if $\mathcal{R} = (\Sigma, E \cup B, R)$, define $\mathcal{R}^{-1} = (\Sigma, E \cup B, R^{-1})$, where $R^{-1} = \{r \rightarrow l \text{ if } \varphi \mid (l \rightarrow r \text{ if } \varphi) \in R\}$. Then note that if \mathcal{R} satisfies conditions (1)–(2) and assuming the rules R^{-1} are ground coherent with the equations E modulo B , then \mathcal{R}^{-1} also satisfies conditions (1)–(2). Here is the key result.

Theorem 4. *Under the above assumptions on \mathcal{R}^{-1} , if $S_0, Q \in \text{PatPred}(\Omega, \Sigma)$ are constrained pattern predicates in standard form with $\text{vars}(S_0) \cap \text{vars}(Q) = \emptyset$, then $\llbracket Q \rrbracket$ is a co-invariant of $(\mathcal{C}_{\mathcal{R}, \text{State}}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ if: (i) $\llbracket Q \rrbracket \cap \llbracket S_0 \rrbracket = \emptyset$, and (ii) $(\mathcal{R}^{-1})_{\text{stop}} \models_{\square}^{\forall} Q\sigma \rightarrow^{\otimes} \llbracket Q \rrbracket$, where σ is a bijective renaming of variables such that $\text{vars}(Q) \cap \text{vars}(Q\sigma) = \emptyset$.*

The reduction of parametric invariants to unparameterized ones applies *mutatis mutandis*, to parametric co-invariants. For example, in the parametric version of the above theorem, where $Y = \text{vars}(S_0) \cap \text{vars}(Q)$, the checking of (i) now becomes checking $\llbracket Q \rrbracket \cap_Y \llbracket S_0 \rrbracket = \emptyset$ (see Section 3); and for (ii), proving the reachability formula $Q\sigma \rightarrow^{\otimes} \llbracket Q \rrbracket$, where σ is a bijective renaming of variables such that $\text{vars}(Q) \cap \text{vars}(Q\sigma) = Y$ and $\sigma(y) = y$ for each $y \in Y$.

5 A Sound Inference System

We present our inference system for all-path reachability, parametric on \mathcal{R} satisfying requirements (1)–(2) in Section 4, with rules $R = \{l_j \rightarrow r_j \text{ if } \phi_j\}_{j \in J}$ such that $l_j, r_j \in T_{\Omega}(X)$, $j \in J$. Variables of rules in R are always assumed disjoint from variables in reachability formulas; this can be ensured by renaming. The inference system has two proof rules. The $\text{STEP}^{\forall} + \text{SUBSUMPTION}$ proof rule allows taking one step of (symbolic) rewriting along all paths according to the rules in \mathcal{R} . The AXIOM proof rule allows the use of a trusted reachability formula to summarize multiple rewrite steps, and thus to handle repetitive behavior.

These proof rules derive sequents of the form $[\mathcal{A}, \mathcal{C}] \vdash_T u \mid \varphi \longrightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$, where \mathcal{A} and \mathcal{C} are finite sets of reachability formulas and T a pattern predicate

defining a set of T -terminating ground states. Formulas in \mathcal{A} are called *axioms* and those in \mathcal{C} are called *circularities*. We furthermore assume that in all reachability formulas $u \mid \varphi \longrightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ we have $\text{vars}(\psi_i) \subseteq \text{vars}(v_i) \cup \text{vars}(u \mid \varphi)$ for each i . According to the implicit quantification of the semantic relation \models_T^\forall , this means that any variable in ψ_i is either universally quantified and comes from the precondition $u \mid \varphi$, or is existentially quantified and comes from v_i only. This property is an invariant preserved by the two inference rules.

Proofs always begin with a set \mathcal{C} of formulas that we want to *simultaneously* prove, so that the proof effort only succeeds if all formulas in \mathcal{C} are proved. \mathcal{C} contains the main properties we want to prove as well as any auxiliary lemmas that may be needed to carry out the proof. The initial set of goals we want to prove is $[\emptyset, \mathcal{C}] \vdash_T \mathcal{C}$, which is a shorthand for the set of goals $\{[\emptyset, \mathcal{C}] \vdash_T u \mid \varphi \longrightarrow^{\otimes} \bigvee_i v_i \mid \psi_i \mid (u \mid \varphi \longrightarrow^{\otimes} \bigvee_i v_i \mid \psi_i) \in \mathcal{C}\}$. Thus, we start *without any axioms* \mathcal{A} , but we shall be able to use the formulas in \mathcal{C} as axioms in their own derivation *after* taking at least one step with the rewrite rules in \mathcal{R} .

A very useful key feature is that sequents $[\emptyset, \mathcal{C}] \vdash_T u \mid \varphi \longrightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$, whose formulas \mathcal{C} have been *postulated* (as the conjectures we want to prove) but not yet justified, are transformed by $\text{STEP}^\forall + \text{SUBSUMPTION}$ into sequents of the form $[\mathcal{C}, \emptyset] \vdash_T u' \mid \varphi' \longrightarrow^{\otimes} \bigvee_i v'_i \mid \psi'_i$, where now the formulas in \mathcal{C} *can be assumed valid*, and can be used in derivations with the AXIOM rule.

Verifying QLOCK's Mutual Exclusion. By Corollary 1, QLOCK's mutual exclusion can be verified by: (i) using pattern subsumption to check the trivial inclusion $\llbracket S_0 \rrbracket \subseteq \llbracket P \rrbracket$, and (ii) proving $\mathcal{R}_{\text{stop}} \models_{\square}^\forall P\sigma \rightarrow^{\otimes} [P]$, where σ is a sort-preserving bijective renaming of variables such that $\text{vars}(P) \cap \text{vars}(P\sigma) = \emptyset$. But since for QLOCK P is a disjunction, in our inference system this means proving from $\mathcal{R}_{\text{stop}}$ that $[\emptyset, \mathcal{C}] \vdash_{\square} \mathcal{C}$, where \mathcal{C} are the conjectures:

$$\langle n' \mid w' \mid i' \mid i' ; q' \rangle \mid \varphi' \rightarrow^{\otimes} [\langle n \mid w \mid i \mid i ; q \rangle \mid \varphi \vee \langle n \mid w \mid mt \mid q \rangle \mid \psi]$$

$$\langle n' \mid w' \mid mt \mid q \rangle \mid \psi' \rightarrow^{\otimes} [\langle n \mid w \mid i \mid i ; q \rangle \mid \varphi \vee \langle n \mid w \mid mt \mid q \rangle \mid \psi].$$

where $\varphi = \text{dupl}(n \ w \ i) \neq tt$, $\psi = \text{dupl}(n \ w) \neq tt$, and φ', ψ' are their obvious renamings.

Before explaining the $\text{STEP}^\forall + \text{SUBSUMPTION}$ proof rule we introduce some notational conventions. Assume T is the pattern predicate $T = \bigvee_j t_j \mid \chi_j$, with $\text{vars}(\chi_j) \subseteq \text{vars}(t_j)$, and let $R = \{l_j \rightarrow r_j \mid \phi_j\}_{j \in J}$, we then define:

$$\text{MATCH}(u, \{v_i\}_{i \in I}) \subseteq \{(i, \beta) \mid \beta \in [\text{vars}(v_i) \setminus \text{vars}(u) \rightarrow T_\Omega(X)] \text{ s.t. } u =_{E_\Omega \cup B_\Omega} v_i \beta\}$$

a *complete* set of $E_\Omega \cup B_\Omega$ -matches of u against the patterns v_i , and

$$\text{UNIFY}(u \mid \varphi', R) \equiv \{(j, \alpha) \mid \alpha \in \text{Unif}_{E_\Omega \cup B_\Omega}(u, l_j) \text{ and } (\varphi' \wedge \phi_j)\alpha \text{ satisfiable in } \mathcal{T}_{\Sigma/E \cup B}\}$$

a complete set of $E_\Omega \cup B_\Omega$ -unifiers of a pattern $u \mid \varphi'$ with the lefthand-sides of the rules in R with satisfiable associated constraints. Consider now the rule:

STEP[∀] + SUBSUMPTION

$$\frac{\bigwedge_{(j,\alpha) \in \text{UNIFY}(u|\varphi', R)} [\mathcal{A} \cup \mathcal{C}, \emptyset] \vdash_T (r_j | \varphi' \wedge \phi_j)\alpha \longrightarrow^{\otimes} \bigvee_i (v_i | \psi_i)\alpha}{[\mathcal{A}, \mathcal{C}] \vdash_T u | \varphi \longrightarrow^{\otimes} \bigvee_i v_i | \psi_i}$$

where $\varphi' \equiv \varphi \wedge \bigwedge_{(i,\beta) \in \text{MATCH}(u, \{v_i\})} \neg(\psi_i\beta)$. This inference rule allows us to take one step with the rules in \mathcal{R} . Intuitively, $u | \varphi'$ characterizes the states satisfying $u | \varphi$ that are not subsumed by any $v_i | \psi_i$; that is, states in the lefthand side of the current goal that have not yet reached the righthand side. Note that, according to Definition 6, $u | \varphi \longrightarrow^{\otimes} \bigvee_i v_i | \psi_i$ is semantically valid iff $u | \varphi' \longrightarrow^{\otimes} \bigvee_i v_i | \psi_i$ is valid. Thus, this inference rule only unifies $u | \varphi'$ with the lefthand sides of rules in R . We impose on this inference rule a side condition that $\bigvee_{j,\gamma \in \text{Unif}_{E_\Omega \cup B_\Omega}(u,t_j)} (\varphi' \wedge \chi_j)\gamma$ is unsatisfiable in $\mathcal{T}_{\Sigma/E \cup B}$, where $T = \bigvee_j t_j | \chi_j$ is the pattern predicate characterizing the chosen T -terminating states. This condition ensures that any state in $u | \varphi'$ has an \mathcal{R} -successor. Thus, a state in $u | \varphi'$ reaches on all T -terminating paths a state in $\bigvee_i v_i | \psi_i$ if all its successors do so. Each \mathcal{R} -successor is covered by one of $(r_j | \varphi' \wedge \phi_j)\alpha$. As an optimization, we check that $(\varphi' \wedge \phi_j)\alpha$ is satisfiable and we drop the ones which are not. Finally, we also assume that $\text{vars}((u | \varphi)\alpha) \cap \text{vars}((\bigvee_i v_i | \psi_i)\alpha) = \text{vars}((r_j | \varphi' \wedge \phi_j)\alpha) \cap \text{vars}((\bigvee_i v_i | \psi_i)\alpha)$. This parameter preservation condition ensures correct implicit quantification. Note that formulas in \mathcal{C} are added to \mathcal{A} , so that from now on they can be used by AXIOM. By using $E_\Omega \cup B_\Omega$ -unification, this inference rule is effectively performing narrowing of $u | \varphi'$ with rules R [47].

AXIOM

$$\frac{\bigwedge_j [\{u' | \varphi' \longrightarrow^{\otimes} \bigvee_j v'_j | \psi'_j\} \cup \mathcal{A}, \emptyset] \vdash_T v'_j\alpha | \varphi \wedge \psi'_j\alpha \longrightarrow^{\otimes} \bigvee_i v_i | \psi_i}{[\{u' | \varphi' \longrightarrow^{\otimes} \bigvee_j v'_j | \psi'_j\} \cup \mathcal{A}, \emptyset] \vdash_T u | \varphi \longrightarrow^{\otimes} \bigvee_i v_i | \psi_i}$$

if $\exists \alpha$ such that $u =_{E_\Omega \cup B_\Omega} u'\alpha$ and $\mathcal{T}_{\Sigma/E \cup B} \models \varphi \Rightarrow \varphi'\alpha$. This inference rule allows us to use a trusted formula in \mathcal{A} to summarize multiple transition steps. This is similar to how several transition steps would apply to a ground term, except that for ground terms we would check that $\varphi'\alpha$ is valid, whereas here we check that the condition φ implies $\varphi'\alpha$. Since φ is stronger than $\varphi'\alpha$, we add φ to $(v'_j | \psi'_j)\alpha$ (the result of using axiom $u' | \varphi' \longrightarrow^{\otimes} \bigvee_j v'_j | \psi'_j$). We assume that $u | \varphi \longrightarrow^{\otimes} \bigvee_i v_i | \psi_i$ and $u' | \varphi' \longrightarrow^{\otimes} \bigvee_j v'_j | \psi'_j$ do not share variables, which can always be guaranteed by renaming. For correct implicit quantification, as in STEP[∀] + SUBSUMPTION, we assume for each j the parameter preservation condition $\text{vars}(u | \varphi) \cap \text{vars}(\bigvee_i v_i | \psi_i) = \text{vars}(v'_j\alpha | \varphi \wedge \psi'_j\alpha) \cap \text{vars}(\bigvee_i v_i | \psi_i)$.

The soundness of STEP[∀] + SUBSUMPTION plus AXIOM is now the theorem:

Theorem 5. (Soundness) *Let \mathcal{R} be a rewrite theory, and \mathcal{C} a finite set of reachability formulas. If \mathcal{R} proves $[\emptyset, \mathcal{C}] \vdash_T \mathcal{C}$ then $\mathcal{R} \models_T^{\forall} \mathcal{C}$.*

QLOCK Proof Details. Using our implementation of the proof system (see Section 6), we can extract traces of completed proofs to help understand how such proofs work in practice. Recall that for QLOCK we had to prove $[\emptyset, \mathcal{C}] \vdash_{\square} \mathcal{C}$, where \mathcal{C} were the two already-discussed reachability formulas with respective preconditions the renamed disjuncts P'_i , $1 \leq i \leq 2$ in the invariant $P = P_1 \vee P_2$, and postcondition $[P]$, where $P_1 = \langle n \mid w \mid i \mid i ; q \rangle \mid \varphi$ and $P_2 = \langle n \mid w \mid mt \mid q \rangle \mid \psi$. As an example, we examine a branch of the proof tree for the sequent $[\emptyset, \mathcal{C}] \vdash_{\square} P'_1 \rightarrow^{\otimes} [P]$, where the dots represent omitted proof branches.

$$\frac{\frac{\frac{\emptyset}{\dots [\mathcal{C}, \emptyset] \vdash_{\square} [n'' \mid w'' \mid mt \mid q''] \mid \varphi' \wedge \psi'' \rightarrow^{\otimes} [P] \dots} \text{subsume}}{\dots [\mathcal{C}, \emptyset] \vdash_{\square} \langle n' \mid i' \mid w' \mid mt \mid q' \rangle \mid \varphi' \rightarrow^{\otimes} [P] \dots} \text{axiom: } P'_2 \rightarrow^{\otimes} [P]}{\dots [\mathcal{C}, \emptyset] \vdash_{\square} \langle n' \mid w' \mid i' \mid i ; q' \rangle \mid \varphi' \rightarrow^{\otimes} [P]} \text{step: } c2n$$

5.1 The Split Auxiliary Rule

The following SPLIT rule is an auxiliary proof rule that uses a formula ϕ to split a goal into two. SPLIT is a *validity-preserving* rule transforming a set \mathcal{G} of reachability logic goals to be proved (understood as a *conjunction*) into a *semantically equivalent* set of goals \mathcal{G}' , so that $\mathcal{R} \models_T^{\forall} \mathcal{G} \Leftrightarrow \mathcal{R} \models_T^{\forall} \mathcal{G}'$. This means that SPLIT *does not affect soundness*.

SPLIT

$$\frac{[\mathcal{A}, \mathcal{C}] \vdash_T u \mid \varphi \wedge \phi \rightarrow^{\otimes} A \quad [\mathcal{A}, \mathcal{C}] \vdash_T u \mid \varphi \wedge \neg\phi \rightarrow^{\otimes} A}{[\mathcal{A}, \mathcal{C}] \vdash_T u \mid \varphi \rightarrow^{\otimes} A}$$

subject to the condition $\text{vars}(u \mid \varphi) \cap \text{vars}(A) = \text{vars}(u \mid \varphi \wedge \phi) \cap \text{vars}(A)$.

Lemma 2. *In the above SPLIT rule, $\mathcal{R} \models_T^{\forall} \mathcal{G} \Leftrightarrow \mathcal{R} \models_T^{\forall} \mathcal{G}'$, where \mathcal{G} is the premise and \mathcal{G}' the conclusion.*

This still leaves open the question of when it would be advantageous to use the SPLIT rule and with what choice of ϕ . One attractive possibility is to use SPLIT to increase success in application attempts for the AXIOM rule. Suppose that we have tried to apply AXIOM with a substitution α such that $u =_{E_{\Omega} \cup B_{\Omega}} u' \alpha$, but the condition $\mathcal{T}_{\Sigma/E \cup B} \models \varphi \Rightarrow (\varphi' \alpha)$ does not hold. Suppose, however, that both $\varphi \wedge (\varphi' \alpha)$ and $\varphi \wedge \neg(\varphi' \alpha)$ are satisfiable in $\mathcal{T}_{\Sigma/E \cup B}$, and that $\text{vars}(u \mid \varphi) \cap \text{vars}(\bigvee_i v_i \mid \psi_i) = \text{vars}(u \mid \varphi \wedge (\varphi' \alpha)) \cap \text{vars}(\bigvee_i v_i \mid \psi_i)$. In such a case, we can first apply SPLIT to split $u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ into $u \mid \varphi \wedge (\varphi' \alpha) \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ and $u \mid \varphi \wedge \neg(\varphi' \alpha) \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$, and then apply AXIOM (checking parameter preservation) to the first of these two reachability formulas.

6 Prototype Implementation and Experiments

We have implemented the proof system in Maude [8]. We exploit the fact that rewriting logic is reflective, so that concepts such as terms, rewrite rules, signatures, and theories are directly expressible as data in the logic. This is supported by Maude’s `META-LEVEL` library. Our prototype tool takes as input (i) a reflected rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R, \phi)$ and (ii) a set of reachability formulas $\mathcal{C} = \{A_i \rightarrow^{\otimes} B_i\}_{i \in I}$ to be simultaneously proved.

The state of a reachability proof is represented as an associative-commutative set of proof sequents as defined in Section 5 plus some global state information (for example, theory \mathcal{R}). Given goal set \mathcal{C} , the initial proof state will be $\{[\emptyset, \mathcal{C}] \vdash_T A_i \rightarrow^{\otimes} B_i\}_{i \in I}$, that is, one sequent for each goal. Given the simplicity of the proof system, we need only perform a very simple proof search: until there are no pending goals, apply `AXIOM` as much as possible and then apply `STEP`^v + `SUBSUMPTION` if possible.

We of course need to mechanize the two proof rules. Internally, each proof rule is represented by a corresponding rewrite rule. Rule application is controlled by a metadata flag that indicates which rule to apply next. Our rules further require a finitary B -unification algorithm as well as an SMT solver to discharge $E \cup B$ constraints. Maude can perform unification modulo commutativity and associativity/commutativity with or without identity and in some cases associativity without identity. For SMT solving we use the variant satisfiability techniques in [30, 45] which allow us to handle any rewrite theory $\mathcal{R} = (\Sigma, E \cup B, R)$ satisfying conditions (1)–(2) in Section 4 and such that the equational theory $(\Sigma, E \cup B)$ has a convergent decomposition satisfying the finite variant property [12], and protects a constructor subtheory which we assume consists only of axioms B_Ω of the above-described kind. Note that this means that both validity and satisfiability of QF formulas in the initial algebra $\mathcal{T}_{\Sigma/E \cup B}$ are decidable [30]. Future versions of the tool will add other decision procedures and will support more general classes of rewrite theories $\mathcal{R} = (\Sigma, E \cup B, R)$. Furthermore, besides further automation, an inductive theorem prover backend will be added to reason about validity of formulas in $\mathcal{T}_{\Sigma/E \cup B}$.

In addition to the above proof representation, two other issues must be addressed. First, to ensure correct applications of unification, we uniquely rename all variables in rules in the theory \mathcal{R} and in goals \mathcal{C} by maintaining a counter recording the next fresh variable name. Second, recall that we assume that the rewrite theory \mathcal{R} has been Ω -abstracted as $\hat{\mathcal{R}}$. Therefore, we have implemented a method that can Ω -abstract many theories \mathcal{R} in practice.

To validate the feasibility of our approach we have verified properties for a collection of examples including various rewrite theories specifying distributed systems such as communication or mutual exclusion protocols and cyber-physical systems, and for some imperative programs using the rewriting logic semantics of the IMP programming language. Table 1 summarizes these experiments.

Table 1. Examples Verified by the Tool

Example Name	# Goals/Invariants	Time
Thermostat	0/4	Long
Bounded Retransmission Protocol	1/1	Short
Dijkstra’s Mutual Ex. Alg.	1/3	Short
Fault-Tolerant Comm. Protocol	1/5	Short
QLOCK Mutual Ex. Alg.	1/2	Short
Readers/Writers	1/2	Short
Fixed-Size Token Ring	1/2	Short
Unbounded Lamport’s Bakery	2/9	Long
IMP Swap Function	1/1	Short
IMP Min Function	1/1	Short
IMP Max Function	1/1	Short

7 Related Work and Conclusions

Reachability logic [43, 42, 46, 36] is a language-generic approach to program verification, parametric on the operational semantics of a programming language. Both Hoare logic and separation logic can be naturally mapped into reachability logic [43, 42]. This work further generalizes reachability logic from a programming-language-generic logic of programs to a rewrite-theory-generic logic to reason about both distributed system designs and programs, based on their rewriting logic semantics. This generalization is non-trivial and requires a number of new concepts and results such as the following: (i) relativization of terminating sequences to a chosen subset $\llbracket T \rrbracket$ of terminating states; (ii) solving the “invariant paradox,” to make it possible to reason about invariants and co-invariants of possibly non-terminating systems, and characterizing such invariants by means of reachability formulas through a theory transformation; and (iii) making it possible to achieve higher levels of automation by systematically basing the state predicates on positive Boolean combination of constrained constructor patterns of the form $u \mid \varphi$ with u a constructor term. In contrast, \mathbb{K} ’s version of reachability logic uses matching logic, which assumes a first-order model \mathcal{M} and its semantic satisfaction relation $\mathcal{M} \models \varphi$ as the basis of the reachability logic proof system. As discussed in Section 3, we choose $T_{\Sigma/E \cup B}$ as the model, rather than some general model \mathcal{M} , and systematically exploit the isomorphism $T_{\Sigma/E \cup B}|_{\Omega} \cong T_{\Omega/E_{\Omega} \cup B_{\Omega}}$, allowing us to use unification, matching, narrowing, and satisfiability procedures based on the typically much simpler initial algebra of constructors $T_{\Omega/E_{\Omega} \cup B_{\Omega}}$. This has the advantage that we can explicitly give the complete details of our inference rules (e.g. how $\text{STEP}^{\forall} + \text{SUBSUMPTION}$ checks the subsumption, or ensures that states have at least a successor), instead of relying on a general satisfaction relation \models on \mathcal{M} . The result is a simpler inference system with only two rules (instead of the eight in reachability logic). On the practical side, reachability logic has been previously implemented as part of the \mathbb{K} framework, and has only been instantiated with operational semantics

of programming languages and used for the purpose of program verification. In particular, the implementation in \mathbb{K} has several hand-crafted heuristics for reasoning about specific features of programming language, such as dynamically allocated memory (the “heap”). In spite of the fact that similar heuristics have not yet been added to the current prototype described in Section 6, the potential for automation of the constructor-based reachability logic approach has been demonstrated by the tool’s capacity to prove safety properties for a significant collection of distributed system designs, including various communication protocols, mutual exclusion protocols, and real-time systems. Of course, this is just a proof of concept: adding reasoning heuristics and further theorem proving support will be crucial to handle a much wider range of applications.

As mentioned in the Introduction, we have been inspired by the work in [26]. We agree on the common goal of making reachability logic rewrite-theory-generic, but differ on the methods used and their applicability. Main differences include: (1) the authors in [26] do not give an inference system but a verification algorithm manipulating goals, which makes it hard to compare both logics. (2) the theories to which the methods in [26] apply seem more restricted than the ones presented here. Roughly, (see their **Assumption 3**) [26] assumes restrictions akin to those imposed in [39] to allow “rewriting modulo SMT,” which limits the equational theories (Σ, E) that can be handled. (3) Matching is used throughout in [26] instead of unification. This means that, unless a formula has been sufficiently instantiated, no matching rule may exist, whereas unification with some rule is always possible in our case. (4) No method for proving invariants is given in [26]; solving the “invariant paradox” provides such a method.

Finally, there is a close connection between this work and various rewriting-based symbolic methods, including: (i) unification modulo FVP theories [16]; (ii) decidable satisfiability (and validity) of quantifier-free formulas in initial algebras [28, 9, 10, 2, 11, 35, 19, 20, 17, 1, 30]; (iii) narrowing-based reachability analysis [47]; (iv) narrowing-based proof of safety properties [37, 38]; (v) patterns and constrained-based conditional narrowing [35, 6]; and (vi) rewriting modulo SMT [39, 40]. Exploiting such connections, particularly with [16, 30, 45], has been essential to achieve the goals of this work.

In conclusion, this work advances the goal of making reachability logic available as a rewrite-theory-generic verification logic. The goals of wide applicability, invariant verification, simplicity, and mechanization of inference rules have been substantially advanced, but much work remains ahead. The feasibility of the approach has been validated with a prototype implementation using a suite of representative examples; but building a robust and highly effective reachability logic tool for rewrite theories specifying both distributed systems and programming languages is a considerably more ambitious goal. This, together with experimentation with a wider class of examples and case studies, will be the main focus of future research.

Acknowledgements. Partially supported by NSF Grants CNS 13-19109 and CNS 14-09416, and AFOSR Contract FA8750-11-2-0084.

References

1. Aoto, T., Stratulat, S.: Decision procedures for proving inductive theorems without induction. In: Proc. PPDP2014. pp. 237–248. ACM (2014)
2. Baader, F., Schulz, K.U.: Combination techniques and decision problems for disunification. *Theor. Comput. Sci.* 142(2), 229–255 (1995)
3. Baader, F., Siekmann, J.H.: Unification theory. In: Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 2, pp. 41–126. Oxford University Press (1994)
4. Baader, F., Snyder, W.: Unification theory. In: Handbook of Automated Reasoning (in 2 volumes), pp. 445–532. Elsevier and MIT Press (2001)
5. Bruni, R., Meseguer, J.: Semantic foundations for generalized rewrite theories. *Theor. Comput. Sci.* 360(1-3), 386–414 (2006)
6. Cholewa, A., Escobar, S., Meseguer, J.: Constrained narrowing for conditional equational theories modulo axioms. *Science of Computer Programming* 112, 24–57 (2015)
7. Cholewa, A., Meseguer, J., Escobar, S.: Variants of variants and the finite variant property. Tech. rep., CS Dept. University of Illinois at Urbana-Champaign (February 2014), available at <http://hdl.handle.net/2142/47117>
8. Clavel, M., Durán, F., Eker, S., Meseguer, J., Lincoln, P., Martí-Oliet, N., Talcott, C.: All About Maude. Springer LNCS Vol. 4350 (2007)
9. Comon, H., Lescanne, P.: Equational problems and disunification. *Journal of Symbolic Computation* 7, 371–425 (1989)
10. Comon, H.: Complete axiomatizations of some quotient term algebras. *Theor. Comput. Sci.* 118(2), 167–191 (1993)
11. Comon, H., Delor, C.: Equational formulae with membership constraints. *Inf. Comput.* 112(2), 167–216 (1994)
12. Comon-Lundth, H., Delaune, S.: The finite variant property: how to get rid of some algebraic properties, in Proc *RTA'05*, Springer LNCS 3467, 294–307, 2005
13. Dershowitz, N., Jouannaud, J.P.: Rewrite systems. In: van Leeuwen, J. (ed.) Handbook of Theoretical Computer Science, Vol. B, pp. 243–320. North-Holland (1990)
14. Durán, F., Meseguer, J.: On the church-rosser and coherence properties of conditional order-sorted rewrite theories. *J. Log. Algebr. Program.* 81(7-8), 816–850 (2012)
15. Ehrig, H., Mahr, B.: Fundamentals of Algebraic Specification 1. Springer (1985)
16. Escobar, S., Sasse, R., Meseguer, J.: Folding variant narrowing and optimal variant termination. *J. Algebraic and Logic Programming* 81, 898–928 (2012)
17. Falke, S., Kapur, D.: Rewriting induction + linear arithmetic = decision procedure. In: Gramlich, B., Miller, D., Sattler, U. (eds.) IJCAR. Lecture Notes in Computer Science, vol. 7364, pp. 241–255. Springer (2012)
18. Futatsugi, K.: Fostering proof scores in CafeOBJ. In: Proc. ICFEM 2010. vol. 6447, pp. 1–20. Springer LNCS (2010)
19. Giesl, J., Kapur, D.: Decidable classes of inductive theorems. In: Proc. IJCAR 2001. vol. 2083, pp. 469–484. Springer LNCS (2001)
20. Giesl, J., Kapur, D.: Deciding inductive validity of equations. In: Proc. CADE 2003. vol. 2741, pp. 17–31. Springer LNCS (2003)
21. Goguen, J., Meseguer, J.: Order-sorted algebra I. *Theoretical Computer Science* 105, 217–273 (1992)
22. Gutiérrez, R., Meseguer, J., Rocha, C.: Order-sorted equality enrichments modulo axioms. *Sci. Comput. Program.* 99, 235–261 (2015)

23. Hullot, J.M.: Canonical forms and unification. In: Bibel, W., Kowalski, R. (eds.) Proceedings, Fifth Conference on Automated Deduction, pp. 318–334. Springer-Verlag (1980), LNCS, Volume 87
24. Jouannaud, J.P., Kirchner, C., Kirchner, H.: Incremental construction of unification algorithms in equational theories. In: Proc. ICALP'83. pp. 361–373. Springer LNCS 154 (1983)
25. Jouannaud, J.P., Kirchner, C.: Solving equations in abstract algebras: A rule-based survey of unification. In: Computational Logic - Essays in Honor of Alan Robinson. pp. 257–321. MIT Press (1991)
26. Lucanu, D., Rusu, V., Arusoaie, A., Nowak, D.: Verifying reachability-logic properties on rewriting-logic specifications. In: Logic, Rewriting, and Concurrency - Essays dedicated to José Meseguer on the Occasion of His 65th Birthday. vol. 9200, pp. 451–474. Springer LNCS (2015)
27. Lucas, S., Meseguer, J.: Normal forms and normal theories in conditional rewriting. *J. Log. Algebr. Meth. Program.* 85(1), 67–97 (2016)
28. Maher, M.J.: Complete axiomatizations of the algebras of finite, rational and infinite trees. In: Proc. LICS '88. pp. 348–357. IEEE Computer Society (1988)
29. Meseguer, J.: Membership algebra as a logical framework for equational specification. In: Proc. WADT'97. pp. 18–61. Springer LNCS 1376 (1998)
30. Meseguer, J.: Variant-based satisfiability in initial algebras. In: Artho, C., Ölveczky, P. (eds.) Proc. FTSCS 2015. pp. 1–32. Springer CCIS 596 (2016), in press.
31. Meseguer, J., Roşu, G.: The rewriting logic semantics project. *Theoretical Computer Science* 373, 213–237 (2007)
32. Meseguer, J.: Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science* 96(1), 73–155 (1992)
33. Meseguer, J.: Twenty years of rewriting logic. *J. Algebraic and Logic Programming* 81, 721–781 (2012)
34. Meseguer, J., Rosu, G.: The rewriting logic semantics project: A progress report. *Inf. Comput.* 231, 38–69 (2013)
35. Meseguer, J., Skeirik, S.: Equational formulas and pattern operations in initial order-sorted algebras. In: Falaschi, M. (ed.) Proc. LOPSTR 2015. vol. 9527, pp. 36–53. Springer LNCS (2015)
36. Park, D., Stefanescu, A., Rosu, G.: KJS: a complete formal semantics of javascript. In: ACM. pp. 346–356. ACM (2015), <http://doi.acm.org/10.1145/2737924.2737991>
37. Rocha, C., Meseguer, J.: Proving safety properties of rewrite theories (2011), in Proc. CALCO 2011, Springer LNCS 6859, 314–328
38. Rocha, C., Meseguer, J.: Mechanical analysis of reliable communication in the alternating bit protocol using the maude invariant analyzer tool. In: Specification, Algebra, and Software - Essays Dedicated to Kokichi Futatsugi. Lecture Notes in Computer Science, vol. 8373, pp. 603–629. Springer (2014)
39. Rocha, C., Meseguer, J., Muñoz, C.A.: Rewriting modulo SMT and open system analysis. In: Proc. Rewriting Logic and Its Applications - 10th International Workshop, WRLA 2014. vol. 8663, pp. 247–262. Springer LNCS (2014)
40. Rocha, C., Meseguer, J., Muñoz, C.A.: Rewriting modulo SMT and open system analysis. Tech. Rep. <http://hdl.handle.net/2142/91638>, University of Illinois Computer Science Department (October 2016), to appear in *Logic and Algebraic Methods in Programming*
41. Rosu, G., Serbanuta, T.: An overview of the K semantic framework. *J. Log. Algebr. Program.* 79(6), 397–434 (2010), <http://dx.doi.org/10.1016/j.jlap.2010.03.012>
42. Rosu, G., Stefanescu, A.: Checking reachability using matching logic. In: Proc. OOPSLA 2012. pp. 555–574. ACM (2012)

43. Rosu, G., Stefanescu, A.: From hoare logic to matching logic reachability. In: Giannakopoulou, D., Méry, D. (eds.) FM. Lecture Notes in Computer Science, vol. 7436, pp. 387–402. Springer (2012)
44. Siekmann, J.H.: Unification theory. J. Symb. Comput. 7(3/4), 207–274 (1989)
45. Skeirik, S., Meseguer, J.: Metalevel algorithms for variant-based satisfiability. In: Lucanu, D. (ed.) Proc. WRLA 2016. Springer LNCS (2016), to appear. Preliminary version in <https://fmse.info.uaic.ro/events/WRLA2016/>
46. Stefanescu, A., Ștefan Ciobăcă, Mereuta, R., Moore, B.M., Serbanuta, T., Rosu, G.: All-path reachability logic. In: Proc. RTA-TLCA 2014. vol. 8560, pp. 425–440. Springer LNCS (2014)
47. Thati, P., Meseguer, J.: Symbolic reachability analysis using narrowing and its application to the verification of cryptographic protocols. J. Higher-Order and Symbolic Computation 20(1–2), 123–160 (2007)

A Proofs of Lemmas and Theorems

Proof of Lemma 1.

Proof. If $[u] \rightarrow_{\mathcal{R}} [v]$ corresponds to the topmost R, B -rewrite $u \rightarrow_{R, B} u'$, performed with a rewrite rule $l \rightarrow r$ if $\phi \in R$ and a ground substitution $\sigma \in [Y \rightarrow T_{\Sigma}]$, with Y the rule's variables, and such that $u =_{B, \Omega} l\sigma$, $u' = r\sigma$, and $[u!] = [v]$, this is also a rewrite with the rule $l \rightarrow r'$ if $\phi \wedge \hat{\theta}$, by extending σ to the fresh variables $X_P = \{x_p \mid p \in P\}$ with the assignments $x_p \mapsto (r\sigma)|_p$, so that we have $[u] \rightarrow_{\hat{\mathcal{R}}} [v]$.

Conversely, if $[u] \rightarrow_{\hat{\mathcal{R}}} [v]$ corresponds to the topmost \hat{R}, B -rewrite $u \rightarrow_{\hat{R}, B} w$, performed with rewrite rule $l \rightarrow r'$ if $\phi \wedge \hat{\theta}$ in \hat{R} and ground substitution $\rho \in [Y \uplus X_P \rightarrow T_{\Sigma}]$, so that $w = r'\rho$ and $[w!] = [v]$, then we can perform a corresponding rewrite with rule $l \rightarrow r$ if $\phi \in R$ and substitution $\rho|_Y$, because $E \cup B \models \phi\rho$. Furthermore, since $E \cup B \models \hat{\theta}\rho$, we must have $[w!] = [(r\rho)!] = [v]$, so that $[u] \rightarrow_{\mathcal{R}} [v]$. \square

Proof of Theorem 3.

Proof. In case (1), a state $\langle u_1, \dots, u_n \rangle \in \mathcal{C}_{\mathcal{R}, State}$ is reachable from $\llbracket S_0 \rrbracket$ iff $\llbracket [u_1, \dots, u_n] \rrbracket$ is reachable from $\llbracket S_0 \rrbracket$ in $\mathcal{C}_{\mathcal{R}, stop}$. Therefore, $\llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ iff $\mathcal{R}_{stop} \models_{\llbracket \cdot \rrbracket}^{\forall} S_0 \rightarrow^{\otimes} \llbracket P \rrbracket$.

In case (2), a state $\langle u_1, \dots, u_n \rangle \in \mathcal{C}_{\mathcal{R}, State}$ reachable from $\llbracket S_0 \rrbracket$ belongs to $\llbracket A \rrbracket$ iff $\llbracket [u_1, \dots, u_n] \rrbracket$ is reachable from $\llbracket S_0 \rrbracket$ in $\mathcal{C}_{\mathcal{R}, stop(A)}$. Therefore, $\llbracket A \rrbracket \Rightarrow \llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket S_0 \rrbracket$ iff $\mathcal{R}_{stop(A)} \models_{\llbracket \cdot \rrbracket}^{\forall} S_0 \rightarrow^{\otimes} \llbracket P \rrbracket$. \square

Proof of Corollary 1.

Proof. Suppose (i) and (ii) hold. Then, since $\llbracket P \rrbracket = \llbracket P\sigma \rrbracket$, we have $\llbracket S_0 \rrbracket \subseteq \llbracket P\sigma \rrbracket$, and by (ii) and Theorem 3, $\llbracket P \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}, State}, \rightarrow_{\mathcal{R}})$ from $\llbracket P\sigma \rrbracket$ and, *a fortiori*, from $\llbracket S_0 \rrbracket$. \square

Proof of Theorem 4.

Proof. Suppose $\llbracket Q \rrbracket$ is not a co-invariant from $\llbracket S_0 \rrbracket$. This exactly means that there are $[u] \in \llbracket S_0 \rrbracket$ and $[v] \in \llbracket Q \rrbracket$ such that $[u] \rightarrow_{\mathcal{R}}^* [v]$, or, equivalently, $[v] \rightarrow_{\mathcal{R}^{-1}}^* [u]$. But since, by (ii) and Theorem 3, $\llbracket Q\sigma \rrbracket$ is an invariant of $(\mathcal{C}_{\mathcal{R}^{-1}, State}, \rightarrow_{\mathcal{R}^{-1}})$ from $\llbracket Q\sigma \rrbracket = \llbracket Q \rrbracket$, we must have $[u] \in \llbracket Q\sigma \rrbracket$, which is impossible by (i). \square

Proof of Theorem 5.

Proof. We begin by introducing the following auxiliary notation

Definition 7. Let $u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ be a reachability formula. By definition, $\mathcal{R} \models_T^{\forall, n} u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ iff for each $[u_0] = [u\rho!] \in \llbracket u \mid \varphi \rrbracket$ and for each T -terminating sequence $[u_0] \rightarrow_{\mathcal{R}} [u_1] \rightarrow_{\mathcal{R}} \dots \rightarrow_{\mathcal{R}} [u_k]$ with $k \leq n$, there exist $0 \leq j \leq k$, i , and $[w] = [v_i\tau!] \in \llbracket v_i \mid \psi_i \rrbracket$ such that $[u_j] = [w]$ and $\rho|_{Y \cap Z} =_{E_{\Omega} \cup B_{\Omega}} \tau|_{Y \cap Z}$, where $Y = \text{vars}(u \mid \varphi)$ and $Z = \text{vars}(v_i \mid \psi_i)$. By convention, $\mathcal{R} \models_T^{\forall, -1} u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ always holds.

With this notation, we state the following auxiliary lemma:

Lemma 3. Let $[\mathcal{A}, \mathcal{C}] \vdash_T u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ be a sequent derived by our inference system for \mathcal{R} . If $\mathcal{R} \models_T^{\forall, n} \mathcal{A}$ and $\mathcal{R} \models_T^{\forall, n-1} \mathcal{C}$, then $\mathcal{R} \models_T^{\forall, n} u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$.

Proof. We prove the lemma by contradiction. Assume it does not hold, and let n_{min} be the minimal n for which the lemma does not hold. Further, let $[\mathcal{A}, \mathcal{C}] \vdash_T u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ be a sequent with a minimal proof tree \mathcal{P} for which the lemma does not hold for n_{min} . Thus, the lemma holds for any $n < n_{min}$, and for any sequent derived by a sub-proof of \mathcal{P} for $n \leq n_{min}$. Finally, let $[u_0] = [u\rho!] \in \llbracket u \mid \varphi \rrbracket$ and $[u_0] \rightarrow_{\mathcal{R}} [u_1] \rightarrow_{\mathcal{R}} \dots \rightarrow_{\mathcal{R}} [u_n]$ a T -terminating path with $n \leq n_{min}$. Next, we show that there exist $0 \leq j \leq n$ and $[w] = [v_i\tau!] \in \llbracket v_i \mid \psi_i \rrbracket$ such that $[u_j] = [w]$ and ρ and τ agree modulo $E_{\Omega} \cup B_{\Omega}$ on $\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i)$, which is a contradiction and completes the proof.

We distinguish the following cases according to the last proof rule applied.

STEP ^{\forall} + SUBSUMPTION. First, notice that

$$\varphi \Leftrightarrow (\varphi \wedge \bigvee_{(i, \beta) \in \text{MATCH}(u, \{v_i\})} \psi_i \beta) \vee \varphi'$$

If ρ satisfies the first part of the disjunction, then it follows that ρ satisfies $\psi_i \beta$ for some i and β . Since β is a matching substitution with domain $\text{vars}(v_i) \setminus \text{vars}(u)$, we can pick any τ extending $\rho|_{\text{vars}(u \mid \varphi)} \uplus \beta\rho$, and we have that τ satisfies ψ_i . Further, since $u_0 =_{E_{\Omega} \cup B_{\Omega}} u\rho$ and $u =_{E_{\Omega} \cup B_{\Omega}} v_i\beta$, we have that $u_0 =_{E_{\Omega} \cup B_{\Omega}} v_i\tau$. Thus, $[u_0] = [v_i\tau!] \in \llbracket v_i \mid \psi_i \rrbracket$, and in this case we are done. In the second case, it must be the case that $[u_0] = [u\tau!] \in \llbracket u \mid \varphi' \rrbracket$.

We notice that $[u_0] \rightarrow_{\mathcal{R}} [u_1]$ by rule $l_j \rightarrow r_j$ if $\phi_j \in \mathcal{R}$ iff there exist a ground substitution τ such that $u_0 =_{E_{\Omega} \cup B_{\Omega}} l_j\tau$ and $E \cup B \models \phi_j\tau$, or equivalently, iff $[u_0] = [l_j\tau!] \in \llbracket l_j \mid \phi_j \rrbracket$. Since $u_0 =_{E_{\Omega} \cup B_{\Omega}} u\rho$, it follows that $u\rho =_{E_{\Omega} \cup B_{\Omega}} l_j\tau$, and since $\rho|_{\text{vars}(u \mid \varphi')} \cap \tau|_{\text{vars}(l_j \mid \phi_j)} = \emptyset$, it follows that $[u_0] = [l_j\tau!] \in \llbracket l_j \mid \phi_j \rrbracket$

iff there exist $\alpha \in \text{Unif}_{E_\Omega \cup B_\Omega}(u, l_j)$ and a ground substitution θ such that $\alpha\theta$ agrees with ρ on $\text{vars}(u \mid \varphi')$ and with τ on $\text{vars}(l_j \mid \phi_j)$ modulo $E_\Omega \cup B_\Omega$. Then $u_0 =_{E_\Omega \cup B_\Omega} u\alpha\theta =_{E_\Omega \cup B_\Omega} l_j\alpha\theta =_{E_\Omega \cup B_\Omega} l_j\tau$, and, likewise, $r_j\tau =_{E_\Omega \cup B_\Omega} r_j\alpha\theta$. Since $[u_1] = [r_j\alpha\theta!_{E_\Omega, B_\Omega}]$, it follows that $[u_1] = [r_j\alpha\theta!] \in \llbracket r_j \mid \varphi' \wedge \phi_j \rrbracket$. Therefore, we can conclude that for any $[u_0] \rightarrow_{\mathcal{R}} [u_1]$, there exist some $(j, \alpha) \in \text{UNIFY}(u \mid \varphi', \mathcal{R})$ and some θ such that $[u_1] = [r_j\alpha\theta!] \in \llbracket (r_j \mid \varphi' \wedge \phi_j)\alpha \rrbracket$.

Recall that we assume that $[u_0]$ is T -terminating, i.e., $[u_0] = [t_j\tau!] \in \llbracket t_j \mid \chi_j \rrbracket$ for some j . Similar to the above reasoning, that holds iff there exists some $\gamma \in \text{Unif}_{E_\Omega \cup B_\Omega}(u, t_j)$ and a ground substitution θ such that $\gamma\theta$ agrees with ρ on $\text{vars}(u \mid \varphi')$ and with τ on $\text{vars}(t_j \mid \chi_j)$ modulo $E_\Omega \cup B_\Omega$. It would then follow that θ satisfies $(\varphi' \wedge \chi_j)\gamma$, which contradicts the side-condition of $\text{STEP}^\forall + \text{SUBSUMPTION}$, which states that $(\varphi' \wedge \chi_j)\gamma$ are unsatisfiable in $\mathcal{T}_{\Sigma/E \cup B}$. Therefore, we can assume that $[u_0]$ has a \mathcal{R} -successor and that $n_{\min} \geq 1$.

Since $\mathcal{R} \models_T^{\forall, n_{\min}} \mathcal{A}$ and $\mathcal{R} \models_T^{\forall, n_{\min}-1} \mathcal{C}$, it follows that $\mathcal{R} \models_T^{\forall, n_{\min}-1} \mathcal{A} \cup \mathcal{C}$. The lemma holds for $n_{\min} - 1$, thus $\mathcal{R} \models_T^{\forall, n_{\min}-1} (r_j \mid \varphi' \wedge \phi_j)\alpha \rightarrow^{\otimes} \bigvee_i (v_i \mid \psi_i)\alpha$. Since $[u_1] \rightarrow_{\mathcal{R}} [u_1] \rightarrow_{\mathcal{R}} \dots \rightarrow_{\mathcal{R}} [u_n]$ is a T -terminating path of length at most $n_{\min} - 1$, we have that there exist i and $1 \leq k \leq n$ and τ such that $[u_k] = [v_j\alpha\tau!] \in \llbracket (v_i \mid \psi_i)\alpha \rrbracket$ and, modulo $E_\Omega \cup B_\Omega$, τ agrees with θ on $\text{vars}((r_j \mid \varphi' \wedge \phi_j)\alpha) \cap \text{vars}((v_i \mid \psi_i)\alpha)$. It follows that $[u_k] = [v_j\alpha\tau!] \in \llbracket v_i \mid \psi_i \rrbracket$. Moreover, since $\text{STEP}^\forall + \text{SUBSUMPTION}$ guarantees that $\text{vars}((\text{vars}(u \mid \varphi) \cap \text{vars}(\bigvee_i v_i \mid \psi_i))\alpha) \subseteq \text{vars}((r_j \mid \varphi' \wedge \phi_j)\alpha)$ and we have that τ agrees with θ on $\text{vars}((\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i))\alpha) \cap \text{vars}((v_i \mid \psi_i)\alpha)$. Further, $\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i) \subseteq \text{vars}(v_i \mid \psi_i)$, we have $\text{vars}((\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i))\alpha) \subseteq \text{vars}((v_i \mid \psi_i)\alpha)$, and thus, modulo $E_\Omega \cup B_\Omega$, τ agrees with θ on $\text{vars}((\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i))\alpha)$. We can conclude that, modulo $E_\Omega \cup B_\Omega$, $\alpha\theta$ agrees with $\alpha\tau$ on $\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i)$, and we are done.

AXIOM. Since $u =_{E_\Omega \cup B_\Omega} u'\alpha$ and $u_0 =_{E_\Omega \cup B_\Omega} u\rho$ we have that $u_0 =_{E_\Omega \cup B_\Omega} u'\alpha\rho$. Further, since $\mathcal{T}_{\Sigma/E \cup B} \models \varphi \Rightarrow \varphi'\alpha$ and $E \cup B \models \varphi\rho$, we have that $E \cup B \models \varphi'\alpha\rho$. Thus, $[u_0] = [u'\alpha\rho!] \in \llbracket u' \mid \varphi' \rrbracket$. Since $\mathcal{R} \models_T^{\forall, n_{\min}} u' \mid \varphi' \rightarrow^{\otimes} \bigvee_j v'_j \mid \psi'_j$, there exists j and $0 \leq k \leq n$ and θ such that $[u_k] = [v'_j\alpha\theta!] \in \llbracket v'_j \mid \psi'_j \rrbracket$ and $\alpha\rho$ and $\alpha\theta$ agree on $\text{vars}(u' \mid \varphi') \cap \text{vars}(v'_j \mid \psi'_j)$ modulo $E_\Omega \cup B_\Omega$. Since $E \cup B \models \varphi\rho$, we can conclude that $[u_k] = [v'_j\alpha\theta!] \in \llbracket v'_j\alpha \mid \varphi \wedge \psi'_j\alpha \rrbracket$. We also have that ρ and θ agree on $\text{vars}((\text{vars}(u' \mid \varphi') \cap \text{vars}(v'_j \mid \psi'_j))\alpha)$ modulo $E_\Omega \cup B_\Omega$. Since the proof tree deriving $v'_j\alpha \mid \varphi \wedge \psi'_j\alpha \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ is a subproof of \mathcal{P} , the lemma must hold. Thus, we have that there exists i and $k \leq m \leq n$ and τ such that $[u_m] = [v_i\tau!] \in \llbracket v_i \mid \psi_i \rrbracket$ and τ agrees with θ on $\text{vars}(v'_j\alpha \mid \varphi \wedge \psi'_j\alpha) \cap \text{vars}(v_i \mid \psi_i)$. Moreover, since **AXIOM** guarantees that $\text{vars}(u \mid \varphi) \cap \text{vars}(\bigvee_i v_i \mid \psi_i) \subseteq \text{vars}(v'_j\alpha \mid \varphi \wedge \psi'_j\alpha)$, we have that, modulo $E_\Omega \cup B_\Omega$, ρ agrees with θ on $\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i)$ and θ agrees with τ on $\text{vars}(u \mid \varphi) \cap \text{vars}(v_i \mid \psi_i)$, and we are done. \square

Now we prove the main result (Theorem 5) using Lemma 3. Indeed, assume by contradiction that the theorem does not hold. Then, there must be a formula $u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i \in \mathcal{C}$ such that $[\emptyset, \mathcal{C}] \vdash_T u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i$ is derived for \mathcal{R} by our inference system, but $\mathcal{R} \not\models u \mid \varphi \rightarrow^{\otimes} \bigvee_i v_i \mid \psi_i \in \mathcal{C}$. Further, let

n_{min} be the minimal natural number n for which $\mathcal{R} \not\models_n^{\forall} u \mid \varphi \longrightarrow^{\otimes} \bigvee_i v_i \mid \psi_i \in \mathcal{C}$. Then $\mathcal{R} \models_T^{\forall, n_{min}-1} \mathcal{C}$ (recall that if $n_{min} = 0$, then $\mathcal{R} \models_T^{\forall, -1} \mathcal{C}$ holds by convention). Thus, by Lemma 3, we have that $\mathcal{R} \models_T^{\forall, n_{min}} \mathcal{C}$, and in particular, that $\mathcal{R} \models_T^{\forall, n_{min}} \varphi \longrightarrow^{\otimes} \bigvee_i \psi_i$. This is a contradiction with the definition of n_{min} , and it completes the proof. \square

Proof of Lemma 2

Proof. Since φ is logically equivalent to $(\varphi \wedge \phi) \vee (\varphi \wedge \neg\phi)$ we have $\llbracket u \mid \varphi \rrbracket = \llbracket u \mid \varphi \wedge \phi \rrbracket \cup \llbracket u \mid \varphi \wedge \neg\phi \rrbracket$. The lemma then follows easily from Definition 6, using the parameter preservation condition. \square