DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN
UNIVERSIDAD POLITÉCNICA DE VALENCIA

P.O. Box: 22012 E-46071 Valencia (SPAIN)

Informe Técnico / Technical Report

Ref. No: DSIC-II/05/07 Pages: 24

Title: Symbolic Model Checking of Infinite-State Systems Using Narrowing

Authors: S. Escobar and J. Meseguer

Date: 02/15/2007

Keywords: symbolic model checking, rewriting logic, narrowing

VºBº
Leader of Reasearch Group

Author (s):

Symbolic Model Checking of Infinite-State Systems Using Narrowing

Santiago Escobar[1] and José Meseguer[2]

[1] Universidad Politécnica de Valencia, Spain. sescobar@dsic.upv.es
[2] University of Illinois at Urbana-Champaign, USA. meseguer@cs.uiuc.edu

Abstract. Rewriting is a general and expressive way of specifying concurrent systems, where concurrent transitions are axiomatized by rewrite rules. Narrowing is a complete symbolic method for model checking reachability properties. We show that this method can be reinterpreted as a *lifting simulation* relating the original system and the symbolic system associated to the narrowing transitions. Since the narrowing graph can be infinite, this lifting simulation only gives us a semi-decision procedure for the failure of invariants. However, we propose new methods for folding the narrowing tree that can in practice result in finite systems that symbolically simulate the original system and can be used to algorithmically verify its properties. We also show how both narrowing and folding can be used to symbolically model check systems which, in addition, have state predicates, and therefore correspond to Kripke structures on which $ACTL^*$ and LTL formulas can be algorithmically verified using such finite symbolic abstractions.

1 Introduction

Model checking techniques have proved enormously effective in concurrent system verification. However, the standard model checking algorithms only work when the set of states reachable from the given initial state is finite. Various model checking techniques for infinite-state systems exist, but they are less developed than finite-state techniques and tend to place stronger limitations on the kind of systems and/or the properties that can be model checked (see Section 1.1).

In this work we adopt the rewriting logic point of view, in which a concurrent system can always be axiomatized as a rewrite theory modulo some equational axioms, with system transitions described by rewrite rules. We then propose a new narrowing-based method for model checking such, possibly infinite-state, systems under reasonable assumptions. The key insight is that the well-known theorem on the completeness of narrowing (which for rewrite theories whose rules need not be convergent has to satisfy a topmost restriction) can be reinterpreted as a *lifting simulation* between two systems, namely, between the initial model associated to the rewrite theory (which describes our system of interest), and a "symbolic abstraction" of such a system by the narrowing relation.

The narrowing relation itself may still lead to an infinite-state system. Even then, narrowing already gives us a semi-decision procedure for finding failures of invariants. To obtain a finite-state abstraction, we then define a second simulation by *folding* the narrowing-based abstraction, using a generalization criterion to fold the possibly infinite narrowing tree into a finite graph. There is no guarantee that such a folding will always be finite. But we think that such foldings can be finite in many practical cases and give several examples of finite concurrent system abstractions of infinite systems that can be obtained in this way and can be used to verify properties of infinite systems.

Our work applies not only to the model checking of invariants, but also to the model checking of $ACTL^*$ and LTL temporal logic formulas; not just for one initial state, but for a possibly infinite, symbolically described set of initial states. We therefore also provide results about the $ACTL^*$ and LTL model checking of concurrent systems axiomatized as rewrite theories. For such temporal logic model checking we have to perform narrowing in two different dimensions: (i) in the dimension of *transitions*, as already explained above; and (ii) in the dimensions of *state predicates*, because they are not defined in general for arbitrary terms with variables, but only for suitable substitution instances. Again, our narrowing techniques, when successful in folding the system into a finite-state abstraction, allow the use of standard model checking algorithms to verify $ACTL^*$ and LTL properties of the corresponding infinite-state systems.

After some preliminaries in Section 2, we consider narrowing for model checking invariants of transition systems in Section 3, and narrowing for model checking temporal logic formulas on Kripke structures in Section 4. We conclude in Section 5. Throughout we use Lamport's infinite-state "bakery" protocol as the source of various examples. Appendix A discusses other examples based on a readers-writers protocol.

1.1 Related work

The idea that narrowing in its reachability sense should be used as a method for analyzing concurrent systems and should fit within a wider spectrum of analysis capabilities, was suggested in [37,16], and was fully developed in [34,35]. The application of this idea to the verification of cryptographic protocols has been further developed by the authors in collaboration with Catherine Meadows and has been used as the basis of the Maude-NPA protocol analyzer [21]. In relation to such previous work, we contribute several new ideas, including the use of lifting simulations, the folding of the narrowing graph by a generalization criterion, and the new techniques for the verification of $ACTL^*$ and LTL properties.

The methods proposed in this paper are complementary to other infinite-state model checking methods, of which narrowing is one. What narrowing has in common with various infinite-state model checking analyses is the idea of representing sets of states *symbolically*, and to perform reachability analysis to verify properties. The symbolic representations vary from approach to approach. String and multiset grammars are often used to symbolically compute reachability sets, sometimes in conjunction with descriptions of the systems as rewrite theories [6,5], and sometimes in conjunction with learning algorithms [44]. Tree automata are also used for symbolic representation [25,40]. In general, like narrowing, some of these methods are only semi-decision procedures; but by restricting the classes of systems and/or the properties being analyzed, and by sometimes using acceleration or learning techniques, actual algorithms can be obtained for suitable subclasses: see the references above and also [7,8,20,24].

Two infinite-state model checking approaches closer in spirit to ours are: (i) the "constraint-based multiset rewriting" of Delzanno [15,14], where the infinity of a concurrent system is represented by the use of constraints (over integer or real numbers) and reachability analysis is performed by rewriting with a constraint store to which more constraints are added and checked for satisfiability or failure; and (ii) the logic-programming approach of [4], where simulations/bisimulations of labeled transition systems and symbolic representations of them using terms with variables and logic programming are studied. In spite of their similarities, the technical approaches taken in (i) and (ii) are quite different from ours. In (i), the analogue of narrowing is checking satisfiability of the constraint store; whereas in (ii) the main focus is on analyzing process calculi and on developing effective techniques using tabled logic programming to detect when a simulation or bisimulation exists.

Our work is also related to abstraction techniques, e.g., [9,31,26,29,41], which can sometimes collapse an infinite-state system into a finite-state one. In particular, it is related to, and complements, abstraction techniques for rewrite theories such as [39,33,23]. In fact, all the simulations we propose, especially the ones involving folding, can be viewed as suitable abstractions. From this point of view, our results provide new methods for automatically defining correct abstractions in a symbolic way.

There is, finally, related work on computing finite representations of the search space associated by narrowing to an expression in a rewrite theory, e.g., for computing regular expressions denoting a possibly infinite set of unifiers in [3], or for partial evaluation in [43,2,30,1]. However, these works have a different motivation and do not consider applications to simulation/bisimulation issues, although they contain notions of correctness/completeness suitable for such applications.

2 Preliminaries

We follow the classical notations and terminology from [17,42] for term rewriting and from [36,38] for rewriting logic and order-sorted notions. We assume an *order-sorted signature* Σ with a finite poset of sorts $(\mathsf{S}, \leq)$ and a finite number of function symbols. We furthermore assume that: (i) each connected component in the poset ordering has a top sort, and for each $\mathsf{s} \in \mathsf{S}$ we denote by $[\mathsf{s}]$ the top sort above s; and (ii) for each operator declaration $f : \mathsf{s}_1 \times \ldots \times \mathsf{s}_n \to \mathsf{s}$ in Σ, there is also a declaration $f : [\mathsf{s}_1] \times \ldots \times [\mathsf{s}_n] \to [\mathsf{s}]$. We assume an S-sorted family $\mathcal{X} = \{\mathcal{X}_\mathsf{s}\}_{\mathsf{s} \in \mathsf{S}}$ of disjoint variable sets with each $\mathcal{X}_\mathsf{s}$ countably infinite. $\mathcal{T}_\Sigma(\mathcal{X})_\mathsf{s}$ is the set of terms of sort s, and $\mathcal{T}_{\Sigma,\mathsf{s}}$ is the set of ground terms of sort s. We write

$\mathcal{T}_{\Sigma}(\mathcal{X})$ and $\mathcal{T}_{\Sigma}$ for the corresponding term algebras. The set of positions of a term t is written $Pos(t)$ and the set of non-variable positions $Pos_{\Sigma}(t)$. The root of a term is $\varLambda$. The subterm of t at position p is $t|_p$ and $t[u]_p$ is the subterm $t|_p$ in t replaced by u. A *substitution* σ is a sorted mapping from a finite subset of $\mathcal{X}$, written $Dom(\sigma)$, to $\mathcal{T}_{\Sigma}(\mathcal{X})$. The set of variables introduced by σ is $Ran(\sigma)$. The identity substitution is *id*. Substitutions are homomorphically extended to $\mathcal{T}_{\Sigma}(\mathcal{X})$. The restriction of σ to a set of variables V is $\sigma|_V$.

A *Σ-equation* is an unoriented pair $t = t'$, where $t, t' \in \mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{s}}$ for some sort $\mathsf{s} \in \mathsf{S}$. Given Σ and a set E of Σ-equations such that $\mathcal{T}_{\Sigma,\mathsf{s}} \neq \emptyset$ for every sort s, order-sorted equational logic induces a congruence relation $=_E$ on terms $t, t' \in \mathcal{T}_{\Sigma}(\mathcal{X})$ (see [38]). Throughout this paper we assume that $\mathcal{T}_{\Sigma,\mathsf{s}} \neq \emptyset$ for every sort s. The *E-subsumption* order on terms $\mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{s}}$, written $t \preccurlyeq_E t'$ (meaning that t' is more general than t), holds if $\exists \sigma : t =_E \sigma(t')$. The E-renaming equivalence on terms $\mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{s}}$, written $t \approx_E t'$, holds if $t \preccurlyeq_E t'$ and $t' \preccurlyeq_E t$. We extend $=_E$, $\approx_E$, and $\preccurlyeq_E$ to substitutions in the expected way. An *E-unifier* for a Σ-equation $t = t'$ is a substitution σ s.t. $\sigma(t) =_E \sigma(t')$. A *complete* set of E-unifiers of an equation $t = t'$ is written $CSU_E(t = t')$. We say $CSU_E(t = t')$ is *finitary* if it contains a finite number of E-unifiers. This notion can be extended to several equations, written $CSU_E(t_1 = t'_1 \wedge \cdots \wedge t_n = t'_n)$.

A *rewrite rule* is an oriented pair $l \to r$, where $l \notin \mathcal{X}$, $l, r \in \mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{s}}$ for some sort $\mathsf{s} \in \mathsf{S}$, and $Var(r) \subseteq Var(l)$. An *(unconditional) order-sorted rewrite theory* is a triple $\mathcal{R} = (\Sigma, E, R)$ with Σ an order-sorted signature, E a set of Σ-equations, and R a set of rewrite rules. A *topmost rewrite theory* is a rewrite theory s.t. for each $l \to r \in R$, $l, r \in \mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$ for a top sort State, and no operator in Σ has State as an argument sort.

The rewriting relation $\to_R$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ is $t \xrightarrow{p}_R t'$ (or $\to_R$) if $p \in Pos_{\Sigma}(t)$, $l \to r \in R$, $t|_p = \sigma(l)$, and $t' = t[\sigma(r)]_p$. The relation $\to_{R/E}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ is $=_E ; \to_R ; =_E$. Note that $\to_{R/E}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ induces a relation $\to_{R/E}$ on $\mathcal{T}_{\Sigma/E}(\mathcal{X})$ by $[t]_E \to_{R/E} [t']_E$ iff $t \to_{R/E} t'$. When $\mathcal{R} = (\Sigma, E, R)$ is a topmost rewrite theory we can safely restrict ourselves to the rewriting relation $\to_{R,E}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$, where $t \xrightarrow{p}_{R,E} t'$ (or $\to_{R,E}$) if $p \in Pos_{\Sigma}(t)$, $l \to r \in R$, $t|_p =_E \sigma(l)$, and $t' = t[\sigma(r)]_p$. Note that $\to_{R,E}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ induces a relation $\to_{R,E}$ on $\mathcal{T}_{\Sigma/E}(\mathcal{X})$ by $[t]_E \to_{R,E} [t']_E$ iff $\exists w \in \mathcal{T}_{\Sigma}(\mathcal{X}) : t \to_{R,E} w$ and $w =_E t'$. The narrowing relation $\leadsto_R$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ is $t \overset{p,\sigma}{\leadsto}_R t'$ (or $\overset{\sigma}{\leadsto}_R$, $\leadsto_R$) if $p \in Pos_{\Sigma}(t)$, $l \to r \in R$, $\sigma \in CSU_{\emptyset}(t = t')$, and $t' = \sigma(t[r]_p)$. Assuming that E has a finitary and complete unification algorithm, the narrowing relation $\leadsto_{R,E}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ is $t \overset{p,\sigma}{\leadsto}_{R,E} t'$ (or $\overset{\sigma}{\leadsto}_{R,E}$, $\leadsto_{R,E}$) if $p \in Pos_{\Sigma}(t)$, $l \to r \in R$, $\sigma \in CSU_E(t|_p = l)$, and $t' = \sigma(t[r]_p)$. Note that $\leadsto_{R,E}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ induces a relation $\leadsto_{R,E}$ on $\mathcal{T}_{\Sigma/E}(\mathcal{X})$ by $[t]_E \overset{\sigma}{\leadsto}_{R,E} [t']_E$ iff $\exists w \in \mathcal{T}_{\Sigma}(\mathcal{X}) : t \overset{\sigma}{\leadsto}_{R,E} w$ and $w =_E t'$. Note that, since we will only consider topmost rewrite theories, we avoid any coherence problems, and, as pointed above for $\to_{R/E}$ and $\to_{R,E}$, the narrowing relation $\leadsto_{R,E}$ achieves the same effect as a more general narrowing relation $\leadsto_{R/E}$ (see [35]).

3 Narrowing-based Reachability Analysis

A rewrite theory $\mathcal{R} = (\Sigma, E, R)$ specifies a transition system $\mathcal{T}_{\mathcal{R}}$ whose states are elements of the initial algebra $\mathcal{T}_{\Sigma/E}$, and whose transitions are specified by R. Before discussing the narrowing-based reachability analysis of the system $\mathcal{T}_{\mathcal{R}}$, we review some basic notions about transition systems.

Definition 1 (Transition System). *A* transition system *is written* $\mathcal{A} = (A, \rightarrow)$, *where* A *is a set of states, and* $\rightarrow$ *is a transition relation between states, i.e.,* $\rightarrow \subseteq A \times A$. *We write* $\mathcal{A} = (A, \rightarrow, I)$ *when* $I \subseteq A$ *is a set of initial states.*

Frequently, we will restrict our attention to a set of initial states in the transition system and, therefore, to the subsystem of states and transitions reachable from the initial states. However, we can obtain a more abstract description of such reachable subsystem by using a *folding relation* in order to shrink the associated transition system, i.e., to collapse several states into a previously seen state according to some criteria.

Definition 2 (Folding Reachable Transition Subsystem). *Given* $\mathcal{A} = (A, \rightarrow, I)$ *and a relation* $G \subseteq A \times A$*, the reachable subsystem from* I *in* A *with folding* G *is written* $\mathcal{R}each^{G}_{\mathcal{A}}(I) = (Reach^{G}_{\rightarrow}(I), \rightarrow^{G}, I)$*, where*

$$Reach^{G}_{\rightarrow}(I) = \textstyle\bigcup_{k \leq n} Frontier^{G}_{\rightarrow}(I)_n,$$

$$\begin{aligned}
&Frontier^{G}_{\rightarrow}(I)_0 = I,\\
&Frontier^{G}_{\rightarrow}(I)_{n+1} = \{y \in A \mid (\exists z \in Frontier^{G}_{\rightarrow}(I)_n : z \rightarrow y) \wedge\\
&\qquad\qquad\qquad\qquad\qquad (\nexists k \leq n, w \in Frontier^{G}_{\rightarrow}(I)_k : y \; G \; w)\},
\end{aligned}$$

$$\rightarrow^{G} = \textstyle\bigcup_{n \in \mathbb{N}} \rightarrow^{G}_{n+1},$$

$$x \rightarrow^{G}_{n+1} y \begin{cases} \textit{if } x \in Frontier^{G}_{\rightarrow}(I)_n, y \in Frontier^{G}_{\rightarrow}(I)_{n+1}, x \rightarrow y; \;\; \textit{or} \\ \textit{if } x \in Frontier^{G}_{\rightarrow}(I)_n, y \notin Frontier^{G}_{\rightarrow}(I)_{n+1}, \\ \quad \exists k \leq n : y \in Frontier^{G}_{\rightarrow}(I)_k, \exists w : (t \rightarrow w \wedge w \; G \; y) \end{cases}$$

In this paper, we consider only folding relations $G \in \{=_E, \approx_E, \preccurlyeq_E\}$ on transition systems whose state set is $\mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{s}}$ for a given sort s. We write $\mathcal{R}each_{\mathcal{A}}(I)$ for the transition system $\mathcal{R}each^{=_E}_{\mathcal{A}}(I)$, which is the standard notion of reachable subsystem. Note that, the more general the relation G, the greater the chances of $\mathcal{R}each^{G}_{\mathcal{A}}(I)$ being a finite transition system. We are furthermore interested in comparisons between different transition systems, for which we use the notions of simulation, lifting simulation, and bisimulation.

Definition 3 (Simulation, lifting simulation, and bisimulation). *Let* $\mathcal{A} = (A, \rightarrow_{\mathcal{A}})$ *and* $\mathcal{B} = (B, \rightarrow_{\mathcal{B}})$ *be two transition systems. A* simulation *from* $\mathcal{A}$ *to* $\mathcal{B}$*, written* $\mathcal{A} \; H \; \mathcal{B}$*, is a relation* $H \subseteq A \times B$ *such that* $a \; H \; b$ *and* $a \rightarrow_{\mathcal{A}} a'$ *implies that there exists* $b' \in B$ *such that* $a' \; H \; b'$ *and* $b \rightarrow_{\mathcal{B}} b'$. *Given* $\mathcal{A} = (A, \rightarrow_{\mathcal{A}}, I_{\mathcal{A}})$ *and* $\mathcal{B} = (B, \rightarrow_{\mathcal{B}}, I_{\mathcal{B}})$*,* H *is a simulation from* $\mathcal{A}$ *to* $\mathcal{B}$ *if* $(A, \rightarrow_{\mathcal{A}}) \; H \; (B, \rightarrow_{\mathcal{B}})$ *and* $\forall a \in I_{\mathcal{A}}, \exists b \in I_{\mathcal{B}}$ *s.t.* $a \; H \; b$. *A simulation* H *from* $(A, \rightarrow_{\mathcal{A}})$ *to* $(B, \rightarrow_{\mathcal{B}})$ *(resp. from* $(A, \rightarrow_{\mathcal{A}}, I_{\mathcal{A}})$ *to* $(B, \rightarrow_{\mathcal{B}}, I_{\mathcal{B}})$*) is a* bisimulation *if* H^{-1} *is a simulation from*

$(B, \rightarrow_{\mathcal{B}})$ *to* $(A, \rightarrow_{\mathcal{A}})$ *(resp. from* $(B, \rightarrow_{\mathcal{B}}, I_{\mathcal{B}})$ *to* $(A, \rightarrow_{\mathcal{A}}, I_{\mathcal{A}})$*). We call a simulation* $(A, \rightarrow_{\mathcal{A}}, I_{\mathcal{A}})\ H\ (B, \rightarrow_{\mathcal{B}}, I_{\mathcal{B}})$ *a* lifting simulation *if for each finite sequence* $b_0 \rightarrow_{\mathcal{B}} b_1 \rightarrow_{\mathcal{B}} b_2 \rightarrow_{\mathcal{B}} \cdots \rightarrow_{\mathcal{B}} b_n$ *with* $b_0 \in I_{\mathcal{B}}$*, there exists a finite sequence* $a_0 \rightarrow_{\mathcal{A}} a_1 \rightarrow_{\mathcal{A}} a_2 \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} a_n$ *with* $a_0 \in I_{\mathcal{A}}$ *such that* $a_i\ H\ b_i$ *for* $0 \leq i \leq n$*.*

It is easy to see that simulations, lifting simulations, and bisimulations compose, that is, if $\mathcal{A}\ H\ \mathcal{B}\ K\ \mathcal{C}$ are simulations (resp. lifting simulations, resp. bisimulations), then $\mathcal{A}\ H; K\ \mathcal{C}$ is a simulation (resp. lifting simulation, resp. bisimulation). In fact, we have associated categories, with transition systems as objects and simulations (resp. lifting simulations, resp. bisimulations) as morphisms.

In rewriting logic we usually specify a concurrent system as a topmost[3] rewrite theory $\mathcal{R} = (\Sigma, E, R)$, where states are E-equivalence classes of ground terms of a concrete top sort State, i.e., elements in $\mathcal{T}_{\Sigma/E,\mathsf{State}}$, and transitions are rewrite rules $l \rightarrow r$ for $l, r \in \mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$ that rewrite states into states. We can describe the operational behavior of the concurrent system by an associated transition system.

Definition 4 ($\mathcal{T}_{\mathcal{R}}$-Transition System). *Let* $\mathcal{R} = (\Sigma, E, R)$ *be a topmost rewrite theory with a top sort* State*. We define the transition system* $\mathcal{T}_{\mathcal{R}} = (\mathcal{T}_{\Sigma/E,\mathsf{State}}, \rightarrow_{R,E})$*.*

Example 1. Consider a simplified version of Lamport's bakery protocol, in which we have several processes, each denoted by a natural number, that achieve mutual exclusion between them by the usual method common in bakeries and deli shops: there is a number dispenser, and customers are served in sequential order according to the number that they hold. This system can be specified as an order-sorted topmost rewrite theory in Maude[4] as follows:

```
fmod BAKERY-SYNTAX is
  sort Nat .
  op 0 : -> Nat .
```

[3] Obviously, not all concurrent systems need to have a topmost rewrite theory specification. However, as explained in [34,35], many concurrent systems of interest, including the vast majority of distributed algorithms, admit topmost specifications. For example, concurrent object-oriented systems whose state is a multiset of objects and messages can be given a topmost specification by enclosing the system state in a top operator. Even hierarchical distributed systems of the "Russian doll" kind can likewise be so specified, provided that the boundaries defining such hierarchies are not changed by transitions.

[4] The Maude syntax is so close to the corresponding mathematical notation for defining rewrite theories as to be almost self-explanatory. The general point to keep in mind is that each item: a sort, a subsort, an operation, an equation, a rule, etc., is declared with an obvious keyword: `sort`, `subsort`, `op`, `eq`, `rl`, etc., with each declaration ended by a space and a period. A rewrite theory $\mathcal{R} = (\Sigma, E, R)$ is defined with the signature Σ using keyword `op`, equations in E are specified using keyword `eq` or keywords `assoc`, `comm` and `id:` (for associativity, commutativity, and identity, respectively) appearing in an operator declaration, and rules in R using keyword `rl`. Another important point is the use of "mix-fix" user-definable syntax, with the argument positions specified by underbars; for example: `if_then_else_fi`. We write the sort of a variable using keyword `var` or after its name and a colon, e.g. `X:Nat`.

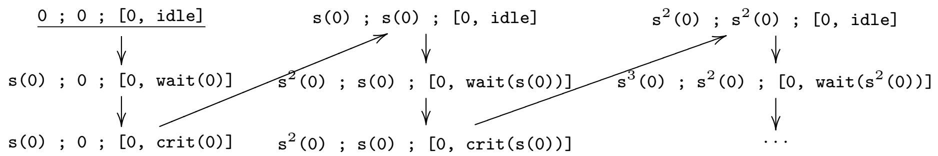
0 ; 0 ; [0, idle]
s(0) ; 0 ; [0, wait(0)]
s(0) ; 0 ; [0, crit(0)]
s(0) ; s(0) ; [0, idle]
s^2(0) ; s(0) ; [0, wait(s(0))]
s^2(0) ; s(0) ; [0, crit(s(0))]
s^2(0) ; s^2(0) ; [0, idle]
s^3(0) ; s^2(0) ; [0, wait(s^2(0))]
...

Fig. 1. Infinite transition system $\mathcal{R}\mathit{each}_{\mathcal{T}_{\mathcal{R}}}$(`0 ; 0 ; [0, idle]`)

```
  op s : Nat -> Nat .
  sorts ModeIdle ModeWait ModeCrit Mode .
  subsorts ModeIdle ModeWait ModeCrit < Mode .
  sorts ProcIdle ProcWait Proc ProcIdleSet ProcWaitSet ProcSet .
  subsorts ProcIdle < ProcIdleSet .
  subsorts ProcWait < ProcWaitSet .
  subsorts ProcIdle ProcWait < Proc < ProcSet .
  subsorts ProcIdleSet < ProcWaitSet < ProcSet .
  op idle : -> ModeIdle .
  op wait : Nat -> ModeWait .
  op crit : Nat -> ModeCrit .
  op [_,_] : Nat ModeIdle -> ProcIdle .
  op [_,_] : Nat ModeWait -> ProcWait .
  op [_,_] : Nat Mode -> Proc .
  op none : -> ProcIdleSet .
  op __ : ProcIdleSet ProcIdleSet -> ProcIdleSet [assoc comm id: none] .
  op __ : ProcWaitSet ProcWaitSet -> ProcWaitSet [assoc comm id: none] .
  op __ : ProcSet ProcSet -> ProcSet [assoc comm id: none] .
  sort State .
  op _;_;_ : Nat Nat ProcSet -> State .
endfm
mod BAKERY is
  protecting BAKERY-SYNTAX .
  var PS : ProcSet .
  vars N M K : Nat .
  rl N ; M ; [K, idle] PS => s(N) ; M ; [K, wait(N)] PS .
  rl N ; M ; [K, wait(M)] PS => N ; M ; [K, crit(M)] PS .
  rl N ; M ; [K, crit(M)] PS => N ; s(M) ; [K, idle] PS .
endm
```

Given the initial state t_1 = "`0 ; 0 ; [0, idle]`", the infinite transition system $\mathcal{R}\mathit{each}_{\mathcal{T}_{\mathcal{R}}}(t_1)$ is depicted in Figure 1. We will graphically identify initial states by underlining them.

Narrowing calculates the most general rewriting sequences associated to a term. We can exploit this generality and use narrowing as a lifting simulation of rewriting. We write $\mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$ for the set of E-equivalence classes of terms of sort State excluding variables, i.e., $\mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}} = \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}} \backslash \mathcal{X}_{\mathsf{State}}$. We can define the transition system associated to narrowing as follows.

Definition 5 ($\mathcal{N}_{\mathcal{R}}$-Transition System). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort* State*. We define a transition system $\mathcal{N}_{\mathcal{R}} = (\mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}, \rightsquigarrow_{R,E})$.*

Note that we exclude variables in Definition 5, since the relation $\rightsquigarrow_{R,E}$ is not defined on them. Note also that for each rule $l \to r$ in a topmost rewrite theory, we have $r \notin \mathcal{X}$, since $l \notin \mathcal{X}$, $\mathit{Var}(r) \subseteq \mathit{Var}(l)$, and $r \in \mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$.

Given a subset $U \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{s}}$, we define the set of *ground instances* of U as $[\![U]\!] = \{[t]_E \in \mathcal{T}_{\Sigma/E,\mathsf{s}} \mid \exists\ [t']_E \in U \text{ s.t. } t \preccurlyeq_E t'\}$. Note that U may be a finite set, whereas $\llbracket U \rrbracket$ can often be an infinite set. This gives us a symbolic way of describing possibly infinite sets of initial states in $\mathcal{T}_\mathcal{R}$, which will be very useful for model checking purposes.

The following result relates the transition systems associated to narrowing and rewriting. Note that we do not have a bisimulation in general, since a term $t \in \mathcal{T}_\Sigma(\mathcal{X})$ may have narrowing steps with incomparable substitutions $\sigma_1, \ldots, \sigma_k$, i.e., given $i \neq j$, $\sigma_i(t)$ may disable the rewriting step performed on $\sigma_j(t)$ and viceversa; see Figure 2. Our results are based on the following result from [35].

Lemma 1 (Topmost Completeness). [35] *For $\mathcal{R} = (\Sigma, E, R)$ a topmost theory, let $t \in \mathcal{T}_\Sigma(\mathcal{X})$ be a term that is not a variable, and let V be a set of variables containing $Var(t)$. For some substitution ρ, let $\rho(t) \rightarrow_{R/E} t'$ using the rule $l \rightarrow r$ in R. Then there are $\sigma, \theta, t"$ such that $t \overset{\sigma}{\leadsto}_{R,E} t"$ using the same rule $l \rightarrow r$, $t"$ is not a variable, $\rho|_V =_E (\sigma \circ \theta)|_V$, and $\theta(t") =_E t'$.*

Theorem 1 (Lifting simulation by narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort State. Let $U \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^\circ_{\mathsf{State}}$. The relation $\preccurlyeq_E$ defines two lifting simulations: $\mathcal{T}_\mathcal{R} \preccurlyeq_E \mathcal{N}_\mathcal{R}$ and $\mathcal{R}each_{\mathcal{T}_\mathcal{R}}(\llbracket U \rrbracket) \preccurlyeq_E \mathcal{R}each_{\mathcal{N}_\mathcal{R}}(U)$.*

Proof. To show that $\preccurlyeq_E$ is a simulation, let $[t]_E, [t']_E \in \mathcal{T}_{\Sigma/E,\mathsf{State}}$ and $[t]_E \rightarrow_{R,E} [t']_E$ using rule $l \rightarrow r \in R$, for each $[w]_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and substitution ρ such that $t =_E \rho(w)$ (i.e., $t \preccurlyeq_E w$), by Lemma 1, there are substitutions σ, θ and $w' \in \mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ such that $w \overset{\sigma}{\leadsto}_{R,E} w'$ using rule $l \rightarrow r \in R$, $t =_E \theta(\sigma(w))$, and $t' =_E \theta(w')$ (i.e., $t' \preccurlyeq_E w'$). To show the lifting property, let $[t_0]_E, \ldots, [t_n]_E \in \mathcal{T}_{\Sigma/E,\mathsf{State}}$ and $t_0 \overset{\sigma_1}{\leadsto}_{R,E} t_1 \overset{\sigma_2}{\leadsto}_{R,E} t_2 \cdots \overset{\sigma_n}{\leadsto}_{R,E} t_n$ using rules $l_1 \rightarrow r_1, \ldots, l_n \rightarrow r_n \in R$. There is at least one substitution ρ such that $(\sigma_1 \circ \sigma_2 \cdots \circ \sigma_n \circ \rho)(t_0) \in \mathcal{T}_{\Sigma,\mathsf{State}}$ and then, let $\theta = \sigma_1 \circ \sigma_2 \cdots \circ \sigma_n \circ \rho$, we have $\theta(t_0) \rightarrow_{R,E} \theta(t_1) \rightarrow_{R,E} \theta(t_2) \cdots \rightarrow_{R,E} \theta(t_n)$ using rules $l_1 \rightarrow r_1, \ldots, l_n \rightarrow r_n \in R$. □

Since $\mathcal{N}_\mathcal{R}$ is typically infinite, for a set $U \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^\circ_{\mathsf{State}}$ of initial states and a relation $G \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^\circ_{\mathsf{State}} \times \mathcal{T}_{\Sigma/E}(\mathcal{X})^\circ_{\mathsf{State}}$, we may be interested in the reachable subsystem from U in $\mathcal{N}_\mathcal{R}$ with folding G, i.e., in the transition system $\mathcal{R}each^G_{\mathcal{N}_\mathcal{R}}(U)$.

Example 2. Consider Example 1 and let t_2 = "`N:Nat ; M:Nat ; [0, MD:Mode]`". The finite transition system $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(t_2)$ is depicted in Figure 2. In the case of narrowing, we will graphically tie the substitution computed by each narrowing step to the proper transition arrow. Also, when a transition step is making use of the folding relation G, i.e., when it is not a normal rewriting/narrowing step but a combination of rewriting/narrowing and folding with the relation G, we mark the arrow with a double head.

Since a transition system usually includes a set of initial states, we can extend Theorem 1 to a folding relation G, to obtain a more specific (and in some sense more powerful) result. For this we need the following characterization of a folding relation G.

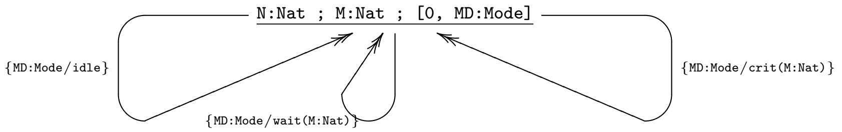

Fig. 2. Finite transition system $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_{\mathcal{R}}}(\texttt{N:Nat ; M:Nat ; [0, MD:Mode]})$

Definition 6 ($\leadsto_{R,E}$-equivalent relation). *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory. The binary relation $G \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X}) \times \mathcal{T}_{\Sigma/E}(\mathcal{X})$ is called $\leadsto_{R,E}$-*equivalent *if for $[t]_E, [t']_E, [w]_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})$ such that $t\ G\ w$ and $t \leadsto_{R,E} t'$ using rule $l \to r$, there is $[w']_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})$ such that $w \leadsto_{R,E} w'$ using rule $l \to r$ and $t'\ G\ w$.*

Lemma 2 ($\leadsto_{R,E}$-equivalence of G). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort* State*. The relations $\{=_E, \approx_E, \preccurlyeq_E\}$ on $\mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ are $\leadsto_{R,E}$-equivalent.*

Proof. We only prove it for $\preccurlyeq_E$. Let $[t]_E, [t']_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ such that $t \overset{\rho}{\leadsto}_{R,E} t'$ using rule $l \to r$. Let $[w]_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and τ such that $t =_E \tau(w)$ (i.e., $t \preccurlyeq_E w$). Note that $\rho(\tau(w)) \to_{R,E} t'$ using rule $l \to r$. By Lemma 1, there are substitutions σ, θ and $w' \in \mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$ such that $w \overset{\sigma}{\leadsto}_{R,E} w'$ using rule $l \to r \in R$, $(\tau \circ \rho)|_{Var(w)} =_E (\sigma \circ \theta)|_{Var(w)}$, $\rho(t) =_E \rho(\tau(w)) =_E \theta(\sigma(w))$, and $t' =_E \theta(w')$ (i.e., $t' \preccurlyeq_E w'$). □

Theorem 2 (Simulation by G-narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort* State*. Let $U \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$ and $G \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}} \times \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$ be $\leadsto_{R,E}$-equivalent. The relation G then defines a simulation $\mathcal{R}each_{\mathcal{N}_{\mathcal{R}}}(U)\ G\ \mathcal{R}each^{G}_{\mathcal{N}_{\mathcal{R}}}(U)$.*

Proof. By Definition 6 and Lemma 2. □

We can obtain a bisimulation when every narrowing step of a transition system computes the identity substitution, which means that every possible rewriting sequence is represented in its most general way in the narrowing tree.

Theorem 3 (Bisimulation by narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort* State*. Let $U \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$. Let each transition in $\mathcal{R}each_{\mathcal{N}_{\mathcal{R}}}(U)$ be of the form $[t]_E \overset{id}{\leadsto}_{R,E} [t']_E$. The relation $\preccurlyeq_E$ then defines a bisimulation $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}}(\llbracket U \rrbracket) \preccurlyeq_E \mathcal{R}each_{\mathcal{N}_{\mathcal{R}}}(U)$.*

Proof. We only prove that $\preccurlyeq_E^{-1}$ is a simulation $\mathcal{R}each_{\mathcal{N}_{\mathcal{R}}}(U) \preccurlyeq_E^{-1} \mathcal{R}each_{\mathcal{T}_{\mathcal{R}}}(\llbracket U \rrbracket)$. If $[w]_E, [w']_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and $[w]_E \overset{id}{\leadsto}_{R,E} [w']_E$ using rule $l \to r \in R$, then for each grounding substitution σ, i.e., $\sigma(w) \in \mathcal{T}_{\Sigma,\mathsf{State}}$, $[\sigma(w)]_E \to_{R,E} [\sigma(w')]_E$. □

Lemma 3 ($\leadsto_{R,E}$-equivalence of G^{-1}). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort* State*. Let $T \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ such that for each $[t]_E, [t']_E \in T$, $[t]_E \overset{\sigma}{\leadsto}_{R,E} [t']_E$ implies $\sigma = id$. The relations $\{{=_E}^{-1}, {\approx_E}^{-1}, {\preccurlyeq_E}^{-1}\}$ on T are $\leadsto_{R,E}$-equivalent.*

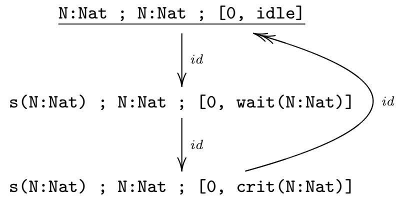
N:Nat ; N:Nat ; [0, idle]
id
s(N:Nat) ; N:Nat ; [0, wait(N:Nat)]
id
id
s(N:Nat) ; N:Nat ; [0, crit(N:Nat)]

Fig. 3. Finite transition system $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(\texttt{N:Nat ; N:Nat ; [0, idle]})$

Proof. We only have to prove it for $\preccurlyeq_E^{-1}$, since $=_E$ and $\approx_E$ are symmetric. Let $[t]_E, [t']_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ such that $t \overset{id}{\rightsquigarrow}_{R,E} t'$ using rule $l \to r$, i.e., $t \to_{R,E} t'$ using rule $l \to r$. Let $[w]_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and τ such that $\tau(t) =_E w$ (i.e., $t \preccurlyeq_E^{-1} w$). Then, $\tau(t) \overset{id}{\rightsquigarrow}_{R,E} \tau(t')$ using rule $l \to r$, i.e., $\tau(t) \to_{R,E} \tau(t')$ using rule $l \to r$. And thus, let $w' = \tau(t')$, we have $t' \preccurlyeq_E^{-1} w'$. □

Theorem 4 (Bisimulation by G-narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with a top sort* State*. Let $G \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}} \times \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$ and G^{-1} be $\rightsquigarrow_{R,E}$-equivalent. Let $U \subseteq \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$. Let each transition in $\mathcal{R}each^{G}_{\mathcal{N}_\mathcal{R}}(U)$ be of the form $[t]_E \overset{id}{\rightsquigarrow}{}^{G}_{R,E} [t']_E$. The relation G then defines a bisimulation $\mathcal{R}each_{\mathcal{N}_\mathcal{R}}(U)\ G\ \mathcal{R}each^{G}_{\mathcal{N}_\mathcal{R}}(U)$.*

Proof. By Definition 6. □

Example 3. Consider Example 1 and t_3 = "`N:Nat ; N:Nat ; [0, idle]`". The finite transition system $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(t_3)$ is depicted in Figure 3. Note that every transition has the *id* substitution. Therefore, by Theorems 1 and 4, we have a bisimulation between the infinite transition system $\mathcal{R}each_{\mathcal{T}_\mathcal{R}}(\texttt{0 ; 0 ; [0, idle]})$ shown in Figure 1 and $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(\texttt{N:Nat ; N:Nat ; [0, idle]})$ in Figure 3.

Note that the narrowing-based methods we have presented allow us to answer *reachability questions* of the form $(\exists \overrightarrow{x})\ t \to^* t'$. That is, given a set of initial states $[\![t]\!]$ we want to know whether from some state in $[\![t]\!]$ we can reach a state in $[\![t']\!]$. The fact that narrowing provides a *lifting* simulation of the system $\mathcal{T}_\mathcal{R}$ means that it is a *complete* semi-decision procedure for answering such reachability questions: the above existential formula holds in $\mathcal{T}_\mathcal{R}$ if and only if from t we can reach by narrowing a term that E-unifies with t'. In particular, narrowing is very useful for verification of *invariants*. Let $p \in \mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ be a pattern representing the set-theoretic complement of an invariant. Then, the reachability formula $\nexists \overrightarrow{x} : t \to^*_{R/E} p$ corresponds to the satisfaction of the invariant for the set of initial states $[\![t]\!]$. Therefore, narrowing provides a semi-decision procedure for the *violation* of invariants. Furthermore, the invariant holds iff p does not E-unify with any term in $\mathcal{R}each_{\mathcal{N}_\mathcal{R}}(t)$. It also holds if p does not E-unify with any term in $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(t)$, which is a decidable question if $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(t)$ is finite. If p does E-unify with some term in $\mathcal{R}each^{\preccurlyeq_E}_{\mathcal{N}_\mathcal{R}}(t)$, in general the invariant may or

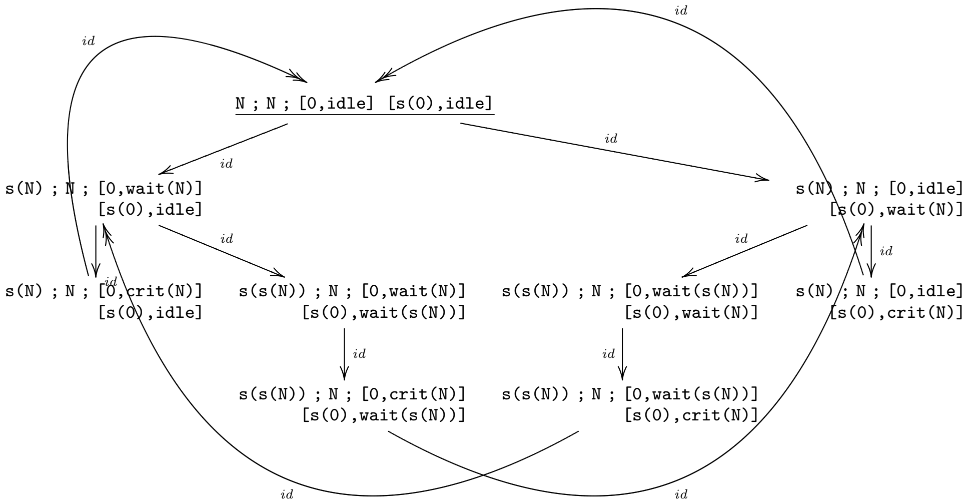
N ; N ; [0,idle] [s(0),idle]
s(N) ; N ; [0,wait(N)] [s(0),idle]
s(N) ; N ; [0,idle] [s(0),wait(N)]
s(N) ; N ; [0,crit(N)] [s(0),idle]
s(s(N)) ; N ; [0,wait(N)] [s(0),wait(s(N))]
s(s(N)) ; N ; [0,wait(s(N))] [s(0),wait(N)]
s(N) ; N ; [0,idle] [s(0),crit(N)]
s(s(N)) ; N ; [0,crit(N)] [s(0),wait(s(N))]
s(s(N)) ; N ; [0,wait(s(N))] [s(0),crit(N)]
id
id
id
id
id
id
id
id
id
id
id
id
id
id

Fig. 4. Finite transition system $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_{\mathcal{R}}}($`N:Nat ; N:Nat ; [0, idle] [s(0), idle]`$)$

may not hold: we need to check whether this corresponds to a real narrowing sequence.

Example 4. Consider Example 1 and the initial state with two processes t_4 = "`N:Nat ; N:Nat ; [0, idle] [s(0), idle]`". The finite transition system $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_{\mathcal{R}}}(t_4)$ is depicted in Figure 4. Note that we have a bisimulation between $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}}([\![t_4]\!])$ and $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_{\mathcal{R}}}(t_4)$. Consider the following pattern identifying that the critical section property has been violated

"`N:Nat ; M:Nat ; [0, crit(C1:Nat)] [s(0), crit(C2:Nat)]`".

We can check that the pattern does not unify with any state in the transition system of Figure 4, and thus this bad pattern is unreachable from any initial state being an instance of t_4. This provides a verification of the *mutual exclusion* property for the infinite-state `BAKERY` protocol, not just from a single initial state, but from an infinite set $[\![t_4]\!]$ of initial states.

Note, finally, that, for U a set of of initial states, even if the transition system $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}}([\![U]\!])$ is finite, the transition system $\mathcal{R}each^{G}_{\mathcal{N}_{\mathcal{R}}}(U)$ can be much smaller. Furthermore, the set U is typically finite, whereas the set $[\![U]\!]$ is typically infinite, making it impossible to model check an invariant from each initial state by finitary methods. In all these ways, narrowing allows algorithmic verification of invariants in many infinite-state systems, and also in finite-state systems whose size make it unfeasible to use standard model checking techniques.

4 Narrowing-based $ACTL^*$ Model Checking

Model checking [10] is the most successful verification technique for temporal logics. When we perform model checking, we use *Kripke structures* to represent the state search space. Kripke structures are the natural models for propositional temporal logic. Essentially, a Kripke structure is a total[5] transition system to which we have added a collection of atomic propositions on its set of states.

Definition 7 (Kripke Structure). *Given a set Π of* atomic propositions*, a* Π-Kripke structure *(or just Kripke structure) is a triple $\mathcal{K} = (A, \to, \mathcal{L})$ such that $(A, \to)$ is a transition system with $\to$ total, and $\mathcal{L} : A \longrightarrow \mathcal{P}(\Pi)$ is a function, called the* labeling function*, assigning to each state a the set $\mathcal{L}(a) \subseteq \Pi$ of atomic propositions that hold in a. We write $\mathcal{K} = (A, \to, I, \mathcal{L})$ when $(A, \to, I)$ defines a transition system with initial states I.*

We consider the *Computation Tree Logic* (CTL*), its universally quantified subset (ACTL*), and a subset of both, *linear temporal logic* (LTL), as the temporal logics for property specification. We refer the reader to [10] for formal definitions of these three temporal logics. Given a set Π of atomic propositions, the semantics of a formula $\varphi \in \mathrm{CTL}^*_\Pi$ (resp. $\varphi \in \mathrm{ACTL}^*_\Pi$, $\varphi \in \mathrm{LTL}_\Pi$) is defined by means of a *satisfaction relation* $\mathcal{K}, a \models \varphi$, where $\mathcal{K} = (A, \to, \mathcal{L})$ is a Π-Kripke structure having Π as its atomic propositions, and $a \in A$ is a state. We refer the reader to [10] for a detailed definition of satisfaction of a CTL* (ACTL*, LTL) formula φ in a Kripke structure.

The following notion of simulation and results, except for lifting simulations, are borrowed from [10,32]. They allow the comparison of Kripke structures.

Definition 8 (Simulation, lifting simulation, and bisimulation of Kripke-structures). [10,32] *Let Π be a set of* atomic propositions*. Let $\mathcal{K}_A = (A, \to_{\mathcal{K}_A}, \mathcal{L}_A)$ and $\mathcal{K}_B = (B, \to_{\mathcal{K}_B}, \mathcal{L}_B)$ be two Π-Kripke structures. A* simulation *H from $\mathcal{K}_A$ to $\mathcal{K}_B$, written $\mathcal{K}_A\ H\ \mathcal{K}_B$, is a simulation of transition systems $(A, \to_{\mathcal{K}_A})\ H\ (B, \to_{\mathcal{K}_B})$ such that a H b implies $\mathcal{L}_A(a) = \mathcal{L}_B(b)$, and is a* bisimulation *if, in addition, $\mathcal{K}_B\ H^{-1}\ \mathcal{K}_A$ is also a simulation. Similarly, given Kripke-structures $\mathcal{K}_A = (A, \to_{\mathcal{K}_A}, I_{\mathcal{K}_A}, \mathcal{L}_A)$ and $\mathcal{K}_B = (B, \to_{\mathcal{K}_B}, I_{\mathcal{K}_B}, \mathcal{L}_B)$, $\mathcal{K}_A\ H\ \mathcal{K}_B$ is a* simulation *if $(A, \to_{\mathcal{K}_A}, I_{\mathcal{K}_A})\ H\ (B, \to_{\mathcal{K}_B}, I_{\mathcal{K}_B})$ is a simulation and a H b implies $\mathcal{L}_A(a) = \mathcal{L}_B(b)$, and is a* bisimulation *if, in addition, $\mathcal{K}_B\ H^{-1}\ \mathcal{K}_A$ is also a simulation. A simulation $(A, \to_{\mathcal{K}_A}, I_{\mathcal{K}_A}, \mathcal{L}_A)\ H\ (B, \to_{\mathcal{K}_B}, I_{\mathcal{K}_B}, \mathcal{L}_B)$ is* lifting *if $(A, \to_{\mathcal{K}_A}, I_{\mathcal{K}_A})\ H\ (B, \to_{\mathcal{K}_B}, I_{\mathcal{K}_B})$ is lifting.*

Satisfiability of formulas in temporal logics is preserved under some conditions when we have a simulation/bisimulation between two Kripke structures. Namely, satisfiability of ACTL* (including LTL) formulas is reflected back by a simulation between two Kripke structures, and satisfiability of CTL* formulas is preserved in both directions in the case of a bisimulation between two Kripke structures.

Theorem 5. [10,32] *Let Π be a set of* atomic propositions*. Let $\mathcal{K}_A = (A, \to_{\mathcal{K}_A}, \mathcal{L}_A)$ and $\mathcal{K}_B = (B, \to_{\mathcal{K}_B}, \mathcal{L}_B)$ be two Π-Kripke structures and let*

[5] A binary relation $R \subseteq A \times A$ on a set A is called *total* if and only if for each $a \in A$ there is at least one $a' \in A$ such that $(a, a') \in R$. If R is not total, it can be made total by defining $R^\bullet = R \cup \{(a, a) \in A^2 \mid \nexists a' \in A.(a, a') \in R\}$.

$\mathcal{K}_A$ H $\mathcal{K}_B$ be a simulation (resp. bisimulation). For each $\varphi \in ACTL^*_\Pi$ (resp. $\varphi \in CTL^*_\Pi$), $a\ H\ b \wedge \mathcal{K}_B, b \models \varphi \implies \mathcal{K}_A, a \models \varphi$ (resp. $a\ H\ b \wedge \mathcal{K}_B, b \models \varphi \iff a\ H\ b \wedge \mathcal{K}_A, a \models \varphi$).

In rewriting logic we usually specify a concurrent system as a topmost rewrite theory $\mathcal{R} = (\Sigma, E, R)$, and the atomic propositions Π as equationally-defined predicates in an equational theory $\mathcal{E}_\Pi = (\Sigma_\Pi, E_\Pi \uplus E)$. As explained in Section 3, the rewrite theory $\mathcal{R}$ contains a top sort State, generating E-equivalence classes $\mathcal{T}_{\Sigma/E,\mathsf{State}}$, and rewrite rules $l \to r \in \mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ denoting system transitions. For the equational theory $\mathcal{E}_\Pi$, we have the following definition (see [13] for further details).

Definition 9 (Bool-equational theory). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort State and rules $l \to r \in \mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$. We define a Bool-equational theory $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ as follows:*

1. *$\mathcal{E}_\Pi$ extends (Σ, E) in a protecting manner; i.e., $\mathcal{T}_{\Sigma_\Pi/(E_\Pi \uplus E)}|_\Sigma = \mathcal{T}_{\Sigma/E}$;*
2. *the signature Σ_Π is defined as $\Sigma_\Pi = \Sigma \uplus \Pi \uplus \{\mathtt{tt}, \mathtt{ff}\}$;*
3. *there is a new top sort Bool with no subsorts containing only constants $\mathtt{tt}$ and $\mathtt{ff}$ and the unary symbols Π such that the operation definition of each $p \in \Pi$ is of the form $p : \mathsf{State} \to \mathsf{Bool}$ and for each equation $t = t' \in E_\Pi$, $t, t' \in \mathcal{T}_{\Sigma_\Pi}(\mathcal{X})_{\mathsf{Bool}}$;*
4. *$\mathcal{E}$ is sufficiently complete (see [17,42]) and protecting of Bool, i.e., $\mathcal{T}_{\Sigma_\Pi/(E_\Pi \uplus E),\mathsf{Bool}}$ contains only two different equivalence classes $[\mathtt{tt}]$ and $[\mathtt{ff}]$;*
5. *the congruence relation $=_{(E_\Pi \uplus E)}$ on $\mathcal{T}_{\Sigma_\Pi}(\mathcal{X})$ is decidable.*

In practice, we concretize the previous definition to the following case, which is the class of equational theories considered in this paper for defining state predicates.

Refinement 1 *When, in addition to Definition 9, equations in E_Π can be oriented into a set $\overrightarrow{E_\Pi}$ of confluent and terminating rules modulo E (see [17,42]), equality questions of the form $p(t) =_{(E_\Pi \uplus E)} \mathtt{tt}$ and $p(t) =_{(E_\Pi \uplus E)} \mathtt{ff}$, for $[t]_E \in \mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and $p \in \Pi$, are decidable (see [38]), i.e., can be decided by whether $p(t) \to^*_{\overrightarrow{E_\Pi},E} w$ with $w =_E \mathtt{tt}$ or $w =_E \mathtt{ff}$.*

We can define a Π-Kripke structure associated to a rewrite theory $\mathcal{R}$ and a Bool-equational theory $\mathcal{E}_\Pi$ defining the atomic propositions Π.

Definition 10 ($\mathcal{T}^\Pi_\mathcal{R}$-Kripke Structure). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort State and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be the Bool-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$. The Π-Kripke structure is defined as the triple $\mathcal{T}^\Pi_\mathcal{R} = (\mathcal{T}_{\Sigma/E,\mathsf{State}}, (\to_{R,E})^\bullet, \mathcal{L}_\Pi)$, where for each $[t]_E \in \mathcal{T}_{\Sigma/E,\mathsf{State}}$ and $p \in \Pi$, we have $p \in \mathcal{L}_\Pi([t]_E) \iff p(t) =_{(E_\Pi \uplus E)} \mathtt{tt}$.*

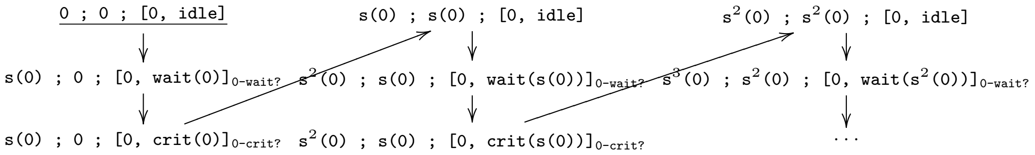
0 ; 0 ; [0, idle]
s(0) ; s(0) ; [0, idle]
s^2(0) ; s^2(0) ; [0, idle]
s(0) ; 0 ; [0, wait(0)]0-wait?
s^2(0) ; s(0) ; [0, wait(s(0))]0-wait?
s^3(0) ; s^2(0) ; [0, wait(s^2(0))]0-wait?
s(0) ; 0 ; [0, crit(0)]0-crit?
s^2(0) ; s(0) ; [0, crit(s(0))]0-crit?
...

Fig. 5. Infinite Kripke structure $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}^{\Pi}}($`0 ; 0 ; [0, idle]`$)$

In what follows we will always assume that $\mathcal{R}$ is *deadlock free*, that is, that the set of $\to_{R,E}$-canonical forms of sort State is empty. As explained in [13,39], this involves no real loss of generality, since $\mathcal{R}$ can always be transformed into a bisimilar $\mathcal{R}^{\mathrm{df}}$ which is deadlock free. Under this assumption the Kripke structure $\mathcal{T}_{\mathcal{R}}^{\Pi}$ then becomes the pair $\mathcal{T}_{\mathcal{R}}^{\Pi} = (\mathcal{T}_{\mathcal{R}}, \mathcal{L}_{\Pi})$.

As in Section 3, given a set $U \subseteq \mathcal{T}_{\Sigma/E,\mathsf{State}}$ of initial states, we abuse the notation and define the reachable sub Π-Kripke structure of $\mathcal{T}_{\mathcal{R}}^{\Pi}$ by $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}^{\Pi}}(U)$.

Example 5. Consider Example 1. We are interested in the atomic propositions $\Pi = \{$`0-wait?`, `0-crit?`$\}$ testing whether process number `0` is in its critical section or waiting, described by the following equational theory[6] in Maude.

```
fmod BAKERY-PROPS is
  protecting BAKERY-SYNTAX .
  sort Bool .
  ops tt ff : -> Bool .
  ops 0-wait? 0-crit? : State -> Bool .
  vars N M : Nat .
  eq 0-wait?(N ; M ; [0, idle]) = ff .
  eq 0-wait?(N ; M ; [0, wait(K)]) = tt .
  eq 0-wait?(N ; M ; [0, crit(K)]) = ff .
  eq 0-crit?(N ; M ; [0, idle]) = ff .
  eq 0-crit?(N ; M ; [0, wait(K)]) = ff .
  eq 0-crit?(N ; M ; [0, crit(K)]) = tt .
endfm
```

Given the initial state $t_1 =$ "`0 ; 0 ; [0, idle]`", the infinite Π-Kripke structure $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}^{\Pi}}(t_1)$ is depicted in Figure 5, where we can model check the formula "$\Box($`0-wait?` $\Rightarrow \Diamond$`0-crit?`$)$" stating that whenever process `0` is waiting, it eventually gets into its critical section.

Note that we can have symbolic states (i.e., terms $\mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}}$) such that the atomic propositions Π cannot be evaluated without further instantiation; check the transition system of Figure 2, where propositions `0-wait?` and `0-crit?` cannot be evaluated in the node "`N:Nat ; M:Nat ; [0, MD:Mode]`". We first characterize the terms for which the atomic propositions can be evaluated.

Definition 11 (**Π-Terms**)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort* State *and $\mathcal{E}_{\Pi} = (\Sigma_{\Pi}, E \uplus E_{\Pi})$ be the* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$. We define the set of Π-*defined *terms as $\mathcal{T}_{\Sigma}^{\Pi}(\mathcal{X})_{\mathsf{State}} = \{t \in \mathcal{T}_{\Sigma/E}(\mathcal{X})^{\circ}_{\mathsf{State}} \mid \forall p \in \Pi : (p(t) =_{(E_{\Pi} \uplus E)} \mathtt{tt}) \vee (p(t) =_{(E_{\Pi} \uplus E)} \mathtt{ff})\}$.*

[6] This equational theory is not sufficiently complete for all possible state terms but only for state terms having a single process with id `0`.

Note that, for a Bool-equational theory $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$, membership in $\mathcal{T}^\Pi_\Sigma(\mathcal{X})_{\mathsf{State}}$ is decidable, since $=_{(E \uplus E_\Pi)}$ is decidable, and $\mathcal{T}_{\Sigma,\mathsf{State}} \subseteq \mathcal{T}^\Pi_\Sigma(\mathcal{X})_{\mathsf{State}}$, since $\mathcal{E}_\Pi$ is sufficiently complete.

For terms in $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}} \setminus \mathcal{T}^\Pi_\Sigma(\mathcal{X})_{\mathsf{State}}$, we need a different relation that instantiates terms as much as necessary in order to belong to the set $\mathcal{T}^\Pi_\Sigma(\mathcal{X})_{\mathsf{State}}$. To define this relation we need a complete and finitary $(E_\Pi \uplus E)$-unification algorithm.

Definition 12 (Π-instantiation relation). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be the* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$. Let $\Pi = \{p_1, \ldots, p_n\}$. Suppose $CSU_{(E_\Pi \uplus E)}(t = t')$ admits a complete and finitary unification algorithm. Then the* instantiation relation *$\rightsquigarrow_\Pi$ is defined as follows*

$$t \overset{\theta}{\rightsquigarrow}_\Pi \theta(t) \iff \theta \in CSU_{(E_\Pi \uplus E)}(p_1(t) = w_1 \wedge \cdots \wedge p_n(t) = w_n)$$
$$\text{where for each } 1 \leq i \leq n,\ w_i \text{ is either } \mathtt{tt} \text{ or } \mathtt{ff}$$

Classes of equational theories $(\Sigma_\Pi, E \uplus E_\Pi)$ with a finitary and complete unification algorithm have been studied in the literature (see [28,18,45]). The class of equational theories considered here is simple but it turns out to be useful in many common cases.

Definition 13 (Simple-Bool-equational theory). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be the* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$. We say $\mathcal{E}_\Pi$ is a* simple-Bool *equational theory if each equation in E_Π is of the form $p(t) = \mathtt{tt}$ or $p(t) = \mathtt{ff}$, where $p \in \Pi$ and $t \in \mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$.*

Assuming that E has a complete and finitary unification algorithm, then we get a finitary and complete set of $(E_\Pi \uplus E)$-unifiers for a simple-Bool-equational theory.

Theorem 6. *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort* State *such that $CSU_E(t = t')$ on $t, t' \in \mathcal{T}_\Sigma(\mathcal{X})$ has a finitary and complete unification algorithm. Let $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be a simple-*Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$. Then $CSU_{(E_\Pi \uplus E)}(t = t')$ on $t, t' \in \mathcal{T}_{\Sigma_\Pi}(\mathcal{X})_{\mathsf{Bool}}$ admits a finitary and complete unification algorithm.*

Now, we can obtain a Π-Kripke structure from a transition system generated by narrowing, since we can safely restrict ourselves to terms in $\mathcal{T}^\Pi_\Sigma(\mathcal{X})_{\mathsf{State}}$ by using the following narrowing relation $\rightsquigarrow_{R,E;\Pi}$.

Definition 14 (Narrowing plus Π-instantiation). *Let $\mathcal{R} = (\Sigma, E, R)$ be a topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. The narrowing relation $\rightsquigarrow_{R,E;\Pi}$ is defined as $\rightsquigarrow_{R,E}; \rightsquigarrow_\Pi$, i.e., $t \overset{\theta}{\rightsquigarrow}_{R,E;\Pi} t'$ iff $\exists w, \sigma, \sigma' : t \overset{\sigma}{\rightsquigarrow}_{R,E} w$, $w \overset{\sigma'}{\rightsquigarrow}_\Pi t'$, and*

$\theta = \sigma \circ \sigma'$. Note that $\rightsquigarrow_{R,E;\Pi}$ on $\mathcal{T}_{\Sigma}(\mathcal{X})$ can be extended to a relation $\overset{\sigma}{\rightsquigarrow}_{R,E;\Pi}$ on $\mathcal{T}_{\Sigma/E}(\mathcal{X})$ as $(\overset{\sigma}{\rightsquigarrow}_{R,E;\Pi});(=_E)$.

Note that if $t \rightsquigarrow_{R,E;\Pi} t'$ and $t \in \mathcal{T}^{\Pi}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$, then $t' \in \mathcal{T}^{\Pi}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$.

The remaining of this section reproduces the results obtained in Section 3 but for Kripke-structures. The proofs of these results are similar to the ones in Section 3, but assume that every node in the corresponding Kripke-structure can be evaluated by the corresponding labeling function $\mathcal{L}_{\Pi}$. We can exploit the generality of narrowing and define a Kripke-structure associated to narrowing.

Definition 15 ($\mathcal{N}^{\Pi}_{\mathcal{R}}$-Kripke Structure). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_{\Pi} = (\Sigma_{\Pi}, E \uplus E_{\Pi})$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. The following triple defines a Π-Kripke structure $\mathcal{N}^{\Pi}_{\mathcal{R}} = (\mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}, \rightsquigarrow_{R/E;\Pi}, \mathcal{L}_{\Pi})$, where for each $[t]_E \in \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and $p \in \Pi$, we have $p \in \mathcal{L}_{\Pi}([t]_E) \iff p(t) =_{(E_{\Pi} \uplus E)} \mathtt{tt}$.*

The following results relate the rewriting and narrowing Kripke-structures associated to a rewrite theory. In practice, we concretize the following results to a simple-Bool-equational theory, which is the class of equational theories considered in this paper.

Theorem 7 (Kripke lifting simulation by narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_{\Pi} = (\Sigma_{\Pi}, E \uplus E_{\Pi})$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. Let $U \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$. The relation $\preccurlyeq_E$ defines two lifting simulations: $\mathcal{T}^{\Pi}_{\mathcal{R}} \preccurlyeq_E \mathcal{N}^{\Pi}_{\mathcal{R}}$ and $\mathcal{R}each_{\mathcal{T}^{\Pi}_{\mathcal{R}}}(\llbracket U \rrbracket) \preccurlyeq_E \mathcal{R}each_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)$.*

Lemma 4 ($\rightsquigarrow_{R,E;\Pi}$-equivalence of G). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_{\Pi} = (\Sigma_{\Pi}, E \uplus E_{\Pi})$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_{\Sigma}(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. The relations $\{=_E, \approx_E, \preccurlyeq_E\}$ on $\mathcal{T}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ are $\rightsquigarrow_{R,E;\Pi}$-equivalent.*

Proof. Since `tt` and `ff` are protected, equations in E_{Π} can be oriented into a set $\overrightarrow{E_{\Pi}}$ of confluent and terminating rules modulo E. Let $t, t' \in \mathcal{T}_{\Sigma_{\Pi}}(\mathcal{X})_{\mathsf{Bool}}$, the set of unifiers $CSU_{(E_{\Pi} \uplus E)}(t = t')$ is defined by $\sigma \in CSU_{(E_{\Pi} \uplus E)}(t = t')$ if $t \simeq t' \overset{\sigma}{\rightsquigarrow}^{*}_{\overrightarrow{E_{\Pi}}^{\bullet},E} \top$, where $\overrightarrow{E_{\Pi}}^{\bullet}$ is the set of rewrite rules $\overrightarrow{E_{\Pi}}^{\bullet} = \overrightarrow{E_{\Pi}} \cup \{x \simeq x \to \top\}$ for a new constant $\top$ of a new top sort NewBool with no subsorts and $\simeq$: Bool $\times$ Bool $\to$ NewBool is a new binary operator. Since $\mathcal{E}_{\Pi}$ is a simple-Bool-equational theory, each sequence $t \simeq t' \overset{\sigma}{\rightsquigarrow}^{*}_{\overrightarrow{E_{\Pi}}^{\bullet},E} \top$ has at most two steps and thus, since the number of rules in $\overrightarrow{E_{\Pi}}$ is finite, $CSU_{(E_{\Pi} \uplus E)}(t = t')$ is finite. Since $\mathcal{E}_{\Pi}$ is topmost, certain additional assumptions such as E-coherence of $\to_{\overrightarrow{E_{\Pi}},E}$ hold (see [27]) and then $(E_{\Pi} \uplus E)$ has a complete unification algorithm by $\rightsquigarrow_{\overrightarrow{E_{\Pi}},E}$, i.e., $CSU_{(E_{\Pi} \uplus E)}(t = t')$ is complete. □

Theorem 8 (Kripke simulation by G-narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. Let $U \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and $G \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}} \times \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ be $\leadsto_{R,E;\Pi}$-equivalent. Then the relation G defines a simulation $\mathcal{R}each_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)\ G\ \mathcal{R}each^{G}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)$.*

And when narrowing steps have identity substitutions, we can have bisimulation as in Section 3.

Theorem 9 (Kripke bisimulation by narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. Let $U \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$. Let each transition in $\mathcal{R}each_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)$ be of the form $[t]_E \overset{id}{\leadsto}_{R,E;\Pi} [t']_E$. Then the relation $\preccurlyeq_E$ defines a bisimulation $\mathcal{R}each_{\mathcal{T}^{\Pi}_{\mathcal{R}}}(U) \preccurlyeq_E \mathcal{R}each_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)$.*

Lemma 5 ($\leadsto_{R,E;\Pi}$-equivalence of G^{-1}). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. Let $T \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ such that for each $[t]_E, [t']_E \in T$, $[t]_E \overset{\sigma}{\leadsto}_{R,E;\Pi} [t']_E$ implies $\sigma = id$. The relations $\{=_E^{-1}, \approx_E^{-1}, \preccurlyeq_E^{-1}\}$ on T are $\leadsto_{R,E;\Pi}$-equivalent.*

Proof. We only have to prove it for $\preccurlyeq_E^{-1}$, since $=_E$ and $\approx_E$ are symmetric. Let $[t]_E, [t']_E \in \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ such that $t \overset{id}{\leadsto}_{R,E;\Pi} t'$ using rule $l \to r$, i.e., $t \to_{R,E} t'$ using rule $l \to r$ and $t' \overset{id}{\leadsto}_{\Pi} t'$. Let $[w]_E \in \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and τ such that $\tau(t) =_E w$ (i.e., $t \preccurlyeq_E^{-1} w$). Then, $\tau(t) \overset{id}{\leadsto}_{R,E;\Pi} \tau(t')$ using rule $l \to r$, i.e., $\tau(t) \to_{R,E} \tau(t')$ using rule $l \to r$ and $\tau(t') \overset{id}{\leadsto}_{\Pi} \tau(t')$. And thus, let $w' = \tau(t')$, we have $t' \preccurlyeq_E^{-1} w'$. □

Theorem 10 (Kripke bisimulation by G-narrowing). *Let $\mathcal{R} = (\Sigma, E, R)$ be a deadlock-free topmost rewrite theory with top sort* State *and $\mathcal{E}_\Pi = (\Sigma_\Pi, E \uplus E_\Pi)$ be a* Bool*-equational theory defining the atomic propositions Π on $\mathcal{T}_\Sigma(\mathcal{X})_{\mathsf{State}}$ that has a complete and finitary unification algorithm. Let $U \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$. Let $G \subseteq \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}} \times \mathcal{T}^{\Pi}_{\Sigma/E}(\mathcal{X})_{\mathsf{State}}$ and G^{-1} be $\leadsto_{R,E;\Pi}$-equivalent. Let each transition in $\mathcal{R}each^{G}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)$ be of the form $[t]_E \overset{id}{\leadsto}{}^{G}_{R,E;\Pi} [t']_E$. Then the relation G defines a bisimulation $\mathcal{R}each_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)\ G\ \mathcal{R}each^{G}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(U)$.*

Example 6. Consider Example 1 again. Now we are interested in the following atomic propositions $\Pi = \{$`ever-wait?`, `ever-crit?`$\}$ expressing that at least one process has been in its waiting (resp. critical) state.

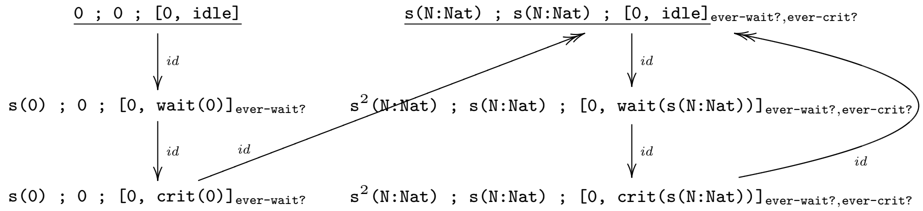
0 ; 0 ; [0, idle]
id
s(0) ; 0 ; [0, wait(0)]ever-wait?
id
id
s(0) ; 0 ; [0, crit(0)]ever-wait?
s(N:Nat) ; s(N:Nat) ; [0, idle]ever-wait?,ever-crit?
id
s2(N:Nat) ; s(N:Nat) ; [0, wait(s(N:Nat))]ever-wait?,ever-crit?
id
id
s2(N:Nat) ; s(N:Nat) ; [0, crit(s(N:Nat))]ever-wait?,ever-crit?

Fig. 6. Finite Kripke structure $\mathcal{R}each^{\preceq_E}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(\{w_1, w_2\})$ with w_1 = "`0 ; 0 ; [0, idle]`" and w_2 = "`s(N:Nat) ; s(N:Nat) ; [0, idle]`"

```
fmod BAKERY-PROPS is
  protecting BAKERY-SYNTAX .
  sort Bool .
  ops tt ff : -> Bool .
  ops ever-wait? ever-crit? : State -> Bool .
  vars N M : Nat .
  vars PS : ProcSet .
  eq ever-wait?(0 ; M ; PS) = ff .
  eq ever-wait?(s(N) ; M ; PS) = tt .
  eq ever-crit?(N ; 0 ; PS) = ff .
  eq ever-crit?(N ; s(M) ; PS) = tt .
endfm
```

Consider the initial state w = "`N:Nat ; N:Nat ; [0, idle]`". The transition system of Figure 3 cannot be transformed into a Π-Kripke structure, since propositions {`ever-wait?`, `ever-crit?`} cannot be evaluated in, for instance, state "`N:Nat ; N:Nat ; [0, idle]`". Therefore, we must instantiate term w using the narrowing relation $\leadsto_{\Pi}$ and obtain terms w_1 = "`0 ; 0 ; [0, idle]`" and w_2 = "`s(N:Nat) ; s(N:Nat) ; [0, idle]`", i.e., $w \leadsto_{\Pi} w_1$ and $w \leadsto_{\Pi} w_2$. The Π-Kripke structure $\mathcal{R}each^{\preceq_E}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(\{w_1, w_2\})$ is depicted in Figure 6, where, since it is a finite-state system, we can use standard LTL model checking techniques to model check the formulas "`ever-wait?` $\Rightarrow \Diamond$`ever-crit?`" and "$\Box$(`ever-crit?` $\Rightarrow$ `ever-wait?`)", which in this case hold in $\mathcal{R}each^{\preceq_E}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(\{w_1, w_2\})$. Therefore, by Theorems 5, 7 and 8, the above LTL formulas also hold for the infinite-state `BAKERY` system $\mathcal{T}^{\Pi}_{\mathcal{R}}$ and the infinite set $[\![\{w_1, w_2\}]\!]$ of initial states. Note that we have a bisimulation and then CTL* formulas can also be verified.

Example 7. Consider Example 1 again. Now we are interested in the following atomic propositions Π = {`ex?`} denoting the mutual exclusion in critical section.

```
fmod BAKERY-PROPS is
  protecting BAKERY .
  sort Bool .
  ops tt ff : -> Bool .
  op ex? : State -> Bool .
  var WS : ProcWaitSet .
  var PS : ProcSet .
  vars N M K : Nat .
  eq ex?(N ; M ; WS) = tt .
  eq ex?(N ; M ; [K1, crit(M1)] WS) = tt .
  eq ex?(N ; M ; [K1, crit(M1)] [K2, crit(M2)] PS) = ff .
endfm
```

Given the initial state t_4 = "`N:Nat ; N:Nat ; [0, idle] [s(0),idle]`", the Π-Kripke structure $\mathcal{R}each^{\preceq_E}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(t_4)$ is similar to Figure 4 but where the atomic proposition `ex?` holds in every state. Then, we can easily verify the formula "$\Box$`ex?`", stating that proposition `ex?` holds in every possible state, using a standard model checking algorithm and without having to explore the infinite-state Π-Kripke structure $\mathcal{T}_{\mathcal{R}}^{\Pi}$ for all initial states in the infinite set of instances $[\![t_4]\!]$, which is, of course, impossible to do with finitary model checking methods.

Similar arguments to those in Section 3 can be given in favor of narrowing for model checking $ACTL^*$ (or CTL^*) properties of systems that are either infinite-state or too big for standard finite-state methods. For example, when a set $U \subseteq \mathcal{T}_{\Sigma/E}^{\Pi}(\mathcal{X})_{\mathsf{State}}$ of initial states is provided, $\mathcal{R}each^{G}_{\mathcal{N}_{\mathcal{R}}^{\Pi}}(U)$ for some G such as $\preccurlyeq_E$ can be finite when $\mathcal{R}each_{\mathcal{T}_{\mathcal{R}}^{\Pi}}([\![U]\!])$ is infinite, or can be much smaller even in the finite-state case. And U can be finite whereas $[\![U]\!]$ may easily be infinite, making it impossible to verify properties by standard model checking algorithms.

5 Concluding Remarks

We have shown that, by specifying possibly infinite concurrent systems as rewrite theories, narrowing gives rise to a lifting simulation and provides a useful semi-decision procedure to answer reachability questions. We have also proposed a method to fold the narrowing graph that, when it yields a finite system, allows algorithmic verification of such reachability questions, including invariants. Furthermore, we have extended these techniques to the verification of $ACTL^*$ and LTL formulas. Much work remains ahead, including:

1. Gaining experience with many more examples: concurrent systems, security protocols, Java program verification, etc.
2. Implementing these techniques in Maude [12,11], taking advantage of its LTL model checker [19].
3. Investigating other folding relations that might further improve the generation of a finite narrowing search space.
4. Allowing more general state predicate definitions, for example with data parameters.
5. Studying how grammar-based techniques [21] and narrowing strategies [22] can be used to further reduce the narrowing search space.

Acknowledgments. We cordially thank Prasanna Thati, with whom we developed the foundations of narrowing-based reachability analysis. We also warmly thank Catherine Meadows for our joint work on narowing-based security verification and the Maude-NPA; this research has provided us with a rich stimulus for investigating the new techniques presented here, which will also be very useful for security verification. S. Escobar has been partially supported by the EU (FEDER) and Spanish MEC TIN-2004-7943-C04-02 project, the Generalitat Valenciana under grant GV06/285, and the Acción Integrada Hispano-Alemana HA2006-0007. J. Meseguer's research has been supported in part by ONR Grant N00014-02-1-0715 and NSF Grant NSF CNS 05-24516.

References

1. Elvira Albert and Germán Vidal. The narrowing-driven approach to functional logic program specialization. *New Generation Comput.*, 20(1):3–26, 2001.
2. M. Alpuente, M. Falaschi, and G. Vidal. Partial Evaluation of Functional Logic Programs. *ACM Transactions on Programming Languages and Systems*, 20(4):768–844, 1998.
3. Sergio Antoy and Zena M. Ariola. Narrowing the narrowing space. In Hugh Glaser, Pieter H. Hartel, and Herbert Kuchen, editors, *PLILP*, volume 1292 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, Berlin, 1997.
4. Samik Basu, Madhavan Mukund, C. R. Ramakrishnan, I. V. Ramakrishnan, and Rakesh M. Verma. Local and symbolic bisimulation using tabled constraint logic programming. In Philippe Codognet, editor, *ICLP*, volume 2237 of *Lecture Notes in Computer Science*, pages 166–180. Springer, 2001.
5. Ahmed Bouajjani. Languages, rewriting systems, and verification of infinite-state systems. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *ICALP*, volume 2076 of *Lecture Notes in Computer Science*, pages 24–39. Springer, 2001.
6. Ahmed Bouajjani and Javier Esparza. Rewriting models of boolean programs. In Frank Pfenning, editor, *RTA*, volume 4098 of *Lecture Notes in Computer Science*, pages 136–150. Springer, 2006.
7. Ahmed Bouajjani and Richard Mayr. Model checking lossy vector addition systems. In *STACS*, pages 323–333, 1999.
8. O. Burkart, D. Caucal, F. Moller, and B. Steffen. Verification over Infinite States. In *Handbook of Process Algebra*, pages 545–623. Elsevier Publishing, 2001.
9. Edmund M. Clarke, Orna Grumberg, and David E. Long. Model checking and abstraction. *ACM Transactions on Programming Languages and Systems*, 16(5):1512–1542, September 1994.
10. E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking*. MIT Press, 2001.
11. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. The Maude 2.0 System. In R. Nieuwenhuis, editor, *Rewriting Techniques and Applications, 12th International Conference, RTA 2003, RTA 2003, Valencia, Spain, June 9-11, 2003, Proceedings*, volume 2706 of *Lecture Notes in Computer Science*, pages 76–87. Springer-Verlag, 2003.
12. Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and José Quesada. Maude: specification and programming in rewriting logic. *Theoretical Computer Science*, 285:187–243, 2002.
13. Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn Talcott. *All About Maude: A High-Performance Logical Framework*. Springer-Verlag, 2007. To appear.
14. Giorgio Delzanno. Constraint multiset rewriting. Technical report, DISI - Università di Genova, 2002.
15. Giorgio Delzanno and Andreas Podelski. Constraint-based deductive model checking. *STTT*, 3(3):250–270, 2001.
16. Grit Denker, José Meseguer, and Carolyn L. Talcott. Protocol specification and analysis in Maude. In N. Heintze and J. Wing, editors, *Proceedings of Workshop on Formal Methods and Security Protocols, June 25, 1998, Indianapolis, Indiana*, 1998. `http://www.cs.bell-labs.com/who/nch/fmsp/index.html`.
17. Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Methods and Semantics, chapter 6, pages 244–320. Elsevier Science, 1990.

18. Nachum Dershowitz, Subrata Mitra, and G. Sivakumar. Decidable matching for convergent systems (preliminary version). In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE-11)*, volume 607 of *Lecture Notes in Computer Science*, pages 589–602. Springer, 1992.
19. Steven Eker, José Meseguer, and Ambarish Sridharanarayanan. The Maude LTL model checker. In F. Gadducci and U. Montanari, editors, *Proc. 4th. Intl. Workshop on Rewriting Logic and its Applications*. ENTCS, Elsevier, 2002.
20. A. Emerson and K. Namjoshi. On model checking for nondeterministic infinite state systems. In *IEEE Symposium on Logic in Computer Science*, 1998.
21. Santiago Escobar, Catherine Meadows, and José Meseguer. A rewriting-based inference system for the NRL Protocol Analyzer and its meta-logical properties. *Theoretical Computer Science*, 367(1-2):162–202, 2006.
22. Santiago Escobar, José Meseguer, and Prasanna Thati. Natural narrowing for general term rewriting systems. In J. Giesl, editor, *Proc. of 16th International Conference on Rewriting Techniques and Applications, RTA'05*, volume 3467 of *Lecture Notes in Computer Science*, pages 279–293. Springer-Verlag, Berlin, 2005.
23. A. Farzan and J. Meseguer. State space reduction of rewrite theories using invisible transitions. In *Proc. AMAST'06*, volume 4019 of *Springer LNCS*, page 142.
24. Alain Finkel and Ph. Schnoebelen. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256(1):63–92, 2001.
25. Thomas Genet and Valérie Viet Triem Tong. Reachability analysis of term rewriting systems with Timbuk. In 8^{th} *International Conference on Logic for Programming*, volume 2250 of *Lecture Notes in Computer Science*, 2001.
26. Susanne Graf and Hassen Saidi. Construction of abstract state graphs with PVS. In Orna Grumberg, editor, *Computer Aided Verification. 9th International Conference, CAV'97, Haifa, Israel, June 22-25, 1997, Proceedings*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83, 1997.
27. Jean-Pierre Jouannaud, Claude Kirchner, and Helene Kirchner. Incremental construction of unification algorithms in equational theories. In 10^{th} *International Colloquium on Automata, Languages and Programming*, volume 154 of *Lecture Notes in Computer Science*, pages 361–373. Springer, 1983.
28. D. Kapur and P. Narendran. Matching, Unification and Complexity. *ACM SIGSAM Bulletin*, 21(4):6–9, 1987.
29. Yonit Kesten and Amir Pnueli. Control and data abstraction: The cornerstones of practical formal verification. *International Journal on Software Tools for Technology Transfer*, 4(2):328–342, 2000.
30. L. Lafave and J.P. Gallagher. Constraint-based Partial Evaluation of Rewriting-based Functional Logic Programs. In *Proc. of 7th International Workshop on Logic-based Program Synthesis and Transformation, LOPSTR'97*, volume 1463 of *Lecture Notes in Computer Science*, pages 168–188. Springer-Verlag, Berlin, 1997.
31. C. Loiseaux, S. Graf, J. Sifakis, A. Bouajjani, and S. Bensalem. Property preserving abstractions for the verification of concurrent systems. *Formal Methods in System Design*, 6:1–36, 1995.
32. P. Manolios. *Mechanical Verification of Reactive Systems*. PhD thesis, University of Texas at Austin, 2001.
33. Narciso Martí-Oliet, José Meseguer, and Miguel Palomino. Theoroidal maps as algebraic simulations. in Proc. WADT'04, Springer LNCS 3423, 126–143. 2005.
34. J. Meseguer and P. Thati. Symbolic reachability analysis using narrowing and its application to the verification of cryptographic protocols. In N. Martí-Oliet, editor, *Proc. 5th. Intl. Workshop on Rewriting Logic and its Applications*. ENTCS, Elsevier, 2004.

35. J. Meseguer and P. Thati. Symbolic reachability analysis using narrowing and its application to verification of cryptographic protocols. *High-Order Symbolic Computation*, 2006. To appear.
36. José Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.
37. José Meseguer. Multiparadigm logic programming. In H. Kirchner and G. Levi, editors, *Proc. 3rd Intl. Conf. on Algebraic and Logic Programming*, pages 158–200. Springer LNCS 632, 1992.
38. José Meseguer. Membership algebra as a logical framework for equational specification. In F. Parisi-Presicce, editor, *Proc. WADT'97*, pages 18–61. Springer LNCS 1376, 1998.
39. José Meseguer, Miguel Palomino, and Narciso Martí-Oliet. Equational abstractions. in Proc. CADE-19, Springer LNCS, Vol. 2741, 2–16, 2003.
40. Hitoshi Ohsaki, Hiroyuki Seki, and Toshinori Takai. Recognizing boolean closed A-tree languages with membership conditional mechanism. In 14th *International Conference on Rewriting Techniques and Applications*, volume 2706 of *Lecture notes in computer science*, pages 483–498. Springer Verlag, 2003.
41. Hassen Saïdi and Natarajan Shankar. Abstract and model check while you prove. In Nicolas Halbwachs and Doron Peled, editors, *Computer Aided Verification. 11th International Conference, CAV'99, Trento, Italy, July 6-10, 1999, Proceedings*, volume 1633 of *Lecture Notes in Computer Science*, pages 443–454, 1999.
42. TeReSe, editor. *Term Rewriting Systems.* Cambridge University Press, Cambridge, 2003.
43. V.F. Turchin. Program Transformation by Supercompilation. In H. Ganzinger and N.D. Jones, editors, *Programs as Data Objects, 1985*, volume 217 of *Lecture Notes in Computer Science*, pages 257–281. Springer-Verlag, Berlin, 1986.
44. A. Vardhan, K. Sen, M. Viswanathan, and G. Agha. Using language inference to verify omega-regular properties. In *Tools and Algorithms for the Construction and Analysis of Systems TACAS '05*, volume 3440 of *Lecture Notes in Computer Science*, pages 45–60. Springer-Verlag, 2005.
45. Emanuele Viola. E-unifiability via narrowing. In Antonio Restivo, Simona Ronchi Della Rocca, and Luca Roversi, editors, *ICTCS*, volume 2202 of *Lecture Notes in Computer Science*, pages 426–438. Springer, 2001.

A Another example of concurrent system

Example 8. Consider a concurrent system counting the number of reader and writer processes accessing a critical resource; the example is borrowed from [13]. Readers and writers can leave the resource at any time, but writers can only gain access to it if nobody else is using it, and readers only if there are no writers. This system can be specified as an order-sorted topmost rewrite theory in Maude as follows:

```
mod READERS-WRITERS is
  sort Nat State .
  op <_,_> : Nat Nat -> State .  --- readers/writers
  op 0 : -> Nat .
  op s : Nat -> Nat .
  vars R W : Nat .

  rl < 0, 0 > => < 0, s(0) > .
```

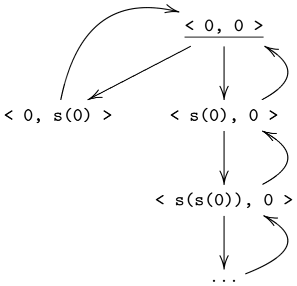

Fig. 7. Infinite transition system $\mathcal{R}each_{\mathcal{T}_\mathcal{R}}(\texttt{< 0, 0 >})$

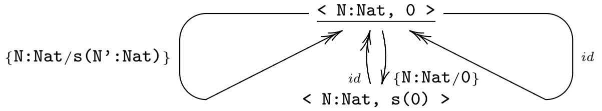

Fig. 8. Finite transition system $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_\mathcal{R}}(\texttt{< N:Nat, 0 >})$

```
  rl < R, s(W) > => < R, W > .
  rl < R, 0 > => < s(R), 0 > .
  rl < s(R), W > => < R, W > .
endm
```

Given the initial state u_1 = "`< 0, 0 >`", the infinite transition system $\mathcal{R}each_{\mathcal{T}_\mathcal{R}}(u_1)$ is depicted in Figure 7.

Example 9. Consider Example 8 and let u_2 = "`< N:Nat, 0 >`". The finite transition system $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_\mathcal{R}}(u_2)$ is depicted in Figure 8. Note that we have a bisimulation between the infinite transition system $\mathcal{R}each_{\mathcal{T}_\mathcal{R}}(\texttt{< 0 , 0 >})$ shown in Figure 7 and $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_\mathcal{R}}(\texttt{< N:Nat, 0 >})$ in Figure 8. Furthermore, we have a bisimulation between the infinite number of infinite transition systems associated to $\mathcal{R}each_{\mathcal{T}_\mathcal{R}}([\![\texttt{< N:Nat , 0 >}]\!])$ and $\mathcal{R}each^{\preceq_E}_{\mathcal{N}_\mathcal{R}}(\texttt{< N:Nat, 0 >})$.

Example 10. Consider Example 8. We are interested in the atomic propositions $\Pi = \{$`one-writer-atmost?`, `some-readers?`, `some-writers?`, `exclusion?`$\}$ testing, respectively, that there is at most one writer in the system, that there is at least one reader in the system, that there is at least one writer in the system, and the mutual exclusion of the critical resource. These atomic propositions are described by the following equational theory in Maude.

```
fmod READERS-WRITERS-PROPS is
  protecting READERS-WRITERS .
  sort Bool .
  ops tt ff : -> Bool .
  op one-writer-atmost? : State -> Bool .
  eq one-writer-atmost?(< N:Nat, 0 >) = tt .
  eq one-writer-atmost?(< N:Nat, s(0) >) = tt .
  eq one-writer-atmost?(< N:Nat, s(s(M:Nat)) >) = ff .
```

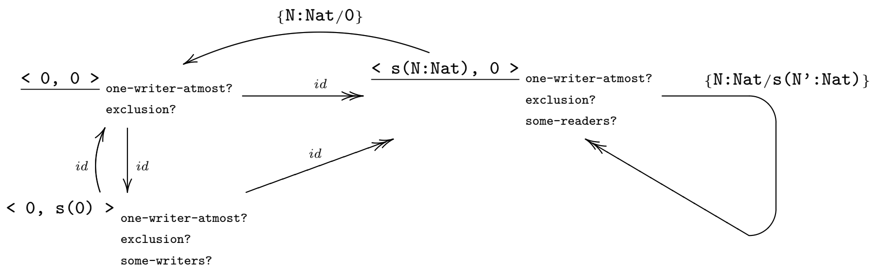

Fig. 9. Finite Kripke-structure $\mathcal{R}each^{\preceq_E}_{\mathcal{N}^{\Pi}_{\mathcal{R}}}(\{$`< 0, 0 >`$,$`< s(N:Nat), 0 >`$\})$

```
  op some-readers? : State -> Bool .
  eq some-readers?(< s(N:Nat), M:Nat >) = tt .
  eq some-readers?(< 0, M:Nat >) = ff .

  op some-writers? : State -> Bool .
  eq some-writers?(< N:Nat, s(N:Nat) >) = tt .
  eq some-writers?(< N:Nat, 0 >) = ff .

  op exclusion? : State -> Bool .
  eq exclusion?(< s(N:Nat), s(M:Nat) >) = ff .
  eq exclusion?(< 0, M:Nat >) = tt .
  eq exclusion?(< N:Nat, 0 >) = tt .
endfm
```

Given the initial state u_2 = "`< N:Nat, 0 >`", the transition system of Figure 8 cannot be transformed into a Π-kripke structure, since some propositions cannot be evaluated, e.g. `some-readers?`. Therefore, we must instantiate term u_2 using the narrowing relation $\leadsto_{\Pi}$ and obtain terms u_2' = "`< 0, 0 >`" and u_2'' = "`< s(N:Nat), 0 >`", i.e., $u_2 \leadsto_{\Pi} u_2'$ and $u_2 \leadsto_{\Pi} u_2''$. The Π-Kripke structure $\mathcal{R}each^{\preceq_E}_{\mathcal{T}^{\Pi}_{\mathcal{R}}}(\{u_2', u_2''\})$ is depicted in Figure 9, where, since it is a finite-state system, we can use standard CTL* model checking techniques to model check the formulas "$A\Box$`one-writer?`", "$A\Box$`exclusion?`", and "$A\Box($`many-readers?` $\Rightarrow$ $(E\Diamond$`one-writer?`$))$", which in this case hold in $\mathcal{R}each^{\preceq_E}_{\mathcal{T}^{\Pi}_{\mathcal{R}}}(\{u_2', u_2''\})$. Therefore, since we have a bisimulation and by Theorems 5, 9, and 10, the above CTL* formulas also hold for the infinite-state `READERS-WRITERS` system $\mathcal{T}^{\Pi}_{\mathcal{R}}$ and the infinite set $[\![u_2]\!]$ of initial states.