NASA/TM–2013-218033

Rewriting Modulo SMT

Camilo Rocha
Escuela Colombiana de Ingeniería, Bogotá, Colombia

José Meseguer
University of Illinois at Urbana-Champaign, Urbana, Illinois

César A. Muñoz
Langley Research Center, Hampton, Virginia

August 2013

NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Report Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at *http://www.sti.nasa.gov*

- E-mail your question to help@sti.nasa.gov

- Fax your question to the NASA STI Information Desk at 443-757-5803

- Phone the NASA STI Information Desk at 443-757-5802

- Write to:
  STI Information Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076-1320

NASA/TM–2013-218033

Rewriting Modulo SMT

Camilo Rocha
Escuela Colombiana de Ingeniería, Bogotá, Colombia

José Meseguer
University of Illinois at Urbana-Champaign, Urbana, Illinois

César A. Muñoz
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

August 2013

The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration

Available from:

NASA Center for AeroSpace Information
7115 Standard Drive
Hanover, MD 21076-1320
443-757-5802

Abstract

Combining symbolic techniques such as: (i) SMT solving, (ii) rewriting modulo theories, and (iii) model checking can enable the analysis of infinite-state systems outside the scope of each such technique. This paper proposes rewriting modulo SMT as a new technique combining the powers of (i)-(iii) and ideally suited to model and analyze infinite-state open systems; that is, systems that interact with a non-deterministic environment. Such systems exhibit both internal non-determinism due to the system, and external non-determinism due to the environment. They are not amenable to finite-state model checking analysis because they typically are infinite-state. By being reducible to standard rewriting using reflective techniques, rewriting modulo SMT can both naturally model and analyze open systems without requiring any changes to rewriting-based reachability analysis techniques for closed systems. This is illustrated by the analysis of a real-time system beyond the scope of timed automata methods.

1 Introduction

Symbolic techniques that represent possibly infinite sets of states by *symbolic constraints* have become essential to make formal verification —using model checking, theorem proving, or combining features from both— much more scalable. They provide high levels of automation when verifying various kinds of infinite-state systems. Such techniques have been vigorously developed, adopted in many systems, and proved highly successful. They include: (i) SAT solving and other decision procedures, and their combination into *Satisfiability Modulo Theories* (SMT) solvers; (ii) *rewriting- and unification-based techniques*, including *rewriting modulo theories* and *narrowing modulo theories*; and (iii) *symbolic model checking* techniques.

A key open research issue limiting the applicability of current symbolic techniques is lack of, or limited support for, *extensibility.* That is, although certain classes of systems can be formalized in ways that enable the application of specific symbolic analysis techniques, many other systems of interest (Section 6 provides an example) fall outside the scope of some existing symbolic techniques. In such cases one would like to *extend and combine* the power of symbolic techniques to analyze the given system.

Certainly, some techniques to combine methods or procedures provide useful ways of broadening the scope of methods and tools. For example: (i) combinations of decision procedures, e.g., [26, 27]; and of unification algorithms, e.g., [5, 10]; (ii) combinations of theorem provers with decision procedures, e.g., [1, 9, 31]; and (iii) integration of SMT solvers in model checkers, e.g., [3, 17, 25, 34, 36]. However, it seems fair to say that at present there is a lack of *general extensibility techniques for symbolic analysis* that can *simultaneously* combine the power of SMT solving, rewriting- and narrowing-based analysis, and symbolic model checking to analyze systems beyond the scope of each separate analysis technique.

The main goal of the present work is to propose a new symbolic technique that seamlessly combines the powers of rewriting modulo theories, SMT solving, and model checking. For brevity, this technique is called *rewriting modulo SMT*, although it could more precisely be called "rewriting modulo SMT+B," where B is any equational theory having a matching algorithm. It complements, and has similarities with, another symbolic technique combining narrowing modulo theories and model checking, namely, *narrowing-based* reachability analysis [24] and its extension to symbolic LTL model checking [7].

Rewriting modulo SMT can be usefully applied to increase the power of equational reasoning, but its full power (including its model checking capabilities) is best exploited when applied to *concurrent open systems.* The key point is that deterministic systems can be naturally specified by equational theories, but specification of concurrent, non-deterministic systems requires *rewrite theories* [21], that is, triples $\mathcal{R} = (\Sigma, E, R)$ with (Σ, E) an equational theory describing system *states* as elements of the initial algebra $\mathcal{T}_{\Sigma/E}$, and R rewrite rules describing the system's *local concurrent transitions.* Although extensive experience and many tools exist to specify and analyze concurrent systems in this way (see the survey [23]), the specification of concurrent *open* systems remains quite challenging. However, as explained below, specification and analysis of open systems becomes easy and unproblematic with rewriting modulo SMT.

An *open* system is a concurrent system that *interacts* with an external, non-deterministic *environment*. When such a system is specified by a rewrite theory $\mathcal{R} = (\Sigma, E, R)$, it has *two* sources of non-determinism, one internal and the other external. Internal non-determinism comes from the fact that in a given system state *different* instances of rules in R may be enabled, and the local transitions thus enabled may lead to completely different states. What is peculiar about an open system is that it also has *external*, and often infinitely-branching, non-determinism due to the environment. That is, the *state* of an open system must include the state changes due to the environment. Technically, this means that, while a system transition in a closed system can be described by a rewrite rule $t{\to}t'$ with $\mathit{vars}(t'){\subseteq}\mathit{vars}(t)$, a transition in an open system is instead modeled by a rule of the form $t(\overrightarrow{x}){\to}t'(\overrightarrow{x}, \overrightarrow{y})$, where $\overrightarrow{y}$ represents fresh new variables. Therefore, a substitution for the variables $\overrightarrow{x}{\uplus}\overrightarrow{y}$ decomposes into *two* substitutions, one, say θ, for the variables $\overrightarrow{x}$ under the control of the system, and another, say ρ, for the variables $\overrightarrow{y}$ under the control of the environment. In rewriting modulo SMT such open systems are described by conditional rewrite rules of the form $t(\overrightarrow{x}) \to t'(\overrightarrow{x}, \overrightarrow{y})$ **if** ϕ, where ϕ is a constraint solvable by an SMT solver. This constraint ϕ may still allow the environment to choose an infinite number of substitutions ρ for the variables $\overrightarrow{y}$, but can exclude choices that the environment will never make.

The non-trivial challenges of modeling and analyzing open systems can now be better explained. They include: (1) the enormous and possibly infinitary non-determinism due to the environment, which typically renders finite-state model checking impossible or unfeasible; (2) the impossibility of *executing* the rewrite theory $\mathcal{R} = (\Sigma, E, R)$ in the standard sense, due to the non-deterministic choice of ρ; and (3) the in general undecidable challenge of checking the rule's condition ϕ, since without knowing ρ, the condition $\phi\theta$ is non-ground, so that its E-satisfiability may be undecidable, even for E confluent and terminating. As further explained in the paper, challenges (1)–(3) are all met successfully by rewriting modulo SMT because: (1) states are represented not as concrete states (i.e., *ground* terms) but as *symbolic constrained terms* $\langle t\,;\varphi\rangle$ with t a term with variables ranging in the domain(s) handled by the SMT solver and φ an SMT-solvable formula, so that the choice of ρ is avoided; (2) rewriting modulo SMT can *symbolically rewrite* such pairs $\langle t\,;\varphi\rangle$ (describing possibly infinite sets of concrete states) to other pairs $\langle t'\,;\varphi'\rangle$; and (3) decidability of $\phi\theta$ (more precisely of $\varphi \wedge \phi\theta$) can be settled by invoking an SMT solver.

How rewriting modulo SMT is seamlessly integrated with a symbolic style of model checking for infinite-state systems, thus combining the power of rewriting, SMT solving, and model checking, is also worth explaining. The essential point (further expanded in Section 5) is that, by exploiting the fact that rewriting logic is *reflective* [14], rewriting modulo SMT can be *reduced to standard rewriting.* Specifically, this means that all the techniques, algorithms, and tools available for model checking closed systems specified as rewrite theories, such as Maude's search-based reachability analysis [13], become *directly available* to perform *symbolic* reachability analysis on systems that are now infinite-state. This is illustrated by the formal analysis of an infite-state real-time system outside the scope of timed-automata techniques in Section 6.

Contributions. The contributions of this paper can be summarized as follows: (1) it presents rewriting modulo SMT as a new symbolic technique combining the powers of rewriting, SMT solving, and model checking; (2) this combined power can be applied to model and analyze systems outside the scope of each individual technique; (3) in particular, it is ideally suited to model and analyze the challenging case of *open systems*; and (4) because of its reflective reduction to standard rewriting, current algorithms and tools for model checking closed systems can be *reused* in this new symbolic setting without requiring any changes to their implementation.

2 Preliminaries

We recall notation on terms, term algebras, and equational theories as in [6, 18].

An *order-sorted signature* Σ is a tuple $\Sigma{=}(S, \leq, F)$ with a finite poset of sorts $(S, \leq)$ and set of function symbols F. The binary relation $\equiv_{\leq}$ denotes the equivalence relation generated by $\leq$ on S and its point-wise extension to strings in S^*. The function symbols in F can be subsort-overloaded and satisfy the condition that, for $(w, s), (w', s') \in S^* \times S$, if $f \in F_{w,s} {\cap} F_{w',s'}$, then $w{\equiv_{\leq}}w'$ implies $s{\equiv_{\leq}}s'$. A *top sort* in Σ is a sort $s \in S$ such that if $s' \in S$ and $s \equiv_{\leq} s'$, then $s' \leq s$. For any sort $s \in S$, the expression $[s]$ denotes the connected component of s, that is, $[s] = [s]_{\equiv_{\leq}}$.

The *variables* X are an S-indexed family $X{=}\{X_s\}_{s \in S}$ of disjoint variable sets with each X_s countably infinite. The *set of terms of sort* s is denoted $T_{\Sigma}(X)_s$ and the *set of ground terms of sort* s is denoted $T_{\Sigma,s}$. $\mathcal{T}_{\Sigma}(X)$ and $\mathcal{T}_{\Sigma}$ denote the corresponding order-sorted Σ-term algebras. All order-sorted signatures are assumed *preregular* [18], i.e., each Σ-term has a *least sort* $ls(t){\in}S$ s.t. $t{\in}T_{\Sigma}(X)_{ls(t)}$. For $S'{\subseteq}S$, a term is called S'*-linear* if no variable with sort in S' occurs in it twice. The *set of variables* of t is written $vars(t)$.

A *substitution* is an S-indexed mapping $\theta : X \longrightarrow T_{\Sigma}(X)$ that is different from the identity only for a finite subset of X. The identity substitution is denoted by id and $\theta|_Y$ denotes the restriction of θ to a family of variables $Y \subseteq X$. $dom(\theta)$ denotes the domain of θ, i.e., the subfamily of X for which $\theta(x) \neq x$, and $ran(\theta)$ denotes the family of variables introduced by $\theta(x)$, for $x \in dom(\theta)$. Substitutions extend homomorphically to terms in the natural way. A substitution θ is called *ground* iff $ran(\theta) = \emptyset$. The application of a substitution θ to a term t is denoted by $t\theta$ and the composition of two substitutions θ_1 and θ_2 is denoted by $\theta_1\theta_2$. A *context* C is a λ-term of the form $C = \lambda x_1, \ldots, x_n.c$ with $c \in T_{\Sigma}(X)$ and $\{x_1, \ldots, x_n\} \subseteq vars(c)$; it can be viewed as a n-ary function $C(t_1, \ldots, t_n) = c\theta$, where $\theta(x_i) = t_i$ for $1 \leq i \leq n$ and $\theta(x) = x$ otherwise.

A Σ*-equation* is an unoriented pair $t = u$ with $t \in T_{\Sigma}(X)_{s_t}$, $u \in T_{\Sigma}(X)_{s_u}$, and $s_t \equiv_{\leq} s_u$. A *conditional* Σ*-equation* is a triple $t = u$ **if** γ, with $t = u$ a Σ-equation and γ a finite conjunction of Σ-equations; it is called *unconditional* if γ is the empty conjunction. An *equational theory* is a tuple (Σ, E), with Σ an order-sorted signature and E a finite collection of (possibly conditional) Σ-equations. We assume throughout that $T_{\Sigma,s} \neq \emptyset$ for each $s \in S$, because this affords a simpler deduction system. An equational theory $\mathcal{E} = (\Sigma, E)$ induces the congruence relation $=_{\mathcal{E}}$ on $T_{\Sigma}(X)$ defined for $t, u \in T_{\Sigma}(X)$ by $t =_{\mathcal{E}} u$ iff $\mathcal{E} \vdash t = u$ by the deduction rules for order-sorted equational logic in [22]. Similarly, $=^1_{\mathcal{E}}$ denotes provable $\mathcal{E}$-equality in *one step* of deduction. The *$\mathcal{E}$-subsumption* ordering $\ll_{\mathcal{E}}$ is the binary relation on $T_\Sigma(X)$ defined for any $t, u \in T_\Sigma(X)$ by $t \ll_{\mathcal{E}} u$ iff there is a substitution $\theta : X \longrightarrow T_\Sigma(X)$ such that $t =_{\mathcal{E}} u\theta$. A set of equations E is called *collapse-free* for a subset of sorts $S' \subseteq S$ iff for any $t = u \in E$ and any substitution $\theta : X \longrightarrow T_\Sigma(X)$ neither $t\theta$ nor $u\theta$ are a variable for some sort $s \in S'$. $\mathcal{T}_{\mathcal{E}}(X)$ and $\mathcal{T}_{\mathcal{E}}$ (also written $\mathcal{T}_{\Sigma/E}(X)$ and $\mathcal{T}_{\Sigma/E}$) denote the quotient algebras induced by $=_{\mathcal{E}}$ on the term algebras $\mathcal{T}_\Sigma(X)$ and $\mathcal{T}_\Sigma$, respectively; $\mathcal{T}_{\Sigma/E}$ is called the *initial algebra* of (Σ, E). A theory inclusion $(\Sigma, E) \subseteq (\Sigma', E')$, with $\Sigma \subseteq \Sigma'$ and $E \subseteq E'$, is called *protecting* iff the unique Σ-homomorphism $\mathcal{T}_{\Sigma/E} \longrightarrow \mathcal{T}_{\Sigma'/E'}|_\Sigma$ to the Σ-reduct of the initial algebra $\mathcal{T}_{\Sigma'/E'}$ is a Σ-isomorphism, written $\mathcal{T}_{\Sigma/E} \simeq \mathcal{T}_{\Sigma'/E'}|_\Sigma$. A set of equations E is called *regular* iff $\mathit{vars}(t) = \mathit{vars}(u)$ for any equation $t = u$ **if** $\gamma \in E$.

Appropriate requirements are needed to make an equational theory $\mathcal{E}$ *admissible*, i.e., *executable* in rewriting language such as Maude [13]. It is assumed that the equations of $\mathcal{E}$ can be decomposed into a disjoint union $E \uplus B$, with B a collection of structural axioms (such as associativity, and/or commutativity, and/or identity) for which there exists a *matching algorithm modulo B* producing a finite number of B-matching solutions, or failing otherwise, and that the equations E can be oriented into a set (of possibly conditional) sort-decreasing, operationally terminating, and confluent conditional rewrite rules $\overrightarrow{E}$ modulo B. $\overrightarrow{E}$ is *sort decreasing* modulo B iff for each $t \rightarrow u$ **if** $\gamma \in \overrightarrow{E}$ and substitution θ, $\mathit{ls}(t\theta) \geq \mathit{ls}(u\theta)$ if $(\Sigma, B, \overrightarrow{E}) \vdash \gamma\theta$. $\overrightarrow{E}$ is *operationally terminating* modulo B iff there is no infinite well-formed proof tree in $(\Sigma, B, \overrightarrow{E})$. $\overrightarrow{E}$ is *confluent* modulo B iff all $t, t_1, t_2 \in T_\Sigma(X)$, if $t \rightarrow^*_{E/B} t_1$ and $t \rightarrow^*_{E/B} t_2$, then there is $u \in T_\Sigma(X)$ such that $t_1 \rightarrow^*_{E/B} u$ and $t_2 \rightarrow^*_{E/B} u$. The term $t{\downarrow_{E/B}} \in T_\Sigma(X)$ denotes the *E-canonical form* of t modulo B so that $t \rightarrow^*_{E/B} t{\downarrow_{E/B}}$ and $t \downarrow_{E/B}$ cannot be further reduced by $\rightarrow_{E/B}$. Under the above assumptions $t{\downarrow_{E/B}}$ is unique up to B-equality.

A *Σ-rule* is a triple $l \rightarrow r$ **if** ϕ, with $l, r \in T_\Sigma(X)_s$, for some sort $s \in S$, and $\phi = \bigwedge_{i \in I} t_i = u_i$ a finite conjunction of Σ-equations. A *rewrite theory* is a tuple $\mathcal{R} = (\Sigma, E, R)$ with (Σ, E) an order-sorted equational theory and R a finite set of Σ-rules. $\mathcal{R}$ induces a rewrite relation $\rightarrow_{\mathcal{R}}$ on $T_\Sigma(X)$ defined for every $t, u \in T_\Sigma(X)$ by $t \rightarrow_{\mathcal{R}} u$ iff there is a rule $(l \rightarrow r$ **if** $\phi) \in R$ and a substitution $\theta : X \longrightarrow T_\Sigma(X)$ satisfying $t =_E l\theta$, $u =_E r\theta$, and $E \vdash \phi\theta$. Relation $\rightarrow_{\mathcal{R}}$ is undecidable in general, unless conditions such as coherence [35] are given. A key point of this paper is to make such a relation decidable when E decomposes as $\mathcal{E}_0 \uplus B_1$, where $\mathcal{E}_0$ is a built-in theory for which formula satisfiability is decidable and B_1 has a matching algorithm. A *topmost rewrite theory* is a rewrite theory $\mathcal{R} = (\Sigma, E, R)$, such that for some top sort *State*, no operator in Σ has *State* as argument sort and each rule $l \rightarrow r$ **if** $\phi \in R$ satisfies $l, r \in T_\Sigma(X)_{\mathit{State}}$ and $l \notin X$.

3 Rewriting Modulo a Built-in Subtheory

The concept of rewriting modulo a built-in equational subtheory is presented. In particular, the notion of rewrite theory modulo a built-in subtheory and its ground rewrite relation are introduced. A canonical representation for rewrite theories modulo built-ins is proposed, and some basic results are proved.

Definition 1 (Signature with Built-ins)**.** *An order-sorted signature $\Sigma = (S, \leq, F)$ is a signature with* built-in subsignature $\Sigma_0 \subseteq \Sigma$ *iff $\Sigma_0 = (S_0, F_0)$ is many-sorted, S_0 is a set of minimal elements in $(S, \leq)$, and if $f : w \longrightarrow s \in F_1$, then $s \notin S_0$ and f has no other typing in F_0, where $F_1 = F \setminus F_0$.*

The notion of built-in subsignature in an order-sorted signature Σ is modeled by a many-sorted signature Σ_0 defining the built-in terms $T_{\Sigma_0}(X_0)$. The restriction imposed on the sorts and the function symbols in Σ w.r.t. Σ_0 provides a clear syntactic distinction between built-in terms (the only ones with built-in sorts) and all other terms.

If $\Sigma \supseteq \Sigma_0$ is a signature with built-ins, then an *abstraction of built-ins* for t is a context $\lambda x_1 \cdots x_n.t^\circ$ such that $t^\circ \in T_{\Sigma_1}(X)$ and $\{x_1, \ldots, x_n\} = \mathit{vars}(t^\circ) \cap X_0$, where $\Sigma_1 = (S, \leq, F_1)$ and $X_0 = \{X_s\}_{s \in S_0}$. Lemma 1 shows that such an abstraction can be chosen so as to provide a canonical decomposition of t with useful properties.

Lemma 1. *Let Σ be a signature with built-in subsignature $\Sigma_0 = (S_0, F_0)$. For each $t \in T_\Sigma(X)$, there exist an abstraction of built-ins $\lambda x_1 \cdots x_n.t^\circ$ for t and a substitution $\theta^\circ : X_0 \longrightarrow T_{\Sigma_0}(X_0)$ such that (i) $t = t^\circ\theta$, (ii) $\{x_1, \ldots, x_n\}$ are pairwise distinct and disjoint from vars(t), and (iii) $\theta(x) = x$ if $x \neq x_i$, for $1 \leq i \leq n$; moreover, (iv) t° can always be selected to be S_0-linear and with $\{x_1, \ldots, x_n\}$ disjoint from an arbitrarily chosen finite subset Y of X_0.*

Proof. By induction on the structure of t. □

In the rest of the paper, for any $t \in T_\Sigma(X)$ and $Y \subseteq X_0$ finite, the expression $\mathit{abstract}_{\Sigma_1}(t, Y)$ denotes the choice of a triple $\langle \lambda x_1 \cdots x_n.t^\circ \,; \theta^\circ \,; \phi^\circ \rangle$ such that the context $\lambda x_1 \cdots x_n.t^\circ$ and the substitution θ° satisfy the properties (i)–(iv) in Lemma 1, and $\phi^\circ = \bigwedge_{i=1}^{n} (x_i = \theta^\circ(x_i))$.

Under certain restrictions on axioms, matching a Σ-term t to a Σ-term u, can be decomposed modularly into Σ_1-matching of the corresponding λ-abstraction and Σ_0-matching of the built-in subterms. This is described in Lemma 2. The proof of this lemma uses the following corollary.

Corollary 1. *Let $\Sigma = (S, \leq, F)$ be a signature with built-in subsignature $\Sigma_0 = (S_0, F_0)$. Let B_0 be a set of Σ_0-axioms and B_1 a set of Σ_1-axioms. For B_0 and B_1 regular, linear, collapse free for any sort in S_0, and sort-preserving, and $t \in T_\Sigma(X_0)$:*

(a) if $t \in T_{\Sigma_0}(X_0)$ and $t =_{B_1}^{1} t'$, then $t = t'$;

(b) if $t \in T_{\Sigma_1}(X_0)$ and $t =_{B_0}^{1} t'$, then $t = t'$;

(c) if $t \in T_{\Sigma_1}(X_0)$ and $t =_{B_1}^{1} t'$, then vars$(t) =$ vars(t') and t is linear iff t' is so;

Proof.

(a) Axioms B_1 do not mention any function symbol in F_0. Therefore, the equation in B_0 can only apply to variables in X_0. But B_1 is collapse-free for any sort in S_0, so that no B_1 equation can be applied to t, forcing $t = t'$.

(b) Same argument as (a).

(c) Trivial consequence of B_1 being regular and linear.

□

Lemma 2. *Let $\Sigma = (S, \leq, F)$ be a signature with built-in subsignature $\Sigma_0 = (S_0, F_0)$. Let B_0 be a set of Σ_0-axioms and B_1 a set of Σ_1-axioms. For B_0 and B_1 regular, linear, collapse free for any sort in S_0, and sort-preserving, if $t \in T_{\Sigma_1}(X_0)$ is linear with $vars(t) = \{x_1, \ldots, x_n\}$, then for each $\theta : X_0 \longrightarrow T_{\Sigma_0}(X_0)$:*

(a) if $t\theta =^1_{B_0} t'$, then there exist $x \in \{x_1, \ldots, x_n\}$ and $w \in T_{\Sigma_0}(X_0)$ such that $\theta(x) =^1_{B_0} w$ and $t' = t\theta'$, with $\theta'(x) = w$ and $\theta'(y) = \theta(y)$ otherwise;

(b) if $t\theta =^1_{B_1} t'$, then there exists $v \in T_{\Sigma_1}(X_0)$ such that $t =^1_{B_1} v$ and $t' = v\theta$; and

(c) if $t\theta =_{B_0 \uplus B_1} t'$, then there exist $v \in T_{\Sigma_1}(X_0)$ and $\theta' : X_0 \longrightarrow T_{\Sigma_0}(X_0)$ such that $t' = v\theta'$, $t =_{B_1} v$, and $\theta =_{B_0} \theta'$ (i.e., $\theta(x) =_{B_0} \theta'(x)$ for each $x \in X_0$).

Proof. (a) It follows from Corollary 1 part (b) that B_0 can only be applied on some built-in subterm $\theta(x)$ of $t\theta$, for some $x \in dom(\theta)$. That is, there is $w \in T_{\Sigma_0}(X_0)$ such that $\theta(x) =^1_{B_0} w$ and, since t is linear, $t' = t\theta'$, where $\theta'(x) = w$ and $\theta'(x) = \theta(x)$ otherwise.

(b) It follows from Corollary 1 part (c) that equational deduction with B_1 can only permute the built-in variables in t and it does not equate built-in subterms such as the ones in $ran(\theta)$. Hence, by Corollary 1part (c), there exists a linear $v \in T_{\Sigma_1}(X_0)$ such that $t =^1_{B_1} v$ and $t' = v\theta$.

(c) Follows by induction on the proof's length in $B_0 \uplus B_1$.

□

Definition 2 introduces the notion of rewriting modulo a built-in subtheory.

Definition 2 (Rewriting Modulo a Built-in Subtheory)**.** *A* rewrite theory modulo the built-in subtheory $\mathcal{E}_0$ *is a topmost rewrite theory $\mathcal{R} = (\Sigma, E, R)$ with:*

(a) Σ=$(S, \leq, F)$ a signature with built-in subsignature Σ_0=(S_0, F_0) and top sort State$\in$S;

(b) $E = E_0 \uplus B_0 \uplus B_1$, where E_0 is a set of Σ_0-equations, B_0 (resp., B_1) are Σ_0-axioms (resp., Σ_1-axioms) satisfying the conditions in Lemma 2, $\mathcal{E}_0 = (\Sigma_0, E_0 \uplus B_0)$ and $\mathcal{E} = (\Sigma, E)$ are admissible, and the theory inclusion $\mathcal{E}_0 \subseteq \mathcal{E}$ is protecting;

(c) R is a set of rewrite rules *of the form $l(\overrightarrow{x_1}, \overrightarrow{y}) \rightarrow r(\overrightarrow{x_2}, \overrightarrow{y})$ **if** $\phi(\overrightarrow{x_3})$ such that $l, r \in T_\Sigma(X)_{State}$, l is $(S \setminus S_0)$-linear, $\overrightarrow{x_i}{:}\overrightarrow{s_i}$ with $\overrightarrow{s_i} \in S_0^*$, for $i \in \{1, 2, 3\}$, $\overrightarrow{y}{:}\overrightarrow{s}$ with $\overrightarrow{s} \in (S \setminus S_0)^*$, and $\phi \in QF_{\Sigma_0}(X_0)$, where $QF_{\Sigma_0}(X_0)$ denotes the set of quantifier-free Σ_0-formulas with variables in X_0.*

Note that, due to the presence of conditions ϕ in the rules of $\mathcal{R}$ that are general quantifier-free formulas, as opposed to a conjunction of atoms, properly speaking $\mathcal{R}$ is somewhat more general than a standard rewrite theory as defined in Section 2.

The binary rewrite relation induced by a rewrite theory $\mathcal{R}$ modulo $\mathcal{E}_0$ on $T_{\Sigma,State}$ is called the *ground rewrite relation* of $\mathcal{R}$.

Definition 3 (Ground Rewrite Relation)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo $\mathcal{E}_0$. The relation $\to_{\mathcal{R}}$ induced by $\mathcal{R}$ on $T_{\Sigma,State}$ is defined for $t, u \in T_{\Sigma,State}$ by $t \to_{\mathcal{R}} u$ iff there is a rule $l \to r$ **if** ϕ in R and a ground substitution $\sigma : X \longrightarrow T_\Sigma$ such that (a) $t =_E l\sigma$, $u =_E r\sigma$, and (b) $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$.*

The ground rewrite relation $\to_{\mathcal{R}}$ is the topmost rewrite relation induced by R modulo E on $T_{\Sigma,State}$. This relation is defined even when a rule in R has extra variables in its righthand side: the rule is then non-deterministic and such extra variables can be arbitrarily instantiated, provided that the corresponding instantiation of ϕ holds. Also, note that non built-in variables can occur in l, but $\phi\sigma$ is a *variable-free formula* in $QF_{\Sigma_0}(\emptyset)$, so that either $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$ or $\mathcal{T}_{\mathcal{E}_0} \not\models \phi\sigma$.

A rewrite theory $\mathcal{R}$ modulo $\mathcal{E}_0$ always has a canonical representation in which all left-hand sides of rules are linear Σ_1-terms.

Definition 4 (Normal Form of a Rewrite Theory Modulo $\mathcal{E}_0$)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo $\mathcal{E}_0$. Its* normal form *$\mathcal{R}^\circ = (\Sigma, E, R^\circ)$ has rules:*

$$R^\circ = \{l^\circ \to r \;\textbf{\textit{if}}\; \phi \wedge \phi^\circ \mid (\exists l \to r \;\textbf{\textit{if}}\; \phi \in R)\langle \lambda \overrightarrow{x}.l^\circ \,;\theta^\circ \,;\phi^\circ\rangle = \mathit{abstract}_\Sigma(l, \mathit{vars}(\{l, r, \phi\}))\}.$$

Lemma 3 (Invariance of Ground Rewriting under Normalization)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo $\mathcal{E}_0$. Then $\to_{\mathcal{R}} = \to_{\mathcal{R}^\circ}$.*

Proof. We show that $\to_{\mathcal{R}} \subseteq \to_{\mathcal{R}^\circ}$ and $\to_{\mathcal{R}^\circ} \subseteq \to_{\mathcal{R}}$.

($\subseteq$) Let $t, u \in T_{\Sigma,State}$. If $t \to_{\mathcal{R}} u$, then there is a rule ($l \to r$ **if** ϕ) $\in R$ and a ground substitution $\sigma : X \longrightarrow T_\Sigma$ such that $t =_E l\sigma$, $u =_E r\sigma$, and $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$. It suffices to prove $t \to_{\mathcal{R}^\circ} u$ with witnesses ($l^\circ \to r$ **if** $\phi \wedge \phi^\circ$) $\in R^\circ$ and $\rho = \theta^\circ\sigma$. Note that $t =_E l\sigma = l^\circ\theta^\circ\sigma = l^\circ\rho$. For $\mathcal{T}_{\mathcal{E}_0} \models (\phi \wedge \phi^\circ)\rho$ first note that $\mathcal{T}_{\mathcal{E}_0} \models \phi\rho$ since $\phi\rho = \phi\theta^\circ\sigma = \phi\sigma$ (because $\mathit{vars}(\phi) \cap \mathit{dom}(\theta^\circ) = \emptyset$) and $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$ by assumption. For $\mathcal{T}_{\mathcal{E}_0} \models \phi^\circ\rho$ notice that $\theta^\circ\theta^\circ = \theta^\circ$ because $\mathit{ran}(\theta^\circ) \cap \mathit{dom}(\theta^\circ) = \emptyset$, and then:

$$\begin{aligned}\phi^\circ\rho = \left(\bigwedge_{i=1}^{n} x_i = \theta^\circ(x_i)\right)\rho = \bigwedge_{i=1}^{n} x_i\rho = \theta^\circ(x_i)\rho = \bigwedge_{i=1}^{n} \theta^\circ(x_i)\sigma = \theta^\circ(x_i)\theta^\circ\sigma \\ = \bigwedge_{i=1}^{n} \theta^\circ(x_i)\sigma = \theta^\circ(x_i)\sigma = \top.\end{aligned}$$

Hence $t \to_{\mathcal{R}^\circ} u$ as desired.

($\supseteq$) Let $t, u \in T_{\Sigma,State}$. If $t \to^\circ_{\mathcal{R}} u$, then there is a rule ($l \to r$ **if** ϕ) $\in R$ and a ground substitution $\sigma : X \longrightarrow T_\Sigma$ such that $t =_E l^\circ\sigma$, $u =_E r\sigma$, and $\mathcal{T}_{\mathcal{E}_0} \models (\phi \wedge \phi^\circ)\sigma$. It suffices to prove $t \to_{\mathcal{R}} u$ with witness ($l \to r$ **if** ϕ) $\in R$.

Let $\langle \lambda x_1 \cdots x_n . l^\circ \,;\theta^\circ\,;\phi^\circ\rangle$ be the abstraction of built-ins for l. Substitution σ can be decomposed into substitutions $\theta : X_0 \longrightarrow T_{\Sigma_0}(X_0)$ and $\rho : X \longrightarrow T_\Sigma$, with $\theta(x) = \sigma(x)$ if $x \in \{x_1, \ldots, x_n\}$ and $\theta(x) = x$ otherwise, such that $\sigma = \theta\rho$. From $\mathcal{T}_{\mathcal{E}_0} \models (\phi \wedge \phi^\circ)\sigma$ it follows that $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$, i.e., $\mathcal{T}_{\mathcal{E}_0} \models \phi\rho$ because $\mathit{vars}(\phi) \cap \mathit{dom}(\theta) = \emptyset$. Also, it follows that $\mathcal{T}_{\mathcal{E}_0} \models \bigwedge_{i=1}^{n} \theta(x_i)\rho = \theta^\circ(x_i)\rho$ which implies that:

$$t =_E l^\circ\sigma = l^\circ\theta\rho =_{E_0 \uplus B_0} l^\circ\theta^\circ\rho = l\rho.$$

Hence $t \to_{\mathcal{R}} u$, as desired.

□

By the properties of the axioms in a rewrite theory modulo built-ins $\mathcal{R} = (\Sigma, E_0 \uplus B_0 \uplus B_1)$, B_1-matching a term $t \in T_\Sigma(X_0)$ to a left-hand side l° of a rule in R° provides a complete unifiability algorithm for ground B_1-unification of t and l°.

Lemma 4 (Matching Lemma)**.** *Let $\mathcal{R} = (\Sigma, E_0 \uplus B_0 \uplus B_1, R)$ be a rewrite theory modulo $\mathcal{E}_0$. For $t \in T_\Sigma(X_0)_{\mathit{State}}$ and l° a left-hand side of a rule in R° such that $\mathit{vars}(t) \cap \mathit{vars}(l^\circ)$, $t \ll_{B_1} l^\circ$ iff $GU_{B_1}(t = l^\circ) \neq \emptyset$ holds, where $GU_{B_1}(t = l^\circ) = \{\sigma : X \longrightarrow T_\Sigma \mid t\sigma =_{B_1} l^\circ\sigma\}$.*

Proof.

($\Longrightarrow$) If $t \ll_{B_1} l^\circ$, then $t =_{B_1} l^\circ\theta$ for some $\theta : X \longrightarrow T_\Sigma(X)$. Let $\rho : X \longrightarrow T_\Sigma$ be any ground substitution. Then $\theta\rho \in GU_{B_1}(t = l^\circ)$.

($\Longleftarrow$) Let $\sigma \in GU_{B_1}(t = l^\circ)$ with $l \to r$ **if** $\phi \in R$. Let $\mathit{vars}(l^\circ) \cap X_0 = \{x_1, \ldots, x_n\}$ and $X_1 = X \setminus X_0$. Note that there are substitutions

$$\begin{aligned} &\alpha : \mathit{vars}(l^\circ) \cap X_1 \longrightarrow T_{\Sigma_1}(X_0) \\ &\rho : X \setminus \mathit{dom}(\alpha) \longrightarrow T_\Sigma \end{aligned}$$

satisfying $\sigma = \alpha\rho$ and such that $(l^\circ\alpha) \in T_{\Sigma_1}(X_0)$ is linear and $\mathit{ran}(l^\circ\alpha) \cap (\mathit{vars}(t, l^\circ)) = \emptyset$. Let $\mathit{ran}(\alpha) = \{y_1, \ldots, y_m\}$. Therefore, by Lemma 2, there exists $u \in T_{\Sigma_1}(X_0)$ such that $u =_{B_1} l^\circ\alpha$, u is linear, and $\mathit{vars}(u) = \mathit{vars}(l^\circ\alpha) = x_1, \ldots, x_n, y_1, \ldots, y_m$, and $u\rho = t$. Moreover, t can be written as

$$u(t_1, \ldots, t_n, t_{n+1}, \ldots, t_{n+m})$$

with $t_i \in T_{\Sigma_0}(X_0)$. Define $\theta : X_0 \longrightarrow T_{\Sigma_0}(X_0)$ by $\theta(x) = t_i$ if $x \in \{x_1, \ldots, x_n\}$, $\theta(x) = t_{i+n}$ if $x \in \{y_1, \ldots, y_m\}$, and $\theta(x) = x$ otherwise. Then we have:

$$\begin{aligned} t &= u(t_1, \ldots, t_n, t_{n+1}, \ldots, t_{m+n}) \\ &= u(x_1, \ldots, x_n, y_1, \ldots, y_m)\theta \\ &=_{B_1} l^\circ\alpha\theta. \end{aligned}$$

Therefore, $t \ll_{B_1} l^\circ$, as desired.

□

4 Symbolic Rewriting Modulo a Built-in Subtheory

We explain how a rewrite theory $\mathcal{R}$ modulo $\mathcal{E}_0$ defines a symbolic rewrite relation on terms in $T_{\Sigma_0}(X_0)_{State}$ constrained by formulas in $QF_{\Sigma_0}(X_0)$. The idea is that, when $\mathcal{E}_0$ is a decidable theory, transitions on the symbolic terms can be performed by rewriting modulo B_1, and satisfiability of the formulas can be handled by an SMT decision procedure. This approach provides an efficiently executable symbolic method called *rewriting modulo SMT* that is sound and complete with respect to the ground rewrite relation of Definition 3 and yields a complete symbolic reachability analysis method.

Definition 5 (Constrained Terms and their Denotation)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo $\mathcal{E}_0$. A* constrained term *is a pair $\langle t\,;\varphi\rangle$ in $T_\Sigma(X_0)_{State} \times QF_{\Sigma_0}(X_0)$. Its* denotation *$[\![t]\!]_\varphi$ is defined as*

$$[\![t]\!]_\varphi = \{t'{\in}T_{\Sigma,State} \mid (\exists\sigma : X_0{\longrightarrow}T_{\Sigma_0})\ t'{=}t\sigma \wedge \mathcal{T}_{\mathcal{E}_0} \models \varphi\sigma\}.$$

The domain of σ in Definition 5 ranges over all built-in variables X_0 and consequently $[\![t]\!]_\varphi \subseteq T_{\Sigma,State}$ for any $t \in T_\Sigma(X_0)_{State}$, even if $vars(t) \not\subseteq vars(\varphi)$. Intuitively, $[\![t]\!]_\varphi$ denotes the set of all ground states that are instances of t and satisfy φ.

Before introducing the symbolic rewrite relation on constrained terms induced by a rewrite theory $\mathcal{R}$ modulo $\mathcal{E}_0$, auxiliary notation for variable renaming is required. In the rest of the paper, the expression *fresh-vars*(Y), for $Y \subseteq X$ finite, represents a variable renaming $\zeta : X \longrightarrow X$ satisfying $Y \cap ran(\zeta) = \emptyset$.

Definition 6 (Symbolic Rewrite Relation)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo built-ins $\mathcal{E}_0$. The* symbolic rewrite relation *$\rightsquigarrow_\mathcal{R}$ induced by $\mathcal{R}$ on $T_\Sigma(X_0)_{State} \times QF_{\Sigma_0}(X_0)$ is defined for $t, u \in T_\Sigma(X_0)_{State}$ and $\varphi, \varphi' \in QF_{\Sigma_0}(X_0)$ by $\langle t\,;\varphi\rangle \rightsquigarrow_\mathcal{R} \langle u\,;\varphi'\rangle$ iff there is a rule $l \to r$ **if** ϕ in R and a substitution $\theta : X \longrightarrow T_\Sigma(X)$ such that (a) $t =_E l\zeta\theta$ and $u = r\zeta\theta$, (b) $\mathcal{E}_0 \vdash (\varphi' \Leftrightarrow \varphi \wedge \phi\zeta\theta)$, and (c) φ' is $\mathcal{T}_{\mathcal{E}_0}$-satisfiable, where $\zeta =$ fresh-vars$(vars(t, \varphi))$.*

The symbolic relation $\rightsquigarrow_\mathcal{R}$ on constrained terms is defined as a topmost rewrite relation induced by R modulo E on $T_\Sigma(X_0)$ with extra bookkeeping of constraints. Note that φ' in $\langle t\,;\varphi\rangle \rightsquigarrow_\mathcal{R} \langle u\,;\varphi'\rangle$, when witnessed by $l \to r$ **if** ϕ and θ, is *semantically equivalent* to $\varphi \wedge \phi\zeta\theta$, in contrast to being *syntactically* equal. This extra freedom allows for simplification of constraints if desired. Also, such a constraint φ' is satisfiable in $\mathcal{T}_{\mathcal{E}_0}$, implying that φ and $\phi\theta$ are both satisfiable in $\mathcal{T}_{\mathcal{E}_0}$, and therefore $[\![t]\!]_\varphi \neq \emptyset \neq [\![u]\!]_{\varphi'}$. Note that, up to the choice of the semantically equivalent φ' for which a fixed strategy is assumed, the symbolic relation $\rightsquigarrow_\mathcal{R}$ is deterministic because the renaming of variables in the rules is fixed by *fresh-vars*. This is key when executing $\rightsquigarrow_\mathcal{R}$, as explained in Section 5.

The important question to ask is whether this symbolic relation soundly and completely simulates its ground counterpart. The rest of this section answers this question in the affirmative for *normalized* rewrite theories modulo built-ins. Thanks to Lemma 3, the conclusion is therefore that $\rightsquigarrow_{\mathcal{R}^\circ}$ soundly and completely simulates $\to_\mathcal{R}$ for any rewrite theory $\mathcal{R}$ modulo built-ins $\mathcal{E}_0$.

The soundness of $\rightsquigarrow_{\mathcal{R}^\circ}$ w.r.t. $\rightarrow_{\mathcal{R}^\circ}$ is stated and proved in Theorem 1. Intuitively, soundness means that a pair $\langle t\,;\varphi\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle u\,;\varphi'\rangle$ is a symbolic underapproximation of all pairs such that $t' \rightarrow_{\mathcal{R}^\circ} u'$ with $t' \in \llbracket t \rrbracket_\varphi$ and $u' \in \llbracket u \rrbracket_{\varphi'}$.

Theorem 1 (Soundness)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo built-ins $\mathcal{E}_0$, $t, u \in T_\Sigma(X_0)_{State}$, and $\varphi, \varphi' \in QF_{\Sigma_0}(X_0)$. If $\langle t\,;\varphi\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle u\,;\varphi'\rangle$, then $t\rho \rightarrow_{\mathcal{R}^\circ} u\rho$ for all $\rho : X_0 \longrightarrow T_{\Sigma_0}$ satisfying $\mathcal{T}_{\mathcal{E}_0} \models \varphi'\rho$.*

Proof. Let $\rho : X_0 \longrightarrow T_{\Sigma_0}$ satisfy $\mathcal{T}_{\mathcal{E}_0} \models \varphi'\rho$. The goal is to show that $t\rho \rightarrow_{\mathcal{R}^\circ} u\rho$. Let $l^\circ \rightarrow r$ **if** $\phi \in R^\circ$ and $\theta : X_0 \longrightarrow T_0(X_0)$ witness $\langle t\,;\varphi\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle u\,;\varphi'\rangle$. Then $t =_E l^\circ\zeta\theta$, $u =_E r\zeta\theta$, $\mathcal{E}_0 \vdash (\varphi' \Leftrightarrow \varphi \wedge \phi\zeta\theta)$, and φ' is $\mathcal{T}_{\mathcal{E}_0}$-satisfiable. Without loss of generality assume that $\theta|_{vars(t,\varphi)} = id|_{vars(t,\varphi)}$ and let $\sigma = \zeta\theta\rho$. Then note that $t\rho =_E (l^\circ\zeta\theta)\rho = l^\circ\zeta\theta\rho = l^\circ\sigma$ and $u\rho =_E (r\zeta\theta)\rho = r\zeta\theta\rho = r\sigma$. Moreover, $\mathcal{T}_{\mathcal{E}_0} \models (\varphi' \Leftrightarrow \varphi \wedge \phi\zeta\theta)$ and $\mathcal{T}_{\mathcal{E}_0} \models \varphi'\rho$ imply $\mathcal{T}_{\mathcal{E}_0} \models \phi\zeta\theta\rho$, i.e., $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$. Therefore, $t\rho \rightarrow_{\mathcal{R}^\circ} u\rho$, as desired. □

The completeness of $\rightsquigarrow_{\mathcal{R}^\circ}$ w.r.t. $\rightarrow_{\mathcal{R}^\circ}$ is stated and proved in Theorem 2, which is a "lifting lemma". Intuitively, completeness states that a symbolic relation yields an over-approximation of its ground rewriting counterpart.

Theorem 2 (Completeness)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo built-ins $\mathcal{E}_0$, $t \in T_\Sigma(X_0)_{State}$, $u' \in T_{\Sigma,State}$, and $\varphi \in QF_{\Sigma_0}(X_0)$. For any $\rho : X_0 \longrightarrow T_{\Sigma_0}$ such that $t\rho \in \llbracket t \rrbracket_\varphi$ and $t\rho \rightarrow_{\mathcal{R}^\circ} u'$, there exist $u \in T_\Sigma(X_0)_{State}$ and $\varphi' \in QF_{\Sigma_0}(X_0)$ such that $\langle t\,;\varphi\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle u\,;\varphi'\rangle$ and $u' \in \llbracket u \rrbracket_{\varphi'}$.*

Proof. By the assumptions there is a rule $(l^\circ \rightarrow r$ **if** $\phi) \in R^\circ$ and a ground substitution $\sigma : X \longrightarrow T_\Sigma$ satisfying $t\rho =_E l^\circ\sigma$, $u' =_E r\sigma$, and $\mathcal{T}_{\mathcal{E}_0} \models \phi\sigma$. Without loss of generality assume $vars(t, \varphi) \cap vars(l^\circ, r, \phi)) = \emptyset$; otherwise l, r, ϕ can be renamed by means of *fresh-vars*. Furthermore, $\sigma = \rho$ can be assumed. The goal is to show the existence of $u \in T_\Sigma(X)_{State}$ and $\varphi' \in QF_{\Sigma_0}(X_0)$ such that (i) $\langle t\,;\varphi\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle u\,;\varphi'\rangle$ and (ii) $u' \in \llbracket u \rrbracket_{\varphi'}$. Since l° is linear and built-in subterms are variables, by Lemma 2 there exists $\alpha : X \longrightarrow T_\Sigma$ satisfying $t\alpha =_{B_1} l^\circ\alpha$. Hence $GU_{B_1}(t = l^\circ) \neq \emptyset$ and, by Lemma 4, there exists $\theta' : X \longrightarrow T_\Sigma(X)$ satisfying $t =_{B_1} l^\circ\theta'$ and a fortiori $t =_{E_0 \uplus B_0 \uplus B_1} l^\circ\theta'$. Let $\theta : X \longrightarrow T_\Sigma(X)$ be defined by $\theta(x) = \theta'(x)$ if $x \in vars(l)$ and $\theta(x) = \rho(x)$ otherwise. Note that $\theta|_{vars(l)}\rho =_{E_0 \uplus B_0} \rho|_{vars(l)}$. Define $u = r\theta$ and $\varphi' = \varphi \wedge \phi\theta$, and then for (i) and (ii) above:

(i) It suffices to prove that $\mathcal{T}_{\mathcal{E}_0} \models \varphi'\rho$, i.e., $\mathcal{T}_{\mathcal{E}_0} \models (\varphi \wedge \phi\theta)\rho$. By assumption $\mathcal{T}_{\mathcal{E}_0} \models \varphi\rho$ and $\mathcal{T}_{\mathcal{E}_0} \models \phi\rho$. Notice that:

$$\phi\theta\rho = (\phi\theta|_{vars(l)})\rho =_{E_0 \uplus B_0} (\phi\rho)\rho = \phi\rho.$$

Hence $\mathcal{T}_{\mathcal{E}_0} \models \phi\theta\rho$.

(ii) By assumption $u' =_{E_0 \uplus B_0 \uplus B_1} r\rho$; also:

$$r\rho =_{E_0 \uplus B_0 \uplus B_1} r\theta|_{vars(l)}\rho = r\theta\rho = u\rho.$$

Hence $u' =_{E_0 \uplus B_0 \uplus B_1} u\rho \in \llbracket u \rrbracket_{\varphi'}$ by part (i).

□

Although the above soundness and completeness theorems, plus Lemma 3, show that $\to_{\mathcal{R}}$ is fully characterized symbolically by $\rightsquigarrow_{\mathcal{R}^\circ}$, for any rewrite theory $\mathcal{R}$ modulo $\mathcal{E}_0$, because of condition (6) in Definition 6, the relation $\rightsquigarrow_{\mathcal{R}^\circ}$ is in general undecidable. However, $\rightsquigarrow_{\mathcal{R}}^{\circ}$ becomes decidable for built-in theories $\mathcal{E}_0$ that can be extended to a *decidable theory* $\mathcal{E}_0^+$ (typically by adding some inductive consequences) such that:

$$(\forall \phi \in QF_{\Sigma_0}(X_0))\ \phi \text{ is } \mathcal{E}_0^+\text{-satisfiable} \iff (\exists \sigma : X_0 \longrightarrow T_{\Sigma_0})\ \mathcal{T}_{\mathcal{E}_0} \models \phi\sigma. \qquad (1)$$

Many decidable theories $\mathcal{E}_0^+$ of interest are supported by SMT solvers satisfying this requirement. For example, $\mathcal{E}_0$ can be the equational theory of natural number addition and $\mathcal{E}_0^+$ Pressburger arithmetic. That is, $\mathcal{T}_{\mathcal{E}_0}$ is the *standard model* of both $\mathcal{E}_0$ and $\mathcal{E}_0^+$, and $\mathcal{E}_0^+$-satisfiability coincides with satisfiability in such a standard model. Under such conditions, satisfiability of $\varphi \wedge \phi\zeta\theta$ (and therefore of φ') in a step $\langle t\,;\varphi\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle u\,;\varphi'\rangle$ becomes decidable by invoking an SMT-solver for $\mathcal{E}_0$, so that $\rightsquigarrow_{\mathcal{R}^\circ}$ can be naturally described as *symbolic rewriting modulo SMT* (and modulo B_1).

The symbolic reachability problems considered for a rewrite theory $\mathcal{R}$ modulo $\mathcal{E}_0$ in this paper, are existential formulas of the form $(\exists \overrightarrow{z})\ t \to^* u \wedge \varphi$, with $\overrightarrow{z}$ the variables appearing in t, u, and φ, $t \in T_\Sigma(X_0)_{\mathit{State}}$, $u \in T_\Sigma(X)_{\mathit{State}}$, and $\varphi \in QF_{\Sigma_0}(X_0)$. By abstracting the Σ_0-subterms of u, the ground solutions of such a reachability problem are those witnessing the model-theoretic satisfaction relation:

$$\mathcal{T}_{\mathcal{R}} \models (\exists \overrightarrow{x} \uplus \overrightarrow{y})\ t(\overrightarrow{x}) \to^* u^\circ(\overrightarrow{y}) \wedge \varphi_1(\overrightarrow{x}) \wedge \varphi_2(\overrightarrow{x}, \overrightarrow{y}) \qquad (2)$$

where $\mathcal{T}_{\mathcal{R}} = (\mathcal{T}_{\Sigma/E}, \to_{\mathcal{R}}^*)$ is the initial reachability model of $\mathcal{R}$ [11], $t \in T_\Sigma(X)$ and $u^\circ \in T_{\Sigma_1}(X)$ are S_0-linear, $\mathit{vars}(t) \subseteq \overrightarrow{x} \subseteq X_0$, and $\overrightarrow{y} \subseteq X$. Thanks to the soundness and completeness results, theorems 1 and 2, the solvability of Condition (2) for $\to_{\mathcal{R}}$ can be achieved by reachability analysis with $\rightsquigarrow_{\mathcal{R}^\circ}$. This is stated and proved in Theorem 3.

Theorem 3 (Symbolic Reachability Analysis)**.** *Let $\mathcal{R} = (\Sigma, E, R)$ be a rewrite theory modulo built-ins $\mathcal{E}_0$. The reachability problem in Condition (2) has a solution iff there exist a term $v \in T_\Sigma(X)_{\mathit{State}}$, a constraint $\varphi' \in QF_{\Sigma_0}(X_0)$, and a substitution $\theta : X \longrightarrow T_\Sigma(X)$, with $dom(\theta) \subseteq \overrightarrow{y}$, such that (a) $\langle t\,;\varphi_1\rangle \rightsquigarrow_{\mathcal{R}^\circ}^* \langle v\,;\varphi'\rangle$, (b) $v =_{B_1} u^\circ\theta$, and (c) $\varphi' \wedge \varphi_2\theta$ is $\mathcal{T}_{\mathcal{E}_0}$-satisfiable.*

Proof. By theorems 1 and 2, and induction on the length of the rewrite derivation. □

In Theorem 3, since $dom(\theta) \subseteq \overrightarrow{y}$, and $\overrightarrow{x}$ and $\overrightarrow{y}$ are disjoint, the variables of $\overrightarrow{x}$ in $\varphi_2\theta$ are left unchanged. Therefore, $\varphi_2\theta$ *links* the requirements for the variables $\overrightarrow{x}$ in the initial state and $\overrightarrow{y}$ in the final state according to both φ_1 and φ_2. Also note that the inclusion of formula φ_1 as a conjunct in the formula in Condition (3) of Theorem 3 is superfluous because $\langle t\,;\varphi_1\rangle \rightsquigarrow_{\mathcal{R}^\circ} \langle v\,;\varphi'\rangle$ implies that φ_1 is a semantic consequence of φ'.

5 Reflective Implementation of $\rightsquigarrow_{\mathcal{R}^\circ}$

The design and implementation of prototype that offers support for rewriting modulo SMT in the Maude system are discussed. The prototype relies on Maude's metalevel features, that implement rewriting logic's reflective capabilities, and on SMT solving for $\mathcal{E}_0^+$ integrated in Maude as CVC3's decision procedures. The extension of Maude with CVC3 is available from the Matching Logic Project [33]. In the rest of this section, $\mathcal{R} = (\Sigma, E_0 \uplus B_0 \uplus B_1, R)$ is a rewrite theory modulo built-ins $\mathcal{E}_0$, where $\mathcal{E}_0$ satisfies Condition (1) in Section 4. The theory mapping $\mathcal{R} \mapsto u(\mathcal{R})$ removes the constraints from the rules in R and interprets the built-in variables X_0 as *constants*.

In Maude, reflection is efficiently supported by its *META-LEVEL* module [13], which provides key functionality for rewriting logic's *universal theory* $\mathcal{U}$ [14]. Rewrite theories $\mathcal{R}$ are meta-represented in $\mathcal{U}$ as terms $\overline{\mathcal{R}}$ of sort *Module*, and a term t in $\mathcal{R}$ is meta-represented in $\mathcal{U}$ as a term $\overline{t}$ of sort *Term*. The key idea of the reflective implementation is to reduce symbolic rewriting with $\rightsquigarrow_{\mathcal{R}^\circ}$ to *standard rewriting* in an associated reflective rewrite theory extending the universal theory $\mathcal{U}$. This is specially important for formal analysis purposes, because it makes available to $\rightsquigarrow_{\mathcal{R}^\circ}$ some formal analysis features provided by Maude for rewrite theories such as reachability analysis by search. This is illustrated by the case study in Section 6.

The prototype defines a parametrized functional module $SAT(\Sigma_0, E_0 \uplus B_0)$ of quantifier-free formulas with Σ_0-equations as atoms. This module extends $(\Sigma_0, E_0 \uplus B_0)$ with new sorts *Atom* and *QFFormula*, and new *constants* $var(X_0)$ identifying the variables X_0. It has, among other functions, a function $sat : QFFormula \longrightarrow Bool$ such that for ϕ, $sat(\phi) = \top$ if ϕ is $\mathcal{E}_0^+$-satisfiable, and $sat(\phi) = \bot$ otherwise.

The process of computing the one-step rewrites of a given constrained term $\langle t \,; \varphi \rangle$ under $\rightsquigarrow_{\mathcal{R}^\circ}$ is decomposed into two conceptual steps using Maude's metalevel. First, all possible triples $\langle u \,; \theta \,; \phi \rangle$ such that $t \rightarrow_{u(\mathcal{R}^\circ)} u$ is witnessed by a matching substitution θ and a rule with constraint ϕ are computed[1]. Second, these triples are filtered out by keeping only those for which the quantifier-free formula $\varphi \wedge \phi\theta$ is $\mathcal{E}_0^+$-satisfiable.

The first step in the process is mechanized by function *next*, available from the parametrized module $NEXT(\overline{\mathcal{R}}, \overline{State}, \overline{QFFormula})$ where $\overline{\mathcal{R}}$, $\overline{State}$, and $\overline{QFFormula}$ are the metalevel representations, respectively, of the rewrite theory module $\mathcal{R}$, the state sort *State*, and the quantifier-free formula sort *QFFormula*. Function *next* uses Maude's *meta-match* function and the auxiliary function *new-vars* for computing fresh variables (see Section 4). The call

$$next(\overline{((S, \leq, F \uplus var(X_0)), E_0 \uplus B_0 \uplus B_1, R^\circ)}, \overline{t}, \overline{\varphi})$$

computes all possible triples $\langle \overline{u} \,; \overline{\theta'} \,; \overline{\phi'} \rangle$ such that $t \rightsquigarrow_{\mathcal{R}^\circ} u$ is witnessed by a substitution θ' and a rule with constraint ϕ'. More precisely, such a call first computes a renaming $\zeta = \textit{fresh-vars}(vars(t, \varphi))$ and then, for each rule$(l^\circ \rightarrow r$ **if** $\phi)$, it uses the function *meta-match* to obtain a substitution

$$\overline{\theta} \in \textit{meta-match}(\overline{((S, \leq, F \uplus var(X_0)), B_0 \uplus B_1)}, \overline{t{\downarrow}_{E_0/B_0 \uplus B_1}}, \overline{l^\circ \zeta}),$$

[1] Note that in $u(\mathcal{R}^\circ)$ variables in X_0 are interpreted as *constants*. Therefore, the number of matching substitutions θ thus obtained is finite.

and returns $\langle \overline{u}\,;\overline{\theta'}\,;\overline{\phi'}\rangle$ with $\overline{u} = \overline{r\zeta\theta}$, $\overline{\theta'} = \overline{\zeta\theta}$, and $\overline{\phi'} = \overline{\phi\zeta\theta}$. Note that by having a *deterministic* choice of fresh variables (including those in the constraint), function *next* is actually a *deterministic* function.

Using the above-mentioned infrastructure, the parametrized module *NEXT* implements the symbolic rewrite relation $\rightsquigarrow_{\mathcal{R}^\circ}$ as a *standard rewrite relation* in the theory *NEXT*, extending *META-LEVEL*, by means of the following conditional rewrite rule:

$$\begin{aligned} &\textbf{ceq} \quad \langle X{:}\mathit{State}\,;\varphi{:}\mathit{QFFormula}\rangle \to \langle Y{:}\mathit{State}\,;\varphi'{:}\mathit{QFFormula}\rangle \\ &\quad\textbf{if} \quad \langle \overline{Y}\,;\overline{\theta}\,;\overline{\phi}\rangle\ S := \mathit{next}(\overline{\mathcal{R}^\bullet}, \overline{X}, \overline{\varphi}) \wedge \mathit{sat}(\varphi \wedge \phi) = \top \wedge \varphi' := \varphi \wedge \phi \end{aligned}$$

where $\mathcal{R}^\bullet = ((S, \leq, F \uplus \mathit{var}(X_0)), B, R^\circ)$. Therefore, a call to an external SMT solver is just an invocation of the function *sat* in $\mathit{SAT}(\Sigma_0, E_0 \uplus B_0)$ in order to achieve the above functionality more efficiently and in a built-in way.

Given that the symbolic rewrite relation $\rightsquigarrow_{\mathcal{R}^\circ}$ is encoded as a standard rewrite relation, symbolic search can be *directly implemented* in Maude by its *search* command. In particular, for terms t, u°, constraints φ_1, φ_2, F a variable of sort *QFFormula*, the following invocation solves the inductive reachability problem in Condition (2):

$$\mathit{search}\ \ \langle t\,;\varphi_1\rangle \to^* \langle u^\circ\,;F\rangle\ \ \mathit{such\ that}\ \ \mathit{sat}(F \wedge \varphi_2).$$

6 Analysis of the CASH algorithm

This section presents an example, developed jointly with Kyungmin Bae, of a real-time system beyond the scope of timed automata [2] that can be symbolically analyzed in the prototype tool integrating Maude and CVC3 described in Section 5. The analysis uses such a prototype to perform model checking based on *rewriting modulo SMT*. Some details are omitted; full details and the prototype tool can be found in [8].

The example involves the symbolic analysis of the CASH scheduling algorithm, developed by Caccamo, Buttazzo, and Sha [12], which attempts to maximize system performance while guaranteeing that critical tasks are executed in a timely manner. This is achieved by maintaining a queue of unused execution budgets that can be reused by other jobs to maximize processor utilization. CASH poses nontrivial modeling and analysis challenges because it contains an unbounded queue. Unbounded data types cannot be modeled in timed-automata formalisms, such as those of UPPAAL [20] or Kronos [37], which assume a finite discrete state.

The CASH algorithm was specified and analyzed in Real-Time Maude by *explicit-state model checking* in an earlier paper by Ölveczky and Caccamo [29], which showed that, under certain variations on both the assumptions and the design of the protocol, it could miss deadlines. But explicit-state model checking has intrinsic limitations which the new analysis by rewriting modulo SMT presented below overcomes. The CASH algorithm is parametrized by: (i) the number N of servers in the system, and (ii) the values of a maximum budget b_i and period p_i, for each server $1 \leq i \leq N$. Even if N is fixed, *there are infinitely many initial states* for N servers, since the maximum budgets b_i and periods p_i range over the natural numbers. Therefore, explicit state model checking cannot perform a full analysis. If a counterexample for N servers exists, it may be found by explicit-state model checking for some chosen initial states, as done in [30], but it could be missed if the wrong initial states are chosen.

Rewriting modulo SMT is useful for symbolically analyzing infinite-state systems like CASH. Infinite sets of states are symbolically described by terms which may involve user-definable data structures such as queues, but whose only variables range over decidable types for which an SMT solving procedure is available. For the CASH algorithm, the built-in data types used are the Booleans (sort `iBool`) and the integers (sort `iInt`). Integer built-in terms are used to model discrete time. Boolean built-in terms are used to impose constraints on integers.

A symbolic state is a pair {`iB,Cnf`} of sort `Sys` consisting of a Boolean constraint `iB`, with *and* denoted `^`, and a multiset configuration of objects `Cnf`, with mutiset union denoted by juxtaposition, where each object is a record like-structure with an object identifier, a class name, and a set of attribute-value pairs. In each object configuration there is a global object (of class `global`) that models the time of the system (with attribute name `time`), the priority queue (with attribute name `cq`), the availability (with attribute name `available`), and a deadline missed flag (with attribute name `deadline-miss`). A configuration can also contain any number of server objects (of class `server`). Each server object models the maximum budget (the maximum time within which a given job will be finished, with attribute name `maxBudget`), period (with attribute name `period`), internal state (with attribute name `state`), time executed (with attribute name `timeExecuted`), budget time used (with attribute name `usedOfBudget`), and time to deadline (with attribute name `timeToDeadline`). The symbolic transitions of CASH are specified by 14 conditional rewrite rules whose conditions specify constraints solvable by the SMT decision procedure. For example, rule `[deadlineMiss]` below models the detection of a deadline miss for a server with nonzero maximum budget.

```
vars AtSG AtS : AttributeSet .
var iB : iBool .
var Cnf : Configuration .
vars iT iT' iNZT : iInt .
var St : ServerState .
vars G S : Oid .
var B : Bool .

crl [deadlineMiss] :
    { iB, < G : global | dead-miss |-> B, AtSG >
          < S : server | state |-> St, usedOfBudget |-> iT,
                timeToDeadline |-> iT',
                maxBudget |-> iNZT, AtS > Cnf }
=> {iB ^ iT >= c(0) ^ iNZT > c(0) ^ iT' >  c(0) ^ iNZT > iT + iT',
          < G : global | dead-miss |-> true, AtSG >
          < S : server | state |-> St, usedOfBudget |-> iT,
                         timeToDeadline |-> iT',
                         maxBudget |-> iNZT, AtS > Cnf }
if St =/= idle /\ check-sat(iB ^ iT >= c(0) ^
                            iNZT > c(0) ^ iT' >  c(0) ^
                            iNZT > iT + iT') .
```

That is, the protocol misses a deadline for server `S` whenever the value of attribute `maxBudget` exceeds the addition of values for `usedOfBudget` and `timeToDeadline` (i.e., `iNZT > iT + iT'`) , so that the allocated execution time cannot be exhausted before the server's deadline.

The goal is to verify *symbolically* the existence of missed deadlines of the CASH algorithm for the *infinite set of initial configurations* containing two server objects s_0 and s_1 with maximum budgets b_0 and b_1 and periods p_0 and p_1 as unspecified natural numbers, and such that each server's maximum budget is strictly smaller than its period (i.e., $0 \leq b_0 < p_0 \wedge 0 \leq b_1 < p_1$). This infinite set of initial states is specified symbolically by the equational definition (not shown) of term `symbinit`. Maude's `search` command can then be used to symbolically check if there is a reachable state for any ground instance of `symbinit` that misses the deadline:

```
search in SYMBOLIC-FAILURE : symbinit =>*
  { iB:iBool,  Cnf:Configuration < g : global |
    AtS:AttributeSet, deadline-miss |-> true > } .
Solution 1 (state 233)
states: 234  rewrites: 60517 in 2865ms cpu
  (2865ms real) (21118 rewrites/second)
iB:iBool --> ((i(0) <= c(0) ^
               i(1) <= c(0)) v i(0) <= c(0) + i(1) ^
               ...
Cnf:Configuration -->
< s1 : server | maxBudget |-> i(0), period |-> i(1),
                state |-> waiting, usedOfBudget |-> c(0),
                timeToDeadline |-> ((i(1) -- c(1)) -- c(1)),
                timeExecuted |-> c(0) >
< s2 : server | maxBudget |-> i(2), period |-> i(3),
                state |-> executing, usedOfBudget |-> c(2),
                timeToDeadline |-> ((i(3) -- c(1)) -- c(1)),
                timeExecuted |-> c(2) >
AtS:AttributeSet --> time |-> c(2), cq |-> emptyQueue,
                     available |-> false
```

A counterexample is found at (modeling) time two, after exploring 233 symbolic states in less than 3 seconds. By using a satisfiability witness of the constraint `iB` computed by the search command, a concrete counterexample is found by exploring only 54 ground states. This result compares favorably, in both time and computational resources, with the ground counterexample found by explicit-state model checking in [29], where more that 52,000 concrete states where explored before finding a counterexample.

7 Related Work and Concluding Remarks

The idea of combining term rewriting/narrowing techniques and constrained data structures is an active area of research, specially since the advent of modern theorem provers with highly efficient decision procedures in the form of SMT solvers. The overall aim of these techniques is to advance applicability of methods in symbolic verification where the constraints are expressed in some logic that has an efficient decision procedure (see [28] for an overview). In particular, the work presented here has strong similarities with the narrowing-based symbolic analysis of rewrite theories initiated in [24] and extended in [7]. The main difference is the replacement of narrowing by SMT solving and the decidability advantages of SMT for constraint solving.

M. Ayala-Rincón [4] investigates, in the setting of many-sorted equational logic, the expressiveness of conditional equational systems whose conditions may use built-in predicates. This class of equational theories is important because the combination of equational and built-in premises yield a type of clauses which is more expressive than purely conditional equations. Rewriting notions like confluence, termination, and critical pairs are also investigated. S. Falke and D. Kapur [15] studied the problem of termination of rewriting with constrained built-ins. In particular, they extended the dependency pairs framework to handle termination of equational specifications with semantic data structures and evaluation strategies in the Maude functional sublanguage. The same authors used the idea of combining rewriting induction and linear arithmetic over constrained terms [16]. Their aim is to obtain equational decision procedures that can handle semantic data types represented by the constrained built-ins. H. Kirchner and C. Ringeissen proposed the notion of constrained rewriting and have used it by combining symbolic constraint solvers [19]. The main difference between their work and rewriting modulo SMT presented in this paper, is that the former uses narrowing for symbolic execution, both at the symbolic 'pattern matching' and the constraint solving levels. In contrast, rewriting modulo SMT solves the symbolic pattern matching task by rewriting while constraint solving is delegated to an SMT decision procedure. More generally, a difference common to [4, 15, 16, 19] is that all of those papers address symbolic reasoning for *equational* theorem proving purposes, but none of them addresses the kind of non-deterministic rewrite rules, which are needed for open system modeling.

This paper has presented rewrite theories modulo built-ins and has shown how they can be used for *symbolically* modeling and analyzing concurrent open systems, where non-deterministic values from the environment can be represented by built-in terms. Under reasonable assumptions, including decidability of $\mathcal{E}_0^+$, a rewrite theory modulo is executable by term rewriting modulo SMT. This feature makes it possible to use, for symbolic analysis, state-of-the-art tools already available for Maude, such as its space search commands, with no change whatsoever required to use such tools. We have proved that the symbolic rewrite relation is sound and complete with respect to its ground counterpart, have presented an overview of the prototype that offers support for rewriting modulo SMT in Maude, and have presented a case study on the symbolic analysis of the CASH scheduling algorithm illustrating the use of these techniques.

Future work on a mature implementation and on extending the idea of rewriting modulo SMT with other symbolic constraint solving techniques such as narrowing modulo should be pursued. Also, the extension to symbolic LTL model checking, together with state space reduction techniques, should be investigated. The ideas presented here extend results in [32] and have been successfully applied to the symbolic analysis of NASA's PLEXIL language to program open cyber-physical systems [32]. Future applications to PLEXIL and other languages should also be pursued.

References

1. Ernst Althaus, Evgeny Kruglov, and Christoph Weidenbach. Superposition modulo linear arithmetic SUP(LA). In Silvio Ghilardi and Roberto Sebastiani, editors, *FroCos*, volume 5749 of *Lecture Notes in Computer Science*, pages 84–99. Springer, 2009.

2. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

3. A. Armando, J. Mantovani, and L. Platania. Bounded model checking of software using SMT solvers instead of SAT solvers. *Model Checking Software*, pages 146–162, 2006.

4. Mauricio Ayala-Rincón. *Expressiveness of Conditional Equational Systems with Built-in Predicates*. PhD thesis, Universität Kaiserslauten, 1993.

5. F. Baader and K. Schulz. Unification in the union of disjoint equational theories: combining decision procedures. *Journal of Symbolic Computation*, 21:211–243, 1996.

6. Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

7. Kyungmin Bae, Santiago Escobar, and José Meseguer. Abstract logical model checking of infinite-state systems using narrowing. In Femke van Raamsdonk, editor, *RTA*, volume 21 of *LIPIcs*, pages 81–96. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

8. Kyungmin Bae and Camilo Rocha. A note on symbolic reachability analysis modulo integer constraints for the CASH algorithm. Available at: `http://maude.cs.uiuc.edu/cases/scash`, 2012.

9. Maria Paola Bonacina, Christopher Lynch, and Leonardo Mendonça de Moura. On deciding satisfiability by theorem proving with speculative inferences. *Journal of Automated Reasoning*, 47(2):161–189, 2011.

10. Alexandre Boudet. Combining unification algorithms. *J. Symb. Comp.*, 16(6):597–626, 1993.

11. Roberto Bruni and José Meseguer. Semantic foundations for generalized rewrite theories. *Theoretical Computer Science*, 360(1-3):386–414, 2006.

12. Marco Caccamo, Giorgio C. Buttazzo, and Lui Sha. Capacity sharing for overrun control. In *IEEE Real-Time Systems Symposium*, pages 295–304. IEEE Computer Society, 2000.

13. Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn L. Talcott, editors. *All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350 of *LNCS*. Springer, 2007.

14. Manuel Clavel, José Meseguer, and Miguel Palomino. Reflection in membership equational logic, many-sorted equational logic, horn logic with equality, and rewriting logic. *Theoretical Computer Science*, 373(1-2):70–91, 2007.

15. Stephan Falke and Deepak Kapur. Operational termination of conditional rewriting with built-in numbers and semantic data structures. *ENTCS*, 237:75–90, 2009.

16. Stephan Falke and Deepak Kapur. Rewriting induction + linear arithmetic = decision procedure. In Bernhard Gramlich, Dale Miller, and Uli Sattler, editors, *IJCAR*, volume 7364 of *Lecture Notes in Computer Science*, pages 241–255. Springer, 2012.

17. M.K. Ganai and A. Gupta. Accelerating high-level bounded model checking. In *ICCAD*, pages 794–801. ACM, 2006.

18. Joseph A. Goguen and José Meseguer. Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105(2):217–273, 1992.

19. Hélène Kirchner and Christophe Ringeissen. Combining symbolic constraint solvers on algebraic domains. *Journal of Symbolic Computation*, 18(2):113–155, 1994.

20. Kim Guldstrand Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *STTT*, 1(1-2):134–152, 1997.

21. José Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.

22. José Meseguer. Membership algebra as a logical framework for equational specification. In Francesco Parisi-Presicce, editor, *WADT*, volume 1376 of *Lecture Notes in Computer Science*, pages 18–61. Springer, 1997.

23. José Meseguer. Twenty years of rewriting logic. *JLAP*, 81(7-8):721–781, 2012.

24. José Meseguer and Prasanna Thati. Symbolic reachability analysis using narrowing and its application to verification of cryptographic protocols. *Higher-Order and Symbolic Computation*, 20(1-2):123–160, 2007.

25. A. Milicevic and H. Kugler. Model checking using SMT and theory of lists. *NASA Formal Methods*, pages 282–297, 2011.

26. Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979.

27. Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL(t). *Journal of the ACM*, 53(6):937–977, 2006.

28. Institute of Computer Science University of Innsbruck. Constrained rewriting and SMT: Emerging trends in rewriting. Available at `http://cl-informatik.uibk.ac.at/research/projects/constrained-rewriting-and-smt-emerging-trends-in`.

29. P. C. Ölveczky and M. Caccamo. Formal simulation and analysis of the CASH scheduling algorithm in Real-Time Maude. In L. Baresi and R. Heckel, editors, *Fundamental Approaches to Software Engineering (FASE'06)*, volume 3922 of *Lecture Notes in Computer Science*, pages 357–372. Springer, 2006.

30. P. C. Ölveczky and J. Meseguer. Semantics and pragmatics of Real-Time Maude. *Higher-Order and Symbolic Computation*, 20(1-2):161–196, 2007.

31. S. Owre, J. M. Rushby, and N. Shankar. PVS: A prototype verification system. In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE)*, volume 607 of *LNAI*, pages 748–752, Saratoga, NY, June 1992. Springer-Verlag.

32. Camilo Rocha. *Symbolic Reachability Analysis for Rewrite Theories*. PhD thesis, University of Illinois at Urbana-Champaign, 2012.

33. Grigore Roşu and Andrei Ştefănescu. Matching Logic: A New Program Verification Approach (NIER Track). In *ICSE'11: Proceedings of the 30th International Conference on Software Engineering*, pages 868–871. ACM, 2011.

34. M. Veanes, N. Bjørner, and A. Raschke. An SMT approach to bounded reachability analysis of model programs. In *FORTE*, pages 53–68. Springer, 2008.

35. P. Viry. Equational rules for rewriting logic. *TCS*, 285:487–517, 2002.

36. D. Walter, S. Little, and C. Myers. Bounded model checking of analog and mixed-signal circuits using an SMT solver. *Automated Technology for Verification and Analysis*, pages 66–81, 2007.

37. Sergio Yovine. Kronos: A verification tool for real-time systems. *STTT*, 1(1-2):123–133, 1997.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 01- 08 - 2013	**2. REPORT TYPE** Technical Memorandum	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE Rewriting Modulo SMT	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Rocha, Camilo; Mesequer, Jose; Munoz, Cesar A.	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER 534723.02.02.07.40
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) NASA Langley Research Center Hampton, VA 23681-2199	**8. PERFORMING ORGANIZATION REPORT NUMBER** L-20315
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001	**10. SPONSOR/MONITOR'S ACRONYM(S)** NASA
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) NASA/TM-2013-218033

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 61
Availability: NASA CASI (443) 757-5802

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Combining symbolic techniques such as: (i) SMT solving, (ii) rewriting modulo theories, and (iii) model checking can enable the analysis of infinite-state systems outside the scope of each such technique. This paper proposes rewriting modulo SMT as a new technique combining the powers of (i)-(iii) and ideally suited to model and analyze infinite-state open systems; that is, systems that interact with a non-deterministic environment. Such systems exhibit both internal non-determinism due to the system, and external non-determinism due to the environment. They are not amenable to finite-state model checking analysis because they typically are infinite-state. By being reducible to standard rewriting using reflective techniques, rewriting modulo SMT can both naturally model and analyze open systems without requiring any changes to rewriting-based reachability analysis techniques for closed systems. This is illustrated by the analysis of a real-time system beyond the scope of timed automata methods.

15. SUBJECT TERMS

Formal Verification; Open Systems; Rewriting Logic; Rewriting Modulo SMT; Satisfiability Modulo Theories; Symbolic Execution

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			STI Help Desk (email: help@sti.nasa.gov)
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	25	(443) 757-5802