Abstract. Program semantics can often be expressed as a (many-sorted) first-order theory $S$, and program properties as sentences $\varphi$ which are intended to hold in the canonical model of such a theory, which is often incomputable. Recently, we have shown that properties $\varphi$ expressed as the existential closure of a boolean combination of atoms can be disproved by just finding a model of $S$ and the negation $\neg \varphi$ of $\varphi$. Furthermore, this idea works quite well in practice due to the existence of powerful tools for the automatic generation of models for (many-sorted) first-order theories. In this paper we extend our previous results to arbitrary properties, expressed as sentences without any special restriction. Consequently, one can prove a program property $\varphi$ by just finding a model of an appropriate theory (including $S$ and possibly something else) and an appropriate first-order formula related to $\varphi$. Beyond its possible theoretical interest, we show that our results can also be of practical use in several respects.

Keywords: First-Order Logic, Logical models, Program analysis.

1 Introduction

Given a first-order theory $S$ and a sentence $\varphi$, finding a model $A$ of $S \cup \{\neg \varphi\}$, i.e., such that $A \models S \cup \{\neg \varphi\}$ holds, shows indeed that $\varphi$ is not a logical consequence of $S$: there is at least one model of $S$ (e.g., $A$) which does not satisfy $\varphi$ (as it satisfies $\neg \varphi$). Provability of $\varphi$ in $S$, i.e., $S \vdash \varphi$, implies (by correctness of the proof calculus) that $\varphi$ is a logical consequence of $S$ (written $S \models \varphi$). Thus, $A \models S \cup \{\neg \varphi\}$ disproves $\varphi$ regarding $S$; this can be written $\neg (S \vdash \varphi)$ by using some metalevel notation. In general, this does not allow us to conclude that $\neg \varphi$ is proved, i.e., $S \vdash \neg \varphi$, or is a logical consequence of $S$, i.e., $S \models \neg \varphi$. What can be concluded about $\neg \varphi$ regarding $S$ from the fact that $A \models S \cup \{\neg \varphi\}$ holds? Can this be advantageously used in a ‘logic-based’ approach to program analysis?

In [14], some answers to these questions are given: a sentence $\varphi$ which is an Existentially Closed Boolean Combination of Atoms (ECBCA for short) does not hold in the initial model $I_S$ of a theory $S$ consisting of a set of ground atoms if we find a model $A$ of $S \cup \{\neg \varphi\}$ [14] Corollary 2. This is useful in program
Our approach in [14] relies on the notion of preservation of a formula under homomorphisms \( h \) between interpretations. Roughly speaking, a homomorphism...
Table 1. Some properties about rewriting-based systems

| Property                                | $\varphi$                                                                 |
|-----------------------------------------|--------------------------------------------------------------------------|
| Ground reducible                        | $(\forall x)(\exists y) f(x) \rightarrow y$                             |
| Completely defined symbol $f$           | $(\forall x)(\exists y)\bigcup_{t \in T} f(x_1, \ldots, x_k) \rightarrow y$ |
| Completely defined TRS                  | $(\forall x)(\exists y) \bigwedge_{t \in T} f(x_1, \ldots, x_{ar(f)}) \rightarrow yf$ |
| Productive                              | $(\forall x)(\exists y) \bigvee_{c \in C} x \rightarrow^* c(y_1, \ldots, y_k)$ |
| Nonterminating                          | $(\exists x)(\forall n \in N)(\exists y)\bigcup_{t \in T} x \rightarrow^n y$ |
| Infinitely root-reducible               | $(\exists x)(\forall n \in N)\bigcup_{t \in T} x \rightarrow^* o(c \cdot n)y$ |
| Normalizing term                        | $(\exists x)\bigcup_{t \rightarrow^* x \wedge \neg(\exists y)\bigwedge_{t \rightarrow^* y} x \rightarrow z)$ |
| Normalizing TRS (WN)                    | $(\forall x)(\exists y)\bigcup_{t \rightarrow^* y \wedge \neg(\exists z)\bigwedge_{t \rightarrow^* z} y \rightarrow z)$ |
| Locally confluent (WCR)                 | $(\forall x, y, z)\bigcup_{t \rightarrow^* y \wedge x \rightarrow z} \Rightarrow (\exists u)\bigcup_{t \rightarrow^* u \wedge z \rightarrow^* u}$ |
| Confluent (CR)                          | $(\forall x, y, z)\bigcup_{t \rightarrow^* y \wedge x \rightarrow^* z} \Rightarrow (\exists u)\bigcup_{t \rightarrow^* u \wedge z \rightarrow^* u}$ |

$h$ preserves a formula $\varphi$ if $\varphi$ is satisfied in the target interpretation of $h$ whenever $\varphi$ is satisfied in its domain interpretation [11] Section 2.4. Homomorphisms preserve ECBCA [11] Theorem 2.4.3(a)); the results in [13] rely on this fact. In this paper we extend [14] to deal with more general program properties. Homomorphisms preserve other first-order sentences if further requirements are imposed: (i) positive sentences (where connective ‘¬’ is absent) are preserved under surjective homomorphisms and (ii) arbitrary sentences are preserved under embeddings [11] Theorem 2.4.3. In contrast to [14] (and [11]), here we focus on many-sorted logic [25] (see Section 2). This has an important advantage: since homomorphisms in many-sorted logic with set of sorts $S$ are actually a family $h_s$ of homomorphisms between components of sort $s$ for each $s \in S$, the preservation requirements for $h_s$ depend on the specific quantification of variables $x : s$ for such a sort. In Section 5 we provide a unique preservation theorem that subsumes the results in [14], and even improves [11]. Section 4 investigates how to guarantee surjectivity of homomorphisms. Section 6 shows several application examples taken from Table 1 which shows some properties of rewriting-based systems that could not be captured in [14] but are able to handle now. Here, $t(x)$ is a term with variables $x$ (or just $t$ if it is ground), $C$ (and $D$) are the constructor (resp. defined) symbols in the TRS, and $\rightarrow^*$ is topmost rewriting. Section 6 discusses the possibility of providing more information about disproved properties by means of refutation witnesses, i.e., (counter)examples of sentences which are synthesized from the models that are used to disprove the property. Section 7 shows how to deal with completely general sentences by means of a simple example. Section 8 discusses some related work. Section 9 concludes.

## 2 Many-Sorted First-Order Logic

Given a set of sorts $S$, a (many-sorted) signature (with predicates) $\Omega = (S, \Sigma, \Pi)$ consists of a set of sorts $S$, an $S^* \times S$-indexed family of sets $\Sigma = \{\Sigma_{w,s}\}_{(w,s) \in S^* \times S}$ containing function symbols $f \in \Sigma_{s_1, \ldots, s_k, s}$, with a rank declaration $f : s_1 \cdots s_k \rightarrow$
s (constant symbols c have rank declaration c : λ → s, where λ denotes the empty sequence), and an $S^{+}$-indexed family of sets $Π = \{Π_w\}_{w ∈ S^{+}}$ of ranked predicates $P : w$. Given an $S$-sorted set $X = \{X_s \mid s ∈ S\}$ of mutually disjoint sets of variables (which are also disjoint from $Σ$), the set $T_Σ(X)_s$ of terms of sort $s$ is the least set such that $X_s ⊆ T_Σ(X)_s$ and for each $t : 1 ≤ i ≤ k$ and $t_i ∈ T_Σ(X)_{s_i}$, $1 ≤ i ≤ k$, $f(t_1, \ldots, t_k) ∈ T_Σ(X)_s$. If $X = \emptyset$, we write $T_Σ$ rather than $T_Σ(∅)$ for the set of ground terms. The set $T_Σ(X)$ of many-sorted terms is $T_Σ(X) = \bigcup_{s ∈ S} T_Σ(X)_s$. For $w = s_1 \cdots s_n ∈ S^{+}$, we write $T_Σ(X)_w$ rather than $T_Σ(X)_{s_1} × \cdots × T_Σ(X)_{s_n}$ and even write $t ∈ T_Σ(X)_w$ rather than $t_i ∈ T_Σ(X)_{s_i}$ for each $1 ≤ i ≤ n$. The formulas $ϕ ∈ Form_{Ω}$ of a signature $Ω$ are built up from atoms $P(t)$ with $P ∈ Π_w$ and $t ∈ T_Σ(X)_w$, logic connectives ($\neg$, $∧$, and also $∨$, $⇒$, ... ) and quantifiers ($∀$ and $∃$) in the usual way. A closed formula, i.e., one whose variables are all universally or existentially quantified, is called a sentence. In the following, substitutions $σ$ are assumed to be $S$-sorted mappings such that for all sorts $s ∈ S$, we have $σ(x) ∈ T_Σ(X)_s$.

An $Ω$-structure $A$ consists of (i) a family $\{A_s \mid s ∈ S\}$ of sets called the carriers or domains together with (ii) a function $f_w^{A_s} ∈ A_w → A_s$ for each $f ∈ Σ_{w,s}$ ($A_w$ is a one point set when $w = λ$ and hence $A_w → A_s$ is isomorphic to $A_s$), and (iii) an assignment to each $P ∈ Π_w$ of a subset $P^A_s ⊆ A_s$; if the identity predicate $= : ss$ is in $Π_{ss}$, then $(=)_{A_s} = \{<a,a> \mid a ∈ A_s\}$, i.e., $= : ss$ is interpreted as the identity on $A_s$.

Let $A$ and $A'$ be $Ω$-structures. An $Ω$-homomorphism $h : A → A'$ is an $S$-sorted function $h = \{h_s : A_s → A'_s \mid s ∈ S\}$ such that for each $f ∈ Σ_{w,s}$ and $P ∈ Π_w$ with $w = s_1, \ldots, s_k$, (i) $h_s(f_{w,s}(a_1, \ldots, a_k)) = f_{w,s}(h_{s_1}(a_1), \ldots, h_{s_k}(a_k))$ and (ii) if $a ∈ P^A_s$, then $h(a) ∈ P^{A'}_{h(s)}$. Given an $S$-sorted valuation mapping $α : X → A$, the evaluation mapping $[α]_A : T_Σ(X) → A$ is the unique $(S,Σ)$-homomorphism extending α. Finally, $[\cdot]_A : Form_{Ω} → Bool$ is given by:

1. $[P(t_1, \ldots, t_n)]_A = true$ (with $P ∈ Π_w$) if and only if $(t_1^α, \ldots, t_n^α) ∈ P^A_w$;
2. $[\neg ϕ]_A = true$ if and only if $[ϕ]_A = false$;
3. $[ϕ ∧ ψ]_A = true$ if and only if $[ϕ]_A = true$ and $[ψ]_A = true$; and
4. $[(∀x) : s] ϕ]_A = true$ if and only if for all $a ∈ A_s$, $[ϕ]_A[x → a] = true$.

A valuation $α ∈ X → A$ satisfies $ϕ$ in $A$ (written $A ≺ ϕ [α]$) if $[ϕ]_A = true$. We then say that $ϕ$ is satisfiable. If $A ≺ ϕ [α]$ for all valuations $α$, we write $A ≺ ϕ$ and say that $A$ is a model of $ϕ$ or that $ϕ$ is true in $A$. We say that $A$ is a model of a set of sentences $S ⊆ Form_{Ω}$ (written $A ≺ S$) if for all $ϕ ∈ S$, $A ≺ ϕ$. Given a sentence $ϕ$, we write $S ≺ ϕ$ if $A ≺ ϕ$ holds for all models $A$ of $S$.

### 3 Preservation of Many-Sorted First-Order Sentences

Every set $S$ of ground atoms has an initial model $I_S$ (or just $I$ if no confusion arises) which consists of the usual (many-sorted) Herbrand Domain of ground terms modulo the equivalence $\sim$ generated by the equations in $S$. There is a unique homomorphism $h : I → A$ from $I$ to any model $A$ of $S$ [9, Section 3.2].
In the following, $h$ refers to such a homomorphism. If $S$ contains no equation, then $I$ is the (many-sorted) Least Herbrand Model of $S$ and $I_s$ is $I_{S_s}$ for each sort $s \in S$. In the following, we consider sentences in prenex form as follows:

$$(Q_1 x_1: s_1) \cdots (Q_k x_k: s_k) \bigvee_{i=1}^{m} \bigwedge_{j=1}^{n} L_{ij}$$

where (i) for all $1 \leq i \leq m$ and $1 \leq j \leq n_i$, $L_{ij}$ are literals, i.e., $L_{ij} = A_{ij}$ or $L_{ij} = \lnot A_{ij}$ for some atom $A_{ij}$ (in the first case, we say that $L_{ij}$ is positive; otherwise, it is negative), (ii) $x_1, \ldots, x_k$ for some $k \geq 0$ are the variables occurring in those literals (of sorts $s_1, \ldots, s_k$, respectively), and (iii) $Q_1, \ldots, Q_k$ are universal/existential quantifiers. A sentence $\varphi$ (equivalent to) (6) is said to be positive if all literals are.

**Theorem 1.** Let $\Omega$ be a signature, $S$ be a set of ground atoms, $\varphi$ be a sentence (6), and $A$ be a model of $S$ such that (a) for all $q, 1 \leq q \leq k$, if $Q_q = \forall$ then $h_{s_q}$ is surjective and (b) for all negative literals $L_{ij} = \lnot P(t)$, with $P \in \Pi_w$, and substitutions $\sigma$, if $h(\sigma(t)) \in P^A$ then $\sigma(t) \in P^I$. Then, $I_S \models \varphi \implies A \models \varphi$.

In order to achieve condition (b) in Theorem 1, given $P \in \Pi_w$, let $N(P) = \mathcal{I}_w - P^I$ be the complement of the (Herbrand) interpretation of $P$. Let $\mathcal{N}(P) = \{\lnot P(t) \mid t \in N(P)\}$ (cf. Reiter’s Closed World Assumption [24]). In general, $\mathcal{N}(P)$ is infinite and uncomputable. In some simple cases, though, we can provide a finite description of $\mathcal{N}(P)$ for the required predicates $P$ (see Section 7).

**Proposition 1.** Let $\Omega$ be a signature, $S$ be a set of ground atoms, $\varphi$ be a sentence (6), $A$ be a model of $S$, and $N = \bigcup_{L_{ij} = \lnot P(t)} \mathcal{N}(P)$ be such that $A \models N$. Let $L_{ij} = \lnot P(t)$ be a negative literal and $\sigma$ be a substitution. If $h(\sigma(t)) \in P^A$, then $\sigma(t) \in P^I$.

Consider a theory $S$ and let $S^+$ be the set of ground atoms obtained as the deductive closure of $S$, i.e., the set of atoms $P(t_1, \ldots, t_n)$ for each $n$-ary predicate symbol $P$ and ground terms $t_1, \ldots, t_n$, such that $S \vdash P(t_1, \ldots, t_n)$. The following result is the basis of the practical applications discussed in the following sections.

**Corollary 1 (Semantic criterion).** Let $\Omega$ be a signature, $S_0$ be a theory, $S = S_0^+$, $\varphi$ be a sentence (6), and $A$ be a model of $S_0$ such that (a) for all $q, 1 \leq q \leq k$, if $Q_q = \forall$ then $h_{s_q}$ is surjective and (b) for all negative literals $L_{ij} = \lnot P(t)$, with $P \in \Pi_w$ and substitutions $\sigma$, if $h(\sigma(t)) \in P^A$ then $\sigma(t) \in P^I$. If $A \models \lnot \varphi$, then $I_S \models \lnot \varphi$.

In the following, we will not distinguish between theories $S$ and their ground deductive closure $S^+$; we rather use $S$ in both cases.

**Remark 1 (Proofs by satisfiability).** We can prove an arbitrary sentence $\varphi$ valid in $I_S$ by satisfiability in some model $A$ of $S$. First define $\overline{\varphi}$ as the negation $\lnot \varphi$ of $\varphi$. Then, find an appropriate structure $A$ satisfying (a) and (b) (with regard to $\overline{\varphi}$) and such that $A \models S \cup \{\overline{\varphi}\}$. By Corollary 1, $I \models \lnot \varphi$ holds. Since $\lnot \varphi$ is equivalent to $\varphi$, $I \models \varphi$ holds.

---

1 A mapping $f : A \rightarrow B$ is surjective if for all $b \in B$ there is $a \in A$ such that $f(a) = b$. 

4 Surjective Homomorphisms

Given $\Omega = (S, \Sigma, \Pi)$, $s \in S$ and $T \subseteq T_{\Sigma_s}$, consider the following sentences:

$$\left(\forall x : s\right) \bigvee_{t \in T} x = t \quad \left(7\right) \quad \bigwedge_{t,u \in T, t \neq u} \neg(t = u) \quad \left(8\right)$$

In the following, we write (7) to make sort $s$ referred in (7) explicit. We do the same in similar formulas below.

**Proposition 2.** Let $\Omega$ be a signature, $S$ be a theory, $A$ be a model of $S$, $s \in S$, and $T \subseteq T_{\Sigma_s}$. (a) If $T \neq \emptyset$ and $A \models (7)$, then $h_s$ is surjective and $|A_s| \leq |T|$. (b) If $A_s \neq \emptyset$ and $A \models (8)$, then $|A_s| \geq |T|$.

In view of Proposition 2(a), denote (7) as $SuH^T_s(\Omega)$ (or just $SuH^T_s$ or $SuH^T$ if no confusion arises). Whenever $T$ is finite, Proposition 2(a) imposes that the interpretation domain $A_s$ for sort $s$ is finite. This is appropriate for tools like Mace4 which generate structures with finite domains only. However, the choice of $T$ in Proposition 2, when used together with a theory $S$ imposing further requirements on symbols, can be crucial for Corollary 1 to succeed. Restricting the attention to finite domains can also be a drawback. In the following, we investigate a different approach which avoids any choice of terms $T$ and is valid for infinite structures as well.

Consider the following sentence:

$$\left(\forall x : s\right)\left(\exists n : \text{Nat}\right) \text{term}_s(x, n) \quad \left(9\right)$$

where $\text{Nat}$ is a new sort, to be interpreted as the set $\mathbb{N}$ of natural numbers, and $\text{term}_s : s \rightarrow \text{Nat}$ is a new predicate for each $s \in S$. The intended meaning of (9) is that, for all $x \in A_s$, there is $t \in T_{\Sigma_s}$ of height at most $n$ such that $x = t^A$. We substantiate this, for each sort $s \in S$, by means of two (families of) formulas:

$$\left(\forall x : s\right)\left(\forall n : \text{Nat}\right) \text{term}_s(x, 0) \Rightarrow \bigvee_{c \in \Sigma_{\lambda,s}} x = c \quad \left(10\right)$$

$$\left(\forall x : s\right)\left(\forall n : \text{Nat}\right)\left(\exists m : \text{Nat}\right) \left(n > 0 \land \text{term}_s(x, n)\right) \Rightarrow$$

$$n > m \land \left(\left.\text{term}_s(x, m)\vee \bigvee_{f \in \Sigma_{w,s}} \left(\exists y : w\right) \left(x = f(y) \land \bigwedge_{s_i \in w} \text{term}_{s_i}(y_i, m)\right)\right) \quad \left(11\right)$$

Thus, by (10), values $x$ satisfying $\text{term}_s(x, 0)$ will be represented by some constant symbol $c$ of sort $s$. Similarly, by (11), values $x$ satisfying $\text{term}_s(x, n)$ for
some \( n > 0 \) will be represented by some ground term \( s \) of height \( m \) for some \( m < n \), or by a term \( t = f(t_1, \ldots, t_k) \), where \( f \) has rank \( w \rightarrow s \) for some \( w \in S^+ \) and \( t_1, \ldots, t_k \) have height \( m \) at most.

The set \( K(s) \) of \( s \)-relevant sorts is the least set satisfying: (i) \( s \in K(s) \) and (ii) if \( f \in \Sigma_{s_1^{s_1} \cdots s_n^{s_n}} \) and \( s' \in K(s) \), then \( \{s_1, \ldots, s_n\} \subseteq K(s) \). Let \( \Omega_{Nat,s} = (S_{Nat}, \Sigma_{Nat}, \Pi_{Nat,K(s)}) \) be an extension of \( \Omega \) where \( S_{Nat} = S \cup \{Nat\} \), \( \Sigma_{Nat} \) extends \( \Sigma \) with a new constant \( 0 : \lambda \rightarrow Nat \), and \( \Pi_{Nat,K(s)} \) extends \( \Pi \) with \( > : Nat Nat \) and a predicate \( \text{term}_{s'} : s' Nat \) for each \( s' \in K(s) \). We let

\[
\text{SuH}_s = \{ (9)_{s'}, (10)_{s'}, (11)_{s'} | s' \in K(s) \}
\]

(12)

**Proposition 3.** Let \( \Omega \) be a signature, \( S \) be a theory, \( s \in S \), and \( A \) be an \( \Omega_{Nat,s} \)-structure which is a model of \( S \). Assume that \( A_{Nat} = N, 0^A = 0 \), and \( m >^A n \leftrightarrow m >_N n \) for all \( m, n \in A_{Nat} \). If \( A \models \text{SuH}_s \), then \( h_{s'} \) is surjective for all \( s' \in K(s) \).

Given an extension \( \Omega' \) of a signature \( \Omega \), every \( \Omega' \)-structure \( A' \) defines an \( \Omega \)-structure \( A \); just take \( A_s = A'_s \) for all \( s \in S \), and then \( f^A_{w,s} = f^{A'}_{w,s} \) and \( P^A_w = P^{A'}_w \) for all \( w \in S^+, s \in S, f \in \Sigma_{w,s}, \) and \( P \in \Pi_w \). Thus, Proposition 3 is used to guarantee surjectivity of \( h : T_{\Sigma_{s'}} \rightarrow A_{s'} \), rather than \( h : T_{\Sigma_{Nat,s'}} \rightarrow A_{s'} \).

## 5 Examples of Application with Positive Sentences

In this section we exemplify the use of Corollary 1 together with the approach in Section 4 to deal with positive sentences (6), i.e., all literals are positive.

### 5.1 Complete Definedness and Commutativity

Consider the following Maude specification (hopefully self-explained, but see [5]) for the arithmetic operations in Example 1 together with function \( \text{head} \), which returns the head of a list of natural numbers:

mod ExAddMulHead is

\[
\text{sorts} \ N \ LN . \ \text{*** Sorts for numbers and lists of numbers} \\
\text{op} \ Z : \rightarrow N . \ \text{op} \ suc : N \rightarrow N . \ \text{ops} \ \text{add} \ \text{mul} : N \ N \rightarrow N . \\
\text{op} \ \text{head} : LN \rightarrow N . \ \text{op} \ \text{nil} : \rightarrow LN . \ \text{op} \ \text{cons} : N \ LN \rightarrow LN . \\
\text{vars} \ x \ y : N . \ \text{var} \ \text{xs} : LN . \\
\text{rl} \ \text{add}(Z,x) \Rightarrow x . \ \text{rl} \ \text{add}(\text{suc}(x),y) \Rightarrow \text{suc}(\text{add}(x,y)) . \\
\text{rl} \ \text{mul}(Z,x) \Rightarrow Z . \ \text{rl} \ \text{mul}(\text{suc}(x),y) \Rightarrow \text{add}(y,\text{mul}(x,y)) . \\
\text{rl} \ \text{head}(\text{cons}(x,\text{xs})) \Rightarrow x . \\
de\text{md}
\]

1) Complete definedness. We claim \( \text{head} \) to be completely defined as follows:

\[
(\forall x : LN)(\exists x : N) \ \text{head}(x) \rightarrow x
\]

(13)
We disprove [13] by using Corollary 1. Due to the universal quantification of $x$s in [13], we need to ensure that $h_{\text{add}}: \mathcal{T}_{\Sigma_{LM}} \to \mathcal{A}_{LM}$ is surjective for any structure $\mathcal{A}$ we may use. We use Proposition 3. Since $K(\text{LN}) = \{N,\text{LN}\}$ due to $\text{cons}$, whose first argument is of sort $N$, $\text{SuH}_{LM}$ consists of the following sentences:

\[
\begin{align*}
(\forall x : N)(\exists n : N) & \quad \text{term}_{\text{add}}(x, n) \\
(\forall x : N) & \quad \text{term}_{\text{add}}(x, 0) \Rightarrow x = Z \\
(\forall x : N)(\forall n : N)(\exists m : N) & \quad (\exists y : N) (\exists z : N)(\exists y, z : \text{LN}) \quad n > 0 \land \text{term}_{\text{add}}(x, n) \Rightarrow n > m \land [\text{term}_{\text{add}}(y, m) \lor \text{term}_{\text{add}}(z, m) \land \text{term}_{\text{add}}(y, m) \land \text{term}_{\text{add}}(z, m) \land (x = \text{suc}(y) \lor x = \text{add}(y, z) \lor x = \text{mul}(y, z) \lor x = \text{head}(y, z)])]
\end{align*}
\]

We obtain a model $\mathcal{A}$ of $\text{ExAddMulHead} \cup \text{SuH}_{LM} \cup \{-[13]\}$ with AGES. Sorts are interpreted as follows: $\mathcal{A}_N = \mathcal{A}_{\text{LM}} = \{-1, 0\}$ and $\mathcal{A}_{\text{Nat}} = N$. For function symbols:

\[
\begin{align*}
Z^\mathcal{A} &= -1 & \text{nil}^\mathcal{A} &= 0 & \text{suc}^\mathcal{A}(x) &= x & \text{add}^\mathcal{A}(x, y) &= 0 & \text{mul}^\mathcal{A}(x, y) &= 0 & \text{cons}^\mathcal{A}(x, x, x) &= -1 & \text{head}^\mathcal{A}(x, x) &= -x - 1
\end{align*}
\]

For predicates, $x \rightarrow^\mathcal{A}_N y \iff x \geq y \land x \geq 0$, $x \rightarrow^\mathcal{A}_N y \iff x = y = -1$, and both $x(\rightarrow^\mathcal{A}_N y)$ and $x(\rightarrow^\mathcal{A}_N y)$ are true. We can check surjectivity of $h_{\text{add}}: \mathcal{T}_{\Sigma_{LM}} \to \mathcal{A}$ (for $s \in \{N,\text{LN}\}$). For instance, we have:

\[
\begin{align*}
[Z]_\mathcal{A} &= -1 & [\text{add}(Z, Z)]_\mathcal{A} &= 0 & \text{for sort } N \\
[\text{cons}(Z, \text{nil})]_\mathcal{A} &= -1 & [\text{nil}]_\mathcal{A} &= 0 & \text{for sort } \text{LN}
\end{align*}
\]

2) Commutativity. It is well-known that both $\text{add}$ and $\text{mul}$ as defined by the rules of $\mathcal{R}$ in Example 1 are commutative on ground terms, i.e., for all ground terms $s$ and $t$, $\text{add}(s, t) =_{\mathcal{R}} \text{add}(t, s)$ and $\text{mul}(s, t) =_{\mathcal{R}} \text{mul}(t, s)$, where $=_{\mathcal{R}}$ is the equational theory induced by the rules $\ell \rightarrow r$ in $\mathcal{R}$ treated as equations $\ell = r$. Actually, by using Birkhoff’s theorem and the fact that $\mathcal{R}$ is confluent, we can rephrase commutativity of $\text{add}$ as joinability as follows:

\[
(\forall x)(\forall y)(\exists z) \quad \text{add}(x, y) \rightarrow^* z \land \text{add}(y, x) \rightarrow^* z
\]

Remark 2. Proving commutativity of $\text{add}$ and $\text{mul}$ when defined by $\mathcal{R}$ in Example 1 by using Corollary 1 is possible (see Remark 1) but unlikely. We should first define $\forall$ as $\neg$[13], i.e., $\forall$ is

\[
(\exists x)(\exists y)(\forall z) \quad \neg(\text{add}(x, y) \rightarrow^* z) \lor \neg(\text{add}(y, x) \rightarrow^* z)
\]

Since [15] contains two negative literals, Corollary 1 requires the use of $\mathcal{N}(\rightarrow^*)$. 

8
Since \texttt{head} is not completely defined, \texttt{add} and \texttt{mul} are not commutative in ExAddMulHead. We prove this fact by disproving the sorted version of (14), i.e.,

\[(\forall x : \mathbb{N})(\forall y : \mathbb{N})(\exists z : \mathbb{N}) \text{add}(x, y) \rightarrow^* z \land \text{add}(y, x) \rightarrow^* z \quad (16)\]

Due to the universal quantification of \texttt{x} and \texttt{y} in (16), we need to ensure that \(h_{\mathbb{N}} : \mathcal{T}_{\mathbb{N}} \rightarrow \mathcal{A}_{\mathbb{N}}\) is surjective. Since \(K(\mathbb{N}) = \{\mathbb{N}, \mathcal{L}_{\mathbb{N}}\}\) due to \texttt{head}, we have \(\text{SuH}_{\mathbb{N}} = \text{SuH}_{\mathcal{L}_{\mathbb{N}}}\) as above. AGES obtain a model \(\mathcal{A}\) of ExAddMulHead \(\cup \text{SuH}_{\mathbb{N}} \cup \{\neg (16)\}\) as follows: \(\mathcal{A}_{\mathbb{N}} = \{0, 1\}, \mathcal{A}_{\mathcal{L}_{\mathbb{N}}} = \{-1, 0\}\) and \(\mathcal{A}_{\text{Nat}} = \mathbb{N}\). Also,

\[
\begin{align*}
Z^A &= 1 \\
\text{nil}^A &= -1 \\
\text{suc}^A &= x \\
\text{add}^A &= (x, y) = y \\
\text{mul}^A &= x \\
\text{cons}^A &= (x, xs) = x - 1 \\
\text{head}^A &= (xs) = xs + 1 \\
\end{align*}
\]

\(x \rightarrow^A_{\mathcal{L}_{\mathbb{N}}} y \iff x = y \quad x \rightarrow^A_{\mathcal{L}_{\mathbb{N}}} y \iff x = y \quad x \rightarrow^A_{\mathcal{L}_{\mathbb{N}}} y \iff x = y \quad x \rightarrow^A_{\mathcal{L}_{\mathbb{N}}} y \iff \text{true}
\]

5.2 Top-Termination

A TRS \(R\) is top-terminating if no infinitary reduction sequence performs infinitely many rewrites at topmost position \(\Lambda\) [7]. From a computational point of view, top-termination is important in the semantic description of lazy languages as it is an important ingredient to guarantee that every initial expression has an infinite normal form [7,8]. Accordingly, given a dummy sort \(S\), the negation of

\[(\exists x : S)(\forall n \in \mathbb{N})(\exists y : S) x \rightarrow^* (\rightarrow^* \circ \Lambda)^n y \quad (17)\]

(which claims for the existence of a term with infinitely many rewriting steps at top) captures top-termination. We introduce a new predicate \(\rightarrow^* \circ A\) for the composition \(\rightarrow^* \circ A\) of the many-step rewriting relation \(\rightarrow^*\) (defined as usual, i.e., by the whole theory \(\mathcal{R}\) associated to \(R\)) and topmost rewriting \(\Lambda\) defined by a theory \(\mathcal{R}_A = \{(\forall x : S) \ell \rightarrow^A r | \ell \rightarrow r \in \mathcal{R}\}\). Sequences \(s \rightarrow^* A t\) meaning that \(s \rightarrow^* A\)-reduces into \(t\) in \(n + 1 \rightarrow^* A\)-steps are defined as follows:

\[
\begin{align*}
(\forall x, y, z : S) \quad x \rightarrow^* y \land y \rightarrow^A z \Rightarrow x \rightarrow^0 A z \\
(\forall x, y, z : S)(\forall n \in \mathbb{N}) \quad x \rightarrow^0 A y \land y \rightarrow^* A z \Rightarrow x \rightarrow^{n+1} A z \\
\end{align*}
\]

Overall, the sentence \(\varphi\) to be disproved is:

\[(\exists x : S)(\forall n : \text{Nat})(\exists y : S) x \rightarrow^*_A y \quad (20)\]

Remark 3. We use \(\mathbb{N}\) in (17) but \(\text{Nat}\) in (20). Indeed, (17) is not a valid sentence because \(\mathbb{N}\) is not first-order axiomatizable, see, e.g., [11] Section 2.2. This is consistent with the well-known fact that termination (or top-termination) cannot be encoded in first-order logic [22] Section 5.1.4. We can use (20) together with Corollary [1] provided that \(\text{Nat}\) is interpreted as \(\mathbb{N}\). This is possible with AGES.
Example 2. Consider the following (nonterminating) TRS $R$ [8, Section 9.5]:

\[
\begin{align*}
\text{non} & \rightarrow f(g, f(\text{non}, g)) \\
g & \rightarrow a \\
f(a, x) & \rightarrow a \\
f(b, b) & \rightarrow b \\
f(b, a) & \rightarrow b
\end{align*}
\]

The associated theory $R_{topT}$ is $R_{topT} = R \cup R_A \cup \{18, 19\}$, where $R_A$ is

\[
\begin{align*}
\text{non} & \overset{\Lambda}{\rightarrow} f(g, f(\text{non}, g)) \\
g & \overset{\Lambda}{\rightarrow} a \\
f(b, b) & \overset{\Lambda}{\rightarrow} b \\
f(b, a) & \overset{\Lambda}{\rightarrow} b
\end{align*}
\]

Note that (20) only requires that the homomorphism mapping terms of sort $\text{Nat}$ to $\mathbb{N}$ is surjective, which is automatically achieved by AGES. The structure $A$ with $A_{\text{b}} = \{-1, 0, 1\}$, $A_{\text{Nat}} = \mathbb{N}$, function symbols interpreted by: $a^A = 1$, $b^A = 1$, $g^A = 0$, $\text{non}^A = -1$, and $f^A(x) = 0$; and predicate symbols as follows:

\[
x \rightarrow^A y \iff y \geq x \land y \geq -1 \\
x(\rightarrow^*)^A y \iff y \geq x
\]

is a model of $R_{topT} \cup \{\neg(20)\}$ and proves top-termination of $R$.

6 Refutation Witnesses

In logic, a witness for an existentially quantified sentence $(\exists x) \varphi(x)$ is a specific value $b$ to be substituted by $x$ in $\varphi(x)$ so that $\varphi(b)$ is true (see, e.g., [2, page 81]). Similarly, we can think of a value $b$ such that $\neg \varphi(b)$ holds as a witness of $(\exists x) \neg \varphi(x)$ or as a refutation witness for $(\forall x) \varphi(x)$; we can also think of $b$ as a counterexample to $(\forall x) \varphi(x)$ [31, page 284]. Note, however, that witnesses that are given as values $b$ belonging to an interpretation domain $A$ can be meaningless for the user who is acquainted with the first-order language $\Omega$ but not so much with abstract values from $A$ (which is often automatically synthesized by using some tool). Users can be happier to deal with terms $t$ which are somehow connected to witnesses $b$ by a homomorphism, so that $t^A = b$. Corollary 1 permits a refutation of $\varphi$ by finding a model $A$ of $\neg \varphi$ to conclude that $I \models \neg \varphi$. We want to obtain instances of $\varphi$ to better understand unsatisfiability of $\varphi$. In this section we investigate this problem.

The negation $\neg(6)$ of (6), i.e., of $(Q_1 x_1 : s_1) \cdots (Q_k x_k : s_k) \bigvee_{i=1}^m \bigwedge_{j=1}^{n_i} L_{ij}$ is

\[
(\overline{Q}_1 x_1 : s_1) \cdots (\overline{Q}_k x_k : s_k) \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n_i} \neg L_{ij}(x_1, \ldots, x_k)
\]

where $\overline{Q}_i$ is $\forall$ whenever $Q_i$ is $\exists$ and $\overline{Q}_i$ is $\exists$ whenever $Q_i$ is $\forall$. We assume $\eta \leq k$ universal quantifiers in (32) with indices $U = \{v_1, \ldots, v_\eta\} \subseteq \{1, \ldots, k\}$ and hence $k - \eta$ existential quantifiers with indices $E = \{\epsilon_1, \ldots, \epsilon_{k-\eta}\} = \{1, \ldots, k\} - \eta$. 



In the following \( \eta \) denotes \( k - \eta \). For each \( \epsilon \in E \), we let \( U_\epsilon = \{ v \in U \mid v < \epsilon \} \) be the (possibly empty) set of indices of universally quantified variables in (32) occurring before \( x_\epsilon \) in the quantification prefix of (32). Let \( \eta_\epsilon = |U_\epsilon| \). Note that \( U_{\epsilon_1} \subseteq U_{\epsilon_2} \subseteq \cdots \subseteq U_{\epsilon_m} \). Let \( U_3 \) be the set of indices of universally quantified variables occurring before some existentially quantified variable in the quantification prefix of (32). Note that \( U_3 \) is empty whenever \( \epsilon_1 > \epsilon_{k-\eta} \) (no existential quantification after a universal quantification); otherwise, \( U_3 = \{ \epsilon_1, \ldots, \epsilon_m \} \) for some \( m \leq \epsilon_\eta \). Accordingly, \( U_\eta = U - U_3 = \{ \epsilon_{\eta + 1}, \ldots, k \} \) is the set of indices of universally quantified variables occurring after all existentially quantified variables in the quantification prefix of (32). Note that \( U_\eta \) is empty whenever \( \epsilon_1 > \epsilon_\eta \) (no universal quantification after an existential quantification).

Most theorem provers transform sentences into universally quantified formulas by Skolemization (see, e.g., [12]). Thus, if \( k > \eta \), i.e., (32) contains existential quantifiers, we need to introduce Skolem function symbols \( sk_\epsilon : w_\epsilon \rightarrow s_\epsilon \) for each \( \epsilon \in E \), where \( w_\epsilon \) is the (possibly empty) sequence of \( \eta_\epsilon \) sorts indexed by \( U_\epsilon \). Note that \( sk_\epsilon \) is a constant if \( \eta_\epsilon = 0 \). The Skolem normal form of (32) is

\[
(\forall x_{v_1} : s_{v_1}) \cdots (\forall x_{v_\eta} : s_{v_\eta}) \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n_i} \neg L_{i\epsilon}(e_1, \ldots, e_k)
\]

(33)

where for all \( 1 \leq q \leq k \), (i) \( e_q \equiv x_q \) if \( q \in U \) and (ii) \( e_q \equiv sk_q(x_q) \) if \( q \in E \), where \( x_q \) is the sequence of variables \( x_{v_1}, \ldots, x_{v_\eta} \). If \( E \neq \emptyset \) (i.e., (32) and (33) differ), then (33) is a sentence of an extended signature \( \Omega^s_k = (S, \Sigma^s_k, T) \) where \( \Sigma^s_k \) extends \( \Sigma \) with skolem functions. Since (33) logically implies (32) [2 Section 19.2], every model \( \mathcal{A} \) of (33) is a model of (32) as well.

**Definition 1 (Set of refutation witnesses).** Using the notation developed in the previous paragraphs, let \( \mathcal{A} \) be an \( \Omega^s_k \)-structure such that \( h_{s_q} \) is surjective for all \( q \in U_3 \cup E \). The \( \Omega^s_k \)-sentence (34) is given a set of refutation witnesses \( \Phi \) consisting of \( \Omega \)-sentences \( \phi_\alpha \) for each valuation \( \alpha \) of the variables \( x_{v_1}, \ldots, x_{v_3} \) indexed by \( U_3 \); each \( \phi_\alpha \) is (nondeterministically) defined as follows:

\[
(\forall x_{v_{\eta + 1}} : s_{v_{\eta + 1}}) \cdots (\forall x_{v_k} : s_k) \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n_i} \neg L_{i\epsilon}(e_1', \ldots, e_k')
\]

(34)

where for all \( 1 \leq q \leq k \), (i) \( e_q' \equiv x_q \) if \( q \in U_\eta \) and (ii) \( e_q' \equiv t \) if \( q \in U_3 \cup E \) and \( t \in T_{\Sigma s_q} \) is such that \( [t]_\mathcal{A} = [e_q']_\mathcal{A} \).

Note that, in Definition 1 we could emph fail to find the necessary terms \( t \in T_{\Sigma s_q} \) if \( h_{s_q} \) is not surjective. Note also that, whenever \( E \) is empty, \( \Phi \) is a singleton consisting of (34) which coincides with (33). We have the following:

**Proposition 4.** For every \( \Omega^s_k \)-structure \( \mathcal{A} \), \( \mathcal{A} \models (33) \) if and only if \( \mathcal{A} \models \Phi \).

Refutation witnesses are built from symbols in the original signature \( \Omega \) only. We can use them as more intuitive counterexamples to the refuted property \( \varphi \).
Proposition 5. Let $\Omega$ be a signature, $S$ be a theory, $\varphi$ be a sentence $\Omega$, and $A$ be a model of $S$ such that for all negative literals $L_{ij} = \neg P(t)$ with $P \in \Pi_w$ and substitutions $\sigma$, if $h(\sigma(t)) \in P^A$ then $\sigma(t) \in P^T$. For all $\phi \in \Phi$, $I \models \phi$.

Corollary 2. If $\Omega$ is positive, then for all refutation witnesses $\phi \in \Phi$, $I \models \phi$.

Example 3. Consider ExAddMulHead in Section 5. The refutation of (13) using AGES actually proceeds by skolemization of the negation of (13), i.e., of

$$ (\exists x : LN)(\forall x : N) \neg(\text{head}(xs) \rightarrow x) $$

(35)

With regard to (35), we have $E = \{1\}$, $U_3 = \emptyset$ and $U_\forall = \{2\}$, where 1 and 2 refer to variables $xs$ and $x$, respectively. Accordingly, $\nu_3 = 0$. The only sort involved in the variables indexed by $U_3 \cup E$ is $\text{LN}$. Since variables of sort $\text{LN}$ are universally quantified in (13), the application of Corollary 1 in Section 5 already required surjectivity of $h_{\text{LN}}$. The Skolem normal form of (35) is:

$$ (\forall x : N) \neg(\text{head}(sk_{xs}) \rightarrow x) $$

(36)

where $sk_{xs}$ is a new constant of sort $\text{LN}$. The structure $A$ computed by AGES is actually a model of $\mathcal{R} \cup \text{SuH}_{\text{LN}} \cup \{\text{nil}\}$, for $\text{SuH}_{\text{LN}}$ in Section 5. For $sk_{xs}$, we have $sk_{xs}^A = 0$. There is a single (empty) valuation $\alpha$ of variables indexed by $U_3$ (which is empty). Hence, $\Phi = \{\phi_\alpha\}$ is a singleton. According to Definition 1 since $[\text{nil}]_A = 0 = [sk_{xs}]_A$, the following sentence could be associated to the refutation witness $\phi_\alpha$:

$$ (\forall x : N) \neg(\text{head}(\text{nil}) \rightarrow x). $$

Example 4. With regard to the computation of refutation witnesses for $\mathcal{R}$ in Example 2 we start with the negation of (17), i.e.,

$$ (\forall x : S)(\exists n : \text{Nat})(\forall y : S) \neg(x(\rightarrow^* \circ A^n)y) $$

(37)

We have $E = \{2\}$, $U_3 = \{1\}$ and $U_\forall = \{3\}$. The Skolem normal form of (37) is

$$ (\forall x : S)(\forall y : S) \neg(x(\rightarrow^* \circ A_1^{sk_n(x)})y) $$

(38)

where $sk_n : S \rightarrow \text{Nat}$ is a new (monadic) function symbol. Since the sorts for variables indexed by $U_3 \cup E$ are $S$ and $\text{Nat}$, we require surjectivity of $h_S$ and $h_{\text{Nat}}$.

This is achieved by using $\text{SuH}_S$ and interpreting $\text{Nat}$ as $\text{N}$ as done in AGES. The structure $A$ in Example 2 is a model of $\mathcal{R}_{\text{topT}} \cup \text{SuH}_S \cup \{\text{nil}\}$. The interpretation obtained for $sk_n$ is

$$ sk_n^A(x) = 1 - x $$

Now we can compute refutation witnesses for (38). Since $U_e = \{1\}$ is a singleton whose index refers to a variable $x$ of sort $S$ and $A_3 = \{-1, 0, 1\}$, we have to deal with three valuation functions for the only variable $x$ to be considered:

$$ \alpha_{-1}(x) = -1 \quad \alpha_0(x) = 0 \quad \alpha_1(x) = 1 $$

We have $\Phi = \{\phi_{-1}, \phi_0, \phi_1\}$, where $\phi_{-1}$ is $(\forall y : S)\neg(\text{non}(\rightarrow^* \circ A^n)y)$, $\phi_0$ is $(\forall y : S)\neg(\text{non}(\rightarrow^* \circ A^n)y)$, and $\phi_1$ is $(\forall y : S)\neg(\text{non}(\rightarrow^* \circ A^n)y)$.

Note that, since $f^A(x) = 0$, we could also write $\phi_0$ as $(\forall y : S)\neg(f(t)(\rightarrow^* \circ A^n)y)$ for every ground term $t$. This gives additional, complementary information.
7 Example of Application with General Sentences

Consider a well-known example of a locally confluent but nonconfluent TRS \( \mathcal{R} \):

\[
\begin{align*}
  b & \to a \\
  b & \to c \\
  c & \to b \\
  c & \to d
\end{align*}
\]

**Example 5 (Local confluence of \( \mathcal{R} \)).** Local confluence corresponds to \( \varphi_{WCR} \) in Table 1. As explained in Remark 1, we start with

\[
(\exists x, y, z : S) (\forall u : S) (x \to y \land x \to z \land \neg (x \to^* u)) \lor (x \to y \land x \to z \land \neg (z \to^* u)) (39)
\]

Due to the universal quantifier, \( h_S : \mathcal{T}_S \to \mathcal{A}_S \) must be surjective. We can achieve this by adding the following sentence \( \text{SuH} \) \( \mathcal{S} \):

\[
(\forall x : S) x = a \lor x = b \lor x = c \lor x = d
\]

Due to the negative literals \( \neg (x \to^* u) \) and \( \neg (z \to^* u) \), we consider \( \mathcal{N} \), representing the forbidden many-step rewriting steps, explicitly given by:

\[
\mathcal{N} = \{ \neg (a \to^* b), \neg (a \to^* c), \neg (a \to^* d), \neg (d \to^* a), \neg (d \to^* b), \neg (d \to^* c) \}
\]

We apply Corollary 1 to prove that \( \overline{\varphi_{WCR}} \) (i.e., \( \varphi_{WCR} \)) holds by obtaining a model of \( \overline{\mathcal{R}} \cup \text{SuH}_S \cup \mathcal{N} \cup \{ \varphi_{WCR} \} \) with Mace4. The structure has domain \( \mathcal{A}_S = \{ 0, 1, 2, 3 \} \): constants are interpreted as follows: \( a^A = 0, b^A = 1, c^A = 3, \) and \( d^A = 2 \). With regard to predicate symbols, we have:

\[
x \to^A y = \{ (1, 0), (1, 3), (3, 1), (3, 2) \} \quad x(\to^*)^A y = \{ (1, x), (3, x) \mid x \in \mathcal{A}_S \}
\]

This proves \( \mathcal{R} \) locally confluent.

**Example 6 (Nonconfluence of \( \mathcal{R} \)).** In order to disprove confluence of \( \mathcal{R} \), which is represented by \( \varphi_{CR} \) in Table 1, we first write \( \varphi_{CR} \) in the form (i.e.,

\[
(\forall x, y, z : S) (\exists u : S) \neg (x \to^* y) \lor \neg (x \to^* z) \lor (y \to^* u \land z \to^* u)
\]

Due to the universal quantification and negative literals, we use \( \text{SuH}_S^T \) and \( \mathcal{N} \) as in Example 5. We obtain a model \( \mathcal{A} \) of \( \mathcal{R} \cup \text{SuH}_S^T \cup \mathcal{N} \cup \{ \varphi_{WCR} \} \) with Mace4. The domain is \( \mathcal{A}_S = \{ 0, 1, 2 \} \) and symbols are interpreted by: \( a^A = 0, b^A = c^A = 1, \) \( d^A = 2, x \to^A y \leftrightarrow x = 1 \), and \( x(\to^*)^A y \leftrightarrow x = y \vee x = 1 \). This proves nonconfluence of \( \mathcal{R} \). With regard to the refutation witnesses, \( \neg \varphi_{CR} \) is

\[
(\exists x, y, z : S) (\forall u : S) x \to^* y \land x \to^* z \land \neg (y \to^* u \land z \to^* u)
\]

and its Skolem normal form is

\[
(\forall u : S) \ sk_x \to^* \ sk_y \land \ sk_x \to^* \ sk_z \land \neg (sk_y \to^* u \land sk_z \to^* u)
\]

Mace4 yields \( sk_x^A = 1, sk_y^A = 0 \) and \( sk_z^A = 2 \); \( \Phi \) consists of a single sentence; e.g.,

\[
(\forall u : S) b \to^* a \land b \to^* d \land \neg (a \to^* u \land d \to^* u)
\]

but also: \( (\forall u : S) c \to^* a \land c \to^* d \land \neg (a \to^* u \land d \to^* u) \). Indeed, they represent the two possible cases of nonconfluent behavior in \( \mathcal{R} \).

---

2 This proves \( \mathcal{R} \) ground locally confluent, i.e., variables in \( \varphi_{WCR} \) refer to ground terms only; since \( \mathcal{R} \) is a ground TRS, local confluence and ground local confluence coincide.
Example 7 (Normalizing TRS). \( \mathcal{R} \) is not terminating, but we can prove it normalizing (i.e., every term has a normal form) by disproving \( \overline{\mathcal{R}}_{WN} \), for \( \varphi_{WN} \) in Table 1. Therefore, \( \overline{\mathcal{R}}_{WN} \) is \( (\exists x : \mathcal{S})(\exists y : \mathcal{S})(\exists z : \mathcal{S})(\neg(x \to^* y) \lor y \to z) \). We guarantee surjectivity by using \( \text{Suh}^T \) in Example 5; we also use \( \mathcal{N} \) in Example 6. Mace4 obtains a model \( \mathcal{A} \) of \( \overline{\mathcal{R}} \cup \text{Suh}^T \cup \mathcal{N} \cup \{\varphi_{WN}\} \) with \( \mathcal{A}_0 = \{0, 1, 2\} \), \( a^A = 0 \), \( b^A = c^A = 1 \), \( d^A = 2 \), \( x \to^A y \Leftrightarrow x = 1 \), and \( x(\to^*)^A y \Leftrightarrow x = y \lor x = 1 \).

8 Related Work

In [14] Section 6 we already compared our approach to existing techniques and tools for the so-called First-Order Theory of Rewriting [6], which applies to restricted classes of TRSs and formulas. In [16], we show that our semantic approach is practical when applied to arbitrary (Conditional) TRSs.

McCune’s Prover9/Mace4 are popular automated systems for theorem proving in first-order and equational logic. Given a theory \( \mathcal{S} \) and a goal or statement \( \varphi \), Prover9 tries to prove that \( \mathcal{S} \vdash \varphi \) holds. The generator of models Mace4 complements Prover9 as follows: “If the statement is the denial of some conjecture, any structures found by Mace4 are counterexamples to the conjecture” [3]. Accordingly, the user introduces \( \varphi \) in the goal section of Mace4, but the system seeks a model of \( \mathcal{S} \cup \{\neg \varphi\} \). Indeed, as discussed in Section 1, if \( \mathcal{A} \models \mathcal{S} \cup \{\neg \varphi\} \) holds, then \( \mathcal{S} \vdash \varphi \) does not hold. But, unless \( \varphi \) is an ECBCA, this does not necessarily mean that \( \varphi \) does not hold of a program \( P \) with \( \mathcal{S} = \overline{P} \). Consider the following ‘misleading’ session with Mace4 that ‘disproves’ commutativity of the addition.

Example 8. Consider \( \mathcal{R} \) in Example 1. Mace4 obtains a model \( \mathcal{A} \) of \( \overline{\mathcal{R}} \cup \{\neg(14)\} \) with domain \( \mathcal{A} = \{0, 1\} \), and function and predicate symbols as follows: \( 0^A = 0 \), \( s^A(x) = x \), \( \text{add}^A(x, y) = \begin{cases} 1 & \text{if } x = 0 \land y = 1 \\ 0 & \text{otherwise} \end{cases} \), \( \text{mul}^A(x, y) = 0 \), and \( \to^A \) and \( (\to^*)^A \) both interpreted as the equality. Additionally, Mace4 also displays the following: \( c1^A = 0 \) and \( c2^A = 1 \). These \( c1 \) and \( c2 \) are new Skolem symbols (but unexpected for most users!). In practice, Mace4 finds a model for the Skolem normal form of \( \neg(14) \), which is

\[
(\forall z) (\neg (\text{add}(c1, c2) \to^* z \land \text{add}(c2, c1) \to^* z) ) \tag{45}
\]

Indeed, \( \mathcal{A} \) is a model of \( \overline{\mathcal{R}} \cup \{45\} \). But we should not conclude (as suggested by the aforementioned sentences in Mace4 manual) that \( \text{add} \) is not commutative! The problem in Example 8 is that \( h : \mathcal{T}_S \to \mathcal{A} \) is not surjective. For instance, no ground term \( t \in \mathcal{T}_S \) satisfies \( t^A = 1 \); note that \( c1, c2 \notin \Sigma \). Since proving validity in \( \mathcal{T}_S \) is not the main purpose of Mace4, no warning in its documentation prevents the prospective user to give credit to the ‘refutation’ of (ground) commutativity for the addition computed by Mace4. We believe that our work is helpful to clarify the use of such tools, and even improve it by adding (for instance) sentences reinforcing surjectivity to avoid the problem discussed above. For instance, Mace4 obtains no model of \( \overline{\mathcal{R}} \cup \text{Suh}^T \cup \{13\} \) with, e.g., \( T = \{0, s(0)\} \).

3: https://www.cs.unm.edu/~mccune/prover9/manual/2009-11A/mace4.html
Proofs by Satisfiability vs. Theorem Proving. In order to further clarify the differences between our approach and the use of first-order theorem proving tools, consider the CTRS $\mathcal{R}$ in Example 1, consisting of the rules

\begin{align*}
  b &\rightarrow a & (46) \\
  a &\rightarrow b & (47)
\end{align*}

Its associated Horn theory $\overline{\mathcal{R}}$ is:

\begin{align*}
  (\forall x) \; x \rightarrow^* x & \quad (48) \\
  (\forall x, y, z) \; x \rightarrow y \land y \rightarrow^* z &\Rightarrow x \rightarrow^* z & (49) \\
  b &\rightarrow a & (50) \\
  c &\rightarrow^* b &\Rightarrow a &\rightarrow b & (51)
\end{align*}

We consider some simple tests regarding goals $b \rightarrow a$ and $a \rightarrow b$ and their negations. We tried such four goals with the following theorem provers: Alt-Ergo$^3$, Prover9/Mace4$^5$, and Princess$^6$ (most of them with a web-interface). Besides attempting a proof of each goal with respect to $\mathcal{R}$, tools Alt-Ergo, Mace4, and Princess can also generate models of the negation of the tested goal when the proof attempt fails. The following table summarizes the results of our test:

| #  | Goal          | Alt-Ergo | Mace4 | PDL-tableau | Princess |
|----|---------------|----------|-------|-------------|----------|
| 1  | $b \rightarrow a$ | true     | N     | N           | Y        |
| 2  | $\neg(b \rightarrow a)$ | false    | Y     | N           | Y        |
| 3  | $a \rightarrow b$ | false    | N     | Y           | N        |
| 4  | $\neg(a \rightarrow b)$ | true     | N     | Y           | N        |

Goal $\neg(a \rightarrow b)$ in row 4 is not directly proved by any tool. Indeed, since $\neg(a \rightarrow b)$ is not a logical consequence of $\overline{\mathcal{R}}$ (see Example 2), $\overline{\mathcal{R}} \vdash \neg(a \rightarrow b)$ does not hold. Our satisfiability approach can be used to formally prove that $\mathcal{R}$ cannot reduce $a$ into $b$, i.e., that $I_{\mathcal{R}} \models \neg(a \rightarrow b)$ (or $a \not\rightarrow_{\mathcal{R}} b$) holds: from row 3 we see that $A \models \neg(a \rightarrow b)$ holds for the models $A$ of $\overline{\mathcal{R}}$ computed by some of the tools. By Corollary 1, the desired conclusion $a \not\rightarrow_{\mathcal{R}} b$ follows. Note also that row 4 reports on the ability of some tools to obtain models of $a \rightarrow b$. However, Corollary 1 cannot be used to conclude that $a \rightarrow_{\mathcal{R}} b$ holds (which is obviously wrong): since $\varphi$ in row 4 is a negative literal, condition (b) in Corollary 1 must be fulfilled before being able to conclude $I_{\mathcal{R}} \models a \rightarrow b$ from $A \models a \rightarrow b$ for some model $A$ of $\overline{\mathcal{R}}$. But this is not the case in our test set.

Although Remark 1 explains how an arbitrary program property $\varphi$ can be proved by using Corollary 1 (see also Section 7), from a practical point of view we better think of our approach as complementary to the use of first-order proof techniques and tools. Provability of $\varphi$ (i.e., $S \vdash \varphi$) implies that $I_S \models \varphi$ holds. Thus, as usual, a proof of $\varphi$ with respect to $\mathcal{S}$ implies that a program $P$ with $S = \mathcal{P}$ has property $\varphi$. However, as discussed above, showing that $S \vdash \varphi$ or $S \vdash \neg \varphi$ holds is often impossible. We can try to prove $I_S \models \neg \varphi$ by using Corollary 1 though. For positive sentences $\varphi$, this is often affordable.

\footnotesize
\begin{itemize}
  \item [3] https://alt-ergo.ocamlpro.com/
  \item [5] http://www.cs.man.ac.uk/~schmidt/pdl-tableau/
  \item [6] http://www.philipp.ruemmer.org/princess.shtml
\end{itemize}
9 Conclusions and Future Work

We have shown how to prove properties $\varphi$ of computational systems whose semantics can be given as a first-order theory $S$. Our proofs by satisfiability proceed (see Remark 1) by just finding a model $A$ of $S \cup Z \cup \{\varphi\}$ where $Z$ is an auxiliary theory representing the requirements (a) and (b) in Corollary 1 (referred to $\neg \varphi$), so that $A \models S \cup Z \cup \{\varphi\}$ implies $I_S \models \varphi$. Surjectivity of the interpretation homomorphisms (requirement (a) in Corollary 1) is ensured if $Z$ includes the appropriate theory $SuH$ (see Section 4); and requirement (b), for dealing with negative literals, is fulfilled if $Z$ includes $N$ in Proposition 1. Our results properly subsume the ones in [14], which concern existentially closed boolean combinations of atoms only. We have also introduced the notion of refutation witness which is useful to obtain counterexamples by using the symbols in the first-order language rather than values of the computed model.

From a theoretical point of view, the idea of proving program properties as satisfiability (see Remark 1) is appealing as it emphasizes the role of abstraction (introduced by semantic structures) in theorem proving and logic-based program analysis. However, the requirement of surjectivity of the interpretation homomorphisms and the use of theories $N$ with negative information about some of the predicates introduce additional difficulties in the model generation process. Investigating methods for the practical implementation of our techniques, and also finding specific areas of application where our approach can be useful (as done in [16], for instance) is an interesting subject for future work.

Also, our research suggests that further investigation on the generation of models for many-sorted theories that combines the use of finite and infinite domains is necessary. For instance, [15] explains how to generate such models by interpreting the sort, function, and predicate symbols by using linear algebra techniques. This is implemented in AGES. Domains are defined as the solutions of matrix inequalities, possibly restricted to an underlying set of values (e.g., $\mathbb{Z}$); thus, finite and infinite domains can be obtained as particular cases of the same technique. Since piecewise definitions are allowed, we could eventually provide fully detailed descriptions of functions and predicates by just adding more pieces to the interpretations. However, such a flexibility is expensive. In contrast, Mace4 is based on a different principle (similar to [12]) and it is really fast, but only finite domains can be generated. This is a problem, for instance, when using Proposition 5 to guarantee surjectivity of homomorphisms $h_s : T_{\Sigma_s} \rightarrow A_s$. Even though $A_s$ is finite, we still need to be able to interpret $\text{Nat}$ as $\mathbb{N}$, which is not possible with Mace4. For this reason, the examples in Section 5 (where the computed structures $A$ have finite domains for the ‘proper’ sorts $\mathbb{N}$, $\mathbb{L}$, and $\mathbb{S}$, and only $\text{Nat}$ is interpreted as an infinite set) could not be handled with Mace4, or with similar tools that are able to deal with sorts (e.g., SEM [24] or the work in [21]) but which generate finite domains only.

Acknowledgements. I thank the anonymous referees for their comments and suggestions. I also thank Philipp Rümmer and Mohamed Iguernlala for their clarifying remarks about the use of Princess and Alt-Ergo, respectively.
References

1. F. Baader and T. Nipkow. Term Rewriting and All That. Cambridge University Press, 1998.
2. G.S. Boolos, J.P. Burgess, and R.C. Jeffrey. Computability and Logic, fourth edition. Cambridge University Press, 2002.
3. R. Bruni and J. Meseguer. Semantic foundations for generalized rewrite theories. *Theoretical Computer Science* 351(1):386-414, 2006.
4. K.L. Clark. Predicate Logic as a Computational Formalism. PhD. Thesis, Research Monograph 79/59 TOC, Department of Computing, Imperial College of Science, and Technology, University of London, December 1979.
5. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. All About Maude – A High-Performance Logical Framework. LNCS 4350, Springer-Verlag, 2007.
6. M. Dauchet and S. Tison. The Theory of Ground Rewrite Systems is Decidable. In *Proc. of LICS’90*, pages 242-248, IEEE Press, 1990.
7. N. Dershowitz, S. Kaplan, and D. Plaisted. Rewrite, rewrite, rewrite, rewrite, rewrite, . . . *Theoretical Computer Science* 83:71-96, 1991.
8. J. Endrullis and D. Hendriks. Lazy productivity via termination. *Theoretical Computer Science* 412:3203-3225, 2011.
9. J. Goguen and J. Meseguer. Models and Equality for Logical Programming. In *Proc. of TAPSOFT’87*, LNCS 250:1-22, Springer-Verlag, 1987.
10. R. Gutiérrez, S. Lucas, and P. Reinoso. A tool for the automatic generation of logical models of order-sorted first-order theories. In *Proc. of PROLE’16*, pages 215-230, 2016. Tool available at [http://zenon.dsic.upv.es/ages/](http://zenon.dsic.upv.es/ages/).
11. W. Hodges. Model Theory. Cambridge University Press, 1993.
12. S. Kim and H. Zhang. ModGen: Theorem Proving by Model Generation. In *Proc. of AAAI’94*, pages 162-167, AAAI Press/MIT Press, 1994.
13. S.C. Kleene. Mathematical Logic. John Wiley & Sons, 1967 (Dover, 2002).
14. S. Lucas. Analysis of Rewriting-Based Systems as First-Order Theories. In *Revised Selected papers from LOPSTR 2017*, LNCS 10855:180-197, 2018.
15. S. Lucas and R. Gutiérrez. Automatic Synthesis of Logical Models for Order-Sorted First-Order Theories. *Journal of Automated Reasoning* 60(4):465–501, 2018.
16. S. Lucas and R. Gutiérrez. Use of logical models for proving infeasibility in term rewriting. *Information Processing Letters*, 136:90-95, 2018.
17. W. McCune Prover9 and Mace4. [http://www.cs.unm.edu/~mccune/prover9/](http://www.cs.unm.edu/~mccune/prover9/), 2005–2010.
18. J. Meseguer. Twenty years of rewriting logic. *Journal of Logic and Algebraic Programming* 81:721-781, 2012.
19. E. Ohlebusch. *Advanced Topics in Term Rewriting*. Springer-Verlag, Apr. 2002.
20. R. Reiter. On Closed World Data Bases. In Logic and Data Bases, pages 119–140, Plenum Press, 1978.
21. G. Reger, M. Suda, and A. Voronkov. Finding Finite Models in Multi-sorted First-Order Logic. In *Proc. of SAT 2016*, LNCS 9710:323-341, 2016.
22. S. Shapiro. Foundations without Foundationalism: A Case for Second-Order Logic. Clarendon Press, 1991.
23. H. Wang. Logic of many-sorted theories. *Journal of Symbolic Logic* 17(2):105-116, 1952.
24. J. Zhang and H. Zhang. Generating Models by SEM (System Description). In *Proc. of CADE’96*, LNCS 1104:308-312, 1996.