Use of Logical Models for Proving Infeasibility in Term Rewriting[☆]

Salvador Lucas[a], Raúl Gutiérrez[a]

[a] *DSIC, Universitat Politècnica de València*

Abstract

Given a (Conditional) Rewrite System $\mathcal{R}$ and terms s and t, we consider the following problem: is there a substitution σ instantiating the variables in s and t such that the *reachability test* $\sigma(s) \to_{\mathcal{R}}^* \sigma(t)$ succeeds? If such a substitution does *not* exist, we say that the problem is *infeasible*; otherwise, we call it *feasible*. Similarly, we can consider *reducibility*, involving a *single* rewriting step. In term rewriting, a number of important problems involve such *infeasibility tests* (e.g., *confluence* and *termination* analysis). We show how to recast infeasibility tests into the problem of *finding a model* of a set of (first-order) sentences representing the operational semantics of $\mathcal{R}$ together with some additional sentences representing the considered property which is formulated as an infeasibility test.

Keywords: Conditional rewriting, confluence, dependency graph, operational termination.

1. Introduction

Conditional Term Rewriting Systems (CTRSs) [19, Section 7] consist of rules $\ell \to r \Leftarrow c$, where the *conditional part* c is a (possibly empty) sequence $s_1 \approx t_n, \ldots, s_n \approx t_n$ of *conditions* whose *satisfaction* is required before being allowed to apply a rewriting step with ℓ and r in the usual way. Several *interpretations* of the satisfiability of conditions are possible [19, Definition 7.1.3]. For instance, dealing with *oriented* CTRSs, the evaluation of c with respect to a substitution σ consists of testing the instances of s_i and t_i for *reachability*, i.e., checking whether $\sigma(s_i)$ rewrites into $\sigma(t_i)$, written $\sigma(s_i) \to_{\mathcal{R}}^* \sigma(t_i)$, for all $1 \leq i \leq n$.

Given $n > 0$, a sequence $(s_i \bowtie_i t_i)_{i=1}^n$ of *rewriting goals* $s_i \bowtie_i t_i$, where s_i and t_i are terms, and $\bowtie_i$ are *predicate symbols* $\to$ or $\to^*$ is called a *feasibility sequence*. Such a sequence is $\mathcal{R}$-*feasible* if there is a substitution σ such that the instantiated goals $\sigma(s_i) \bowtie_i \sigma(t_i)$ are satisfied when $\bowtie_i$ is interpreted as the one-step or rewriting relations $\to_{\mathcal{R}}$ and $\to_{\mathcal{R}}^*$ for $\mathcal{R}$, respectively; otherwise, the sequence is called $\mathcal{R}$-*infeasible* (see Section 2 for a formal definition).

[☆] Partially supported by the EU (FEDER), Spanish MINECO project TIN2015-69175-C4-1-R and GV project PROMETEOII/2015/013.

Example 1. *Consider the CTRS $\mathcal{R}$ [19, Example 7.2.45]:*

$$\mathsf{a} \to \mathsf{a} \quad \Leftarrow \quad \mathsf{b} \approx x, \mathsf{c} \approx x \qquad (1) \qquad\qquad \mathsf{c} \to \mathsf{d} \quad \Leftarrow \quad \mathsf{d} \approx x, \mathsf{e} \approx x \qquad (3)$$
$$\mathsf{b} \to \mathsf{d} \quad \Leftarrow \quad \mathsf{d} \approx x, \mathsf{e} \approx x \qquad (2)$$

where $\mathsf{a}, \ldots, \mathsf{e}$ *are* constants *and* x *is a* variable*. Since* d *and* e *are irreducible, the only way for* $\mathsf{d} \to^* x, \mathsf{e} \to^* x$ *to be* $\mathcal{R}$*-feasible is instantiating* x *to both* d *and* e *at the same time, which is* not *possible. Thus,* (2) *and* (3) cannot *be used in any rewriting step. They are called* infeasible *rules and may be removed (without changing the induced rewrite relation). Actually, (1) is infeasible too.*

$\mathcal{R}$*-infeasibility* can be used to (i) disable the use of *conditional rules* in reductions or even *remove* them,[1] (ii) discard *conditional dependency pairs* $u \to v \Leftarrow c$ in the analysis of *operational termination* of CTRSs [12], (iii) discard *conditional critical pairs* $u \downarrow v \Leftarrow c$ that arise in the analysis of *confluence* of CTRSs [19, 20, 21], (iv) prove *root-stability* of a term t (i.e., the absence of any rewriting sequence from t to an instance of a left-hand side ℓ of a rewrite rule $\ell \to r \Leftarrow c$) as the $\mathcal{R}$-infeasibility of $t \to^* \ell$ for each $\ell \to r \Leftarrow c \in \mathcal{R}$, (v) prove *irreducibility* of ground terms t (which is undecidable for CTRSs) as the $\mathcal{R}$-infeasibility of $t \to x$ for a variable x, (vi) prove the *non-joinability* of terms s and t as the $\mathcal{R}$-infeasibility of $s \to^* x, t \to^* x$ (with x not occurring in s or t), or (vii) discard *arcs* in the *dependency* graphs that are obtained during the analysis of termination using dependency pairs, see, e.g., [2] for TRSs and [11] for CTRSs.

In Section 3, we prove that $\mathcal{R}$-infeasibility problems can be translated into the problem of finding a *model* $\mathcal{A}$ of the set of sentences $\overline{\mathcal{R}}$ representing the operational semantics of the CTRS $\mathcal{R}$ plus a sentence $\neg(\exists \vec{x})\ \bigwedge_{i=1}^{n} s_i \bowtie_i t_i$ where all symbols (including $\to$ and $\to^*$ as predicate symbols) can be freely interpreted in a first-order structure $\mathcal{A}$. In Section 4 we show by means of examples how to apply our method to problems (i)-(vii). We assume familiarity with the basic notions, terminology and notations of (conditional) term rewriting (see, e.g., [3, 19] for TRSs and [19, Section 7] for CTRSs) and first-order logic [16].

The research in this paper was first presented in [8] (a restricted, non-systematic use in proofs of operational termination of CTRSs is sketched in [7, Section 11.1]) and then settled by the first author in a first-order logic framework in [6]. In this paper we have extended the treatment of [8] to more general properties of rewrite systems (e.g., reducibility or root-stability, see Section 4.3). Also, in contrast to [6], we show that focusing on CTRSs and term rewriting enables the use of specific refinements available for CTRSs only (e.g., usable rules, see Section 2). This allows us to deal with more applications and examples. Interestingly, our semantic approach together with the aforementioned improvements applies to *all* the examples solved in papers developing different specific techniques to deal with problems (i)-(vii) [1, 17, 20, 21].

[1] Sometimes, infeasible rules *cannot* be removed without changing relevant properties of a CTRS. For instance, $\mathcal{R} = \{\mathsf{b} \to \mathsf{c}, \mathsf{a} \to \mathsf{b} \Leftarrow \mathsf{a} \approx \mathsf{b}\}$ is *not* operationally terminating (see Section 4.1 below) due to the conditional rule, which is *infeasible*. However, after removing it, an *operationally terminating* CTRS is obtained.

$$\text{(R)}\quad \frac{}{x \to^* x} \qquad \text{(C)}\quad \frac{x_i \to y_i}{f(x_1,\ldots,x_i,\ldots,x_k) \to f(x_1,\ldots,y_i,\ldots,x_k)} \text{ for all } f \in \mathcal{F} \text{ and } 1 \le i \le k = arity(f)$$

$$\text{(T)}\quad \frac{x \to y \qquad y \to^* z}{x \to^* z} \qquad \text{(Rl)}\quad \frac{s_1 \to^* t_1 \quad \cdots \quad s_n \to^* t_n}{\ell \to r} \text{ for } \ell \to r \Leftarrow s_1 \approx t_1, \ldots, s_n \approx t_n \in \mathcal{R}$$

Figure 1: Inference rules for conditional rewriting with a CTRS $\mathcal{R}$ with signature $\mathcal{F}$

2. Infeasibility problems

Borrowing [19, Definition 7.1.8(3)] we introduce the following.

Definition 2. *Let $\mathcal{R}$ be a CTRS. A sequence $(s_i \bowtie_i t_i)_{i=1}^n$, where s_i and t_i are terms and $\bowtie_i \in \{\to, \to^*\}$ for all $1 \le i \le n$ is called a* feasibility sequence. *It is $\mathcal{R}$-*feasible *if there is a substitution σ such that, for all $1 \le i \le n$, $\sigma(s_i) \to_{\mathcal{R}} \sigma(t_i)$ if $\bowtie_i$ is $\to$, and $\sigma(s_i) \to_{\mathcal{R}}^* \sigma(t_i)$ if $\bowtie_i$ is $\to^*$. Otherwise, it is $\mathcal{R}$-*infeasible.

In Definition 2 we write $s \to_{\mathcal{R}}^* t$ or $s \to_{\mathcal{R}} t$ for terms s and t iff there is a proof tree for $s \to^* t$ (resp. $s \to t$) using $\mathcal{R}$ in the inference system of Figure 1 [9]. All rules in the inference system in Figure 1 are *schematic* in that each inference rule $\frac{B_1 \ \cdots \ B_n}{A}$ can be used under any *instance* $\frac{\sigma(B_1) \ \cdots \ \sigma(B_n)}{\sigma(A)}$ of the rule by a substitution σ. For instance, (Rl) actually establishes that, for every rule $\ell \to r \Leftarrow s_1 \approx t_1, \ldots, s_n \approx t_n$ in the CTRS $\mathcal{R}$, every instance $\sigma(\ell)$ by a substitution σ rewrites into $\sigma(r)$ provided that, for each $s_i \approx t_i$, with $1 \le i \le n$, the reachability condition $\sigma(s_i) \to^* \sigma(t_i)$ can be *proved*. We have the following:

Proposition 3. *Let $\mathcal{R}$ be a CTRS. A feasibility sequence $(s_i \bowtie_i t_i)_{i=1}^n$ is $\mathcal{R}$-*infeasible *if $(s_i \bowtie_i t_i)_{i \in I}$ is $\mathcal{R}$-*infeasible *for some $I \subseteq \{1,\ldots,n\}$.*

Aoto observes the following: for TRSs $\mathcal{R}$, the so-called *usable rules for reachability* associated to a term s, $\mathcal{U}(\mathcal{R}, s)$ [1, Definition 3], are the only ones we need in any rewriting sequence starting from s, i.e., $s \to_{\mathcal{R}}^* t$ iff $s \to_{\mathcal{U}(\mathcal{R},s)}^* t$ [1, Lemma 4]. This also holds for the usable rules $\mathcal{U}(\mathcal{R}, s)$ for CTRSs $\mathcal{R}$ and terms s in [13, Definition 11] that we introduce below. First let

$$RULES(\mathcal{R}, t) \quad = \quad \{\ell \to r \Leftarrow c \in R \mid \exists p \in \mathcal{P}os(t), root(\ell) = root(t|_p)\}$$

Note that $RULES(\mathcal{R}, t)$ contains the rules of $\mathcal{R}$ which are potentially applicable to the subterms in t. Now, the set of *usable rules* for a term t is:

$$\mathcal{U}(\mathcal{R}, t) = RULES(\mathcal{R}, t) \cup \bigcup_{l \to r \Leftarrow c \in RULES(\mathcal{R},t)} \left(\mathcal{U}(\mathcal{R}^\sharp, r) \cup \bigcup_{s \approx t \in c} \mathcal{U}(\mathcal{R}^\sharp, s) \right)$$

where $\mathcal{R}^\sharp = \mathcal{R} - RULES(\mathcal{R}, t)$. That is: we consider both unconditional and conditional rules and add the rules that could be *used* to evaluate the left-hand

sides and the conditions when trying to apply those rules to subterms of t. Given terms $s_1, \ldots, s_n$, let $\mathcal{U}(\mathcal{R}, s_1, \ldots, s_n) = \bigcup_{i=1}^{n} \mathcal{U}(\mathcal{R}, s_i)$. We have the following.

Proposition 4. *Let $\mathcal{R}$ be a CTRS. If $(s_i \bowtie_i t_i)_{i=1}^n$ is $\mathcal{U}(\mathcal{R}, s_1, \ldots, s_n)$-feasible, then it is $\mathcal{R}$-feasible. If terms s_i are* ground *for all $1 \leq i \leq n$ and the sequence is* $\mathcal{R}$-feasible*, then it is $\mathcal{U}(\mathcal{R}, s_1, \ldots, s_n)$-*feasible*.*

$\mathcal{R}$-infeasibility of $(s_i \bowtie_i t_i)_{i=1}^n$ can be proved as $\mathcal{U}(\mathcal{R}, s_1, \ldots, s_n)$-*infeasibility*, provided that all terms s_i are *ground*. This requirement cannot be dropped, in general. For $\mathcal{R} = \{\mathsf{a} \to \mathsf{b}\}$, the sequence $x \to^* \mathsf{a}, x \to^* \mathsf{b}$ is $\mathcal{R}$-feasible (just instantiate variable x to a). However, it is *not* $\mathcal{U}(\mathcal{R}, x, x)$-feasible because $\mathcal{U}(\mathcal{R}, x, x)$ contains no rule.

3. Use of logical models for proving infeasibility

In the logic of CTRSs, with binary *predicate symbols* $\to$ (for one-step rewriting) and $\to^*$ (for many-step rewriting), the first-order theory $\overline{\mathcal{R}}$ for $\mathcal{R} = (\mathcal{F}, R)$ is obtained from the inference rules in Figure 1 by *specializing* $(C)_{f,i}$ for each $f \in \mathcal{F}$ and i, $1 \leq i \leq ar(f)$ and $(Rl)_\rho$ for all $\rho : \ell \to r \Leftarrow c \in R$. Inference rules $\frac{B_1 \;\cdots\; B_n}{A}$ become *sentences* $(\forall x_1, \ldots, x_m)\, B_1 \wedge \cdots \wedge B_n \Rightarrow A$, where $\{x_1, \ldots, x_m\}$ is the (possibly empty) set of variables occurring in the atoms $B_1, \ldots, B_n$ and A [10, Section 2]. If such a set is empty, we write $B_1 \wedge \cdots \wedge B_n \Rightarrow A$.

Example 5. *For $\mathcal{R}$ in Example 1, the associated theory $\overline{\mathcal{R}}$ is:*

$$\forall x\ (x \to^* x) \quad (4)$$

$$\forall x, y, z\ (x \to y \wedge y \to^* z \Rightarrow x \to^* z) \quad (5)$$

$$\forall x\ (\mathsf{b} \to^* x \wedge \mathsf{c} \to^* x \Rightarrow \mathsf{a} \to \mathsf{a}) \quad (6)$$

$$\forall x\ (\mathsf{d} \to^* x \wedge \mathsf{e} \to^* x \Rightarrow \mathsf{b} \to \mathsf{d}) \quad (7)$$

$$\forall x\ (\mathsf{d} \to^* x \wedge \mathsf{e} \to^* x \Rightarrow \mathsf{c} \to \mathsf{d}) \quad (8)$$

Deductions with $\overline{\mathcal{R}}$ proceed *à la Hilbert* by using *modus ponens* and *generalization* as inference rules, the usual set of *logical axioms*, and $\overline{\mathcal{R}}$ as the set of *proper* axioms [16, Section 2.3]. In this way, we can also prove goals $s \to t$ and $s \to^* t$, written $\overline{\mathcal{R}} \vdash s \to t$ and $\overline{\mathcal{R}} \vdash s \to^* t$, respectively. For all terms s and t, we have (i) $s \to_{\mathcal{R}} t$ iff $\overline{\mathcal{R}} \vdash s \to t$ and (ii) $s \to^*_{\mathcal{R}} t$ iff $\overline{\mathcal{R}} \vdash s \to^* t$.

By a *structure* $\mathcal{A}$ for a first-order language we mean an interpretation of the function and predicate symbols ($f, g, \ldots$ and $P, Q, \ldots$, respectively) as mappings $f^{\mathcal{A}}, g^{\mathcal{A}}, \ldots$ and relations $P^{\mathcal{A}}, Q^{\mathcal{A}}, \ldots$ on a given set (carrier) also denoted $\mathcal{A}$. Then, the usual interpretation of first-order formulas with respect to the structure is considered. A model for a set $\mathcal{S}$ of first-order sentences (i.e., formulas whose variables are all *quantified*) is just a structure that makes them all true, written $\mathcal{A} \models \mathcal{S}$. The *Herbrand model* $\mathcal{H}_{\mathcal{R}}$ for a CTRS $\mathcal{R}$ (or just $\mathcal{H}$ if no confusion arises) consists of the usual Herbrand interpretation with $\mathcal{T}(\mathcal{F})$ as domain, constants interpreted as themselves, i.e., $a^{\mathcal{H}} = a$ for all $s \in \mathcal{F}$ with $ar(a) = 0$, k-ary functions $f \in \mathcal{F}$ interpreted by $f^{\mathcal{H}}(t_1, \ldots, t_k) = f(t_1, \ldots, t_k)$ for all $t_1, \ldots, t_k \in \mathcal{T}(\mathcal{F})$ and $\to^{\mathcal{H}}$ and $(\to^*)^{\mathcal{H}}$ given by $s \to^{\mathcal{H}} t$ iff $s \to_{\mathcal{R}} t$ and $s (\to^*)^{\mathcal{H}} t$ iff $s \to^*_{\mathcal{R}} t$ for all $s, t \in \mathcal{T}(\mathcal{F})$. By *correctness* of the first-order

predicate calculus, for all models $\mathcal{A}$ of a theory $\mathcal{S}$, whenever $\mathcal{S} \vdash \varphi$ holds for a sentence φ, we have $\mathcal{A} \models \varphi$. We use these facts in the following result.

Theorem 6. *Let $\mathcal{R}$ be a CTRS, $(s_i \bowtie_i t_i)_{i=1}^n$ be a feasibility sequence, and $\mathcal{A}$ be a structure with nonempty domain. If $\mathcal{A} \models \overline{\mathcal{R}} \cup \{\neg(\exists \vec{x}) \bigwedge_{i=1}^n s_i \bowtie_i t_i\}$, then the sequence is $\mathcal{R}$-infeasible. If $\mathcal{T}(\mathcal{F}) \neq \emptyset$, then $\mathcal{R}$-infeasibility implies that $\mathcal{H}_\mathcal{R} \models \overline{\mathcal{R}} \cup \{\neg(\exists \vec{x}) \bigwedge_{i=1}^n s_i \bowtie_i t_i\}$ holds.*

Proof. We prove the first claim by contradiction. If $(s_i \bowtie_i t_i)_{i=1}^n$ is $\mathcal{R}$-feasible, then there is a substitution σ such that, for all $1 \leq i \leq n$, $\sigma(s_i) \to_\mathcal{R} \sigma(t_i)$ if $\bowtie_i$ is $\to$ and $\sigma(s_i) \to_\mathcal{R}^* \sigma(t_i)$ if $\bowtie_i$ is $\to^*$. Since $\mathcal{A}$ is a model of $\overline{\mathcal{R}}$, by *correctness* we have $\mathcal{A} \models (\forall \vec{y}_i)\ \sigma(s_i) \bowtie_i \sigma(t_i)$ for all $1 \leq i \leq n$, with $\vec{y}_i$ the variables in $\mathcal{V}ar(\sigma(s_i)) \cup \mathcal{V}ar(\sigma(t_i))$. Thus, $\mathcal{A} \models (\forall \vec{y})\ (\bigwedge_{i=1}^n \sigma(s_i) \bowtie_i \sigma(t_i))$, with $\vec{y}$ the variables in $\bigcup_{i=1}^n \mathcal{V}ar(\sigma(s_i)) \cup \mathcal{V}ar(\sigma(t_i))$. Hence, for all $\nu : \vec{y} \to \mathcal{A}$, the interpretation in $\mathcal{A}$ of the universally quantified formula above, i.e., $[\bigwedge_{i=1}^n \sigma(s_i) \bowtie_i \sigma(t_i)]_\nu^\mathcal{A}$, is *true*. Let $\vec{x}$ be the variables in $\bigcup_{i=1}^n \mathcal{V}ar(s_i) \cup \mathcal{V}ar(t_i)$. Since the domain of $\mathcal{A}$ is not empty, given an arbitrary valuation $\nu : \vec{y} \to \mathcal{A}$, there is $\nu' : \vec{x} \to \mathcal{A}$ given by $\nu'(x) = [\sigma(x)]_\nu^\mathcal{A}$ for all variables x in $\vec{x}$, such that $[\bigwedge_{i=1}^n s_i \bowtie_i t_i]_{\nu'}^\mathcal{A}$ is true. This contradicts the assumption $\mathcal{A} \models \neg(\exists \vec{x})(\bigwedge_{i=1}^n s_i \bowtie_i t_i)$. For the second claim, if $(s_i \bowtie_i t_i)_{i=1}^n$ is $\mathcal{R}$-infeasible, then for all ground substitutions σ (which exist due to the assumption $\mathcal{T}(\mathcal{F}) = \emptyset$) there is i, $1 \leq i \leq n$ such that $\sigma(s_i) \bowtie \sigma(t_i)$ does not hold. By definition of $\mathcal{H}_\mathcal{R}$, and since valuations in $\mathcal{H}_\mathcal{R}$ are just ground substitutions (and vice versa), we have that $\mathcal{H}_\mathcal{R} \models \overline{\mathcal{R}} \cup \{\neg(\exists \vec{x}) \bigwedge_{i=1}^n s_i \bowtie_i t_i\}$ holds. □

In order to guarantee that $\mathcal{A}$ in Theorem 6 has a nonempty domain, we just add $(\exists x)\ x = x$ to $\overline{\mathcal{R}} \cup \{\neg\varphi\}$ (this is not necessary if $\mathcal{T}(\mathcal{F}) \neq \emptyset$). Structures $\mathcal{A}$ can be automatically generated from $\overline{\mathcal{R}}$ and sentence(s) $\neg(\exists \vec{x})(\bigwedge_{i=1}^n s_i \bowtie_i t_i)$ by using tools like AGES [5], implementing the methods in [7]. Essentially, AGES interprets the sort, function and predicate symbols of an input (many-sorted) first-order theory as *parametric* domains, functions and predicates. Sentences in the theory are then transformed into *constraints* over such parameters which are then solved by using standard constraint solving methods and tools (based on SAT, SMT, etc.). We also use Mace4 [15], which only computes *finite* models. Details about the use of AGES and Mace4 to solve the examples in the paper can be found in Appendix A.

Example 7. *For $\mathcal{R}$ in Example 1 and $\overline{\mathcal{R}}$ in Example 5,* AGES *computes a structure $\mathcal{A}$ with domain $\mathbb{N} \cup \{-1\}$ where $\mathsf{a}^\mathcal{A} = \mathsf{b}^\mathcal{A} = 0$, $\mathsf{c}^\mathcal{A} = \mathsf{d}^\mathcal{A} = -1$, $\mathsf{e}^\mathcal{A} = 1$, $x \to^\mathcal{A} y$ iff $x = y = -1$ and $x\ (\to^*)^\mathcal{A}\ y$ iff $x = y$. This is a model of $\overline{\mathcal{R}} \cup \{\neg(\exists x)\ (\mathsf{b} \to^* x \wedge \mathsf{c} \to^* x), \neg(\exists x)\ (\mathsf{d} \to^* x \wedge \mathsf{e} \to^* x)\}$. By Theorem 6, both $\mathsf{b} \to^* x, \mathsf{c} \to^* x$ and $\mathsf{d} \to^* x, \mathsf{e} \to^* x$ are $\mathcal{R}$-infeasible.*

4. Applications

In this section we explore the use of Theorem 6 in the application areas (i)–(vii) in the introduction. We mostly consider examples from the literature.

4.1. Infeasible rules and conditional dependency pairs

Rules $\ell \to r \Leftarrow c$ with a conditional part c whose associated sequence $s_1 \to^* t_1, \ldots, s_n \to^* t_n$ is $\mathcal{R}$-infeasible are called *infeasible rules*. Example 7 shows that *all rules* of $\mathcal{R}$ in Example 1 are *infeasible*.

A CTRS $\mathcal{R}$ is *operationally terminating* iff no term t has an infinite proof tree using the inference system in Figure 1 [9]. As for TRSs [2], the dependency pair approach provides a suitable way to mechanize proofs of operational termination of CTRSs [12, 14]. Dependency pairs are just conditional rules $u \to v \Leftarrow c$ which are obtained from the CTRS $\mathcal{R}$. Proving infeasibility of such rules is also useful to prove operational termination of CTRSs, see [14].

4.2. Infeasible critical pairs

A conditional critical pair (CCP) $\langle s, t\rangle \Leftarrow \sigma(c), \sigma(c')$ is obtained from (renamed versions of) rules $\ell \to r \Leftarrow c$ and $\ell' \to r' \Leftarrow c'$ iff there is a nonvariable subterm t of ℓ such that $\sigma(\ell) = \sigma(t)$ for a most general unifier σ [19, Definition 7.1.8]. A critical pair $\langle s, t\rangle \Leftarrow s_1 \approx t_1, \ldots, s_n \approx t_n$ is infeasible if $(s_i \to^* t_i)_{i=1}^n$ is $\mathcal{R}$-infeasible. Joinability of feasible CCPs can be used to prove confluence of CTRSs [19, Section 7.3]. Infeasible CCPs can be dismissed, though.

Example 8. *The following CTRS $\mathcal{R}$ [21, Example 23]:*

$$\mathsf{g}(x) \to \mathsf{f}(x,x) \quad (9) \qquad \mathsf{g}(x) \to \mathsf{g}(x) \Leftarrow \mathsf{g}(x) \approx \mathsf{f}(\mathsf{a},\mathsf{b}) \quad (10)$$

has a conditional critical pair $\langle \mathsf{f}(x,x), \mathsf{g}(x)\rangle \Leftarrow \mathsf{g}(x) \approx \mathsf{f}(\mathsf{a},\mathsf{b})$. With AGES*, we obtain a model of $\overline{\mathcal{R}} \cup \{\neg(\exists x)\ \mathsf{g}(x) \to^* \mathsf{f}(\mathsf{a},\mathsf{b})\}$ which proves infeasibility of the conditional critical pair. In [21, Example 23] this is proved by using unification tests together with a transformation. It is discussed that the alternative tree automata techniques investigated in the paper do* not *work for this example.*

4.3. Root-stability and irreducibility

A term t is *root-stable* (with respect to a TRS $\mathcal{R}$) if t cannot be reduced to a redex, i.e., there is no rule $\ell \to r \in \mathcal{R}$ such that $t \to^* \sigma(\ell)$ for some substitution σ [18]. Root-stability is important in the semantic description of lazy languages where *infinite normal forms* are allowed and approximated as sequences of root-stable terms [18]. Clearly, we can prove a *ground* term t root-stable if we (often independently) show $\mathcal{R}$-infeasibility of $t \to^* \ell$ for all $\ell \to r \in \mathcal{R}$.

Example 9. *Consider the following TRS $\mathcal{R}$:*

$$\mathsf{a} \to \mathsf{b} \quad (11) \qquad \mathsf{b} \to \mathsf{a} \quad (12) \qquad \mathsf{f}(x,x) \to \mathsf{c} \quad (13)$$

We prove root-stability of $\mathsf{f}(\mathsf{a},\mathsf{c})$ by showing $\mathcal{R}$-infeasibility of $\mathsf{f}(\mathsf{a},\mathsf{c}) \to^* \mathsf{f}(x,x)$, $\mathsf{f}(\mathsf{a},\mathsf{c}) \to^* \mathsf{a}$, and $\mathsf{f}(\mathsf{a},\mathsf{c}) \to^* \mathsf{b}$. With Mace4 *we obtain a model of*

$$\overline{\mathcal{R}} \cup \{\neg(\exists x)\ \mathsf{f}(\mathsf{a},\mathsf{c}) \to^* \mathsf{f}(x,x), \neg(\mathsf{f}(\mathsf{a},\mathsf{c}) \to^* \mathsf{a}), \neg(\mathsf{f}(\mathsf{a},\mathsf{c}) \to^* \mathsf{b})\}$$

A term t is irreducible if no rewriting step is possible on it. For ground terms, we can prove irreducibility of t as the $\mathcal{R}$-infeasibility of $(\exists x)\, t \to x$.

Example 10. *Consider the following CTRS $\mathcal{R}$ [13, Example 3]*

$$\begin{array}{rcll} \mathsf{g}(\mathsf{a}) & \to & \mathsf{c}(\mathsf{b}) & (14) \\ \mathsf{b} & \to & \mathsf{f}(\mathsf{a}) & (15) \end{array} \qquad \mathsf{f}(x) \to x \Leftarrow \mathsf{g}(x) \to \mathsf{c}(y) \quad (16)$$

Mace4 *obtains a model of $\overline{\mathcal{R}} \cup \{\neg(\exists x)\ \mathsf{f}(\mathsf{c}(\mathsf{a})) \to x\}$, i.e., $\mathsf{f}(\mathsf{c}(\mathsf{a}))$ is irreducible.*

4.4. Non-joinable terms

We can see the usual joinability problem $s \downarrow t$, i.e., the existence of a term u such that $s \to_{\mathcal{R}}^* u$ and $t \to_{\mathcal{R}}^* u$, as a specific feasibility sequence.

Theorem 11. *Let $\mathcal{R}$ be a CTRS, s, t be terms, and x be a fresh variable not occurring in s or t. If s and t are joinable, then $s \to^* x, t \to^* x$ is $\mathcal{R}$-feasible. If s and t are* ground *and $s \to^* x, t \to^* x$ is $\mathcal{R}$-feasible, then s and t are joinable.*

Proof. If s and t are joinable, then $s \to_{\mathcal{R}}^* u$ and $t \to_{\mathcal{R}}^* u$ for some term u. Hence, $s \to^* x, t \to^* x$ is $\mathcal{R}$-feasible (use σ such that $\sigma(x) = u$ and $\sigma(y) = y$ for any other variable $y \neq x$). On the other hand, if s and t are ground and $s \to^* x, t \to^* x$ is $\mathcal{R}$-feasible, then there is a substitution σ such that $\sigma(s) = s \to_{\mathcal{R}}^* u = \sigma(x)$ and $\sigma(t) = t \to_{\mathcal{R}}^* u = \sigma(x)$, i.e., s and t are joinable. □

By Theorem 11, $\mathcal{R}$-infeasibility of $s \to^* x, t \to^* x$ implies non-joinability of s and t (where s and t may contain variables).

Example 12. *Consider the following CTRS $\mathcal{R}$ [19, Example 7.3.3]:*

$$\mathsf{a} \to \mathsf{b} \quad (17) \qquad \mathsf{f}(x) \to \mathsf{c} \Leftarrow x \approx \mathsf{a} \quad (18)$$

There is no CCP but $\mathcal{R}$ is not *confluent: $\mathsf{f}(\mathsf{a}) \to_{\mathcal{R}} \mathsf{f}(\mathsf{b})$ and $\mathsf{f}(\mathsf{a}) \to_{\mathcal{R}} \mathsf{c}$ but c and $\mathsf{f}(\mathsf{b})$ are* not *joinable. With* AGES *we obtain a model of $\overline{\mathcal{U}(\mathcal{R}, \mathsf{f}(\mathsf{b}), \mathsf{c})} \cup \{\neg(\exists x)\,(\mathsf{f}(\mathsf{b}) \to^* x \wedge \mathsf{c} \to^* x)\}$, where $\mathcal{U}(\mathcal{R}, \mathsf{f}(\mathsf{b}), \mathsf{c}) = \{(18)\}$ (see Proposition 4). Therefore, $\mathsf{f}(\mathsf{b})$ and c are proved non-joinable.*

4.5. Approximation of dependency graphs

In the dependency pair approach for termination analysis of TRSs $\mathcal{R}$, the notion of *chain* of dependency pairs is paramount [2]: a sequence of rules $u_1 \to v_1, u_2 \to v_2, \ldots$ (called *dependency pairs*) is said to be a *chain* if there is a substitution σ such that $\sigma(v_i) \to_{\mathcal{R}}^* \sigma(u_{i+1})$ for all $i \geq 1$. Termination of TRSs is characterized as the absence of infinite chains of dependency pairs. For finite TRSs, infinite chains can always be represented as *cycles* in a *dependency graph* whose nodes are dependency pairs; there is an arc from a 'node' $u \to v$ to another node $u' \to v'$ (previously renamed, to share no variable with $u \to v$) if there is a substitution σ such that $\sigma(v) \to_{\mathcal{R}}^* \sigma(u')$. Thus, the $\mathcal{R}$-infeasibility of $v \to^* u'$ implies the *absence of an arc* from $u \to v$ to $u' \to v'$. The dependency graph is

not computable due to the undecidability of the involved reachability problems. Several overapproximations are available [2, 4]. However, avoiding 'false' arcs is essential to obtain more efficient termination proofs: we pay attention to dependency pairs in *cycles* of the graph only.

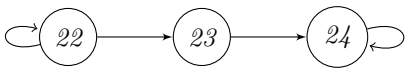
Example 13. *Consider the following TRS [17, Example 21]:*

$$\begin{array}{lcll} \mathsf{f}(x,\mathsf{a}) & \to & \mathsf{f}(x,\mathsf{g}(x,\mathsf{b})) & (19) \\ \mathsf{g}(\mathsf{a},y) & \to & y & (20) \end{array} \qquad \mathsf{g}(\mathsf{h}(x),y) \to \mathsf{g}(x,\mathsf{h}(y)) \quad (21)$$

There are three dependency pairs:

$$\begin{array}{lcll} \mathsf{F}(x,\mathsf{a}) & \to & \mathsf{F}(x,\mathsf{g}(x,\mathsf{b})) & (22) \\ \mathsf{F}(x,\mathsf{a}) & \to & \mathsf{G}(x,\mathsf{b}) & (23) \end{array} \qquad \mathsf{G}(\mathsf{h}(x),y) \to \mathsf{G}(x,\mathsf{h}(y)) \quad (24)$$

The estimated *graph which is obtained by using [4] (usually implemented in termination tools) is the following:*



Middeldorp discusses several refinements that use tree automata techniques for the approximation of the dependency graph. By using one of them (the so-called nv *approximation [17, Definition 8]) he obtains the graph above. By using a second one, he shows that the arcs from* (22) *to itself and from* (22) *to* (23) *can be removed safely, as they do not belong to the dependency graph (there is no substitution such that* $\sigma(\mathsf{F}(x,\mathsf{g}(x,\mathsf{b}))) \to_{\mathcal{R}}^{*} \sigma(\mathsf{F}(y,\mathsf{a}))$*, note the renaming introduced in the second term). With* AGES *we obtain a model of* $\overline{\mathcal{R}} \cup \{\neg(\exists x)\ \mathsf{F}(x,\mathsf{g}(x,\mathsf{b})) \to^{*} \mathsf{F}(y,\mathsf{a})\}$ *which shows that these two arcs do not belong to the dependency graph.*

5. Conclusions

We have presented a semantic approach to prove infeasibility in term rewriting: in order to prove the infeasibility of a sequence $(s_i \bowtie_i t_i)_{i=1}^{n}$ of rewriting goals, we try to *find a model* of a set $\overline{\mathcal{R}}$ of first-order sentences representing the operational semantics of $\mathcal{R}$ together with a sentence $\neg(\exists \vec{x}) \bigwedge_{i=1}^{n} s_i \bowtie_i t_i$.

We have shown the usefulness of this approach by an exhaustive consideration of examples from several recent papers addressing the role of infeasibility when applied to such problems. In particular, we could deal with all of Aoto's examples in [1] (regarding proofs of non-joinability of *ground* terms in TRSs), see Appendix B. In [17], Middeldorp uses different approximations to the estimation of the dependency graph for TRSs in the analysis of termination using dependency pairs. We were also able to deal with all examples reported in [17] (see Example 13 and Appendix C). We also dealt with all examples solved in [20] (see Appendix D), and [21] (see Example 8 and Appendix E). Note that these papers explore several *alternative* methods and, as reported by the authors, some of them *fail* in specific examples which require a different approach.

In contrast to all aforementioned works, we are using a single technique to deal with all kinds of reported problems. Of course, we do not claim to actually outperform them. However, in view of these good results, we believe that the technique is promising.

We do not have a dedicated, fully automated 'infeasibility' checker yet. Instead, we just encode the problem we want to deal with as a specific infeasibility sequence and then use AGES or Mace4 to apply Theorem 6. In AGES, the generation of a *single* model is completely automatic, but in order to *succeed* on a given problem, we may need to try different configurations for the generation of several models. In contrast to Mace4, whose generation scheme is simpler and where this is automatic, AGES currently requires *user assistance* to change settings like the dimension of the domains or the use of piecewise interpretations (like in Example 8), which are usually 'expensive' and not part of the *default* configuration. Thus, further investigation developing heuristics and *strategies* for an efficient use of the technique is envisaged as a subject for future work.

Acknowledgments. We thank the anonymous referees for many remarks and suggestions that led to improve the paper.

References

[1] T. Aoto. Disproving Confluence of Term Rewriting Systems by Interpretation and Ordering. In P. Fontaine, C. Ringeissen, and R.A. Schmidt, editors, *Proc. of the 9th International Symposium on Frontiers of Combining Systems, FroCoS'13*, LNCS 8152:311-326, 2013.

[2] T. Arts and J. Giesl. Termination of Term Rewriting Using Dependency Pairs. *Theoretical Computer Science*, 236(1–2):133–178, 2000.

[3] F. Baader and T. Nipkow. Term Rewriting and All That. Cambridge University Press, 1998.

[4] J. Giesl, R. Thiemann, and P. Schneider-Kamp. Proving and Disproving Termination of Higher-Order Functions. In B. Gramlich, editor, *Proc. of 5th International Workshop on Frontiers of Combining Systems, FroCoS'05*, LNAI 3717:216-231, 2005.

[5] R. Gutiérrez, S. Lucas, and P. Reinoso. A tool for the automatic generation of logical models of order-sorted first-order theories. In A. Villanueva, editor, *Proc. of the XVI Jornadas sobre Programación y Lenguajes, PROLE'16*, pages 215-230, 2016. `http://hdl.handle.net/11705/PROLE/2016/018`. Tool available at `http://zenon.dsic.upv.es/ages/`.

[6] S. Lucas. Analysis of Rewriting-Based Systems as First-Order Theories. In F. Fioravanti and J.P. Gallagher, editors, *Revised Selected papers from the 27th International Symposium on Logic-Based Program Synthesis and Transformation, LOPSTR 2017*, LNCS to appear, 2018.

[7] 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.

[8] S. Lucas and R. Gutiérrez. A Semantic Criterion For Proving Infeasibitlity In Conditional Rewriting. In B. Accatoli and B. Felgenhauer, editors, *Proc. of the 6th International Workshop on Confluence, IWC 2017*, pages 15-19, 2017.

[9] S. Lucas, C. Marché, and J. Meseguer. Operational termination of conditional term rewriting systems. *Information Processing Letters*, 95:446–453, 2005.

[10] S. Lucas and J. Meseguer. Models for Logics and Conditional Constraints in Automated Proofs of Termination. In G.A. Aranda-Corral, J. Calmet, and F.J. Martín-Mateos, editors, Proc. of *the 12th International Conference on Artificial Intelligence and Symbolic Computation, AISC'14*, LNAI 8884:9-20, 2014.

[11] S. Lucas and J. Meseguer. 2D Dependency Pairs for Proving Operational Termination of CTRSs. In S. Escobar, editor, *Proc. of the 10th International Workshop on Rewriting Logic and its Applications, WRLA'14*, LNCS 8663:195-212, 2014.

[12] S. Lucas and J. Meseguer. Dependency pairs for proving termination properties of conditional term rewriting systems. *Journal of Logical and Algebraic Methods in Programming*, 86:236-268, 2017.

[13] S. Lucas and J. Meseguer. Normal forms and normal theories in conditional rewriting. *Journal of Logical and Algebraic Methods in Programming*, 85(1):67-97, 2016.

[14] S. Lucas, J. Meseguer, and R. Gutiérrez. Extending the 2D DP Framework for Conditional Term Rewriting Systems. In M. Proietti and H. Seki, editors, *Selected papers of the 24th International Symposium on Logic-Based Program Synthesis and Transformation LOPSTR'14*, LNCS 8981:113-130, 2015.

[15] W. McCune Prover9 and Mace4. `http://www.cs.unm.edu/~mccune/prover9/`, 2005–2010.

[16] E. Mendelson. Introduction to Mathematical Logic. Fourth edition. Chapman & Hall, 1997.

[17] A. Middeldorp. Approximating Dependency Graphs Using Tree Automata Techniques. In R. Goré, A. Leitsch, and T. Nipkow, editors, *Proc. of the First International Joint Conference on Automated Reasoning, IJCAR'01*, LNAI 2083:593–610, 2001.

[18] A. Middeldorp. Call by Need Computations to Root-Stable Form. In *Proc. of the 24th ACM SIGPLAN-SIGACT symposium on Principles of Programming Languages, POPL'97*, pages 94–105. ACM, 1997.

[19] E. Ohlebusch. *Advanced Topics in Term Rewriting*. Springer-Verlag, Apr. 2002.

[20] T. Sternagel and A. Middeldorp. Infeasible Conditional Critical Pairs. In A. Tiwari and T. Aoto, editors, *Proc. of the 4th International Workshop on Confluence, IWC'15*, pages 13–18, 2014.

[21] C. Sternagel and T. Sternagel. Certifying Confluence Of Almost Orthogonal CTRSs Via Exact Tree Automata Completion. In D. Kesner and B. Pientka, editors, *Proc. of the 1st International Conference on Formal Structures for Computation and Deduction, FSCD'16*, LIPIcs 52, Article No. 85; pp. 85:1–85:16, 2016.

Appendix A. Generation of models for the examples in the paper with AGES and Mace4

Example 7: infeasible rules

AGES *specification.*

```
mod ExOhl02_7_2_45 is
  sort S .
  ops a b c d e : -> S .
  var x : S .
  crl a => a if b => x /\ c => x .
  crl b => d if d => x /\ e => x .
  crl c => d if d => x /\ e => x .
endm
```

AGES *goal.* The current version of AGES does *not* accept explicit quantification in goals; universal quantification is *assumed*. Thus, $\neg(\exists \vec{x})\ (s_1 \rightarrow^* t_1 \wedge \cdots \wedge s_n \rightarrow^* t_n)$ must be introduced as $\neg(s_1 \rightarrow^* t_1 \wedge \cdots \wedge s_n \rightarrow^* t_n)$, which actually corresponds to the equivalent universally quantified version $(\forall \vec{x})\ \neg(s_1 \rightarrow^* t_1 \wedge \cdots \wedge s_n \rightarrow^* t_n)$ of the original sentence.

```
~(b ->* x:S /\ c ->* x:S)
~(d ->* x:S /\ e ->* x:S)
```

AGES *output.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[a]| = 0
|[b]| = 0
|[c]| =  - 1
|[d]| =  - 1
|[e]| = 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((0 >= 2+2.x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{-1\}$. Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = \mathsf{b}^{\mathcal{A}} = 0 \qquad \mathsf{c}^{\mathcal{A}} = \mathsf{d}^{\mathcal{A}} = -1 \qquad \mathsf{e}^{\mathcal{A}} = 1$$
$$x \rightarrow^{\mathcal{A}} y \Leftrightarrow x = y = -1 \qquad x\ (\rightarrow^*)^{\mathcal{A}}\ y \Leftrightarrow x = y$$

Example 8: infeasible critical pairs

AGES *specification.*

```
mod Ex23_SS16 is
  sorts S .
  ops a b : -> S .
  op f : S S -> S [N=2].
  op g : S -> S .
  var x : S .
  rl g(x) => f(x,x) .
  crl g(x) => g(x) if g(x) => f(a,b) .
endm
```

AGES *goal.*

```
~(g(x:S) ->* f(a,b))
```

AGES *output.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[g(x_1_1:S)]| = 1 + x_1_1:S
|[a]| = 0
|[b]| =  - 1
|[f(x_1_1:S,x_2_1:S)]| =
  1 + x_1_1:S  if (x_2_1:S >= x_1_1:S)
- 1 + x_1_1:S  otherwise

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{-1\}$. Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = 0 \qquad \mathsf{b}^{\mathcal{A}} = -1 \qquad \mathsf{f}^{\mathcal{A}}(x,y) = \begin{cases} x+1 & \text{if } y \geq x \\ x-1 & \text{otherwise} \end{cases}$$

$$\mathsf{g}^{\mathcal{A}}(x) = x+1 \qquad x \to^{\mathcal{A}} y \Leftrightarrow x = y \qquad x\ (\to^*)^{\mathcal{A}}\ y \Leftrightarrow x = y$$

Example 9: root-stability

Mace4 *assumptions.*

```
reds(x,x).
(red(x,y) & reds(y,z)) -> reds(x,z).
red(x,y) -> red(f(x,z),f(y,z)).
red(x,y) -> red(f(z,x),f(z,y)).
red(f(x,x),c).
red(a,b).
red(b,a).
```

Mace4 *goal.*

```
exists x reds(f(a,c),f(x,x)).
reds(f(a,c),a).
reds(f(a,c),b).
```

Mace4 *output.*

```
a = 0.

b = 0.

c = 1.

f(0,0) = 2.
f(0,1) = 1.
f(0,2) = 1.
f(1,0) = 0.
f(1,1) = 2.
f(1,2) = 2.
f(2,0) = 0.
f(2,1) = 2.
f(2,2) = 2.

  red(0,0).
- red(0,1).
- red(0,2).
- red(1,0).
  red(1,1).
- red(1,2).
- red(2,0).
  red(2,1).
  red(2,2).

  reds(0,0).
- reds(0,1).
```

```
- reds(0,2).
- reds(1,0).
  reds(1,1).
- reds(1,2).
- reds(2,0).
  reds(2,1).
  reds(2,2).
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{0, 1, 2\}$. Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = \mathsf{b}^{\mathcal{A}} = 0 \qquad \mathsf{f}^{\mathcal{A}}(x, y) = \begin{cases} 2 & \text{if } x, y \geq 1 \vee x = y = 0 \\ 1 & \text{if } x = 0 \wedge y \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\mathsf{c}^{\mathcal{A}} = 1 \qquad x \to^{\mathcal{A}} y \Leftrightarrow x \, (\to^*)^{\mathcal{A}} \, y \Leftrightarrow x = y \vee (x = 2 \wedge y = 1)$$

Example 10: irreducibility

Mace4 *assumptions.*

```
reds(x,x).
(red(x,y) & reds(y,z)) -> reds(x,z).
red(x,y) -> red(c(x),c(y)).
red(x,y) -> red(f(x),f(y)).
red(x,y) -> red(g(x),g(y)).
red(g(a),c(b)).
red(b,f(a)).
reds(g(x),c(y)) -> red(f(x),x).
```

Mace4 *goal.*

```
exists x red(f(c(a)),x).
```

Mace4 *output.*

```
a = 0.

b = 1.

c(0) = 2.
c(1) = 1.
c(2) = 1.

f(0) = 1.
f(1) = 1.
f(2) = 0.

g(0) = 1.
g(1) = 1.
g(2) = 0.

- red(0,0).
- red(0,1).
- red(0,2).
  red(1,0).
  red(1,1).
  red(1,2).
- red(2,0).
- red(2,1).
- red(2,2).

  reds(0,0).
- reds(0,1).
- reds(0,2).
```

```
  reds(1,0).
  reds(1,1).
  reds(1,2).
- reds(2,0).
- reds(2,1).
  reds(2,2).
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{0, 1, 2\}$. Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = 0 \qquad \mathsf{c}^{\mathcal{A}}(x) = \begin{cases} 2 & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases} \qquad \mathsf{f}^{\mathcal{A}}(x) = \mathsf{g}^{\mathcal{A}}(x) = \begin{cases} 0 & \text{if } x = 2 \\ 1 & \text{otherwise} \end{cases}$$

$$\mathsf{b}^{\mathcal{A}} = 1 \qquad x \to^{\mathcal{A}} y \Leftrightarrow x = 1 \qquad x \, (\to^*)^{\mathcal{A}} \, y \Leftrightarrow x = y \vee x = 1$$

Example 12: non-joinable terms

AGES *specification.*

```
mod Ex7_3_3_Ohl02 is
  sort S .
  ops a b c : -> S .
  op f : S -> S .
  var x : S .
  crl f(x) => c if x => a .
endm
```

AGES *goal.*

```
~(f(b) ->* x:S /\ c ->* x:S)
```

AGES *output.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[f(x_1_1:S)]| = x_1_1:S
|[a]| = 0
|[b]| =  - 1
|[c]| = 0

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{-1\}$. Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = 0 \qquad \mathsf{b}^{\mathcal{A}} = -1 \qquad \mathsf{c}^{\mathcal{A}} = 0 \qquad \mathsf{f}^{\mathcal{A}}(x) = x$$
$$x \to^{\mathcal{A}} y \Leftrightarrow x = y \qquad x \,(\to^*)^{\mathcal{A}}\, y \Leftrightarrow x = y$$

Example 13: refined dependency graph

AGES *specification.*

```
mod Ex21_Mid01 is
  sort S .
  ops a b : -> S .
  ops f g : S S -> S .
  op h : S -> S .
  ops F : S S -> S .
  vars x y : S .
  rl g(a,y) => y .
  rl f(x,a) => f(x,g(x,b)) .
  rl g(h(x),y) => g(x,h(y)) .
endm
```

AGES *goal.*

```
~(F(x:S,g(x:S,b)) ->* F(y:S,a))
```

AGES *output.*

```
Domains:
S: |N \ {0}

Function Interpretations:
|[f(x_1_1:S,x_2_1:S)]| = x_1_1:S
|[g(x_1_1:S,x_2_1:S)]| = x_2_1:S
|[F(x_1_1:S,x_2_1:S)]| = x_2_1:S
|[a]| = 1
|[b]| = 2
|[h(x_1_1:S)]| = x_1_1:S

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> (x_2_1:S >= x_1_1:S)
 x_1_1:S -> x_2_1:S <=> (x_2_1:S >= x_1_1:S)
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} - \{0\}$. Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = 1 \qquad \mathsf{b}^{\mathcal{A}} = 2 \qquad \mathsf{f}^{\mathcal{A}}(x,y) = x \qquad \mathsf{g}^{\mathcal{A}}(x,y) = y$$
$$\mathsf{h}^{\mathcal{A}}(x) = x \qquad \mathsf{F}^{\mathcal{A}}(x,y) = y \qquad x \to^{\mathcal{A}} y \Leftrightarrow y \geq x \qquad x\,(\to^*)^{\mathcal{A}}\,y \Leftrightarrow y \geq x$$

Appendix B. Additional examples from [1]

Appendix B.1. Example 8 (non-joinability)
AGES *specification (complete).*

```
mod Ex8_Aoto13 is
  sort S .
  ops a c : -> S .
  ops f g h : S -> S .
  var x : S .
  rl a => h(c) .
  rl a => h(f(c)) .
  rl h(x) => h(h(x)) .
  rl f(x) => f(g(x)) .
endm
```

AGES *goal.*

```
~(h(c) ->* x:S /\ h(f(c)) ->* x:S)
```

We restrict the attention to the *usable rules* (Proposition 4):

AGES *specification.*

```
mod Ex8_Aoto13 is
  sort S .
  ops a c : -> S .
  ops f g h : S -> S .
  var x : S .
  rl h(x) => h(h(x)) .
  rl f(x) => f(g(x)) .
endm
```

AGES *output.*

```
Domains:
S: {0 , 1}

Function Interpretations:
|[f(x_1_1:S)]| = 1 - x_1_1:S
|[h(x_1_1:S)]| = x_1_1:S
|[a]| = 1
|[c]| = 0
|[g(x_1_1:S)]| = x_1_1:S

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{0, 1\}$. Symbols are interpreted as shown above.

Appendix B.2. Example 9 (non-joinability)
AGES *specification (complete).*

```
mod Ex9_Aoto13 is
  sort S .
  ops a c : -> S .
  ops f g h : S -> S .
  var x : S .
  rl a => f(c) .
  rl a => h(c) .
  rl f(x) => h(g(x)) .
  rl h(x) => f(g(x)) .
endm
```

AGES *goal.*

```
~(f(c) ->* x:S /\ h(c) ->* x:S)
```

We restrict the attention to the *usable rules* (Proposition 4):

AGES *specification.*

```
mod Ex9_Aoto13 is
  sort S .
  ops a c : -> S .
  ops f g h : S -> S .
  var x : S .
  rl f(x) => h(g(x)) .
  rl h(x) => f(g(x)) .
endm
```

AGES *output.*

```
Domains:
S: {-1 , 0}

Function Interpretations:
|[f(x_1_1:S)]| =  - 1 - x_1_1:S
|[h(x_1_1:S)]| = x_1_1:S
|[a]| = 0
|[c]| = 0
|[g(x_1_1:S)]| =  - 1 - x_1_1:S

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{-1, 0\}$. Symbols are interpreted as shown above.

Appendix B.3. Example 15 (non-joinability)

Mace4 *assumptions.*

```
reds(x,x).
(red(x,y) & reds(y,z)) -> reds(x,z).
red(x,y) -> red(f(x,z),f(y,z)).
red(x,y) -> red(f(z,x),f(z,y)).
red(x,y) -> red(g(x),g(y)).
red(x,y) -> red(h(x,z),h(y,z)).
red(x,y) -> red(h(z,x),h(z,y)).
red(c,f(c,d)).
red(c,h(c,d)).
red(f(x,y),h(g(y),x)).
red(h(x,y),f(g(y),x)).
```

Mace4 *goal.*

```
exists x (reds(h(f(c,d),d),x) & reds(f(c,d),x)).
```

Mace4 *output.*

```
c = 0.

d = 1.

g(0) = 0.
g(1) = 1.
g(2) = 2.

f(0,0) = 0.
f(0,1) = 2.
f(0,2) = 0.
f(1,0) = 0.
f(1,1) = 2.
f(1,2) = 1.
f(2,0) = 0.
f(2,1) = 2.
f(2,2) = 0.

h(0,0) = 0.
h(0,1) = 0.
h(0,2) = 0.
h(1,0) = 2.
h(1,1) = 2.
h(1,2) = 2.
h(2,0) = 0.
h(2,1) = 1.
```

```
h(2,2) = 0.

  red(0,0).
  red(0,1).
  red(0,2).
- red(1,0).
  red(1,1).
- red(1,2).
- red(2,0).
- red(2,1).
  red(2,2).

  reds(0,0).
  reds(0,1).
  reds(0,2).
- reds(1,0).
  reds(1,1).
- reds(1,2).
- reds(2,0).
- reds(2,1).
  reds(2,2).
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{0, 1, 2\}$. Symbols are interpreted as shown above.

Appendix B.4. Example 16 (non-joinability)
Mace4 *assumptions.*

```
reds(x,x).
(red(x,y) & reds(y,z)) -> reds(x,z).
red(x,y) -> red(f(x,z),f(y,z)).
red(x,y) -> red(f(z,x),f(z,y)).
red(f(x,f(y,z)),f(f(x,y),f(x,z))).
red(f(f(x,y),z),f(f(x,z),f(y,z))).
red(f(f(x,y),f(y,z)),y).
```

Mace4 *goal.*

```
exists x (reds(f(a,a),x) & reds(a,x)).
```

Mace4 *output.*

```
a = 0.
f(0,0) = 1.
f(0,1) = 2.
f(0,2) = 2.
f(1,0) = 2.
f(1,1) = 2.
f(1,2) = 2.
f(2,0) = 2.
f(2,1) = 2.
f(2,2) = 2.

- red(0,0).
- red(0,1).
- red(0,2).
- red(1,0).
- red(1,1).
- red(1,2).
  red(2,0).
  red(2,1).
  red(2,2).

  reds(0,0).
- reds(0,1).
- reds(0,2).
- reds(1,0).
  reds(1,1).
- reds(1,2).
  reds(2,0).
  reds(2,1).
  reds(2,2).
```

The computed structure $\mathcal{A}$ has domain $\{0, 1, 2\}$. Symbols interpreted as above.

Appendix C. Additional examples from [17]

Appendix C.1. Example 13 (arcs in dependency graph)
AGES *specification.*

```
mod Ex13_Mid01 is
  sorts S .
  ops a b : -> S .
  ops f g : S -> S .
  op F : S -> S .
  var x : S .
  rl a => b .
  rl f(g(a)) => f(a) .
endm
```

AGES *goal.*

```
~(F(a) ->* F(g(a)))
```

AGES *output.*

```
Domains:
S: {-1 , 0}

Function Interpretations:
|[a]| = 0
|[f(x_1_1:S)]| = x_1_1:S
|[F(x_1_1:S)]| = x_1_1:S
|[b]| = 0
|[g(x_1_1:S)]| =  - 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((0 >= x_1_1:S) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((0 >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{-1, 0\}$. Symbols are interpreted as shown above.

Appendix C.2. Example 14 (arcs in dependency graph)

AGES *specification.*

```
mod Ex14_Mid01 is
  sorts S .
  ops a b : -> S .
  ops f : S S -> S .
  op F : S S -> S [N=2] .
  vars x : S .
  rl f(x,x) => f(a,b) .
endm
```

AGES *goal.*

```
~(F(a,b) ->* F(x:S,x:S))
```

AGES *output.*

```
Domains:
S: {-1 , 0}

Function Interpretations:
|[f(x_1_1:S,x_2_1:S)]| = 0
|[F(x_1_1:S,x_2_1:S)]| =
   - 1  if (x_1_1:S + x_2_1:S >= 0)
  x_1_1:S  otherwise
|[a]| = 0
|[b]| =  - 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{-1, 0\}$. Symbols are interpreted as shown above.

Appendix C.3. Example 20 (arcs in dependency graph)

Mace4 *assumptions.*

```
reds(x,x).
(red(x,y) & reds(y,z)) -> reds(x,z).
red(x,y) -> red(f(x,u,v),f(y,u,v)).
red(x,y) -> red(f(u,x,v),f(u,y,v)).
red(x,y) -> red(f(u,v,x),f(u,v,y)).
red(x,y) -> red(F(x,u,v),F(y,u,v)).
red(x,y) -> red(F(u,x,v),F(u,y,v)).
red(x,y) -> red(F(u,v,x),F(u,v,y)).
red(f(a,b,x),f(x,x,x)).
red(a,c).
```

Mace4 *goal.*

```
exists x exists y reds(F(x,x,x),F(a,b,y)).
```

Mace4 *output.*

```
a = 0.

b = 1.

c = 0.

F(0,0,0) = 0.
F(0,0,1) = 0.
F(0,1,0) = 1.
F(0,1,1) = 1.
F(1,0,0) = 0.
F(1,0,1) = 0.
F(1,1,0) = 0.
F(1,1,1) = 0.

f(0,0,0) = 0.
f(0,0,1) = 0.
f(0,1,0) = 0.
f(0,1,1) = 0.
f(1,0,0) = 0.
f(1,0,1) = 0.
f(1,1,0) = 0.
f(1,1,1) = 0.

  red(0,0).
- red(0,1).
- red(1,0).
```

```
  red(1,1).

  reds(0,0).
- reds(0,1).
- reds(1,0).
  reds(1,1).
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{0, 1\}$. Symbols are interpreted as shown above.

Appendix C.4. Example 30 (arcs in dependency graph)
Mace4 *assumptions.*

```
reds(x,x).
(red(x,y) & reds(y,z)) -> reds(x,z).
red(x,y) -> red(f(x),f(y)).
red(x,y) -> red(g(x),g(y)).
red(x,y) -> red(h(x,z),h(y,z)).
red(x,y) -> red(h(z,x),h(z,y)).
red(x,y) -> red(G(x),G(y)).
red(f(a),g(h(a,b))).
red(g(g(a)),f(b)).
red(h(x,x),g(a)).
```

Mace4 *goal.*

```
reds(G(h(a,b)),G(g(a))).
```

Mace4 *output.*

```
a = 0.

b = 1.

G(0) = 0.
G(1) = 1.

f(0) = 0.
f(1) = 0.

g(0) = 0.
g(1) = 0.

h(0,0) = 0.
h(0,1) = 1.
h(1,0) = 0.
h(1,1) = 0.

  red(0,0).
- red(0,1).
- red(1,0).
  red(1,1).

  reds(0,0).
- reds(0,1).
- reds(1,0).
  reds(1,1).
```

The computed structure $\mathcal{A}$ has domain $\{0, 1\}$. Symbols are interpreted as above.

Appendix D. Additional examples from [20]

Appendix D.1. Introduction (infeasible critical pair)

AGES *specification.*

```
mod SM15_Sec1 is
  sorts S .
  ops a b c : -> S .
  ops f g : S -> S .
  var x : S .
  crl f(g(x)) => b if x => a .
  crl g(x) => c if x => c .
endm
```

AGES *goal.*

```
~(x:S ->* a /\ x:S ->* b)
```

AGES *output.*

```
Domains:
S: {-1 , 0}

Function Interpretations:
|[f(x_1_1:S)]| = x_1_1:S
|[g(x_1_1:S)]| =  - 1
|[a]| = 0
|[b]| =  - 1
|[c]| =  - 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((0 >= 1 + x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{-1, 0\}$. Symbols are interpreted as shown above.

Appendix D.2. Example 3.3 (infeasible critical pair)

AGES *specification.*

```
mod SM15_Example_3_3 is
  sorts S .
  ops a b : -> S .
  op f : S -> S .
  var x : S .
  crl f(x) => a if a => x .
  crl f(x) => b if b => x .
endm
```

AGES *goal.*

```
~(a ->* x:S /\ b ->* x:S)
```

AGES *output.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[f(x_1_1:S)]| = x_1_1:S
|[a]| = 1
|[b]| = 0

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{-1\}$. Symbols are interpreted as shown above.

Appendix D.3. Example 5.1 (infeasible critical pairs)
AGES *specification.*

```
mod SM15_Example_5_1 is
  sorts S [m=1 n=1] .
  ops Z tt : -> S .
  op s : S -> S .
  op leq : S S -> S [N=2] .
  op gt : S S -> S [N=2] .
  op div : S S -> S .
  op minus : S S -> S [N=2] .
  op pair : S S -> S .
  var q r x y : S .
  rl leq(Z,x) => tt .
  rl leq(s(x),s(y)) => leq(x,y) .
  rl gt(s(x),Z) => tt .
  rl gt(s(x),s(y)) => gt(x,y) .
  rl minus(x,Z) => x .
  rl minus(Z,x) => Z .
  rl minus(s(x),s(y)) => minus(x,y) .
  crl div(x,y) => pair(Z,y) if gt(y,x) => tt .
  crl div(x,y) => pair(s(q),r) if
        leq(y,x) => tt /\ div(minus(x,y),y) => pair(q,r) .
endm
```

AGES *goal.*

```
~(leq(x:S,w:S) ->* tt /\ div(minus(w:S,x:S),x:S) ->* pair(y:S,z:S)
      /\ gt(x:S,w:S) ->* tt)
```

By Proposition 3, we can use the following simpler goal:

```
~(leq(x:S,w:S) ->* tt /\ gt(x:S,w:S) ->* tt)
```

AGES *output.* Fix the interpretation of $\rightarrow$ to be the *equality.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[div(x_1_1:S,x_2_1:S)]| = 1
|[gt(x_1_1:S,x_2_1:S)]| =
   - 1  if (x_1_1:S >= 1 + x_2_1:S)
   - x_1_1:S + x_2_1:S  otherwise
|[leq(x_1_1:S,x_2_1:S)]| =
  0  if (x_1_1:S >= 1 + x_2_1:S)
   - 1  otherwise
```

```
|[minus(x_1_1:S,x_2_1:S)]| =
   - 1  if (x_2_1:S >= 1 + x_1_1:S)
   - 1 + x_1_1:S - x_2_1:S  otherwise
|[Z]| =  - 1
|[pair(x_1_1:S,x_2_1:S)]| = 1
|[s(x_1_1:S)]| = 1 + x_1_1:S
|[tt]| =  - 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((1 + x_1_1:S >= 0) /\ (x_2_1:S >= x_1_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{-1\}$. Symbols are interpreted as shown above.

Appendix E. Additional examples from [21]

Appendix E.1. Example 3 (infeasible critical pair)

AGES *specification.*

```
mod Ex3_SS16infCCP is
  sorts S .
  ops a b : -> S .
  op f : S -> S .
  var x : S .
  crl f(x) => a if x => a .
  crl f(x) => b if x => b .
endm
```

AGES *goal.*

```
~(x:S ->* a /\ x:S ->* b)
```

AGES *output.*

```
Domains:
S: |N

Function Interpretations:
|[f(x_1_1:S)]| = x_1_1:S
|[a]| = 0
|[b]| = 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N}$. Symbols are interpreted as shown above.

Appendix E.2. Example 16 (infeasible critical pair)

AGES *specification.*

```
mod Ex16_SS16 is
  sorts S .
  ops a b c d : -> S .
  op g : S S -> S .
  op f : S S -> S .
  var x : S .
  rl f(a,x) => a .
  rl f(b,x) => b .
  rl c => c .
  crl g(a,x) => c if f(x,a) => a .
  crl g(x,a) => d if f(x,b) => b .
endm
```

AGES *goal.*

```
~(f(a,b) ->* b /\ f(a,a) ->* a)
```

AGES *output.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[c]| =  - 1
|[f(x_1_1:S,x_2_1:S)]| = x_1_1:S
|[g(x_1_1:S,x_2_1:S)]| = x_2_1:S
|[a]| =  - 1
|[b]| = 1
|[d]| =  - 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> (1 + x_1_1:S >= x_2_1:S)
 x_1_1:S -> x_2_1:S <=> (x_1_1:S >= x_2_1:S)
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N}$. Symbols are interpreted as shown above.

Appendix E.3. Example 17 (infeasible rule)

AGES *specification.*

```
mod Ex17_SS16 is
  sorts S .
  ops a b c : -> S .
  ops g h : S -> S .
  var x : S .
  rl h(x) => a .
  rl g(x) => x .
  rl c => c .
  crl g(x) => a if h(x) => b .
endm
```

AGES *goal.*

```
~(h(x:S) ->* b)
```

AGES *output.*

```
Domains:
S: {-1 , 0 , 1}

Function Interpretations:
|[c]| = 0
|[g(x_1_1:S)]| = x_1_1:S
|[h(x_1_1:S)]| = 0
|[a]| = 0
|[b]| =  - 1

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> (x_2_1:S >= x_1_1:S)
 x_1_1:S -> x_2_1:S <=> (x_2_1:S >= x_1_1:S)
```

Computed model. The computed structure $\mathcal{A}$ has domain $\{-1, 0, 1\}$. Symbols are interpreted as shown above.

Appendix E.4. Example 23 (infeasible critical pair)

AGES *specification.*

```
mod Ex23_SS16 is
  sorts S .
  ops a b : -> S .
  op f : S S -> S [N=2] .
  op g : S -> S .
  var x : S .
  rl g(x) => f(x,x) .
  crl g(x) => g(x) if g(x) => f(a,b) .
endm
```

AGES *goal.*

```
~(g(x:S) ->* f(a,b))
```

AGES *output.*

```
Domains:
S: |N U {-1}

Function Interpretations:
|[g(x_1_1:S)]| = 1 + x_1_1:S
|[a]| = 0
|[b]| =  - 1
|[f(x_1_1:S,x_2_1:S)]| =
  1 + x_1_1:S  if (x_2_1:S >= x_1_1:S)
   - 1 + x_1_1:S  otherwise

Predicate Interpretations:
 x_1_1:S ->* x_2_1:S <=> ((x_2_1:S >= x_1_1:S) /\ (x_1_1:S >= x_2_1:S))
 x_1_1:S -> x_2_1:S <=> ((x_1_1:S >= x_2_1:S) /\ (x_2_1:S >= x_1_1:S))
```

Computed model. The computed structure $\mathcal{A}$ has domain $\mathbb{N} \cup \{-1\}$. Symbols are interpreted as shown above.