AIST-PS-2005-013

Sufficient Completeness Checking with Propositional Tree Automata

Joe Hendrix[1], Hitoshi Ohsaki[2], and José Meseguer[1]

[1]University of Illinois at Urbana-Champaign
{jhendrix,meseguer}@uiuc.edu

[2]National Institute of Advanced Industrial Science and Technology, &
PRESTO - Japan Science and Technology Agency

ohsaki@ni.aist.go.jp

Sufficient Completeness Checking with Propositional Tree Automata

Joe Hendrix[1], Hitoshi Ohsaki[2,3], and José Meseguer[1]

[1]University of Illinois at Urbana-Champaign
{jhendrix,meseguer}@uiuc.edu

[2]National Institute of Advanced Industrial Science and Technology, and
[3]PRESTO, Japan Science and Technology Agency
ohsaki@ni.aist.go.jp

Abstract. Sufficient completeness means that enough equations have been specified, so that the functions of an equational specification are fully defined on all relevant data. This is important for both debugging and formal reasoning. In this work we extend sufficient completeness methods to handle expressive specifications involving: (i) partiality; (ii) conditional equations; and (iii) deduction *modulo* axioms. Specifically, we give useful characterizations of the sufficient completeness property for membership equational logic (MEL) specifications having features (i)–(iii). We also propose a kind of equational tree automata [18, 22], called *propositional tree automata* (PTA) and identify a class of MEL specifications (called PTA-checkable) whose sufficient completeness problem is equivalent to the emptiness problem of their associated PTA. When the reasoning modulo involves only symbols that are either associative and commutative (AC) or free, we further show that the emptiness of AC-PTA is decidable, and therefore that the sufficient completeness of AC-PTA-checkable specifications is decidable. The methods presented here can serve as a basis for building a next-generation sufficient completeness tool for MEL specifications having features (i)–(iii). These features are widely used in practice, and are supported by languages such as Maude and other advanced specification and equational programming languages.
Keywords: sufficient completeness, equational logic, tree automata

1 Introduction

Sufficient completeness of an equational specification means that enough equations have been specified, so that the functions of the algebraic data type defined by the specification are fully defined on all relevant data elements. This is an important property to check, both to debug and formally reason about specifications and equational programs. For example, many inductive theorem proving techniques are based on the constructors building up the data and crucially depend on the specification being sufficiently complete.

In practice, there is a need to have expressive equational specification formalisms that match the needs of real applications, and correspondingly to extend sufficient completeness methods to handle such formalisms. This work presents new contributions extending sufficient completeness methods in several useful directions, namely: (i) to handle partiality; (ii) to allow conditional specifications; and (iii) to support equational reasoning *modulo* axioms. These extensions are needed in practice because: (i) functions defined on data types are often *partial*; (ii) many languages support conditional specifications; and (iii) functions often assume algebraic properties of their underlying data. For example, functions on sets or multisets are much more simply specified using the fact that set and multiset union are associative and commutative.

Of course, there is tension between expressiveness of specifications and decidability of sufficient completeness. It is well-known that checking sufficient completeness is in general undecidable even for unconditional specifications [7, 8]. Partiality makes decidability even harder to get, and in the presence of associativity axioms decidability is lost for non-linear confluent and terminating equations [13]. In our view, the best way to deal with this tension is *not to give up* on the expressive specifications that are often needed in practice and for which sufficient completeness is undecidable. We advocate a two-pronged approach. First, the sufficient completeness problem should be studied for increasingly more expressive formalisms, and the set of decidable specifications should likewise be expanded as much as possible. Second, sufficient completeness checking algorithms should be coupled with inductive theorem proving techniques that can discharge proof obligations generated when the input specification falls outside of the decidable classes. We refer the reader to [9, 11] for ideas on coupling sufficient completeness and inductive theorem proving.

In this paper, we focus on advancing the first prong in several ways. Our first contribution is to characterize sufficient completeness in a more general setting to support the extensions (i)–(iii) mentioned above. For this purpose, we use membership equational logic (MEL) [2, 16] to allow conditional specification of partial functions (see [16] for a survey of partial specification formalisms and the use of MEL in this context). In MEL atomic sentences are either equations $t = t'$, or memberships $t : s$ stating that a term t has a sort s, where t having a sort is equivalent to t being *defined*. The key idea is that a partial function's domain is axiomatized by conditional membership axioms. We precisely define the sufficient completeness property for conditional MEL specifications which can have extra variables in their conditions and can be defined modulo a set E of unconditional equations. We define sufficient completeness for both MEL theories of this kind and their corresponding conditional rewrite theories when the equations are used as rewrite rules. We also characterize specifications for which both notions coincide. Finally for a large class of systems, we give conditions equivalent to sufficient completeness which can be the basis of a checking algorithm. These theoretical developments directly apply to the analysis of functional modules in the Maude language [3], which are MEL specifications supporting deduction modulo axioms such as associativity, commutativity, and identity.

Our second contribution is to identify a class of specifications whose sufficient completeness problem is equivalent to the emptiness problem of an associated equational tree automaton. The class in question, called PTA-checkable specifications, consists of those MEL specifications that are left-linear, unconditional, ground weakly normalizing, ground confluent, and ground sort-decreasing. An important aspect of this is the definition of a kind of equational tree automata, which we call *propositional tree automata* (PTA). PTA are closed under boolean operations and can encode the sufficient completeness problem in a more compact way than conventional tree automata.

As the axioms E used to rewrite modulo can be arbitrary, the emptiness problem for general PTA is undecidable. However in practice, we need not consider arbitrary axioms E, but rather focus on widely applicable axioms such as associativity, commutativity, and identity for which matching algorithms exist. This leads us to our third contribution, namely a proof of the decidability of the PTA emptiness problem — and therefore of the sufficient completeness of PTA-checkable specifications — when the set E of equational axioms specifies operators which are either both associative and commutative (AC) or free.

Related Work

Instead of a comprehensive survey on sufficient completeness, we mention some related work to place things in context. Sufficient completeness of MEL specifications was first studied in [2]. The definition and methods presented in this paper substantially extend and generalize those in [9], which in turn had generalized those in [2], allowing a much wider class of MEL specifications to be checked. Sufficient completeness itself goes back to Guttag's thesis [7] (but see [8] for a more accessible treatment). An early algorithm for handling unconditional linear specifications is due to Nipkow and Weikum [17]. For a good review of the literature up to the late 80s, as well as some important decidability/undecidability and complexity results, see [12, 13]. A more recent development is the casting of sufficient completeness as tree automata (TA) decision problems: see Chapter 4 of [4] and references there. Our work can be seen as a further contribution to the TA approach to sufficient completeness, particularly in proposing PTA as a new tree automata framework ideally suited for sufficient completeness checking and in extending TA methods and results to the equational cases.

In addition to the TA approach, a widely used alternative approach to sufficient completeness is based on the incremental constructor-based narrowing of patterns. Sufficient completeness checkers in this category include Spike [1], the RRL [11] theorem prover, and the current Maude Sufficient Completeness Checker (SCC), which is integrated with the Maude inductive theorem prover [9]. Although constructor-based narrowing methods have been extended to the AC case in the context of unsorted specifications by Jouannaud and Kounalis [10], it seems much harder to apply this extension in the context of order-sorted or MEL specifications. For this reason, an important practical motivation of this work is to eventually replace the current Maude SCC, which cannot handle the AC case, by a next-generation tool based on the PTA methods presented here.

The paper is organized as follows. In sections 2 and 3, we recall membership equational logic, and introduce a very expressive class of rewrite systems called conditional equational rewriting/membership (CERM) systems. We define sufficient completeness in both these contexts, and give conditions under which these definitions coincide. In Section 4, we give conditions on which to base methods for checking sufficient completeness, and introduce a new class of equational tree automata, called *propositional tree automata* (PTA). In Section 5, we show how the sufficient completeness problem can be formulated as a PTA emptiness problem for a fairly broad class of specifications. Finally in Section 6, we show how to decide emptiness of PTA for the AC case.

2 Preliminaries

A *membership equational logic* (MEL) *signature* Σ is a triple $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$, in which: $\mathcal{K}$ is a set of *kinds*; $\mathcal{F} = \{\mathcal{F}_{\vec{k},k}\}_{(\vec{k},k)\in\mathcal{K}^*\times\mathcal{K}}$ is a $\mathcal{K}$-kinded family of function symbols such that $\mathcal{F}_{\vec{k},k}$ and $\mathcal{F}_{\vec{k},k}$ are disjoint for distinct $k, k' \in \mathcal{K}$; and $\mathcal{S} = \{\mathcal{S}_k\}_{k\in\mathcal{K}}$ is a disjoint $\mathcal{K}$-kinded family of sets of *sorts*. A $\mathcal{K}$-kinded list of variables $\vec{x} = x_1 : k_1, \ldots x_n : k_n$ is *Σ-distinct* when the $x_1, \ldots x_n$ are pairwise disjoint and also distinct from any constant in $\mathcal{F}$, but the kinds $k_1, \ldots k_n$ in the list can be repeated. For each $k \in \mathcal{K}$, $T_\Sigma(\vec{x})_k$ denotes the set of well-kinded terms with kind k formed from the function symbols in $\mathcal{F}$ and variables in $\vec{x}$. $T_\Sigma(\vec{x}) = \bigcup_{k\in\mathcal{K}} T_\Sigma(\vec{x})_k$ denotes the set of all well-kinded terms. The set of all *ground terms*, T_Σ, and set of all ground terms with kind k, $T_{\Sigma,k}$, are sets of terms without variables. The function $\mathrm{vars} : T_\Sigma(\vec{x}) \to \mathcal{P}(\vec{x})$ is the mapping from a term $t \in T_\Sigma(\vec{x})$ to the variables appearing in t. Let $\Box : k$ be a distinguished variable called a *hole*. A *k-context* is a term $\mathbb{C} \in T_\Sigma(\vec{x}, \Box : k)$ with a single occurrence of the variable $\Box : k$. For any $t \in T_\Sigma(\vec{x})_k$, we denote by $\mathbb{C}[t] \in T_\Sigma(\vec{x})$ the term obtained by replacing $\Box$ by t.

A *$\Sigma(\vec{x})$-equation* is a formula $t = u$ with $t, u \in T_\Sigma(\vec{x})_k$ for some $k \in \mathcal{K}$. A *$\Sigma(\vec{x})$-membership* is a formula $t : s$ with $t \in T_\Sigma(\vec{x})_k$ and $s \in \mathcal{S}_k$. *Σ-sentences* are universally quantified Horn clauses of the form $(\forall \vec{x})\ A$ **if** $A_1 \wedge \cdots \wedge A_n$ where A and A_i $(1 \leq i \leq n)$ are either $\Sigma(\vec{x})$-equations or $\Sigma(\vec{x})$-memberships. If A is a $\Sigma(\vec{x})$-equation, the sentence is a *conditional equation*; if A is a $\Sigma(\vec{x})$-membership, the sentence is a *conditional membership*. A MEL theory is a pair $\mathcal{E} = (\Sigma, \Gamma)$ with Σ a MEL signature and Γ a set of Σ-sentences. As explained in [2, 16], there is a sound and complete inference system to derive all theorems of a MEL theory (Σ, Γ).

3 Conditional Equational Rewrite/Membership Systems

Existing techniques for checking sufficient completeness require considering the rewrite system used to define the equational specification. Our approach likewise requires considering the conditional rewrite system that expresses the operational semantics of a MEL theory when executed in a language such as Maude.

Standard conditional term rewriting systems, also called *join* term rewriting systems, interpret equations $t = u$ appearing in a condition as join pairs $t \downarrow u$, in which t and u are considered equal if they rewrite to the same term. However, in order to consider a larger class of term rewriting systems, we use *oriented* systems, in which conditions may contain formulas of the form $t \to u$. An oriented

$$\text{Equivalence} \quad \frac{t =_E u}{t \to u}$$

$$\text{Replacement} \quad \frac{t =_E \mathbb{C}[l\theta] \quad u_1\theta \to v_1\theta \ldots u_m\theta \to v_m\theta \quad w_1\theta : s_1 \ldots w_n\theta : s_n}{t \to^1 \mathbb{C}[r\theta]}$$

$$\text{with } (\forall \vec{y})\, l \to r \textbf{ if } \bigwedge_{i=1}^{m} u_i \to v_i \wedge \bigwedge_{j=1}^{n} w_j : s_j \text{ in } \Gamma.$$

$$\text{Transitivity} \quad \frac{t \to^1 u \qquad u \to v}{t \to v}$$

$$\text{Membership} \quad \frac{t =_E l\theta \quad u_1\theta \to v_1\theta \ldots u_m\theta \to v_m\theta \quad w_1\theta : s_1 \ldots w_n\theta : s_n}{t :^0 s}$$

$$\text{with } (\forall \vec{y})\, l : s' \textbf{ if } \bigwedge_{i=1}^{m} u_i \to v_i \wedge \bigwedge_{j=1}^{n} w_j : s_j \text{ in } \Gamma \text{ and } s' < s$$

$$\text{Subject Reduction} \quad \frac{t \to u \qquad u :^0 s}{t : s}$$

$$\text{with } t, u, v \in T_\Sigma(\vec{x}) \text{ and } \theta : \vec{y} \to T_\Sigma(\vec{x})$$

Fig. 1. Inference rules for $\mathcal{R} = (\Sigma, E, <, \Gamma)$ with respect to Σ-distinct variables $\vec{x}$

term rewriting system can simulate a join term rewriting system by introducing a new kind B, a new constant tt : $\to$ B, and for each $k \in \mathcal{K}$, a binary operator eq : $k\ k \to$ B with a new rule $(\forall x : k)\,\mathrm{eq}(x, x) \to \mathrm{tt}$. Join conditions of the form $t \downarrow u$ can then be expressed as oriented conditions $\mathrm{eq}(t, u) \to \mathrm{tt}$.

Definition 1. *A* conditional equational rewrite/membership (CERM) system *is a tuple $\mathcal{R} = (\Sigma, E, <, \Gamma)$ in which: $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$ is a MEL signature where $T_{\Sigma,k}$ is nonempty for each $k \in \mathcal{K}$, E is a set of unconditional Σ-equations; $< = \{<_k\}_{k \in \mathcal{K}}$ is a $\mathcal{K}$-indexed family of binary relations such that $<_k$ is a strict order over $\mathcal{S}_k$ for each $k \in \mathcal{K}$; and Γ is a set containing two types of rules:*

$$(\forall \vec{y})\, t : s \textbf{ if } \bigwedge u_i \to v_i \wedge \bigwedge w_j : s_j \qquad\qquad (\forall \vec{y})\, t \to t' \textbf{ if } \bigwedge u_i \to v_i \wedge \bigwedge w_j : s_j$$

$$\text{Membership Rule} \qquad\qquad\qquad \text{Rewrite Rule}$$

with $\vec{y}$ a set of Σ-distinct variables, $t, t' \in T_\Sigma(\vec{y})_k$ and $s \in \mathcal{S}_k$ with $k \in \mathcal{K}$, each $u_i, v_i \in T_\Sigma(\vec{y})_{k_i}$ with $k_i \in \mathcal{K}$, and each $w_j \in T_\Sigma(\vec{y})_{k_j}$ and $s_j \in \mathcal{S}_{k_j}$ with $k_j \in \mathcal{K}$.

In the rest of this section, $\mathcal{R} = (\Sigma, E, <, \Gamma)$ is a CERM system where $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$ and $\vec{x}$ is a list of Σ-distinct variables. $\Sigma(\vec{x})$-*atomic formulas* are expressions of the form $t \to u$ and $t : s$, where $t, u \in T_\Sigma(\vec{x})_k$ and $s \in \mathcal{S}_k$ with $k \in \mathcal{K}$.

When A is a $\Sigma(\vec{x})$-atomic formula, $\mathcal{R} \vdash A$ denotes that A can be derived from the inferences rules in Fig. 1 for $\mathcal{R}$. We will additionally use the notation $\mathcal{R} \vdash t \downarrow u$ to abbreviate $(\exists v)\ \mathcal{R} \vdash t \to v \wedge \mathcal{R} \vdash u \to v$. This inference system refines and extends the one used in Fig. 7 of [2]. Specifically, we define refined notions of rewriting and membership directly in the proof theory, combine several rules into a single rule to allow simpler proofs later on, and allow rewrites to take place *modulo* the axioms E. Note that $\to^1$ corresponds to the notion of a one-step rewrite modulo the equations E, and $:^0$ is a more restricted notion of membership in which we disallow subject reduction.

A term $t \in T_\Sigma(\vec{x})$ is *$\mathcal{R}$-reducible* when there is a $u \in T_\Sigma(\vec{x})$ where $\mathcal{R} \vdash t \to^1 u$ and *$\mathcal{R}$-irreducible* when it is not $\mathcal{R}$-reducible. $\mathcal{R}$ is *pattern-based* when for each rule in $\mathcal{R}$ of the form $(\forall \vec{y})\, A$ **if** $\bigwedge_{1 \le i \le m} u_i \to v_i \wedge \bigwedge_{1 \le j \le n} w_j : s_j$ with A of the form $l \to r$ or $l : s$, we have for each $v_i \in \{v_1, \ldots, v_m\}$:

– The variables in v_i are fresh[1], i.e.

$$\mathrm{vars}(v_i) \cap \big(\mathrm{vars}(l) \cup \bigcup_{1 \le k \le i} \mathrm{vars}(u_k) \cup \bigcup_{1 \le k < i} \mathrm{vars}(v_k)\big) = \varnothing$$

– Any rewriting of a term $v_i\theta$ occurs below the pattern v_i, i.e., for Σ-distinct variables $\vec{x}$ and substitution $\theta : \vec{y} \to T_\Sigma(\vec{x})$, if there is a term $t \in T_\Sigma$ such that $\mathcal{R} \vdash v_i\theta \to t$, then there is a substitution $\tau : \vec{y} \to T_\Sigma(\vec{x})$ where $t =_E v_i\tau$.

$\mathcal{R}$ is *confluent* relative to variables $\vec{x}$ when for every $t, u, v \in T_\Sigma(\vec{x})$, if $\mathcal{R} \vdash t \to u$ and $\mathcal{R} \vdash t \to v$, then $\mathcal{R} \vdash u \downarrow v$. $\mathcal{R}$ is *sort-decreasing* relative to variables $\vec{x}$ when for every $t, u \in T_\Sigma(\vec{x})$ and $s \in \mathcal{S}$ if $\mathcal{R} \vdash t \to u$ and $\mathcal{R} \vdash t : s$, then $\mathcal{R} \vdash u : s$. In particular, when $\mathcal{R}$ is confluent (resp. sort-decreasing) relative to $\varnothing$, the empty set of variables, we call $\mathcal{R}$ *ground confluent* (resp. *ground sort-decreasing*).

Definition 2. *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be a CERM system with $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$. $\mathcal{E}_\mathcal{R} = (\Sigma, E \cup \Gamma' \cup M_<)$ denotes the* underlying MEL specification *of $\mathcal{R}$ where Γ' contains the same axioms as Γ, but with each oriented atomic formula $t \to u$ replaced with an equality $t = u$, and $M_<$ is a set of conditional memberships which axiomatize the subsort relation $<$, i.e.*

$$M_< = \big\{ (\forall x : k)\, x : s \textbf{ if } x : s' \;\big|\; s, s' \in \mathcal{S}_k : s' < s \big\}.$$

$\mathcal{R}$ is *weakly-convergent* relative to variables $\vec{x}$ when $\mathcal{R}$ is pattern-based, confluent, and sort-decreasing relative to $\vec{x}$. $\mathcal{R}$ is *ground weakly-convergent* when $\mathcal{R}$ is weakly-convergent relative $\varnothing$. There is an equivalence between formulas provable in a weakly-convergent system $\mathcal{R}$ and its underlying specification $\mathcal{E}_\mathcal{R}$.

Theorem 1 (Inference Equivalence). *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be a CERM system that is weakly-convergent relative to Σ-distinct variables $\vec{x}$. For all $t, u \in T_\Sigma(\vec{x})$ and $s \in \mathcal{S}$,*

$$\mathcal{E}_\mathcal{R} \vdash t = u \iff \mathcal{R} \vdash t \downarrow u \qquad \textit{and} \qquad \mathcal{E}_\mathcal{R} \vdash t : s \iff \mathcal{R} \vdash t : s.$$

The inference system of $\mathcal{E}_\mathcal{R}$ can be found in Fig. 4 of [2]. □

A model theoretic definition of sufficient completeness for MEL specifications was given in [9]. We provide here proof theoretic definitions of sufficient completeness for both CERM systems and their underlying MEL specifications, and show under what conditions these two definitions coincide.

[1] Without the additional requirement that $\mathrm{vars}(r) \subseteq l \cup \bigcup_{1 \le k \le m} v_k$ and each $\mathrm{vars}(u_i) \subseteq l \cup \bigcup_{1 \le k < i} v_k$, pattern-based systems are not directly executable, unless a strategy to instantiate the additional free variables in r and each u_i is given. Nevertheless, our results hold in generality without this extra requirement.

Definition 3. *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be a CERM system in which $\Gamma = R \cup M$, where R contains the rewrite rules and M contains the membership rules.*

Given a subset of memberships $M_\Omega \subseteq M$, let $\mathcal{R}_\Omega = (\Sigma, E, <, R \cup M_\Omega)$ be the associated CERM system. $\mathcal{R}$ is sufficiently complete *with respect to $\mathcal{R}_\Omega$ when for all $t, u \in T_\Sigma$ and $s \in \mathcal{S}$:*

$$\mathcal{R} \vdash t{\downarrow}u \iff \mathcal{R}_\Omega \vdash t{\downarrow}u \qquad \text{and} \qquad \mathcal{R} \vdash t : s \iff \mathcal{R}_\Omega \vdash t : s.$$

Additionally, $\mathcal{E}_\mathcal{R}$ is sufficiently complete *with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$ when for all $t, u \in T_\Sigma$ and $s \in \mathcal{S}$:*

$$\mathcal{E}_\mathcal{R} \vdash t = u \iff \mathcal{E}_{\mathcal{R}_\Omega} \vdash t = u \qquad \text{and} \qquad \mathcal{E}_\mathcal{R} \vdash t : s \iff \mathcal{E}_{\mathcal{R}_\Omega} \vdash t : s.$$

Moreover, if $\mathcal{E}_\mathcal{R}$ is sufficiently complete with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$, M_Ω are its constructor memberships*, $\mathcal{E}_{\mathcal{R}_\Omega}$ is a* constructor subspecification *of $\mathcal{E}_\mathcal{R}$, and $M_\Delta = M_\Sigma - M_\Omega$ are its* defined memberships*.*

Theorem 2 (Suff. Comp. Equivalence). *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$, and let $M_\Omega \subseteq M$ be a subset of memberships in $\mathcal{R}$.*

- *If $\mathcal{R}$ is ground weakly-convergent and sufficiently complete with respect to $\mathcal{R}_\Omega$, then $\mathcal{E}_\mathcal{R}$ is sufficiently complete with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$.*
- *If $\mathcal{R}_\Omega$ is ground weakly-convergent, and $\mathcal{E}_\mathcal{R}$ is sufficiently complete with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$, then $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$.* □

4 Sufficient Completeness and Propositional Tree Automata

Techniques for checking sufficient completeness of unconditional specifications typically require that the specification is weakly normalizing, i.e., every term t rewrites to an irreducible term u. In the context of conditional rewriting, this condition is not strong enough. We need to develop a stronger notion of weak normalization in the context of CERM systems. Recall that a *reduction order* $\succ$ on $T_\Sigma(\vec{x})$ is a strict order that is noetherian, congruent with $=_E$, and closed with respect to context and substitution.

Definition 4. *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$, and let $\vec{x}$ be a set of Σ-distinct variables.*

A proof is said to be reductive *relative to a reduction order $\succ$ when every use of Replacement and Membership inferences:*

$$\frac{t =_E \mathbb{C}[l\theta] \quad \bigwedge u_i\theta \to v_i\theta \quad \bigwedge w_j\theta : s_1}{t \to^1 \mathbb{C}[r\theta]} \qquad \frac{t =_E l\theta \quad \bigwedge u_i\theta \to v_i\theta \quad \bigwedge w_j\theta : s_j \quad s' < s}{t :^0 s}$$

satisfies that $t \succ \mathbb{C}[r\theta]$, $t \succ u_i\theta$ for each u_i, and $t \succ w_j\theta$ for each w_j.

We say that $\mathcal{R}$ is weakly normalizing *relative to $\vec{x}$ when:*

- *For each $t \in T_\Sigma(\vec{x})$, there is a $\mathcal{R}$-irreducible $u \in T_\Sigma(\vec{x})$ such that $\mathcal{R} \vdash t \to u$*
- *There is a reductive order $\succ_\mathcal{R}$ where for every $\Sigma(\vec{x})$-sentence A derivable from $\mathcal{R}$, i.e. $\mathcal{R} \vdash A$, there is a proof of A that is reductive relative to $\succ_\mathcal{R}$.*

$\mathcal{R}$ is ground weakly normalizing *when it is weakly normalizing with $\vec{x} = \varnothing$.*

For a broad class of specifications, there is a characterization of sufficient completeness which is often easier to check.

Theorem 3 (Checking Theorem). *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$ be a ground weakly normalizing and ground sort-decreasing CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$. Given a set of memberships $M_\Omega \subseteq M$, $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$ iff for every membership $m \in M_\Delta$,*

$$(\forall \vec{y})\, t : s \textbf{ if } \bigwedge u_i \to v_i \wedge \bigwedge w_j : s_j,$$

and every substitution $\theta : \vec{y} \to T_\Sigma$ in which $\mathcal{R} \vdash u_i\theta \to v_i\theta$ and $\mathcal{R} \vdash w_j\theta : s_j$ for each i and j, if $t\theta$ is $\mathcal{R}_\Omega$-irreducible, then $\mathcal{R}_\Omega \vdash t\theta : s$. □

In Section 5, a particular subclass of specifications for which sufficient completeness can be formulated as a tree language problem using a new type of tree automata, proposed below, which we call *propositional tree automata* (PTA).

Definition 5. *A* propositional tree automaton *is a tuple $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, E, \Delta)$ where:*

- *$(\mathcal{K}, \mathcal{F})$ is a many-kinded signature, i.e., a pair with a set $\mathcal{K}$ of kinds and a $\mathcal{K}$-kinded family $\mathcal{F}$ of function symbols;*
- *$\mathcal{Q} = \{\mathcal{Q}_k\}_{k \in \mathcal{K}}$ is a $\mathcal{K}$-kinded family of sets of* states *with $\mathcal{Q}_k \cap \mathcal{F}_{\epsilon,k'} = \varnothing$ for each $k, k' \in \mathcal{K}$;*
- *$\Phi = \{\phi_k\}_{k \in \mathcal{K}}$ is a $\mathcal{K}$-indexed family of propositional formulas, where the atomic propositions in each ϕ_k are states in $\mathcal{Q}_k$;*
- *E is a set of unconditional $(\mathcal{K}, \mathcal{F})$-equations; and*
- *Δ contains rewrite rules, called* transition rules, *each of which is of one of the following forms:*

  $$\text{(Type 1)} \quad f(p_1, \ldots, p_n) \to q \qquad\qquad \text{(Type 2)} \quad p \to q$$

  where for some $k \in \mathcal{K}$, $p, q \in \mathcal{Q}_k$, $f \in \mathcal{F}_{(k_1,\ldots,k_n),k}$, and each $p_i \in \mathcal{Q}_{k_i}$.

Propositional tree automata extend traditional tree automata [4, 5] in several different directions: terms are over a many-kinded signature instead of over an unsorted signature; recognition is done modulo the unconditional equations E; and a term $t \in T_{\Sigma,k}$ is accepted if the complete set of states t can reach is a model of ϕ_k. They can also be viewed as a special class of CERM systems.

Definition 6. *Let $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, E, \Delta)$ be a PTA, and let $\Sigma_\mathcal{A} = (\mathcal{K}, \mathcal{F} \cup \mathcal{Q}, \varnothing)$. We call $\mathcal{R}_\mathcal{A} = (\Sigma_\mathcal{A}, E, \varnothing, \Delta)$ the* associated CERM system *of $\mathcal{A}$ where: (1) the signature $\Sigma_\mathcal{A}$ contains the operations in $\mathcal{F}$ with the states in each $\mathcal{Q}_k$ added as constants of kind k, and no sorts; (2) the subsort relation is empty; (3) the equations E are left unchanged; (4) the rules in $\mathcal{A}$ become rewrite rules in $\mathcal{R}_\mathcal{A}$.*

A move relation *$\to_\mathcal{A}$ on terms in $T_{\Sigma_\mathcal{A}}$ is the binary relation in which, for terms $t, u \in T_{\Sigma_\mathcal{A}}$, $t \to_\mathcal{A} u$ iff $\mathcal{R}_\mathcal{A} \vdash t \to u$. Additionally, for each $k \in \mathcal{K}$, we define the mapping $\mathsf{reach}_{\mathcal{A},k} : T_{\Sigma_\mathcal{A},k} \to \mathcal{P}(\mathcal{Q}_k)$ as follows:*

$$\mathsf{reach}_{\mathcal{A},k}(t) = \{\, q \in \mathcal{Q}_k \mid t \to_\mathcal{A} q \,\}.$$

The language accepted *by $\mathcal{A}$, $\mathcal{L}(\mathcal{A}) = \{\mathcal{L}(\mathcal{A})_k\}_{k \in \mathcal{K}}$, is a $\mathcal{K}$-kinded family of* *subsets of $T_{\Sigma_{\mathcal{A}},k}$ in which for each $k \in \mathcal{K}$:*

$$\mathcal{L}(\mathcal{A})_k = \{\, t \in T_{\Sigma_{\mathcal{A}},k} \mid \mathsf{reach}_{\mathcal{A},k}(t) \models \phi_k \,\}$$

where boolean formulas are evaluated using their standard interpretations:

$$S \models p \;\; \textit{if} \;\; p \in S, \quad S \models \phi_1 \vee \phi_2 \;\; \textit{if} \;\; S \models \phi_1 \;\; \textit{or} \;\; S \models \phi_2, \quad S \models \neg\, \phi \;\; \textit{if} \;\; S \not\models \phi.$$

As an example, let us consider the PTA $\mathcal{A}$ with a single kind k, the propositional formula $\phi_k = \mathsf{q_a} \wedge \neg \mathsf{q_b}$, and the transition rules:

$$\mathsf{a} \to \mathsf{q_a} \qquad \mathsf{b} \to \mathsf{q_b} \qquad \mathsf{f}(\mathsf{q_a}) \to \mathsf{q_a} \qquad \mathsf{f}(\mathsf{q_b}) \to \mathsf{q_b} \qquad \mathsf{f}(\mathsf{q_a}) \to \mathsf{q} \qquad \mathsf{f}(\mathsf{q_b}) \to \mathsf{q}.$$

Then a is accepted by $\mathcal{A}$, because $\mathsf{reach}_{\mathcal{A},k}(\mathsf{a}) = \{\, \mathsf{q_a} \,\}$ and $\{\, \mathsf{q_a} \,\} \models \mathsf{q_a} \wedge \neg \mathsf{q_b}$. Similarly, $\mathsf{f}(\mathsf{a})$ is accepted as $\mathsf{reach}_{\mathcal{A},k}(\mathsf{f}(\mathsf{a})) = \{\, \mathsf{q_a}, \mathsf{q} \,\}$ and $\{\, \mathsf{q_a}, \mathsf{q} \,\} \models \mathsf{q_a} \wedge \neg \mathsf{q_b}$. However, b is not accepted, because $\mathsf{reach}_{\mathcal{A},k}(\mathsf{b}) = \{\, \mathsf{q_b} \,\}$ and $\{\, \mathsf{q_b} \,\} \not\models \mathsf{q_a} \wedge \neg \mathsf{q_b}$.

PTA are closed under boolean operations. Intersecting two automata simply requires intersecting the propositional formulas in the two automata. Complementing an automaton simply requires complementing the propositional formula in that automata.

One can observe that, given a term t and a PTA $\mathcal{A}$ with a set $\mathcal{Q}$ of states, when $t \to_{\mathcal{A}} q$ is decidable for each $q \in \mathcal{Q}$, $\mathsf{reach}(t)$ is effectively computable, and consequently the membership problem for PTA is decidable. When the emptiness problem for PTA is decidable, other typical decision problems, such as inclusion, universality, and intersection-emptiness are all decidable due to the boolean closure properties of PTA.

5 PTA-Construction for Sufficient Completeness

Definition 7. *A CERM system $\mathcal{R} = (\Sigma, E, <, \Gamma)$ in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$ is* PTA-checkable *when it satisfies:*

(a) $\mathcal{R}$ is ground weakly normalizing and ground weakly-convergent.
(b) For each $k, k' \in \mathcal{K}$, $\mathcal{S}_k \cap \mathcal{F}_{\epsilon,k'} = \varnothing$.
(c) Each equation in E is linear.
(d) Every axiom in Γ is of one of the forms:

$$(\forall \vec{y})\, f(t_1, \ldots, t_n) \to r \;\textbf{if}\; \bigwedge_{y_i \in \vec{y}} y_i : s_i \qquad (\forall \vec{y})\, f(t_1, \ldots, t_n) : s \;\textbf{if}\; \bigwedge_{y_i \in \vec{y}} y_i : s_i$$

where $f(t_1, \ldots, t_n)$ is linear and $\mathrm{vars}(f(t_1, \ldots, t_n)) = \vec{y}$.

Every PTA-checkable system satisfies the conditions in Theorem 3. Henceforth, we will often abbreviate a conditional axiom $(\forall \vec{x})\, A$ **if** $\bigwedge_{x_i \in \vec{x}} x_i : s_i$ in a PTA-checkable system with the syntax $(\forall \vec{x} : \vec{s})\, A$, where $\vec{s} = s_1, \ldots, s_{|\vec{x}|}$.

The class of PTA-checkable systems is general enough to include all unconditional, left-linear, order sorted specifications where E is linear. As a simple example of a PTA-checkable system, we can specify non-empty multisets of natural numbers in the following Maude functional module:

```
fmod MSET is
  sorts Nat MSet .             subsort Nat < MSet .
  op 0 : -> Nat [ctor].        op s : Nat -> Nat [ctor].
  op __ : MSet MSet -> MSet [ctor assoc comm].
  op |_| : MSet -> Nat .
  var N : Nat                  var M : MSet .
  eq | N | = s(0) .            eq | N M | = s(| M |) .
endfm
```

This module defines the sorts `Nat` and `MSet` with `Nat` $<$ `MSet`. Because of this subsort declaration, `Nat` and `MSet` are implicitly in the same kind, denoted `[MSet]`. The operators `0` and `s` denote zero and successor respectively. The subsort declaration specifies that every natural number is also a multiset, and the `__` operator denotes the union of two multisets. These operators are labeled with the `ctor` attribute to identify them as constructors, and `__` is additionally labeled with the `assoc` and `comm` attributes to introduce associativity and commutativity axioms for it, i.e., $(xy)z = x(yz)$ and $xy = yx$. A single defined operation is introduced `|_|` which returns the number of elements in a multiset. The variable declarations associate the variable `N` with the the sort `Nat`, and the variable `M` with the sort `MSet`. By this, `M` and `N` are declared to have kind `[MSet]`, and when either variable appears in a rule, an implicit condition is added to the rule that the variable only ranges over terms with the associated sort. The other equations, `| N | = s(0)` and `| N M | = s(| M |)`, are interpreted as rewrite rules. In the CERM system corresponding to `MSET`, E only contains the associativity and commutativity axioms of `__`. Γ contains the memberships and other equations.

The specification contains membership rules as well, but these are implicit in the operator declaration section. In fact, the previous operation and equation declarations are just syntactic sugar for:

```
op 0 : -> [MSet] .             op s : [MSet] -> [MSet] .
op __ : [MSet] [MSet] -> [MSet] [assoc comm].
op |_| : [MSet] -> [MSet] .
mb 0 : Nat .                   cmb s(X) : Nat if X : Nat .
cmb X Y : MSet if X : MSet /\ Y : MSet .
cmb | X | : Nat if X : MSet .
ceq | N | = s(0) if N : Nat .
ceq | N M | = s(| M |) if N : Nat /\ M : MSet .
```

We can check the sufficient completeness of a PTA-checkable system by testing the emptiness of a propositional tree automaton. To do this, given a PTA-checkable CERM system $\mathcal{R}$ annotated with a constructor subsystem $\mathcal{R}_\Omega$, we construct an automata with states such that a term t will rewrite to a specific state: (1) when t is reducible; (2) when t is a constructor with sort $s \in \mathcal{S}$; and (3) when t matches a defined membership with sort $s \in \mathcal{S}$ and has constructor subterms.

Definition 8. *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be an PTA-checkable system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$ and $\Gamma = R \cup M$, where R contains the rewrite rules and M contains the membership rules. Let $T_\Sigma(\mathcal{S})$ denote the set of terms obtained by adding for each $k \in \mathcal{K}$ and sort $s \in \mathcal{S}_k$, a distinct constant of kind k to Σ. Given terms*

$t \in T_{\Sigma,k}$ and $u \in T_\Sigma(\mathcal{S})_k$ with $k \in \mathcal{K}$, we say that u subsumes *t, $\mathcal{R} \vdash u \supseteq t$, when $u \supseteq t$ is derivable from the rules:*

$$\frac{\mathcal{R} \vdash t : s}{\mathcal{R} \vdash s \supseteq t} \qquad\qquad \frac{\mathcal{R} \vdash u_1 \supseteq t_1 \quad \ldots \quad \mathcal{R} \vdash u_n \supseteq t_n}{\mathcal{R} \vdash f(u_1, \ldots, u_n) \supseteq f(t_1, \ldots, t_n)}$$

Given Σ-distinct variables $\vec{x} = x_1 : k_1, \ldots, x_n : k_n$ and sorts $\vec{s} = s_1, \ldots, s_n$ with $s_i \in \mathcal{S}_{k_i}$ for each $s_i \in \vec{s}$, the variable sort mapping, *$\mathrm{s}_{\vec{x}:\vec{s}} : T_\Sigma(\vec{x}) \to T_\Sigma(\mathcal{S})$, returns terms in which each variable x_i is replaced by s_i, i.e., $\mathrm{s}_{\vec{x}:\vec{s}}(f(t_1, \ldots, t_n)) = f(\mathrm{s}_{\vec{x}:\vec{s}}(t_1), \ldots, \mathrm{s}_{\vec{x}:\vec{s}}(t_n))$ and $\mathrm{s}_{\vec{x}:\vec{s}}(x_i) = s_i$.*

The intermediate terms *of $\mathcal{R}$, $\mathrm{I}_\mathcal{R} \subseteq T_\Sigma(\mathcal{S})$, are the terms obtained by applying the $\mathrm{s}_{\vec{x}:\vec{s}}$ to the strict subterms of the left-hand side* lhs *of each rule in Γ:*

$$\mathrm{I}_\mathcal{R} = \left\{ \mathrm{s}_{\vec{x}:\vec{s}}(t) \;\middle|\; (\forall \vec{x} : \vec{s})\, A \in \Gamma \wedge (\exists\, \mathbb{C} \neq \square)\ \mathrm{lhs}(A) = \mathbb{C}[t] \right\}$$

Given a choice of potential constructor memberships $M_\Omega \subseteq M$, the sufficient completeness automaton *for $\mathcal{R}$ and M_Ω is the automaton:*

$$\mathcal{A}_{\mathcal{R}/M_\Omega} = \left(\mathcal{K}, \mathcal{F}, \{\mathcal{Q}_k^\Omega\}_{k \in \mathcal{K}}, \{\phi_k^\Omega\}_{k \in \mathcal{K}}, E, \Delta^\Omega \right)$$

in which for each $k \in \mathcal{K}$, the components $\mathcal{Q}_k^\Omega$, ϕ_k^Ω, and Δ^Ω are defined by:

$$\begin{aligned}
\mathcal{Q}_k^\Omega &= \{q_k, r_k\} \cup \{q_s : s \in \mathcal{S}_k\} \cup \{d_s : s \in \mathcal{S}_k\} \cup \{q_t \mid t \in \mathrm{I}_\mathcal{R}\} \\
\phi_k^\Omega &= \neg r_k \wedge \textstyle\bigvee_{s \in \mathcal{S}_k} (d_s \wedge \neg q_s) \\
\Delta^\Omega &= \left\{ f(q_{k_1}, \ldots, q_{k_n}) \to q_k \;\middle|\; f \in \Sigma_{k_1 \ldots k_n, k} \right\} \\
&\cup \left\{ q_s \to q_{s'} \;\middle|\; s < s' \right\} \\
&\cup \left\{ f(q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_1)}, \ldots, q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_n)}) \to q_s \;\middle|\; (\forall \vec{x} : \vec{s})\ f(t_1, \ldots, t_n) : s \in M_\Omega \right\} \\
&\cup \left\{ f(q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_1)}, \ldots, q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_n)}) \to d_s \;\middle|\; (\forall \vec{x} : \vec{s})\ f(t_1, \ldots, t_n) : s \in M_\Delta \right\} \\
&\cup \left\{ f(q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_1)}, \ldots, q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_n)}) \to q_{\mathrm{s}_{\vec{x}:\vec{s}}(f(t_1, \ldots, t_n))} \;\middle|\; (\forall \vec{x} : \vec{s})\, f(t_1, \ldots, t_n) \in \mathrm{I}_\mathcal{R} \right\} \\
&\cup \left\{ f(q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_1)}, \ldots, q_{\mathrm{s}_{\vec{x}:\vec{s}}(t_n)}) \to r_k \;\middle|\; (\forall \vec{x} : \vec{s}) f(t_1, \ldots, t_n) \to u \in R \wedge f \in \mathcal{F}_{\vec{k},k} \right\} \\
&\cup \left\{ f(q_{k_1} \ldots, q_{k_{i-1}}, r_{k_i}, q_{k_{i+1}}, \ldots, q_{k_n}) \to r_k \;\middle|\; f \in \mathcal{F}_{k_1,\ldots,k_n,k} \wedge 1 \leq i \leq n \right\}
\end{aligned}$$

The following theorem assures the correctness of the previous definition.

Theorem 4 (SCA Theorem). *Let $\mathcal{R} = (\Sigma, E, <, R \cup M_\Omega \cup M_\Delta)$ be a PTA-checkable CERM system, and let $\mathcal{R}_\Omega = (\Sigma, E, <, R \cup M_\Omega)$. $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$ iff $\mathcal{L}(\mathcal{A}_{\mathcal{R}/M_\Omega}) = \varnothing$.* □

As an example, the sufficient completeness automaton for the `MSET` specification can be defined as $\mathcal{A}_{\texttt{MSET}} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, E, \Delta)$ in which:

- $\mathcal{K}$ is the single kind `[MSet]`.
- $\mathcal{F}$ contains `0`, `s`, `__`, and `|_|`.
- $\mathcal{Q} = \{\mathtt{q}_{\texttt{[MSet]}}, \mathtt{r}_{\texttt{[MSet]}}, \mathtt{q}_{\texttt{Nat}}, \mathtt{q}_{\texttt{MSet}}, \mathtt{d}_{\texttt{Nat}}, \mathtt{d}_{\texttt{MSet}}, \mathtt{q}_{\texttt{NM}}\}$.
- $\Phi = \neg \mathtt{r}_{\texttt{[MSet]}} \wedge ((\mathtt{d}_{\texttt{Nat}} \wedge \neg \mathtt{q}_{\texttt{Nat}}) \vee (\mathtt{d}_{\texttt{MSet}} \wedge \neg \mathtt{q}_{\texttt{MSet}}))$.
- E contains the associativity and commutativity axioms for `__`.
- Δ contains the following rules:

$$\begin{aligned}
\mathtt{0} &\to \mathtt{q}_{\mathtt{[MSet]}} & \mathtt{s}(\mathtt{q}_{\mathtt{[MSet]}}) &\to \mathtt{q}_{\mathtt{[MSet]}} && (1)\\
\mathtt{q}_{\mathtt{[MSet]}}\ \mathtt{q}_{\mathtt{[MSet]}} &\to \mathtt{q}_{\mathtt{[MSet]}} & \big|\mathtt{q}_{\mathtt{[MSet]}}\big| &\to \mathtt{q}_{\mathtt{[MSet]}} && (2)\\
\mathtt{0} &\to \mathtt{q}_{\mathtt{Nat}} & \mathtt{s}(\mathtt{q}_{\mathtt{Nat}}) &\to \mathtt{q}_{\mathtt{Nat}} && (3)\\
\mathtt{q}_{\mathtt{MSet}}\ \mathtt{q}_{\mathtt{MSet}} &\to \mathtt{q}_{\mathtt{MSet}} & \mathtt{q}_{\mathtt{Nat}} &\to \mathtt{q}_{\mathtt{MSet}} && (4)\\
|\mathtt{q}_{\mathtt{MSet}}| &\to \mathtt{d}_{\mathtt{Nat}} & \mathtt{q}_{\mathtt{Nat}}\ \mathtt{q}_{\mathtt{MSet}} &\to \mathtt{q}_{\mathtt{NM}} && (5)\\
|\mathtt{q}_{\mathtt{Nat}}| &\to \mathtt{r}_{\mathtt{[MSet]}} & |\mathtt{q}_{\mathtt{NM}}| &\to \mathtt{r}_{\mathtt{[MSet]}} && (6)\\
\mathtt{s}(\mathtt{r}_{\mathtt{[MSet]}}) &\to \mathtt{r}_{\mathtt{[MSet]}} & \mathtt{q}_{\mathtt{[MSet]}}\ \mathtt{r}_{\mathtt{[MSet]}} &\to \mathtt{r}_{\mathtt{[MSet]}} && (7)\\
\mathtt{r}_{\mathtt{[MSet]}}\ \mathtt{q}_{\mathtt{[MSet]}} &\to \mathtt{r}_{\mathtt{[MSet]}} & \big|\mathtt{r}_{\mathtt{[MSet]}}\big| &\to \mathtt{r}_{\mathtt{[MSet]}}. && (8)
\end{aligned}$$

Due to (1) and (2), for every term t, we have $t \to \mathtt{q}_{\mathtt{[MSet]}}$. Due to (3) for every natural number n, we have $t \to \mathtt{q}_{\mathtt{Nat}}$. Due to (4) for every multiset m, we have $m \to \mathtt{q}_{\mathtt{MSet}}$. Due to the first rule on (5). $|m| \to \mathtt{d}_{\mathtt{Nat}}$ for each multiset m, and due to the second on (5), we have $nm \to \mathtt{q}_{\mathtt{NM}}$ for each natural number n and multiset m. Due to (6). terms matching equations are rewritten to $\mathtt{r}_{\mathtt{[MSet]}}$, and finally due to (7) and (8), the set of terms rewritable to $\mathtt{r}_{\mathtt{[Mset]}}$ is closed under context.

6 Flattened Automata for Emptiness Testing

In some sense, PTA are an inherently complete and deterministic type of tree automata. This is due to the acceptance criteria of an automaton being determined by the complete set of states reachable from a term. Our approach to solving the emptiness problem for PTA relies on converting a PTA into an equivalent complete and deterministic automaton with a conventional notion of acceptance.

This technique only can be used for PTA with particular classes of equations E. Thus in the remainder of this paper, we focus on the cases where $E = \varnothing$, and the case where E contains the axioms of *associativity and commutativity* (AC) for some of the binary function symbols $f \in \mathcal{F}$, that is,

$$f(f(x,y),z) = f(x, f(y,z)) \qquad \text{and} \qquad f(x,y) = f(y,x).$$

Each symbol in $\mathcal{F}$ must be either free or both associative and commutative.

In the free case ($E = \varnothing$), a PTA can be encoded into a regular tree automaton that accepts an equivalent language. Given a PTA $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, \varnothing, \Delta)$, we define a *many-kinded* TA $\mathcal{A}^d = ((\mathcal{K}, \mathcal{F}), \mathcal{Q}^d, \mathcal{Q}^d_{\mathsf{f}}, \Delta^d)$ in which: $\mathcal{Q}^d$ contains each set of state symbols in $\mathcal{Q}$, i.e. $\mathcal{Q}^d = \{\, \mathcal{P}(\mathcal{Q}_k)\, \}_{k \in \mathcal{K}}$; $\mathcal{Q}^d_{\mathsf{f}}$ consists of subsets of $\mathcal{Q}$ which model the appropriate formula in Φ, i.e. $\mathcal{Q}^d_{\mathsf{f}} = \{\, \mathcal{P}^d_k \,\}_{k \in \mathcal{K}}$ where $\mathcal{P}^d_k = \{P \subseteq \mathcal{Q}_k \mid P \models \phi_k\}$; and finally:

$$\Delta^d = \{\, f(P_1, \ldots, P_n) \to \mathsf{reach}(f, P_1, \ldots, P_n) \mid f \in \mathcal{F}_{\vec{k},k} \wedge (\forall i)\, P_i \subseteq \mathcal{Q}_{k_i} \}$$

where $\mathsf{reach}(f, P_1, \ldots, P_n) = \{\, q \in \mathcal{Q}_k \mid (\exists p_1 \in P_1, \ldots, p_n \in P_n)\, f(p_1, \ldots, p_n) \to_{\mathcal{A}} q \,\}$.

The case of PTA with AC axioms is more complex, since a deterministic regular tree automaton may not still be deterministic when AC axioms are added to its signature. This problem is often resolved for AC symbols by introducing a *flattened* tree structure where the binary AC symbols are replaced by variadic symbols which take a variable number of arguments, and terms are flattened so that no immediate subterm has the same variadic symbol. These tree automata are closely related to tree automata with arithmetical constraints such as multi-tree automata [14] or Presburger tree automata [21], and the emptiness problem for the class of these automata is known to be decidable [15]. Our definition below uses semi-linear sets as described in [6].

Definition 9. *A* flattened tree automaton *(f-TA) is a tuple $\mathcal{A} = (\Sigma, \mathcal{Q}, \mathcal{Q}_{\mathsf{f}}, \Delta)$ where:*

- *$\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{F}_{\mathrm{AC}})$ where $(\mathcal{K}, \mathcal{F})$ is a many-kinded signature and $\mathcal{F}_{\mathrm{AC}}$ is a $\mathcal{K}$-kinded family of variadic AC symbols disjoint from the symbols in $\mathcal{F}$.*
- *$\mathcal{Q} = \{ \mathcal{Q}_k \}_{k \in \mathcal{K}}$ is a $\mathcal{K}$-kinded family of sets of states.*
- *$\mathcal{Q}_{\mathsf{f}} = \{ \mathcal{P}_k \}_{k \in \mathcal{K}}$ is a $\mathcal{K}$-kinded family of sets of final states with each $\mathcal{P}_k \subseteq \mathcal{Q}_k$.*
- *Δ is a set of transition rules, each of which is of one of the following forms:*

  $$f(p_1, \ldots, p_n) \to q \qquad p \to q \qquad f(X) \to q \textbf{ if } \#(X) \in S$$

  where the first two rules follow the restrictions on (Type 1) and (Type 2) rules, and in the third rule $f \in \mathcal{F}_{\mathrm{AC}\,k}$, X is a variable matching any list of states in $\mathcal{Q}_k^*$, and $S \subseteq \mathbb{N}^{\mathcal{Q}_k}$ is a semi-linear set. The function $\# : \mathcal{Q}_k^* \to \mathbb{N}^{\mathcal{Q}_k}$, called the Parikh mapping *[6] maps a list of states to a vector representing the number of occurrences of each state in the list.*

The language accepted by the f-TA $\mathcal{A}$, $\mathcal{L}(\mathcal{A}) = \{ \mathcal{L}(\mathcal{A})_k \}_{k \in \mathcal{K}}$, is defined as usual:

$$\mathcal{L}(\mathcal{A})_k = \{ t \in T_{\Sigma,k} \mid (\exists q \in \mathcal{P}_k)\ t \to_{\mathcal{A}} q \}$$

To define a notation of equivalence between a signature with AC symbols and flattened languages, we define the mapping $\mathsf{flat} : T_{(\mathcal{K},\mathcal{F})} \to T_{(\mathcal{K},\mathcal{F} \cup \mathcal{F}_{\mathrm{AC}})}$ by the equations $\mathsf{flat}(g(t_1, \ldots, t_n)) = g(\mathsf{flat}(t_1), \ldots, \mathsf{flat}(t_n))$ if g is a free symbol, and $\mathsf{flat}(\mathbb{C}[t_1, \ldots, t_n]) = f(\mathsf{flat}(t_1), \ldots, \mathsf{flat}(t_n))$ when $\mathbb{C}$ is a maximal context involving only occurrences of an AC-symbol f. For each AC-PTA $\mathcal{A}$, there is a flattened f-TA $\mathcal{A}_{\mathsf{det}}$ accepting an equivalent language which can be constructed.

Definition 10. *Let $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, E, \Delta)$ be a AC-PTA. For each AC symbol $f \in \mathcal{F}_{kk,k}$ and state $q \in \mathcal{Q}_k$, let $\mathcal{G}_{fq} = (\mathcal{Q}_k, \mathcal{P}(\mathcal{Q}_k), q, \Lambda)$ be the context free grammar in which $\mathcal{Q}_k$ is the set of non-terminal symbols, $\mathcal{P}(\mathcal{Q}_k)$ is the set of terminal symbols, q is the initial symbol, and Λ contains the rules:*

$$\Lambda = \{q \to p_1 p_2 \mid f(p_1, p_2) \to q \in \Delta\} \cup \{q \to P \mid P \subseteq \mathcal{Q} \wedge q \in P\}\,.$$

From a theorem of Parikh [20], we know there is a semilinear set equal to the commutative closure of a context free language. Accordingly we will use $S_{fq} \subseteq \mathbb{N}^{\mathcal{P}(\mathcal{Q})}$ to denote the the commutative closure of the language $\mathcal{L}(\mathcal{G}_{fq})$.

Let $\mathcal{A}_{\mathsf{det}} = (\Sigma, \mathcal{F}_{\mathrm{AC}}, \mathcal{Q}^d, \mathcal{Q}_{\mathsf{f}}^d, \Delta^d)$ in which: $\Sigma = (\mathcal{K}, \mathcal{F}^d, \mathcal{F}_{\mathrm{AC}})$ where the $\mathcal{F}^d$ contains the free symbols and $\mathcal{F}_{\mathrm{AC}}$ the AC symbols of $\mathcal{F}$; as in the free case, $\mathcal{Q}^d = \{ \mathcal{P}(\mathcal{Q}_k) \}_{k \in \mathcal{K}}$ and $\mathcal{Q}_{\mathsf{f}}^d = \{ P_k^d \}_{k \in \mathcal{K}}$ with $P_k^d = \{P \subseteq \mathcal{Q}_k \mid P \models \phi_k\}$; and finally $\Delta^d = \Delta_1 \cup \Delta_2$:

$$\Delta_1 = \{ f(P_1, \ldots, P_n) \to \mathsf{reach}(f, P_1, \ldots, P_n) \mid f \in \mathcal{F}_{\vec{k},k} \wedge (\forall i)\, P_i \subseteq \mathcal{Q}_{k_i} \}$$

$$\Delta_2 = \{ f(X) \to P \textbf{ if } \sharp(X) \in \Big(\bigcap_{q \in A} S_{fq} \cap \bigcap_{q \in \mathcal{Q} \setminus A} (S_{fq})^{\mathsf{c}} \cap S_{\geqslant 2} \Big) \mid f \in \mathcal{F}_{\mathrm{AC}\,k} \wedge P \subseteq \mathcal{Q}_k \}$$

Theorem 5 (Flattening Theorem). *Let $\mathcal{A}$ be a AC-PTA. For each $t \in T_{\Sigma}$,*

$$t \in \mathcal{L}(\mathcal{A}) \iff \mathsf{flat}(t) \in \mathcal{L}(\mathcal{A}_{\mathsf{det}}).$$

It can be easily shown that emptiness of f-TA is decidable, and consequentially that sufficient completeness of AC-PTA checkable system is decidable.

7 Conclusions

We have presented three contributions advancing methods for proving sufficient completeness to handle conditional specifications involving partial functions and where deduction is performed modulo axioms. Specifically, we have studied the sufficient completeness of such specifications and their associated rewriting systems in greater generality than it had been done up to now, have introduced a new kind of equational tree automata (PTA), and have identified a subclass of specifications whose sufficient completeness problem is equivalent to the emptiness problem for their associated PTA. Furthermore, we have shown that the emptiness problem is decidable for AC-PTA, thus making the sufficient completeness of AC-PTA-checkable specifications decidable.

A number of further extensions of this work seem worth investigating. First, the results for AC-PTA should be extended to allow symbols with different combinations of associativity, commutativity, and identity axioms. Such axioms are used quite frequently and supported by languages such as Maude. This would allow us to decide sufficient completeness for a much wider class of PTA-checkable specifications. A problematic case would be specifications with symbols that are associative but not commutative, because of well-known undecidability results for the corresponding equational tree automata [19]. However, decidability seems unproblematic in all other cases. A second important topic is the generation of counterexamples to show lack of sufficient completeness: ground term counterexamples are practical and easy to generate, but investigating ways of symbolically describing sets of counterexamples may be quite useful for other purposes, such as generating induction schemes for theorem provers. A third topic worth investigating is what we called the "second prong" in the introduction, namely, integrating sufficient completeness checking and inductive theorem proving in order to handle specifications outside the decidable subclasses. Further advances in these three areas should provide both foundations and algorithms in which to build a next-generation PTA-based sufficient completeness tool for MEL specifications modulo axioms. This would make sufficient completeness checking available for a very wide class of specifications in Maude and other equational languages for specification and programming with advanced features.

References

1. A. Bouhoula and M. Rusinowitch: *SPIKE: A System for Automatic Inductive Proofs*, Proc. of 4th AMAST, Montreal (Canada), LNCS 936, pp. 576–577, 1995.
2. A. Bouhoula, J.P. Jouannaud and J. Meseguer: *Specification and Proof in Membership Equational Logic*, TCS 236, pp. 35–132, 2000.

3. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer and J.F. Quesada: *Maude Specification and Programming in Rewriting Logic*, TCS 285, pp. 187–243, 2002.
4. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison and M. Tommasi: *Tree Automata Techniques and Applications (TATA)*, draft, 2002. Available at URL: http://www.grappa.univ-lille3.fr/tata
5. F. Gécseg and M. Steinby: *Tree Languages*, Handbook of Formal Languages 3, pp. 1–68 (Chap. 1), Springer-Verlag, 1996.
6. S. Ginsburg: *The Mathematical Theory of Context-Free Languages*, McGraw-Hill, 1966.
7. J.V. Guttag: *The Specification and Application to Programming of Abstract Data Types*, Ph.D. thesis, Computer Science Department, University of Toronto, 1975.
8. J.V. Guttag and J.J. Horning: *The Algebraic Specification of Abstract Data Types*, Acta Inf. 10, pp. 27–52, 1978.
9. J. Hendrix, M. Clavel and J. Meseguer: *A Sufficient Completeness Reasoning Tool for Partial Specifications*, Proc. of 16th RTA, Nara (Japan), LNCS 3467, pp. 165–174, 2005.
10. J.P. Jouannaud and E. Kounalis: *Automatic Proofs by Induction in Equational Theories without Constructors*, Inf. Comput. 81(1), pp. 1–33, 1989.
11. D. Kapur: *An Automated Tool for Analyzing Completeness of Equational Specifications*, Proc. of ISSTA'94, Seattle (USA), pp. 28–43, ACM Press, 1994.
12. D. Kapur, P. Narendran and H. Zhang: *On Sufficient-Completeness and Related Properties of Term Rewriting Systems*, Acta Inf. 24, pp. 395–415, 1987
13. D. Kapur, P. Narendran, D.J. Rozenkrantz and H. Zhang: *Sufficient-Completeness, Ground-Reducibility and Their Complexity*, Acta Inf. 28, pp. 311–350, 1991.
14. D. Lugiez and S. Dal Zilio: *XML Schema, Tree Logic and Sheaves Automata*, Proc. of 14th RTA, Valencia (Spain), LNCS 2706, pp. 246–263, 2003.
15. D. Lugiez, S. Dal Zilio and C. Meyssonnier: *A Logic You can Count on*, Proc. of 31st POPL, Venice (Italy), pp. 135–146, ACM Press, 2004.
16. J. Meseguer: *Membership Algebra as a Logical Framework for Equational Specification*, Proc. of WADT'97, Tarquinia (Italy), LNCS 1376, pp. 18–61, 1997.
17. T. Nipkow and G. Weikum: *A Decidability Result about Sufficient Completeness of Axiomatically Specified Abstract Data Types*, Proc. of 6th GI Conference, Dortmund (Germany), LNCS 145, pp. 257–268, 1983,
18. H. Ohsaki: *Beyond Regularity: Equational Tree Automata for Associative and Commutative Theories*, Proc. of 15th CSL, Paris (France), LNCS 2142, pp. 539–553, 2001.
19. H. Ohsaki and T. Takai: *Decidability and Closure Properties of Equational Tree Languages*, Proc. of 13th RTA, Copenhagen (Denmark), LNCS 2378, pp. 114–128, 2002.
20. R.J. Parikh: *On Context-Free Languages*, JACM 13(4), pp. 570–581, 1966.
21. H. Seidl, T. Schwentick and A. Muscholl: *Numerical Document Queries*, Proc. of 22nd PODS, San Diego (USA), pp. 155–166, ACM Press, 2003.
22. K.N. Verma: *Two-Way Equational Tree Automata for AC-Like Theories: Decidability and Closure Properties*, Proc. of 14th RTA, Valencia (Spain), LNCS 2706, pp. 165–179, 2003.

A Appendix: Proofs

This section contains proofs of each theorem stated in the previous sections. The first theorem proved will be the Equivalence Theorem (Theorem 1), but first we introduce the following lemmas:

Lemma 1 (General Transitivity). *$\mathcal{R} \vdash t \to u$ and $\mathcal{R} \vdash u \to v$ imply $\mathcal{R} \vdash t \to v$.*

Proof. Use structural induction on a proof tree of $\mathcal{R} \vdash t \to u$. $\square$

Corollary 1 (General Subject Reduction). *$\mathcal{R} \vdash t \to u$ and $\mathcal{R} \vdash u : s$ imply $\mathcal{R} \vdash t : s$.*

Proof. There must be a v such that $\mathcal{R} \vdash u \to v$ and $\mathcal{R} \vdash v :^0 s$. $\mathcal{R} \vdash t \to v$ by General Transitivity, and $\mathcal{R} \vdash t : s$ by Subject Reduction $\square$

Lemma 2 (Subsort Lemma). *$\mathcal{R} \vdash t : s$ and $s < s'$ imply $\mathcal{R} \vdash t : s'$.*

Proof. There must be a v such that $\mathcal{R} \vdash t \to v$ and $\mathcal{R} \vdash v :^0 s$. Any proof of $\mathcal{R} \vdash v :^0 s$ is also a proof $\mathcal{R} \vdash v :^0 s'$, and thus $\mathcal{R} \vdash t : s'$.

Lemma 3 (Context Lemma). *$\mathcal{R} \vdash t \to u$ implies $\mathcal{R} \vdash \mathbb{C}[t] \to \mathbb{C}[u]$ for any context $\mathbb{C}$.*

Proof. Use structural induction on a proof tree of $\mathcal{R} \vdash t \to u$. $\square$

Corollary 2 (General Congruence). *$\mathcal{R} \vdash t_1 \to u_1, \ldots, \mathcal{R} \vdash t_n \to u_n$ implies $\mathcal{R} \vdash f(t_1, \ldots, t_n) \to f(u_1, \ldots, u_n)$.*

Proof. Induct on the number of distinct positions j such that $t_i \neq u_i$. If $j = 0$, use Equivalence. Otherwise, assume that the conjecture holds for j positions and that there are $j+1$ distinct positions such that $t_i \neq u_i$. If we fix i to a particular position where $t_i \neq u_i$, by induction we have

$$\mathcal{R} \vdash f(t_1, \ldots, t_n) \to f(u_1, \ldots, u_{i-1}, t_i, u_{i+1}, \ldots, u_n).$$

By the conditions of the theorem and the context lemma, we have

$$\mathcal{R} \vdash f(u_1, \ldots, u_{i-1}, t_i, u_{i+1}, \ldots, u_n) \to f(u_1, \ldots, u_n).$$

We are done by General Transitivity. $\square$

Corollary 3 (Substitution Congruence). *Let $\vec{x}$ and $\vec{y}$ be Σ-distinct variables, let $t \in T_\Sigma(\vec{x})$, and let $\theta, \tau : \vec{x} \to T_\Sigma(\vec{y})$ be substitutions. If $\mathcal{R} \vdash \theta(x) \to \tau(x)$ for each $x \in \vec{x}$, then $\mathcal{R} \vdash t\theta \to t\tau$.*

Proof. Use structural induction on t. If t is a variable $x \in \vec{x}$, then by assumption, $\mathcal{R} \vdash \theta(x) \to \tau(x)$. Otherwise, t is of the form $f(t_1, \ldots, t_n)$ and by our induction hypothesis $\mathcal{R} \vdash t_i\theta \to t_i\tau$ for each $i \in \{1, \ldots, n\}$. By General Congruence, $\mathcal{R} \vdash t\theta \to t\tau$. $\square$

In a way similar to our definition of pattern-based CERM systems, we define what it means for a conjunction of atomic formulas to be pattern based. Specifically a conjunction of atomic formulas, $u_1 \to v_1 \wedge \cdots \wedge u_n \to v_n$, is *pattern based* when for each v_i:

- The variables in v_i are fresh[2], i.e.

$$\mathrm{vars}(v_i) \cap \big(\mathrm{vars}(l) \cup \bigcup_{1 \le k \le i} \mathrm{vars}(u_k) \cup \bigcup_{1 \le k < i} \mathrm{vars}(v_k)\big) = \varnothing$$

– Any rewriting of a term $v_i\theta$ occurs below the pattern v_i, i.e., for Σ-distinct variables $\vec{x}$ and substitution $\theta : \vec{y} \to T_\Sigma(\vec{x})$, if there is a term $t \in T_\Sigma$ such that $\mathcal{R} \vdash v_i\theta \to t$, then there is a substitution $\tau : \vec{y} \to T_\Sigma(\vec{x})$ where $t =_E v_i\tau$.

Lemma 4. *Let $\mathcal{R}$ be a CERM that is confluent relative to variables $\vec{x}$. Given a pattern-based conjunction of atomic formulas $t_1 \to u_1 \wedge \cdots \wedge t_n \to u_n$ containing variables $\vec{y}$, if $\theta : \vec{y} \to T_\Sigma(\vec{x})$ is a substitution such that $\mathcal{R} \vdash t_i\theta \downarrow u_i\theta$ for each $i \in \{1, \dots, m\}$, then there is a substitution $\tau : \vec{y} \to T_\Sigma(\vec{x})$ such that:*

$$\begin{aligned} &\mathcal{R} \vdash \theta(y) \to \tau(y) && \textit{for each } y \in \vec{y} \\ &\mathcal{R} \vdash t_i\tau \to u_i\tau && \textit{for each } i \in \{1, \dots, m\} \end{aligned}$$

Proof. For a substitution ϕ, let k_ϕ be the smallest index i such that $\mathcal{R} \nvdash t_i\phi \to u_i\phi$. If no such index exists, then let $k_\phi = n + 1$. We prove this theorem by an induction scheme in which we assume the theorem holds for every substitution ϕ where $k_\phi > k_\theta$.

If $k_\theta = n + 1$, then observe that $\tau = \theta$ satisfies the required properties, and we are done. Otherwise, let $k_\theta = i$ with $i \leq n$. By the confluence of $\mathcal{R}$ and the condition that the conjunction is pattern-based, we know there exists a substitution ϕ such that $\mathcal{R} \vdash t_i\theta \to u_i\phi$ and $\mathcal{R} \vdash u_i\theta \to u_i\phi$. Let τ be the substitution:

$$\begin{aligned} &\tau(y) = \phi(y) && \text{if } y \in \text{vars}(u_i) \\ &\tau(y) = \theta(y) && \text{otherwise} \end{aligned}$$

Observe that $t_j\theta = t_j\tau$ for $j \leq i$, $u_j\theta = u_j\tau$ for $j < i$, and $u_i\phi = u_i\tau$. Collectively this implies that $\mathcal{R} \vdash t_1\tau \to u_1\tau, \dots, \mathcal{R} \vdash t_i\tau \to u_i\tau$. Thus, $k_\tau > i$ and by induction we are done. □

Finally, we are ready to prove Theorem 1.

Theorem (Inference Equivalence). *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be a CERM system that is weakly-convergent relative to Σ-distinct variables $\vec{x}$. For all $t, u \in T_\Sigma(\vec{x})$ and $s \in \mathcal{S}$,*

$$\mathcal{E}_\mathcal{R} \vdash t = u \iff \mathcal{R} \vdash t \downarrow u \qquad \textit{and} \qquad \mathcal{E}_\mathcal{R} \vdash t : s \iff \mathcal{R} \vdash t : s.$$

The inference system of $\mathcal{E}_\mathcal{R}$ can be found in Fig. 4 of [2].

Proof. As a typographical convention we will use **Bold** for rules in Fig. 4 of [2] and Sans Serif for rules in Fig. 1. The $\Leftarrow$ direction can be easily shown by structural induction on proof formed from the rules in Fig. 1. The $\Rightarrow$ direction can be shown by structural induction on proofs formed from the rules in Fig. 4 of [2]. We next split into different cases depending on which inference rule is used at the top of the proof tree.

Subject Reduction: $\dfrac{t = u \qquad u : s}{t : s}$

By induction, $\mathcal{R} \vdash u : s$ and there must be a $v \in T_\Sigma(\vec{x})$ such that $\mathcal{R} \vdash t \to v$ and $\mathcal{R} \vdash u \to v$. As $\mathcal{R}$ is sort-decreasing, $\mathcal{R} \vdash v : s$, and thus $\mathcal{R} \vdash t : s$.

Membership:

$$\frac{u_1\theta = v_1\theta \ \ldots\ u_m\theta = v_m\theta \qquad w_1\theta : s_1 \ \ldots\ w_n\theta : s_n}{t\theta : s}$$

If the membership used is in $M_<$, then it is of the form $(\forall x : k)\, x : s$ **if** $x : s'$ with $s, s' \in \mathcal{S}_k$ and $s' < s$. By induction, $\mathcal{R} \vdash t\theta : s'$, and therefore $\mathcal{R} \vdash t\theta : s$. Otherwise, by induction $\mathcal{R} \vdash u_i\theta \downarrow v_i\theta$ for $i \in \{1, \ldots, m\}$, and $\mathcal{R} \vdash w_j\theta : s_j$ for $j \in \{1, \ldots, n\}$. There must exist a membership in Γ with the form:

$$(\forall \vec{y})\, t : s \textbf{ if } \bigwedge_{i=1}^{m} u_i \to v_i \wedge \bigwedge_{j=1}^{n} w_j : s_j$$

As $\mathcal{R}$ is pattern-based, by Lemma 4, there is a substitution $\tau : \vec{y} \to T_\Sigma(\vec{x})$ such that $\mathcal{R} \vdash \theta(y) \to \tau(y)$ for each $y \in \vec{y}$, and $\mathcal{R} \vdash u_i\tau \to v_i\tau$ for each $i \in \{1, \ldots, m\}$. By Corollary 3, $\mathcal{R} \vdash w_j\theta \to w_j\tau$ for $j \in \{1, \ldots, n\}$. As $\mathcal{R}$ is sort-decreasing, $\mathcal{R} \vdash w_j\tau \to s_j$ for $j \in \{1, \ldots, n\}$ We then have $\mathcal{R} \vdash t\tau : s$. $\mathcal{R} \vdash t\theta \to t\tau$ by Corollary 3, and therefore $\mathcal{R} \vdash t\theta : s$.

Reflexivity: $\dfrac{}{t = t}$

Clearly $\mathcal{R} \vdash t \to t$.

Symmetry: $\dfrac{t = u}{u = t}$

By induction we have $\mathcal{R} \vdash t \downarrow u$, which is commutative, and thus $\mathcal{R} \vdash u \downarrow t$.

Transitivity: $\dfrac{t = u \qquad u = v}{t = v}$

$\mathcal{R} \vdash t \downarrow u$ and $\mathcal{R} \vdash u \downarrow v$ by induction. Using confluence, we can construct the diagram:

$$\begin{tikzcd}
t \arrow[dr] & & u \arrow[dl] \arrow[dr] & & v \arrow[dl] \\
& w \arrow[dr] & & x \arrow[dl] & \\
& & y & &
\end{tikzcd}$$

Congruence:

$$\frac{t_1 = u_1 \ \ldots\ t_n = u_n}{f(t_1, \ldots, t_n) = f(u_1, \ldots, u_n)}$$

By induction, for each $i \in \{1, \ldots, n\}$, there is a term $v_i \in T_\Sigma(\vec{x})$ such that $\mathcal{R} \vdash t_i \to v_i$ and $\mathcal{R} \vdash u_i \to v_i$. Thus by General congruence, $\mathcal{R} \vdash f(t_1, \ldots, t_n) \downarrow f(u_1, \ldots, u_n)$.

Replacement:

$$\frac{u_1\theta = v_1\theta \ \ldots\ u_m\theta = v_m\theta \qquad w_1\theta : s_1 \ \ldots\ w_n\theta : s_n}{t\theta = t'\theta}$$

If the equation used is in E, we are done by Equivalence. Otherwise using an inductive argument identical to that of the **Membership** rule, we can show there must exist a substitution τ such that $\mathcal{R} \vdash t\theta \to t\tau$, $\mathcal{R} \vdash t'\theta \to t'\tau$, and $\mathcal{R} \vdash t\tau \to t'\tau$, and we are done. □

We next prove Theorem 2.

Theorem (Suff. Comp. Equivalence). *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$, and let $M_\Omega \subseteq M$ be a subset of memberships in $\mathcal{R}$.*

– *If $\mathcal{R}$ is ground weakly-convergent and sufficiently complete with respect to $\mathcal{R}_\Omega$, then $\mathcal{E}_\mathcal{R}$ is sufficiently complete with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$.*
– *If $\mathcal{R}_\Omega$ is ground weakly-convergent, and $\mathcal{E}_\mathcal{R}$ is sufficiently complete with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$, then $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$.* □

Proof. Recalling Definition 3, $\mathcal{R}$ is *sufficiently complete* with respect to $\mathcal{R}_\Omega$ when for every $t, u \in T_\Sigma$ and $s \in \mathcal{S}$:

$$\mathcal{R} \vdash t{\downarrow}u \iff \mathcal{R}_\Omega \vdash t{\downarrow}u \qquad \text{and} \qquad \mathcal{R} \vdash t : s \iff \mathcal{R}_\Omega \vdash t : s \tag{9}$$

Additionally, $\mathcal{E}_\mathcal{R}$ is *sufficiently complete* with respect to $\mathcal{E}_{\mathcal{R}_\Omega}$ when for every $t, u \in T_\Sigma$ and $s \in \mathcal{S}$:

$$\mathcal{E}_\mathcal{R} \vdash t = u \iff \mathcal{E}_{\mathcal{R}_\Omega} \vdash t = u \qquad \text{and} \qquad \mathcal{E}_\mathcal{R} \vdash t : s \iff \mathcal{E}_{\mathcal{R}_\Omega} \vdash t : s \tag{10}$$

Observe that in both (9) and (10), the $\Leftarrow$ direction is trivial, and so we only need to prove the $\Rightarrow$ direction. To prove the first part of the theorem, note:

$$\begin{aligned}
\mathcal{E}_\mathcal{R} \vdash t = u &\iff \mathcal{R} \vdash t{\downarrow}u \iff \mathcal{R}_\Omega \vdash t{\downarrow}u \Rightarrow \mathcal{E}_{\mathcal{R}_\Omega} \vdash t = u\\
\mathcal{E}_\mathcal{R} \vdash t : s &\iff \mathcal{R} \vdash t : s \iff \mathcal{R}_\Omega \vdash t : s \Rightarrow \mathcal{E}_{\mathcal{R}_\Omega} \vdash t : s
\end{aligned}$$

To prove the second part, note:

$$\begin{aligned}
\mathcal{R} \vdash t{\downarrow}u \Rightarrow \mathcal{E}_\mathcal{R} \vdash t = u &\iff \mathcal{E}_{\mathcal{R}_\Omega} \vdash t = u \iff \mathcal{R}_\Omega \vdash t{\downarrow}u\\
\mathcal{R} \vdash t : s \Rightarrow \mathcal{E}_\mathcal{R} \vdash t : s &\iff \mathcal{E}_{\mathcal{R}_\Omega} \vdash t : s \iff \mathcal{R}_\Omega \vdash t : s
\end{aligned}$$

□

To prove the Checking Theorem (Theorem 3), we first define the noetherian relation $\gg$ over $\Sigma(\vec{x})$-sentences.

Definition 11. *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be a CERM system that is weakly normalizing relative to Σ-distinct variables $\vec{x}$, and let $\succ_\mathcal{R}$ be the reductive order used to show $\mathcal{R}$ is weakly normalizing. The noetherian strict $\gg$ over $\Sigma(\vec{x})$-sentences can be defined as follows:*

$$\begin{aligned}
t \to u &\gg v \to w && \textit{if } t \succ_\mathcal{R} v, \textit{ or } t =_E v \textit{ and } u \succ_\mathcal{R} w\\
t \to u &\gg v : s' && \textit{if } t \succ_\mathcal{R} v\\
t : s &\gg v \to w && \textit{if } t \succ_\mathcal{R} v, \textit{ or } t =_E v \textit{ and } t \succ_\mathcal{R} w\\
t : s &\gg v : s' && \textit{if } t \succ_\mathcal{R} v
\end{aligned}$$

with $t, u, v, w \in T_\Sigma(\vec{x})$ and $s, s' \in \mathcal{S}$

We also prove the following lemma:

Lemma 5. *Let $\mathcal{R}$ be a CERM that is weakly normalizing relative to Σ-distinct variables $\vec{x}$, and let $\succ_{\mathcal{R}}$ be the reductive order used to show $\mathcal{R}$ is weakly normalizing. If $\mathcal{R} \vdash t \to u$ with $t, u \in T_\Sigma(\vec{x})$, then $t \succ_{\mathcal{R}} u$ iff $t \neq_E u$.*

Proof. By induction on proof trees that are reductive with respect to $\succ_{\mathcal{R}}$ using the noetherian order $\gg$, the order of $\Sigma(\vec{x})$-sentences formed from $\succ_{\mathcal{R}}$.

Finally, we can prove the Checking Theorem

Theorem (Checking Theorem). *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$ be a ground weakly normalizing and ground sort-decreasing CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$. Given a set of memberships $M_\Omega \subseteq M$, let $\mathcal{R}_\Omega = (\Sigma, E, <, R \cup M_\Omega)$. $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$ iff for every membership*

$$(\forall \vec{y})\, t : s \textbf{ if } \bigwedge_{i=1}^{m} u_i \to v_i \wedge \bigwedge_{j=1}^{n} w_j : s_j \tag{11}$$

in M_Δ, and every substitution $\theta : \vec{y} \to T_\Sigma$, if $t\theta$ is $\mathcal{R}_\Omega$-irreducible, $\mathcal{R}_\Omega \vdash u_i\theta \to v_i\theta$ for every $i \in \{1, \ldots, m\}$, and $\mathcal{R}_\Omega \vdash w_j\theta : s_j$ for every $j \in \{1, \ldots, n\}$, then $\mathcal{R}_\Omega \vdash t\theta : s$.

Proof. For every membership of form (11) and substitution $\theta : \vec{y} \to T_\Sigma$ satisfying the above requirements, $\mathcal{R} \vdash t\theta : s$. If $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$, then clearly $\mathcal{R}_\Omega \vdash t\theta : s$. To prove the theorem in the other direction, we show that for every $t, u \in T_\Sigma$ and $s \in \mathcal{S}$:

$$\mathcal{R} \vdash t \to u \Rightarrow \mathcal{R}_\Omega \vdash t \to u \qquad\qquad \mathcal{R} \vdash t : s \Rightarrow \mathcal{R}_\Omega \vdash t : s \tag{12}$$

If this is true, then $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$. As $\mathcal{R}$ is weakly normalizing, for every atomic formula A such that $\mathcal{R} \vdash A$, A can be proved with a proof that is reductive with respect to $\succ_{\mathcal{R}}$. Let $\gg$ is the order of atomic formulas formed from $\succ_{\mathcal{R}}$. We prove (12) by induction over $\gg$ on $\mathcal{R}$-proofs that are reductive with respect to $\succ_{\mathcal{R}}$ If the top of the proof is an Equivalence rule:

$$\frac{t =_E u}{t \to u}$$

clearly $\mathcal{R}_\Omega \vdash t \to u$. If the top of the proof tree is a Transitivity rule, the left antecedent must be a Rewrite rule:

$$\frac{\dfrac{t =_E \mathbb{C}[l\theta] \qquad \bigwedge_{i=1}^{m} u_i\theta \to v_i\theta \qquad \bigwedge_{j=1}^{n} w_j\theta : s_j}{t \to^1 \mathbb{C}[r\theta]} \qquad \mathbb{C}[r\theta] \to t'}{t \to t'}$$

As the proof is in reductive relative to $\succ_{\mathcal{R}}$, we have $t \to t' \gg \mathbb{C}[r\theta] \to t'$, $t \to t' \gg u_i\theta \to v_i\theta$ for each $i \in \{1, \ldots, m\}$, $t \to t' \gg w_j\theta : s_j$ for each $j \in \{1, \ldots, n\}$. Thus, by induction they are provable in $\mathcal{R}_\Omega$, and $\mathcal{R}_\Omega \vdash t \to t'$. If the top of the proof is a Subject Reduction rule, the right antecedent must be a Membership rule:

$$\frac{t \to u \qquad \dfrac{u =_E l\theta \qquad \bigwedge_{i=1}^{m} u_i\theta \to v_i\theta \qquad \bigwedge_{j=1}^{n} w_j\theta : s_j}{u :^0 s}}{t : s}$$

If $t \succ_{\mathcal{R}} u$, by our induction hypothesis, $\mathcal{R}_\Omega \vdash t \to u$ and $\mathcal{R}_\Omega \vdash u : s$. By General Subject Reduction, we are done. Otherwise, by Lemma 5, $t =_E u$. By induction, $\mathcal{R}_\Omega \vdash u_i\theta \to v_i\theta$ for each $i \in \{1,\dots,m\}$, and $\mathcal{R}_\Omega \vdash w_j\theta : s_j$ for each $j \in \{1,\dots,n\}$. If the membership used to prove $\mathcal{R} \vdash u :^0 s$ is in M_Ω, $\mathcal{R}_\Omega \vdash u :^0 s$ and consequentially $\mathcal{R}_\Omega \vdash t : s$. Otherwise, the membership used to prove $\mathcal{R} \vdash u :^0 s$ is in M_Δ. If t is $\mathcal{R}$-irreducible, then u must be $\mathcal{R}_\Omega$-irreducible. Then by the induction hypothesis $\mathcal{R}_\Omega \vdash u : s$ and consequentially $\mathcal{R}_\Omega \vdash t : s$. If t is $\mathcal{R}$-reducible, then as $\mathcal{R}$ is ground weakly normalizing and ground sort-decreasing of $\mathcal{R}$, there is an irreducible $t' \in T_\Sigma$ such that $\mathcal{R} \vdash t \to t'$ and $\mathcal{R} \vdash t' : s$. As t is $\mathcal{R}$-reducible, t' is $\mathcal{R}$-irreducible, $t \neq_E t'$. By Lemma 5, $t \succ_\mathcal{R} t'$. Thus, $t : s \gg t \to t'$ and $t : s \gg t' : s$. By our induction hypothesis, $\mathcal{R}_\Omega \vdash t \to t'$ and $\mathcal{R}_\Omega \vdash t' : s$. Thus by General Subject Reduction, $\mathcal{R}_\Omega \vdash t : s$. □

The next theorem to prove is the correctness proof of the construction of the sufficient completeness PTA for a PTA-checkable system. Given a PTA $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, E, \Delta)$, let $\mathcal{A}^\varnothing = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, \varnothing, \Delta)$ be the PTA in which the equations E in $\mathcal{A}$ have been removed. The following lemma was proved in [18]. It has been generalized for propositional tree automata.

Lemma 6. *Let $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Phi, E, \Delta)$ be a PTA. For every t $inT_{(\mathcal{K},\mathcal{F})}$ and $q \in \mathcal{Q}$:*

$$t \to_\mathcal{A} u \iff (\exists t' \in T_{(\mathcal{K},\mathcal{F})})\, t =_E t' \wedge t' \to_{\mathcal{A}^\varnothing} u$$

□

Lemma 7. *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$ be a PTA-checkable CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$, let $M_\Omega \subseteq M$, and let $\mathcal{R}_{M_\Omega} = (\Sigma, E, <, M_\Omega)$. For every $t \in T_\Sigma$, $s \in \mathcal{S}$, and $u \in \mathrm{I}_\mathcal{R}$:*

$$t \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} q_s \iff \mathcal{R}^\varnothing_{M_\Omega} \vdash t : s \qquad \text{and} \qquad t \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} q_u \iff \mathcal{R}^\varnothing_{M_\Omega} \vdash u \supseteq t$$

Proof. By structural induction on t. □

Corollary 4. *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$ be a PTA-checkable CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$, let $M_\Omega \subseteq M$, and let $\mathcal{R}_\Omega = (\Sigma, E, <, R \cup M_\Omega)$. For every $t \in T_\Sigma$, $k \in \mathcal{K}$:*

$$t \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} r_k \iff t \text{ is } \mathcal{R}^\varnothing_\Omega\text{-reducible}$$

Proof. By simultaneous structural induction on t and $\mathbb{C}$, we can show that for every $k \in \mathcal{K}$ and $t \in T_\Sigma$, $t \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} r_k$ when there is a context $\mathbb{C} \in T_\Sigma(\Box : k')_k$, a rule $(\forall \vec{y} : \vec{s})\, l \to r \in R$ with $l \in T_{\Sigma,k'}$, and a substitution $\theta : \vec{y} \to T_\Sigma$ such that $t = \mathbb{C}[l\theta]$ and $\theta(y_i) \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} q_{s_i}$ for each $y_i \in \vec{y}$,

Let $\mathcal{R}_{M_\Omega} = (\Sigma, E, <, M_\Omega)$. By Lemma 7, we know that $\theta(y_i) \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} q_{s_i}$ iff $\mathcal{R}^\varnothing_{M_\Omega} \vdash \theta(y_i) : s_i$ for each $y_i \in \vec{y}$. Thus, we can show that $\mathbb{C}[l\theta] \to_{\mathcal{A}^\varnothing_{\mathcal{R}/M_\Omega}} r_k$ iff $\mathcal{R}^\varnothing_\Omega \vdash \mathbb{C}[l\theta] \to^1 \mathbb{C}[r\theta]$, and consequently, t is $\mathcal{R}^\varnothing$-reducible. □

Corollary 5. *Let $\mathcal{R} = (\Sigma, E, <, R \cup M)$ be a PTA-checkable CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$, let $M_\Omega \subseteq M$, and let $\mathcal{R}_{M_\Omega} = (\Sigma, E, <, M_\Omega)$. For every*

$t \in T_\Sigma$, $s \in \mathcal{S}$:

$$t \rightarrow_{\mathcal{A}^{\varnothing}_{\mathcal{R}/M_\Omega}} d_s \iff (\exists\ (\forall \vec{x} : \vec{s})\, l : s \in M_\Delta)\ \mathcal{R}^{\varnothing}_{M_\Omega} \vdash \mathrm{s}_{\vec{x}:\vec{s}}(l) \supseteq t$$

Proof. By definition of $\mathcal{A}_{\mathcal{R}/M_\Omega}$ and Lemma 7.

Lemma 8. *Let $\mathcal{R} = (\Sigma, E, <, M)$ be a PTA-checkable CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$ and M is a set only containing conditional memberships. For every $t \in T_\Sigma$ and $s \in \mathcal{S}$:*

$$\mathcal{R} \vdash t : s \iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge \mathcal{R}^{\varnothing} \vdash t' : s$$

Proof. The $\Leftarrow$ direction is trivial. The $\Rightarrow$ direction can be proved by structural induction on the proof used to show $\mathcal{R} \vdash t : s$.

Corollary 6. *Let $\mathcal{R} = (\Sigma, E, <, M)$ be a PTA-checkable CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$ and M is a set only containing conditional memberships. For every $t \in T_\Sigma$ and $u \in T_\Sigma(\mathcal{S})$:*

$$\mathcal{R} \vdash u \supseteq t \iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge \mathcal{R}^{\varnothing} \vdash u \supseteq t'$$

Proof. By induction on u and Lemma 8. $\square$

Corollary 7. *Let $\mathcal{R} = (\Sigma, E, <, \Gamma)$ be a PTA-checkable CERM system. For every $t \in T_\Sigma$:*

$$t \text{ is } \mathcal{R}\text{-reducible} \iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge t \text{ is } \mathcal{R}^{\varnothing}\text{-reducible}$$

Proof. By considering the definition of $\rightarrow^1$ and Lemma 8.

Lemma 9. *Let $\mathcal{R} = (\Sigma, E, <, R \cup M_\Omega \cup M_\Delta)$ be a PTA-checkable CERM system in which $\Sigma = (\mathcal{K}, \mathcal{F}, \mathcal{S})$, and let $\mathcal{R}_{M_\Omega} = (\Sigma, E, <, M_\Omega)$. For each $k \in \mathcal{K}$, $t \in T_{\Sigma,k}$, $s \in \mathcal{S}_k$, and $u \in \mathrm{I}_\mathcal{R}$:*

$$t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_k \tag{13}$$

$$t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_s \iff \mathcal{R}_{M_\Omega} \vdash t : s \tag{14}$$

$$t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_u \iff \mathcal{R}_{M_\Omega} \vdash u \supseteq t \tag{15}$$

$$t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} r_k \iff t \text{ is } \mathcal{R}_\Omega\text{-reducible} \tag{16}$$

$$t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} d_s \iff (\exists\ (\forall \vec{x} : \vec{s})\, l : s \in M_\Delta)\mathcal{R}_{M_\Omega} \vdash \mathrm{s}_{\vec{x}:\vec{s}}(l) \supseteq t \tag{17}$$

Proof. Equation (13) can be proved by structural induction on t.
To prove (14) - (17), note:

$$\begin{aligned} t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_s &\iff (\exists t' \in T_\Sigma(\mathcal{Q}))\, t =_E t' \wedge t' \rightarrow_{\mathcal{A}^{\varnothing}_{\mathcal{R}/M_\Omega}} q_s && \text{by Lemma 6} \\ &\iff (\exists t' \in T_\Sigma(\mathcal{Q}))\, t =_E t' \wedge \mathcal{R}^{\varnothing}_{M_\Omega} \vdash t' : s && \text{by Lemma 7} \\ &\iff \mathcal{R}_{M_\Omega} \vdash t : s && \text{by Lemma 8} \end{aligned}$$

$$
\begin{aligned}
t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_u &\iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge t' \rightarrow_{\mathcal{A}^{\varnothing}_{\mathcal{R}/M_\Omega}} q_u && \text{by Lemma 6}\\
&\iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge \mathcal{R}^{\varnothing}_{M_\Omega} \vdash u \supseteq t' && \text{by Lemma 7}\\
&\iff \mathcal{R}_{M_\Omega} \vdash u \supseteq t && \text{by Corollary 6}
\end{aligned}
$$

$$
\begin{aligned}
t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} r_k &\iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge t' \rightarrow_{\mathcal{A}^{\varnothing}_{\mathcal{R}/M_\Omega}} r_k && \text{by Lemma 6}\\
&\iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge t' \text{ is } \mathcal{R}^{\varnothing}_{\Omega}\text{-reducible} && \text{by Corollary 4}\\
&\iff t \text{ is } \mathcal{R}_\Omega\text{-reducible} && \text{by Corollary 7}
\end{aligned}
$$

$$
\begin{aligned}
t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} d_s &\iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge t' \rightarrow_{\mathcal{A}^{\varnothing}_{\mathcal{R}/M_\Omega}} d_s && \text{by Lemma 6}\\
&\iff (\exists t' \in T_\Sigma)\, t =_E t' \wedge \\
&\qquad (\exists\ (\forall \vec{x} : \vec{s})\, l : s \in M_\Delta)\ \mathcal{R}^{\varnothing}_{M_\Omega} \vdash \mathrm{s}_{\vec{x}:\vec{s}}(l) \supseteq t' && \text{by Corollary 5}\\
&\iff (\exists\ (\forall \vec{x} : \vec{s})\, l : s \in M_\Delta)\ \mathcal{R}_{M_\Omega} \vdash \mathrm{s}_{\vec{x}:\vec{s}}(l) \supseteq t && \text{by Corollary 6}
\end{aligned}
$$

□

Theorem (SCA Theorem). *Let $\mathcal{R} = (\Sigma, E, <, R \cup M_\Omega \cup M_\Delta)$ be a PTA-checkable CERM system, and let $\mathcal{R}_\Omega = (\Sigma, E, <, R \cup M_\Omega)$. $\mathcal{R}$ is sufficiently complete with respect to $\mathcal{R}_\Omega$ iff $\mathcal{L}(\mathcal{A}_{\mathcal{R}/M_\Omega}) = \varnothing$. Let $\mathcal{R} = (\Sigma, E, <, R \cup M_\Omega \cup M_\Delta)$ be a*

Proof. If $\mathcal{R}$ is sufficiently complete to $\mathcal{R}_\Omega$, using the characterization of sufficient completeness in Theorem 3, for each term $t \in T_{\Sigma,k}$ and $s \in \mathcal{S}_k$ if there exists a membership $(\forall \vec{x} : \vec{s})\ u : s$ in M_Δ such that $\mathrm{s}_{\vec{x}:\vec{s}}(u) \supseteq t$, then either t is $\mathcal{R}_\Omega$-reducible or $\mathcal{R}_\Omega \vdash t :^0 s$. Therefore, by using the characterization of $\mathcal{A}_{\mathcal{R}/M_\Omega}$ from Lemma 9, for each such $t \in T_{\Sigma,k}$ and $s \in \mathcal{S}_k$, if $t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} d_s$ then then either $t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} r_k$ or $t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_s$. Clearly then, there is no $t \in \mathcal{L}(\mathcal{A}_{\mathcal{R}/M_\Omega})$.

If $\mathcal{L}(\mathcal{A}_{\mathcal{R}/M_\Omega}) = \varnothing$, then we know for every $t \in T_{\Sigma,k}$, either $t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} r_k$, or for every $s \in \mathcal{S}_k$, if $t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} d_s$, then $t \rightarrow_{\mathcal{A}_{\mathcal{R}/M_\Omega}} q_s$. Using the characterizations given for these states in Lemma 9 and Theorem 3, we see that $\mathcal{R}$ must be sufficiently complete with respect to $\mathcal{R}_\Omega$. □

Finally, we conclude with a proof of the flattening theorem:

Lemma 10. *Let $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Psi, E, \Delta)$ be an AC-PTA. For a term t, there exists a unique set of states $P \subseteq \mathcal{Q}$ such that $\mathsf{flat}(t) \rightarrow_{\mathcal{A}_{\mathsf{det}}} P$.*

Proof. Prove by structural induction on t. □

Lemma 11. *Let $\mathcal{A} = (\mathcal{K}, \mathcal{F}, \mathcal{Q}, \Psi, E, \Delta)$ be an AC-PTA. For each $t \in T_{(\mathcal{K},\mathcal{F}),k}$, and $q \in \mathcal{Q}$,*

$$t \rightarrow_{\mathcal{A}} q \iff (\exists Q \subseteq \mathcal{Q}_k)\ \mathsf{flat}(t) \rightarrow_{\mathcal{A}_{\mathsf{det}}} Q \wedge q \in Q.$$

Proof. Prove by structural induction on t. □

Theorem 6 (Flattening Theorem). *Let $\mathcal{A}$ be a AC-PTA. For each $t \in T_\Sigma$,*

$$t \in \mathcal{L}(\mathcal{A}) \iff \mathsf{flat}(t) \in \mathcal{L}(\mathcal{A}_{\mathsf{det}}).$$

Proof. By the definition of $\mathcal{A}_{\mathsf{det}}$ and Lemma 11. □

Sufficient Completeness Checking with Propositional Tree Automata

AIST Programming Science Group Technical Report
August 23, 2005
Research Center for Verification and Semantics
Amagasaki site 一 AIST Kansai
National Institute of Advanced Industrial Science and Technology (AIST)
Nakoji 3−11−46, Amagasaki, Hyogo 661-0974, Japan
E-mail : informatics-inquiry@m.aist.go.jp

Sufficient Completeness Checking with Propositional Tree Automata

産業技術総合研究所算譜科学グループ研究速報
発行日：2005 年 8 月 23 日
編集・発行：独立行政法人産業技術総合研究所関西センター尼崎事業所
　　　　　　システム検証研究センター
連絡先：〒661−0974 兵庫県尼崎市若王寺 3−11−46
E-mail：informatics-inquiry@m.aist.go.jp