Sound Over-Approximation of Equational Reasoning with Variable-Preserving Rules Parameterized by Derivation Depth

Mateus de Oliveira Oliveira

Department of Computer and Systems Sciences, Stockholm University, Sweden
Department of Informatics, University of Bergen, Norway
oliveira@dsv.su.se

Abstract

Equational reasoning is one of the most intuitive and widely used types of symbolic reasoning. In this setting, the goal is to determine whether a given ground equation $t \simeq t'$ follows as a consequence of a set of equational axioms E using the process of replacing equals with equals. An equation $t \simeq t'$ is *variable-preserving* if each variable occurring in t occurs in t' and vice versa. Such equations have widely applicability in algebra, since they can be used to express properties such as commutativity, associativity, distributivity, among others.

In this work, we show that for each fixed set E of variable-preserving equations, the set of ground equations derivable from E in depth at most d is *soundly* over-approximable in fixed-parameter tractable time. More specifically, we devise an algorithm that takes as input a set E of variable-preserving equations and a target ground equation $t \simeq t'$, and always halts with a YES or NO answer.

1. If equation $t \simeq t'$ can be derived from E in depth at most d, the algorithm always halts with a YES.
2. If equation $t \simeq t'$ does not belong to the equational closure of E, then the algorithm always halts with a NO.

In other words, the set of YES instances contains the set of ground equations that can be deduced from E in depth at most d, and possibly other equations that require derivations of higher depth. However, this set contains no ground equation that is not in the equational closure of E. For this reason, the algorithm is sound. Our algorithm works in time $f_E(d) \cdot |t| \cdot |t'|$, where $|t|$ and $|t'|$ are the number of symbols in t and t' respectively, d is the depth parameter, and $f_E(d)$ is a function whose growth depends only on d and on parameters associated with the equations in E.

1 Introduction

Equational reasoning, or the process of replacing equals by equals, is one of the most intuitive and widely used methods of formal reasoning. Given a set E of equations, called *axioms*, and terms t and t', the goal is to determine whether the equation $t \simeq t'$ follows as a logical consequence of the axioms in E (Plaisted 1993; Echenim, Peltier, and Tourret 2013; Fish and Lisitsa 2014; Bachmair and Ganzinger 1998). A celebrated theorem due to Birkhoff (Birkhoff 1935) states that such a logical implication holds if and only if term t can be transformed into term t' by a chain of replacements of equals with equals.

In this work, we consider an enhanced version of this calculus where at each inference step, several axioms can be applied to a term simultaneously as long as they do not interfere with one another. When ignoring complexity theoretic considerations, this parallel variant of equational reasoning has the same expressiveness as the sequential variant. Nevertheless, the minimum number of parallel inference steps to derive a given equation $t \simeq t'$ may be drastically smaller than then the minimum number of sequential steps. For example, consider the axiom $x+y \simeq y+x$ stating that addition is commutative. For each $n \in \mathbb{N}$, there is a ground equation $t_n \simeq t'_n$ that is derivable from this axiom in *a single parallel step*, but whose sequential derivations require $\Omega(n)$ steps (Observation 7).

It is worth noting that the problem of determining whether a given ground equation $t \simeq t'$ can be inferred from equations in a set E of axioms is, in general, undecidable (McNulty 1992). The problem becomes NEXP-complete if instead we ask whether $t \simeq t'$ can be derived in d steps, for a given d specified in binary [1]. In this work, we cope with the intractability of the equational inference problem using the framework of the theory of fixed-parameter tractability (Downey and Fellows 1995). A computational problem is said to be *fixed-parameter tractable* with respect to parameters $k_1, ..., k_r$ if there is an algorithm that solves instances of size n in time at most $f(k_1, ..., k_r) \cdot n^c$ for some constant c and some function f depending only on the parameters k_i.

Our main result (Theorem 11) states that for each fixed set E of variable-preserving equations, the set of ground equations derivable from E in depth at most d is *soundly* over-approximable in fixed-parameter tractable time. More specifically, we devise an algorithm that takes as input a set of variable-preserving equations E and a target ground equation $t \simeq t'$, and always halts with a YES or NO answer.

1. If the equation $t \simeq t'$ is derivable from E in depth at most d, the algorithm always halts with a YES.
2. If the equation $t \simeq t'$ does not belong to the equational closure of E, then the algorithm always halts with a NO.

[1] NEXP is the class of problems solvable in nondeterministic exponential time. If d is specified in unary, then the problem becomes NP-complete.

In other words, the set of YES instances contains the set of ground equations that are derivable from E in depth at most d, and possibly other equations that require derivations of higher depth. Nevertheless, our algorithm guarantees that all ground equations in this set belong to the equational closure of E. For this reason, the algorithm is sound. Our algorithm works in time $f_E(d)\cdot|t|\cdot|t'|$, where $|t|$ and $|t'|$ are the number of symbols in t and t' respectively, d is the depth parameter, and $f_E(d)$ is a function whose growth depends only on d and on parameters associated with the equations in E.

It is worth noting that in the context of equational reasoning, a *sound* over-approximation is an advantage because, in practice, what is relevant is determining whether a given equation (for instance, representing a theorem) is provable from a set of equational axioms. In this sense, while our algorithm is able to certify the provability of all ground equations derivable in depth at most d, as a positive side effect, it may also certify the provability of some ground equations that are provable in depth greater than d. The important thing is that the algorithm never certifies the provability of a ground equation that is not provable from the given set of equational axioms.

An interesting aspect of our algorithm is that it does not construct an explicit derivation of the equation $t \simeq t'$ from the equations in E. Instead, by leveraging on tree-automata completion techniques (Feuillade, Genet, and Tong 2004; Genet 1998; Korp and Middeldorp 2009; Felgenhauer and Thiemann 2014; Genet and Rusu 2010; Iosif, Rogalewicz, and Vojnar 2014; Bachmair and Dershowitz 1989) we will show how to efficiently construct a tree automaton $\mathcal{A}(E, d, t)$ whose language soundly over-approximates the set of terms that can be reached from t in depth at most d, using rewriting rules extracted from equations in E. Our algorithm answers YES if t' is accepted by $\mathcal{A}(E, d, t)$, and NO otherwise.

2 Preliminaries

2.1 Basic Notation

We let $\mathbb{N} \doteq \{0, 1, \dots\}$ denote the set of natural numbers, including 0, and $\mathbb{N}_+ = \mathbb{N}\backslash\{0\}$ denote the set of positive natural numbers. For each $n \in \mathbb{N}_+$, we let $[n] = \{1, \dots, n\}$. We may write $[0]$ to denote the empty set $\emptyset$.

2.2 Term Rewriting Systems

In this section, we define some basic concepts related to the theory of term rewriting systems. We refer (Baader and Nipkow 1999; N. Dershowitz 1990) for a detailed treatment on this subject.

A *ranked alphabet* is a finite set Σ whose elements are called *function symbols*, together with an arity function $\mathfrak{a} : \Sigma \to \mathbb{N}$. Intuitively, the arity $\mathfrak{a}(f)$ of a symbol $f \in \Sigma$ specifies the number of inputs of f. A function symbol of arity 0 is called a *constant symbol*. We let $\mathfrak{a}(\Sigma) = \max\{\mathfrak{a}(f) \mid f \in \Sigma\}$ be the maximum arity of a function symbol in Σ.

Let Σ be a ranked alphabet, and X be a countable set of variables disjoint from Σ. We let $Ter(\Sigma, X)$ be the smallest set satisfying the following three conditions:

1. each function symbol $a \in \Sigma$ of arity 0 is a term in $Ter(\Sigma, X)$;
2. each variable $x \in X$ is a term in $Ter(\Sigma, X)$;
3. for each $r \in \mathbb{N}$, each function symbol $f \in \Sigma$ of arity r, and each sequence of terms $t_1, t_2, \dots, t_r \in Ter(\Sigma, X)$, $f(t_1, t_2, \dots, t_r)$ is a term in $Ter(\Sigma, X)$.

Given such a term t, we let $var(t)$ be the set of variables occurring in t. We say that t is a *ground term* if it has no variables (that is, $var(t) = \emptyset$). We let $Ter(\Sigma)$ be the set of all ground terms in $Ter(\Sigma, X)$.

The set of positions of a term $t \in Ter(\Sigma, X)$ is the set $Pos(t)$ of strings over the set $[\mathfrak{a}(\Sigma)]$ inductively defined as follows:

1. if $t = a$, for some function symbol a of arity 0, or $t = x$ for some variable $x \in X$, then $Pos(t) = \{\varepsilon\}$, where ε is the empty string;
2. if $t = f(t_1, \dots, t_r)$ for some function symbol f of arity r, then

$$Pos(t) = \{\varepsilon\} \cup \bigcup_{j=1}^{r} \{jp \,:\, p \in Pos(t_j)\}.$$

We let $|t| = |Pos(t)|$ denote the *size* of the term t. We say that the empty string ε is the *root position* of t. For each $p \in Pos(t)$, we let $t(p)$ be the function symbol of t at position p. The symbol $t(\varepsilon)$ is called the *root symbol* of t. For each position $p \in Pos(t)$, we let $t|_p$ denote the subterm of t rooted at position p. That is to say, $Pos(t|_p) = \{p' \,:\, pp' \in Pos(t)\}$ and for each $p' \in Pos(t|_p)$, $t|_p(p') = t(pp')$. We say that a term t' is a *proper subterm* of t if $t' = t|_p$ for some position $p \in Pos(t)$ with $p \neq \varepsilon$. If s and t are terms and p is a position in $Pos(t)$ then we denote by $t[s]_p$ the term obtained from t by replacing the subterm $t|_p$ with the term s. A (Σ, X)-*substitution* is a function $\sigma : X \to Ter(\Sigma, X)$ mapping variables in X to terms in $Ter(\Sigma, X)$. If t is a term in $Ter(\Sigma, X)$, and σ is a substitution, then we denote by t^σ the term obtained from t by replacing each occurrence of each variable $x \in X$ with the term $\sigma(x)$.

A *rewriting rule* over $Ter(\Sigma, X)$ is a pair $l \to r$ where l and r are terms in $Ter(\Sigma, X)$ such that l is not a variable and $var(r) \subseteq var(l)$. We say that $l \to r$ is *variable-preserving* if $var(r) = var(l)$. A term rewriting system on $Ter(\Sigma, X)$ is any finite set $\mathfrak{R}$ of rewriting rules. Let t be a term in $Ter(\Sigma, X)$, p be a position in $Pos(t)$, and $l \to r$ be a rewriting rule in $\mathfrak{R}$. We say that $l \to r$ can be applied to t at position p if there is a substitution $\sigma : X \to Ter(\Sigma, X)$ such that $t|_p = l^\sigma$. In this case, we let $t' = t[r^\sigma]_p$ be the term obtained from t by replacing the subterm $t|_p$ with the term r^σ. We write $t \to_{\mathfrak{R}} t'$ to denote that t' can be obtained from t by applying some rewriting rule $l \to r \in \mathfrak{R}$ at some position p of t. We say that $\to_{\mathfrak{R}}$ is the relation induced by $\mathfrak{R}$ on the set $Ter(\Sigma, X)$. We let $\to^*_{\mathfrak{R}}$ be the transitive closure of $\to_{\mathfrak{R}}$. In other words, $t \to^*_{\mathfrak{R}} t'$ if and only if t' is obtained from t by applying a finite number of rewriting rules from $\mathfrak{R}$.

2.3 Tree Automata

In this subsection, we briefly define some notation related to tree automata theory. An extensive treatment on the subject can be found in (Comon et al. 2007).

A *bottom-up tree automaton* is a tuple $\mathcal{A} = (Q, \Sigma, F, \Delta)$ where Q is a finite set of *states*, Σ is a ranked alphabet, $F \subseteq Q$ is a set of *final states* and Δ is a finite set of rewriting rules of the form $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$, where f is a function symbol in Σ and $q_1, ..., q_{\mathfrak{a}(f)}$ and q are states in Q. Here, we should regard Δ as a term rewriting system acting on terms in $Ter(\Sigma, Q)$. That is to say, we regard the states in Q as variables. We let $\to_\Delta \subseteq Ter(\Sigma, Q) \times Ter(\Sigma, Q)$ be the rewriting relation induced by Δ, and let $\to_\Delta^*$ be the transitive closure of $\to_\Delta$. For each state $q \in Q$, we let $\mathcal{L}(\mathcal{A}, q) = \{t \in Ter(\Sigma) \mid t \to_\Delta^* q\}$ be the set of ground terms that reach q. The *language* of $\mathcal{A}$ is defined as the set $\mathcal{L}(\mathcal{A}) = \bigcup_{q \in F} \mathcal{L}(\mathcal{A}, q)$ of ground terms reaching some final state of $\mathcal{A}$. We let $|\mathcal{A}| = |Q| + |\Delta|$ be the *size* of $\mathcal{A}$. We may write $Q(\mathcal{A})$, $F(\mathcal{A})$ and $\Delta(\mathcal{A})$ to denote the set of states, the set of final states, and the set of transitions of $\mathcal{A}$ respectively. Let $q \in Q$ be a state in Q. A *q-context* is a term $t \in Ter(\Sigma, Q)$ where there is exactly one leaf position $p \in Pos(t)$ with $t(p) = q$, and for every leaf position $p' \in Pos(t)$ distinct from p, $t(p') \in \Sigma$. We say that $\mathcal{A}$ is *reachable* if for each state $q \in Q$, the language $\mathcal{L}(\mathcal{A}, q)$ is nonempty. We say that $\mathcal{A}$ is *co-reachable* if for each state $q \in Q$, there is a q-context t and a final state q' such that $t \to_\Delta^* q'$.

A tree-automaton $\mathcal{A}$ is said to be *deterministic* if for each function symbol $f \in \Sigma$ and each sequence of states $q_1, \ldots, q_{\mathfrak{a}(f)} \in Q$ there is at most one state $q \in Q$ such that the transition $f(q_1, \ldots, q_{\mathfrak{a}(f)}) \to q$ belongs to Δ. We observe that if $\mathcal{A}$ is a deterministic tree automaton, then each term $t \in Ter(\Sigma)$ reaches at most one state in Q.

3 Equational Reasoning in Bounded Depth

3.1 Equational Reasoning vs Term Rewriting

In this section, we let X be a countable set of variables. Let Σ be a ranked alphabet. A Σ-equation is a pair $t \simeq t'$ of terms in $Ter(\Sigma, X)$. Next, we define the notion of *equational closure* of a finite set of Σ-equations.

Definition 1 (Equational Closure)**.** *Let Σ be a ranked alphabet and E be a set of Σ-equations. The equational closure of E, denoted by $\mathsf{Closure}_\Sigma(E)$ is the smallest set S of Σ-equations closed under the following rules of inference:*

1. Reflexivity: *for each term $t \in Ter(\Sigma, X)$, $t \simeq t \in S$.*
2. Symmetry: *for each $t, t' \in Ter(\Sigma, X)$, if $t \simeq t'$ belongs to S, then $t' \simeq t$ belongs to S.*
3. Transitivity: *for each $t, u, t' \in Ter(\Sigma, X)$, if $t \simeq u$ and $u \simeq t'$ belong to S then $t \simeq t'$ belongs to S.*
4. Congruence: *for each function symbol $f \in \Sigma$ of arity $\mathfrak{a}$, if equations $t_1 \simeq t'_1, \ldots, t_\mathfrak{a} \simeq t'_\mathfrak{a}$ belong to S, then $f(t_1, \ldots, t_\mathfrak{a}) \simeq f(t'_1, \ldots, t'_\mathfrak{a})$ belongs to S.*
5. Substitution: *for each substitution $\sigma : X \to Ter(\Sigma, X)$, and each two terms $s, t \in Ter(\Sigma, X)$, if $s \simeq t \in S$, then $s^\sigma \simeq t^\sigma$ is in S.*

An equation $t \simeq t'$ is said to be *a logical consequence of* E if $s \simeq t$ belongs to $\mathsf{Closure}_\Sigma(E)$. The rules of inference defined above may be used to certify that a given equation $t \simeq t'$ is implied by E. More specifically, $t \simeq t'$ belongs to $\mathsf{Closure}_\Sigma(E)$ if and only if there is a sequence of terms $s_0, s_1, ..., s_n$ such that $t = s_0$, $t' = s_n$, and for each $i \in [n]$, s_i is obtained from previous terms in the sequence by the applications of one of the rules $(1) - (5)$.

A more intuitive, and widely used, inference system is the so-called *replacement of equals by equals*. In this system, equations are essentially used as rewriting rules. More specifically, given a Σ-equation $s \simeq t$, we define the following set of rewriting rules:

$$R(s \simeq t) = \begin{cases} \{s \to t\} \text{ if } var(t) \subsetneq var(s) \\ \{t \to s\} \text{ if } var(s) \subsetneq var(t) \\ \{s \to t, t \to s\} \text{ if } var(t) = var(s) \end{cases} \quad (1)$$

Going further, given a set E of Σ-equations, we let $R(E) = \bigcup_{s \simeq t \in E} R(s \simeq t)$ be the term rewriting system derived from E. We let $\to_{\mathfrak{R}(E)}^*$ denote the reflexive, transitive closure of $\mathfrak{R}(E)$, and $\leftrightarrow_{\mathfrak{R}(E)}^*$ denote the reflexive, symmetric, transitive closure of $\mathfrak{R}(E)$. Birkhoff's theorem, a classical result in the field of equational reasoning, establishes a close connection between the equational inference system described in Definition 1 and replacement of equals by equals (see for instance Theorem 3.1.12 of (Baader and Nipkow 1999)).

Theorem 2 ((Birkhoff 1935))**.** *Let E be a set of Σ-equations and $t \simeq t'$ be a Σ-equation. Then $t \simeq t'$ belongs to $\mathsf{Closure}_\Sigma(E)$ if and only if $t \leftrightarrow_{\mathfrak{R}(E)}^* t'$.*

If all equations in E are variable-preserving, then for each equation $s \simeq t$ in E, $\mathfrak{R}(s \simeq t) = \{s \to t, t \to s\}$, and therefore, in this case, $\mathfrak{R}(E)$ is a symmetric relation. Since this implies that $\leftrightarrow_{\mathfrak{R}(E)}^*$ is the same relation as $\to_{\mathfrak{R}(E)}^*$, we have the following corollary of Theorem 2.

Corollary 3. *Let E be a set of variable-preserving Σ-equations and $t \simeq t'$ be a Σ-equation. Then $t \simeq t'$ belongs to $\mathsf{Closure}_\Sigma(E)$ if and only if $t \to_{\mathfrak{R}(E)}^* t'$.*

3.2 Parallel Rewriting

Next, we define the notion of multi-step rewriting as introduced by Oostrom in (van Oostrom 1999) (see also Chapter 4 of (Jan Willem Klop 2003)). Intuitively, multi-steps can be used to give a formal definition for the notion of parallel rewriting, where several term rewriting rules are allowed to be applied simultaneously, as long as they do not interfere with each other.

Definition 4 (Multi-Step)**.** *Let $\mathfrak{R}$ be a term rewriting system. The multi-step relation $\multimap \subseteq Ter(\Sigma) \times Ter(\Sigma)$ induced by $\mathfrak{R}$ is inductively defined as follows.*

1. *Let $f \in \Sigma$ be a function symbol of arity $\mathfrak{a}(f) = 0$. Then $f \multimap f$.*
2. *Let $f \in \Sigma$ be a function symbol of arity $\mathfrak{a}(f) \geq 1$, and $t_1, \ldots, t_{\mathfrak{a}(f)}$ and $t'_1, \ldots, t'_{\mathfrak{a}(f)}$ be sequences of terms such that $t_i \multimap t'_i$ for each $i \in [\mathfrak{a}(f)]$. Then*

$$f(t_1, ..., t_{\mathfrak{a}(f)}) \multimap f(t'_1, ..., t'_{\mathfrak{a}(f)}).$$

3. *Let $l \to r$ be a rewriting rule $\mathfrak{R}$, and $\sigma, \theta : X \to Ter(\Sigma)$ be substitutions such that $\sigma(x) \multimap\!\!\to \theta(x)$ for each variable $x \in var(l)$. Then $l^\sigma \multimap\!\!\to r^\theta$.*

We note that items 1 and 2 of Definition 4 imply that $t \multimap\!\!\to t$ for each ground term $t \in Ter(\Sigma)$. We say that a term t' is *reachable* from a term t in depth at most d if there is a sequence of multi-steps $t_0 \multimap\!\!\to t_1 \multimap\!\!\to ... \multimap\!\!\to t_d$ such that $t_0 = t$ and $t_d = t'$. We write $t \overset{d}{\multimap\!\!\to} t'$ to denote that t' is reachable from t in depth at most d. Next, we use the notion of multistep to define a suitable notion of provability in bounded depth.

Definition 5 (Provability in Depth d)**.** *Let E be a set of Σ-equations, $\multimap\!\!\to$ be the multistep relation induced by $\mathfrak{R}(E)$, and t and t' be ground terms in $Ter(\Sigma)$. We say that equation $t \simeq t'$ is provable from E in depth at most d if $t \overset{d}{\multimap\!\!\to} t'$.*

If all equations in E are variable-preserving then we have the following observation.

Observation 6. *Let E be a set of variable-preserving Σ-equations and $t \simeq t'$ be a Σ-equation. Then $t \simeq t'$ belongs to $\mathsf{Closure}_\Sigma(E)$ if and only if $t \overset{d}{\multimap\!\!\to} t'$ for some $d \in \mathbb{N}$.*

3.3 Parallel vs Sequential Rewriting

Next, we illustrate the applicability of parallel equational reasoning with two examples. First, we show that equational reasoning in depth at most 1 can already be *unboundedly* more powerful than sequential equational reasoning. Second, we show that equational reasoning in depth 1 is already sufficient to model non-trivial combinatorial problems, such as isomorphism of labeled trees of bounded degree.

Example 1. Let $\Sigma = \{+, a, b\}$ where $+$ is a symbol of arity 2 and a and b are constant symbols. Let $E = \{x + y = y + x\}$ be the set of equations that contain a unique axiom that expresses the commutativity of $+$. For each $n \in \mathbb{N}$, where n is a power of 2 and each $\alpha \in \{0,1\}^n$, we let s_α denote the term that has the structure of a complete binary tree with $2n$ leaves, where all internal nodes are labeled with the symbol $+$, and where for each $i \in [n]$, the i-th deepest internal node, from left to right, has the left child labeled with a and the right child with b, if $\alpha_i = 0$, and vice versa, if $\alpha_i = 1$. Then, we have the following observation.

Observation 7.

1. *For each $\alpha, \alpha' \in \{0,1\}^n$, the equality $s_\alpha \simeq s_{\alpha'}$ can be proved from E in depth at most 1.*
2. *The equality $s_{0^n} \simeq s_{1^n}$ requires sequential proofs of length $\Omega(n)$.*

Item 1 of Observation 7 provides us with a concrete example where exponentially many terms can be derived from a single term in depth at most 1. In particular, for each $\alpha \in \{0,1\}^n$, s_α can be derived from s_{0^n} in a single parallel derivation step. Contrast this with the fact that in sequential rewriting using the rules of any term rewriting system $\mathfrak{R}$, a term with n nodes can reach at most $|\mathfrak{R}| \cdot n$ other terms in a single sequential step, since each rewriting rule can be matched in at most one way at each node. On the other hand, Item 2 of Observation 7 provides us with a concrete example of an equation that can be derived in a single parallel step, but that can only be derived using a linear number of sequential steps.

3.4 Term Isomorphism in a Single Parallel Rewriting Step

Example 2. Let Σ be a ranked alphabet and s and t be ground terms in $Ter(\Sigma)$. We say that two terms s and t in $Ter(\Sigma)$ are *isomorphic* if there is a bijection $\phi : Pos(s) \to Pos(t)$ such that $\phi(\mathtt{root}(s)) = \phi(\mathtt{root}(t))$, for each position $p \in Pos(s)$, $s(p) = t(\phi(p))$, and for each two positions p and p' in $Pos(s)$, p' is a child of p in s if and only if $\phi(p)$ is a child of $\phi(p')$ in t. Intuitively, two terms are isomorphic if they are isomorphic as labeled tree structures, where the ordering on the children of each node is forgotten.

Theorem 8. *Let Σ be a ranked alphabet with symbols of arity at most r. Then there is a set E_Σ with $O(|\Sigma| \cdot r!)$ variable-preserving equations with the property that for each two Σ-terms s and t in $Ter(\Sigma)$, s and t are isomorphic if and only if the equation $s \simeq t$ is provable from E_Σ in depth at most 1.*

We note that isomorphism of labeled trees of degree at most r can be reduced straightforwardly to the isomorphism of terms of arity at most r. We also note that tree isomorphism is a well-understood computational problem that can be solved in polynomial even if no restriction on maximum degree is imposed (Jenner, McKenzie, and Torán 1998). However, the fact that isomorphic terms (over alphabets of bounded arity) can be reached from each other in single parallel step suggests that derivation in bounded depth is relevant in settings where isomorphic terms should be identified, such as in unification modulo commutative theories (Neugebauer and Petermann 1998).

4 Ground Equational Reasoning is FPT with Respect to Depth of Derivations

In this section, we state and prove our main result (Theorem 11). Intuitively, this result states that ground equational reasoning can be soundly over-approximated in fixed-parameter tractable time when parameterized by the depth of derivations, provided all equations in the set of equational axioms are variable-preserving. We start by defining the notion of a tree automaton compatible with a set of equations (Definition 9).

Definition 9. *Let $\mathcal{A}$ be a tree automaton over Σ. We say that $\mathcal{A}$ is E-compatible if for each state $q \in Q(\mathcal{A})$, and each two terms $t, t' \in \mathcal{L}(\mathcal{A}, q)$, the equation $t \simeq t'$ belongs to the equational closure of E.*

The proof of our main result will depend crucially on the following lemma.

Lemma 10 (Main Technical Lemma)**.** *Let Σ be a ranked alphabet, E be a finite set of variable-preserving Σ-equations, and $d \in \mathbb{N}$. For each ground term $t \in Ter(\Sigma)$, one can construct in time $f_E(d) \cdot |t|$ an E-compatible tree automaton $\mathcal{A}(E, d, t)$ with a single accepting state such that*

$$\{t' \mid t \overset{d}{\multimap\!\!\to} t'\} \subseteq \mathcal{L}(\mathcal{A}(E, d, t)). \tag{2}$$

In Lemma 10, $f_E(d)$ is a function that depends only on the depth parameter and on parameters associated to the set of equations E and alphabet Σ. Intuitively, the language $\mathcal{L}(\mathcal{A}(E,d,t))$ is a *sound* over-approximation of the set of terms that can be reached from t in depth at most d. On the one hand, Equation 2 guarantees that every term reachable from t in depth at most d belongs to $\mathcal{L}(\mathcal{A}(E,d,t))$. On the other hand, the fact that $\mathcal{A}(E,d,t)$ has a unique accepting state, combined with the fact that it is E-compatible implies that for each term $t' \in \mathcal{L}(\mathcal{A}(E,d,t))$, the equation $t \simeq t'$ belongs to the equational closure of E. We will prove Lemma 10 in the next three subsections. Before that, we will use this lemma to prove Theorem 11.

Below, we formally state our main theorem.

Theorem 11 (Main Theorem)**.** *Let Σ be a ranked alphabet, and E be a set of variable-preserving Σ-equations. There is an algorithm that takes as input a positive integer d and two Σ-terms t and t' and always halts with a* YES *or with a* NO*.*

1. *If the equation $t \simeq t'$ is derivable from E in depth at most d, the algorithm always halts with a* YES*.*
2. *If the equation $t \simeq t'$ does not belong to the equational closure of E, then the algorithm always halts with a* NO*.*

Furthermore, the algorithm works in time $f_E(d) \cdot |t| \cdot |t'|$ for some function $f_E(d)$ depending only on the depth parameter d and on parameters associated to the set E.

Proof. The algorithm proceeds as follows when given a ground equation $t \simeq t'$ as input: first, it constructs the tree automaton $\mathcal{A}(E,d,t)$. Subsequently it tests whether t' belongs to $\mathcal{L}(\mathcal{A}(E,d,t))$. If t' does belong to $\mathcal{L}(\mathcal{A}(E,d,t))$, the algorithm halts with a YES. Otherwise, it halts with a NO.

Now, we show that this algorithm satisfies Condition 1 and Condition 2 of Theorem 11. First, we note that if t can be derived from E in depth at most d then, by Lemma 10, t' belongs to $\mathcal{L}(\mathcal{A}(E,d,t))$. Therefore, if $t \overset{d}{\leftrightarrow} t'$, then the algorithm always halts with a YES, as required by Condition 1. Now, suppose that the equation $t \simeq t'$ does not belong to the equational closure of E. We claim that t' does not belong to $\mathcal{L}(\mathcal{A}(E,d,t))$. Therefore, in this case, the algorithm always answers NO, as required by Condition 2. To prove our claim, assume, for the sake of contradiction, that t' belongs to $\mathcal{L}(\mathcal{A}(E,d,t))$. Since $\mathcal{A}(E,d,t)$ has a unique final state, and both t and t' are accepted by $\mathcal{A}(E,d,t)$, both t and t' reach this unique final state. But since $\mathcal{A}(E,d,t)$ is E-compatible, the equation $t \simeq t'$ belongs to the equational closure of E. This contradicts our initial supposition that $t \simeq t'$ does not belong to the equational closure of E, and therefore, we conclude that t' does not belong to $\mathcal{L}(\mathcal{A}(E,d,t))$.

Finally, we analyze the running time of our algorithm. By Lemma 10, $\mathcal{A}(E,d,t)$ can be constructed in time $f_E(d) \cdot |t|$ for some suitable function $f_E(d)$. Therefore, this automaton has size at most $f_(d) \cdot |t|$. This implies that one can test in time $f_E(d) \cdot |t||t'|$ whether t' belongs to $\mathcal{L}(\mathcal{A}(E,d,t))$. This concludes the proof of Theorem 11. □

4.1 Construction of the Automaton $\mathcal{A}(E,d,t)$

Given a term $t \in Ter(\Sigma, X)$, we let

$$\mathrm{Sub}(t) = \{s \ : \ s \text{ is a subterm of } t\}$$

be the set of *subterms* of t. We note that a term s may occur multiple times as a subterm of t, and this term is counted only once in $\mathrm{Sub}(t)$. We let $\widehat{\mathrm{Sub}}(t) = \mathrm{Sub}(t) \backslash var(t)$ be the set of subterms of t that are not a variable.

Definition 12 (Tree Automaton Associated with a Term)**.** *Given a term $t \in Ter(\Sigma, X)$, we let $\mathcal{B}(t)$ be the tree automaton over Σ defined as follows.*

$$Q(\mathcal{B}(t)) = \{q_s \mid s \in \mathrm{Sub}(t)\} \qquad F(\mathcal{B}(t)) = \{q_t\}$$

$$\Delta(\mathcal{B}(t)) = \{f(q_{s_1}, ..., q_{s_{\mathfrak{a}(f)}}) \to q_s \mid s \in \widehat{\mathrm{Sub}}(t), \\ s = f(s_1, \ldots, s_{\mathfrak{a}(f)})\}.$$

The tree automaton $\mathcal{B}(t)$ has one state q_s for each subterm in $\mathrm{Sub}(t)$, and one transition $f(q_{s_1}, ..., q_{s_{\mathfrak{a}(f)}}) \to q_s$ for each subterm $s \in \widehat{\mathrm{Sub}}(t)$ whose leading symbol is f and whose children are $s_1, \ldots, s_{\mathfrak{a}(f)}$. The only final state is the state q_t corresponding to the term t itself. We note that the language of $\mathcal{B}(t)$ is empty if t is not a ground term because states associated with variables of t are unreachable in $\mathcal{B}(t)$. On the other hand, when t is a ground term then t is the unique term accepted by $\mathcal{B}(t)$.

Proposition 13. *Let t be a ground term in $Ter(\Sigma)$.*

1. *$\mathcal{B}(t)$ is deterministic, reachable and co-reachable.*
2. *$\mathcal{L}(\mathcal{B}(t)) = \{t\}$.*
3. *For any set E of Σ-equations, $\mathcal{B}(t)$ is E-compatible.*

Let $\mathcal{A}$ be a tree automaton over Σ. A *configuration* for $\mathcal{A}$ is a term $u \in Ter(\Sigma, Q(\mathcal{A}))$. In other words, the variables of u are states of $\mathcal{A}$. Now, let l be a term in $Ter(\Sigma, X)$ for some X disjoint from $Q(\mathcal{A})$, and let $\mathfrak{m} : Y \to Q(\mathcal{A})$ be a map from some set of variables Y with $var(l) \subseteq Y \subseteq X$ to states of $\mathcal{A}$. Then, we let $l \circ \mathfrak{m}$ be the configuration for $\mathcal{A}$ obtained from l by replacing each occurrence of each variable $x \in var(l)$ with the state $\mathfrak{m}(x)$.

Definition 14 (Matching)**.** *Let $\mathcal{A}$ be a tree automaton and l be a term in $Ter(\Sigma, X)$. A matching for l in $\mathcal{A}$ is a pair $(q, \mathfrak{m})$ where q is a state in $Q(\mathcal{A})$, and $\mathfrak{m} : var(l) \to Q(\mathcal{A})$ is a map such that the configuration $l \circ \mathfrak{m}$ reaches state q in $\mathcal{A}$ (that is to say, $l \circ \mathfrak{m} \to_{\mathcal{A}}^{*} q$).*

We let $\mathcal{M}(\mathcal{A}, q, l)$ be the set of all maps from variables of l to states of $\mathcal{A}$ with the property that (q, m) is a matching for l in $\mathcal{A}$.

$$\mathcal{M}(\mathcal{A}, q, l) = \{\mathfrak{m} : var(l) \to Q(\mathcal{A}) | (q, \mathfrak{m}) \text{ is a matching for } l \text{ in } \mathcal{A}\}.$$

Let t be a term in $Ter(\Sigma, X)$, and $\mathfrak{m} : Y \to Q(\mathcal{A})$ for some set of variables Y with $var(t) \subset Y$. We say that t is *matchable* by $\mathfrak{m}$ if there is a state $q \in Q(\mathcal{A})$ such that $(q, \mathfrak{m}|_{var(t)})$ is a matching for t in $\mathcal{A}$, where $\mathfrak{m}|_{var(t)}$ is the restriction of $\mathfrak{m}$ to the variables of t. Given a configuration $\mathfrak{r}(q_1, \ldots, q_k)$ with state variables $q_1, ..., q_k$, and given ground terms $t_1, \ldots, t_k$ in $Ter(\Sigma)$, we let $\mathfrak{r}(t_1, \ldots, t_k)$ be the ground term obtained by replacing each occurrence of q_i in $\mathfrak{r}$ with the term t_i.

Definition 15 (Residuos)**.** *Given a term $r \in Ter(\Sigma)$, and a function $\mathfrak{m} : Y \to Q(\mathcal{A})$ with $var(r) \subseteq Y$, we say that a configuration $\mathfrak{r}(q_1, \ldots, q_k)$ is an $\mathfrak{m}$-residuo of r in $\mathcal{A}$ if there are proper subterms $t_1, \ldots, t_k$ of r satisfying the following conditions:*

1. *$r = \mathfrak{r}(t_1, \ldots, t_k)$,*
2. *for each $i \in [k]$, t_i is matchable by $\mathfrak{m}$,*
3. *for each $i \in [k]$, no proper subterm of t that is a proper superterm of t_i is matchable by $\mathfrak{m}$, and*
4. *for each $i \in [k]$, $\mathfrak{m} \circ t_i \to_\Delta^* q_i$.*

Intuitively, the terms $t_1, \ldots, t_k$ are the maximal proper subterms of r that can be reduced to some state in $Q(\mathcal{A})$ when their variables are replaced according to $\mathfrak{m}$. Furthermore for each $i \in [k]$, the configuration $\mathfrak{m} \circ t_i$ reaches state q_i. Note that the fact that the terms $t_1, \ldots, t_k$ are required to be proper subterms of t implies that an $\mathfrak{m}$-residuo always exists. Given such a residuo $\mathfrak{r} = \mathfrak{r}(q_1, \ldots, q_k)$, let $\mathcal{B}(\mathfrak{r})$ be the tree automaton associated with the configuration $\mathfrak{r}$ as specified in Definition 12. Note that, since $\mathfrak{r}$ is a term in $Ter(\Sigma, Q(\mathcal{A}))$, the elements in $Q(\mathcal{A})$ act as variables. We let $\mathcal{C}(\mathcal{A}, q, \mathfrak{r})$ be the tree automaton obtained from $\mathcal{B}(\mathfrak{r})$ by performing the following modifications: first, rename the state $q_\mathfrak{r}$ of $\mathcal{B}(\mathfrak{r})$ to the state q of $\mathcal{A}$; subsequently, set the set of final states of $\mathcal{C}(\mathcal{A}, q, \mathfrak{r})$ to the empty set.

Now, consider the tree automaton

$$\mathcal{N}(\mathcal{A}, E) = \mathcal{A} \cup \bigcup_{q,\mathfrak{r}} \mathcal{C}(\mathcal{A}, q, \mathfrak{r}). \tag{3}$$

obtained by taking the union of $\mathcal{A}$ with all $\mathcal{C}(\mathcal{A}, q, \mathfrak{r})$ where q is a state in $Q(\mathcal{A})$, and $\mathfrak{r}$ is a configuration with the property that there exists a rule $l \to r$ in $\mathfrak{R}(E)$, and a map $\mathfrak{m} : var(l) \to Q(\mathcal{A})$ in $\mathcal{M}(\mathcal{A}, q, l)$, such that $\mathfrak{r}$ is an $\mathfrak{m}$-residuo for r in $\mathcal{A}$.

The next lemma states that the language of the tree automaton $\mathcal{N}(\mathcal{A}, E)$ contains every term that can be reached from some term in $\mathcal{L}(\mathcal{A})$ by the application of one $R(E)$-multi-step. Additionally, if $\mathcal{A}$ is E-compatible, then so is $\mathcal{N}(\mathcal{A}, E)$.

Lemma 16. *Let Σ be a ranked alphabet, E be a set of variable-preserving Σ-equations, and $\mathcal{A}$ be reachable, co-reachable, and E-compatible tree automaton. Let $\multimap\!\!\to$ be the multistep relation induced by $R(E)$. Then $\mathcal{N}(\mathcal{A}, E)$ is reachable, co-reachable, and E-compatible. Additionally,*

$$\{t' \mid \exists t \in \mathcal{L}(\mathcal{A}),\ t \multimap\!\!\to t'\} \subseteq \mathcal{L}(\mathcal{N}(\mathcal{A}, E)). \tag{4}$$

Proof. Let $\mathcal{A} = (Q, \Sigma, F, \Delta)$ be a reachable, co-reachable, E-compatible tree automaton, and let $\mathcal{N}(\mathcal{A}, E) = (Q \cup Q', \Sigma, F, \Delta \cup \Delta')$ where $Q \cap Q' = \emptyset$ and $\Delta \cap \Delta' = \emptyset$. Note that every state of $\mathcal{A}$ is a state of $\mathcal{N}(\mathcal{A}, E)$, and every transition of $\mathcal{A}$ is a transition of $\mathcal{N}(\mathcal{A}, E)$. Therefore, for each state $q \in Q$, we have that $\mathcal{L}(\mathcal{A}, q) \subseteq \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$. Additionally, since $\mathcal{A}$ and $\mathcal{N}(\mathcal{A}, E)$ have the same final states, we have that $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{N}(\mathcal{A}, E))$.

We will separate the remainder of the proof in two parts. In the first part (Completeness Proof), we show that each term derivable in a single parallel step from some term in $\mathcal{L}(\mathcal{A})$ also belongs to $\mathcal{L}(\mathcal{N}(\mathcal{A}, E))$. In the second part (Soundness Proof), we show that if two terms s and t reach the same state in $\mathcal{N}(\mathcal{A}, E)$, the equation $s \simeq t$ belongs to the equational closure of E.

Completeness Proof. Let $q \in Q$, $t \in \mathcal{L}(\mathcal{A}, q)$, and $t' \in Ter(\Sigma)$ be a ground term such that $t \multimap\!\!\to t'$. We claim that $t' \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$. Since $\mathcal{A}$ and $\mathcal{N}(\mathcal{A}, E)$ have the same set of final states, our claim implies that $\{t' \mid \exists t \in \mathcal{L}(\mathcal{A}),\ t \multimap\!\!\to t'\} \subseteq \mathcal{L}(\mathcal{N}(\mathcal{A}, E))$. The proof of the claim is by induction on the structure of t, using the definition of multi-step (Definition 4). In the base case, $t = f$ for some function symbol f of arity 0. There are two subcases to be analyzed. In the first subcase, $t' = f$, by Condition 1 of Definition 4, and therefore t' is already in $\mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$. In the second subcase, the rewriting rule $t \to t'$ belongs to E, by Condition 3 of Definition 4 (with any substitution, since t has no variable an empty substitution). Since t' reaches state q in the component $\mathcal{C}(\mathcal{A}, q, \mathfrak{r})$, we have that t' reaches state q in $\mathcal{N}(\mathcal{A}, E)$. This concludes the proof of the claim in the base case.

Now, the inductive step is split into two cases. One corresponding to Condition 2 of Definition 4 and one corresponding to Condition 3.

For the inductive step corresponding to Condition 2, let $t = f(t_1, ..., t_{\mathfrak{a}(f)})$ and $t' = f(t'_1, ..., t'_{\mathfrak{a}(f)})$ where $t_i \multimap\!\!\to t'_i$ for each $i \in \{1, ..., \mathfrak{a}(f)\}$. Since $t \in \mathcal{L}(\mathcal{A}, q)$, there exist states $q_1, \ldots, q_{\mathfrak{a}(f)}$, and a transition $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$ in $\mathcal{A}$ such that t_i reaches q_i for each $i \in \{1, ..., \mathfrak{a}(f)\}$. Since by assumption $t_i \multimap\!\!\to t'_i$, using the inductive hypothesis we have that $t'_i \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q_i)$. This implies that $f(t'_1, ..., t'_{\mathfrak{a}(f)}) \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$.

For the inductive step corresponding to Condition 3, let $l \to r$ be a rewriting rule in $R(E)$ and $\sigma, \theta : X \to Ter(\Sigma)$, be substitutions such that $t = l^\sigma$, $t' = r^\theta$, and for each variable $x \in var(l)$, $\sigma(x) \multimap\!\!\to \theta(x)$. Since $t \in \mathcal{L}(\mathcal{A}, q)$, there exists a state substitution $\gamma : var(l) \to Q$ such that $l^\gamma \to_\Delta^* q$ and for each $x \in var(l)$, $\sigma(x) \to_\Delta^* \gamma(x)$. Since, by assumption, $\sigma(x) \multimap\!\!\to \theta(x)$, we have that by the induction hypothesis, $\theta(x) \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), \gamma(x))$ for each $x \in var(l)$. Additionally, $r^\gamma \to_{\Delta'}^* q$ by using transitions in the component $\mathcal{C}(\mathcal{A}, q, \mathfrak{r})$ corresponding to some residuo $\mathfrak{r}$. This implies that $t' \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$.

Soundness Proof. We claim that for each state q of $\mathcal{A}$, if $t' \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$ then there is a ground term $t \in \mathcal{L}(\mathcal{A}, q)$ such that the equation $t \simeq t'$ is implied by E. By transitivity and symmetry this implies that for each two terms t' and t'' in $\mathcal{L}(\mathcal{N}(\mathcal{A}, E), q)$, the equation $t' = t''$ is implied by E. Therefore, the tree automaton $\mathcal{N}(\mathcal{A}, E)$ is compatible with E.

The proof of the claim is by well founded induction with terms ordered by the strict subterm relation. In the base case, $t' = a$ for some function symbol $a \in \Sigma$ of arity 0. Let $a \to q$ be a transition in $\mathcal{N}(\mathcal{A}, E)$. If $a \to q$ belongs to Δ then $a \in \mathcal{L}(\mathcal{A}, q)$ and therefore by Condition 1 of Definition 4, we have that $a \multimap\!\!\to a$. On the other hand, if $a \to q$ belongs to Δ', then there is some rewriting rule $l \to r$ in $R(E)$ and some state-substitution $\gamma : var(l) \to Q$ such

that $a \to q$ is a transition in $\mathcal{C}(\mathcal{A}, q, \mathfrak{r})$ for some residuo $\mathfrak{r}$. Let $\sigma : var(l) \to Ter(\Sigma)$ be a substitution that associates with each variable $x \in var(l)$ an arbitrary term $\sigma(x)$ in $\mathcal{L}(\mathcal{A}, \gamma(x))$. Note that such a term is guaranteed to exist since by assumption, $\mathcal{A}$ is reachable. Then the term l^σ belongs to $\mathcal{L}(\mathcal{A}, q)$. Additionally, by Condition 3 of Definition 4, we have that $l^\sigma \multimap a^\tau$ where $\tau : \emptyset \to Ter(\Sigma)$ is the empty substitution. This verifies the claim in the base case.

Now, let $t' = f(t'_1, ..., t'_{\mathfrak{a}(f)})$ where f has arity at least 1. Then there exist states $q_1, \ldots, q_{\mathfrak{a}(f)}$ such that the transition $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$ belongs to $\Delta \cup \Delta'$ and $t'_i \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), q_i)$ for each $i \in \{1, ..., \mathfrak{a}(f)\}$. There are two cases to be analysed.

1. If $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$ belongs to Δ, then all states $q_1, ..., q_{\mathfrak{a}(f)}$ belong to Q. In this case, by the induction hypothesis, there exist terms $t_1, ..., t_{\mathfrak{a}(f)}$ such that for each $i \in \{1, ..., \mathfrak{a}(f)\}$, $t_i \in \mathcal{L}(\mathcal{A}, q_i)$ and $t_i \simeq t'_i$ is implied by E. This implies that the term $t = f(t_1, ..., t_{\mathfrak{a}(f)})$ is in $\mathcal{L}(\mathcal{A}, q)$. Finally, by congruence (Definition 1.4), we have that the equation $t \simeq t'$ is implied by E.
2. If $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$ belongs to Δ' then the states $q_1, ..., q_{\mathfrak{a}(f)}$ belong to some right component of $\mathcal{N}(\mathcal{A}, E)$. In other words, there is some rewriting rule $l \to r$ in $\mathfrak{R}$, some substitution $\theta : var(r) \to Ter(\Sigma)$ with $t' = r^\theta$, and some state-substitution $\gamma : var(l) \to Q$ satisfying the following conditions: (a) $l^\gamma \to^*_\Delta q$. (b) $r^\gamma \to^*_{\Delta'} q$. (c) For each $x \in var(r)$, $\theta(x) \in \mathcal{L}(\mathcal{N}(\mathcal{A}, E), \gamma(x))$. By the induction hypothesis, for each $x \in var(r) \subseteq var(l)$ there is a term s_x that belongs to $\mathcal{L}(\mathcal{A}, \gamma(x))$ such that the equation $s_x \simeq \theta(x)$ is implied by E. Additionally, since $\mathcal{A}$ is reachable, for each variable $y \in var(l) \backslash var(r)$ there is at least one term s_y in $\mathcal{L}(\mathcal{A}, \gamma(y))$. Let $\sigma : var(l) \to Ter(\Sigma)$ be the substitution that sets $\sigma(x) = s_x$ for each variable $x \in var(r)$, and which sets $\sigma(y) = s_y$ for each variable $y \in var(l) \backslash var(r)$. Then the term l^σ belongs to $\mathcal{L}(\mathcal{A}, q)$. Since $l \to r$ is a rewriting rule in $R(E)$, $t = l^\sigma$, $t' = r^\theta$, and $\sigma(x) \simeq \theta(x)$ is implied by E for each $x \in var(l) \backslash var(r)$, we have that the equation $t \simeq t'$ is implied by E. This concludes the proof of the lemma.

□

Now, let $t \in Ter(\Sigma)$ be a ground term over Σ, and let E be a set of Σ-equations. Let $\mathcal{B}(t)$ be the tree automaton associated with t according to Definition 12. For each $d \in \mathbb{N}$ we inductively define the tree automaton $\mathcal{A}(E, d, t)$ as follows.

$$\mathcal{A}(E, d, t) = \begin{cases} \mathcal{B}(t) \text{ if } d = 0 \\ \\ \mathcal{N}(\mathcal{A}(E, d-1, t),\ E) \qquad \text{if } d \geq 1. \end{cases} \tag{5}$$

By Proposition 13, $\mathcal{B}(t)$ is reachable, co-reachable and E-compatible. Additionally, $\mathcal{L}(\mathcal{A}(t)) = \{t\}$. Using Lemma 16, if follows by induction on d that $\mathcal{A}(E, d, t)$ is E-compatible and that Equation 2 holds:

$$\{t' \mid t \overset{d}{\multimap} t'\} \subseteq \mathcal{L}(\mathcal{A}(E, d, t)).$$

This almost concludes the proof of Lemma 10. The only caveat is that it is not immediately clear that the automaton $\mathcal{A}(E, d, t)$ can be constructed in time $f_E(d) \cdot |t|$ for some function $f_E(d)$ independent of the size of t (Subsection 4.2). Proving this upper bound will require performing a careful analysis of the ratio between the size of a tree automaton $\mathcal{A}$ and the size of the tree automaton $\mathcal{N}(\mathcal{A}, E)$. The complete analysis is carried out in the full version of this paper.

4.2 Time Analysis

Let $\mathcal{A} = (\Sigma, Q, \Delta, F)$ be a tree automaton, and let $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$ be a transition in Δ. We say that q is the *consequent* of $f(q_1, ..., q_{\mathfrak{a}(f)}) \to q$. The *in-degree* of a state q in Q, denoted by $\delta(q)$, is the number of transitions in Δ that have q as a consequent. The *maximum state in-degree* of $\mathcal{A}$, defined as $\delta(\mathcal{A}) = \max_{q \in Q} \delta(q)$, is the maximum in-degree of a state in $\mathcal{A}$.

Given a set E of Σ-equations, we consider the following parameters: We let $\mathfrak{a}$ be the maximum arity of a function symbol in Σ, s_1 be the maximum size of the left-hand side of a rule in $\mathfrak{R}(E)$, s_2 be the maximum size of a right-hand side of a rule in $\mathfrak{R}(E)$, and ρ be the maximum number of rules in $\mathfrak{R}(E)$ with the same left-hand side. We note that all four parameters listed above are fixed for fixed E, and often small.

Lemma 17. *Let Σ be a ranked alphabet, E be a set of variable-preserving Σ-equations, and t be a ground term in $Ter(\Sigma)$. The tree automaton $\mathcal{A}(t, E, d)$ can be constructed in time $s_2^d \cdot (e \cdot \rho \cdot \mathfrak{a})^{s_1^{2 \cdot d}} \cdot (\log |\mathfrak{R}|) \cdot |t|$.*

5 Conclusion

In this work, we introduced a novel framework for variable-preserving equational reasoning. More specifically, we showed that sound over-approximation equational reasoning using variable-preserving rules is fixed-parameter tractable when parameterized by the depth of derivations.

We obtain our main result by leveraging tree automata completion techniques, to show that given a tree automaton $\mathcal{A}$ compatible with a set of equations E, one can construct a tree automaton $\mathcal{N}(\mathcal{A}, E)$ whose language $\mathcal{L}(\mathcal{N}(\mathcal{A}, E))$ soundly over-approximates the set of terms derivable in a single parallel step from the terms in $\mathcal{L}(\mathcal{A})$. It is worth noting that in this context, over-approximation in the construction of $\mathcal{N}(\mathcal{A}, E)$ is a necessity, given that in general, the task of constructing a tree automaton whose language precisely matches the set of terms derivable from $\mathcal{L}(\mathcal{A})$ in a single parallel step is uncomputable.

Interestingly, in many contexts such as automated theorem proving, satisfiability modulo theories, etc, over-approximation is an advantage, because in these applications, the goal is to prove equivalence between terms modulo some set of equational axioms irrespectively of the derivation depth. In this sense, although our algorithm is always guaranteed to certify that a given equation is provable in depth at most d, as a bonus it can also certify the provability of some equations that are indeed logical consequences of the set of equational axioms E but whose demonstration using rewriting rules require larger depth.

Acknowledgements

This work received financial support from the Research Council of Norway, grant numbers 288761 and 326537.

References

Baader, F.; and Nipkow, T. 1999. *Term rewriting and all that*. Cambridge university press.

Bachmair, L.; and Dershowitz, N. 1989. Completion for rewriting modulo a congruence. *Theoretical Computer Science*, 67(2-3): 173–201.

Bachmair, L.; and Ganzinger, H. 1998. Equational reasoning in saturation-based theorem proving. *Automated deduction—a basis for applications*, 1: 353–397.

Birkhoff, G. 1935. On the Structure of Abstract Algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(4): 433–454.

Comon, H.; Dauchet, M.; Gilleron, R.; Löding, C.; Jacquemard, F.; Lugiez, D.; Tison, S.; and Tommasi, M. 2007. Tree Automata Techniques and Applications. Available on: http://www.grappa.univ-lille3.fr/tata. Release October, 12th 2007.

Downey, R. G.; and Fellows, M. R. 1995. Fixed-parameter tractability and completeness II: On completeness for W [1]. *Theoretical Computer Science*, 141(1-2): 109–131.

Echenim, M.; Peltier, N.; and Tourret, S. 2013. An approach to abductive reasoning in equational logic. In *IJCAI 2013-International Joint Conference on Artificial Intelligence*, 531–537. AAAI Press.

Felgenhauer, B.; and Thiemann, R. 2014. Reachability analysis with state-compatible automata. In *Proc. of the 8th International Conference on Language and Automata Theory and Applications*, LNCS, 347–359. Springer.

Feuillade, G.; Genet, T.; and Tong, V. V. T. 2004. Reachability analysis over term rewriting systems. *Journal of Automated Reasoning*, 33(3-4): 341–383.

Fish, A.; and Lisitsa, A. 2014. Detecting unknots via equational reasoning, I: Exploration. In *International Conference on Intelligent Computer Mathematics*, 76–91. Springer.

Genet, T. 1998. Decidable approximations of sets of descendants and sets of normal forms. In *Proc. of the 9th International Conference on Rewriting Techniques and Applications*, LNCS, 151–165.

Genet, T.; and Rusu, V. 2010. Equational approximations for tree automata completion. *Journal of Symbolic Computation*, 45(5): 574–597.

Iosif, R.; Rogalewicz, A.; and Vojnar, T. 2014. Deciding entailments in inductive separation logic with tree automata. In *International Symposium on Automated Technology for Verification and Analysis*, 201–218. Springer.

Jan Willem Klop, R. d. V., Vincent van Oostrom. 2003. Orthogonality. *Term Rewriting Systems (Chapter 4). Cambridge Tracts in Theoretical Computer Science*, 55.

Jenner, B.; McKenzie, P.; and Torán, J. 1998. A note on the hardness of tree isomorphism. In *Proceedings. Thirteenth Annual IEEE Conference on Computational Complexity (Formerly: Structure in Complexity Theory Conference)(Cat. No. 98CB36247)*, 101–105. IEEE.

Korp, M.; and Middeldorp, A. 2009. Match-bounds revisited. *Information and Computation*, 207(11): 1259–1283.

McNulty, G. F. 1992. A field guide to equational logic. *Journal of symbolic computation*, 14(4): 371–397.

N. Dershowitz, J. P. J. 1990. Rewrite systems. *Handbook of Theoretical Computer Science (Chapter 6)*, 243–320.

Neugebauer, G.; and Petermann, U. 1998. Specifications of Inference Rules: Extensions of the PTTP Technique. In *Automated Deduction—A Basis for Applications*, 167–188. Springer.

Plaisted, D. A. 1993. Equational reasoning and term rewriting systems. *Handbook of logic in artificial intelligence and logic programming*, 1: 273–364.

van Oostrom, V. 1999. Normalisation in weakly orthogonal rewriting. In *Proc. of the 10th International Conference on Rewriting Techniques and Applications*, LNCS, 60–74.