A Complete Axiomatization of Strict Equality over Infinite Trees (Extended Version)* Technical Report SIC-03-2009 UCM

Javier Álvez and Francisco J. López-Fraguas

Universidad Complutense de Madrid
javieralvez@fdi.ucm.es fraguas@sip.ucm.es

Abstract. Computing with data values that are some kind of trees — finite, infinite, rational— is at the core of declarative programming, either logic, functional or functional-logic. Understanding the logic of trees is therefore a fundamental question with impact in different aspects, like language design, including constraint systems or constructive negation, or obtaining methods for verifying and reasoning about programs. The theory of *true* equality over finite or infinite trees is quite well known. In particular, a seminal paper by Maher proved its decidability and gave a complete axiomatization of the theory. However, the sensible notion of equality for functional and functional-logic languages with a lazy evaluation regime is *strict* equality, a computable approximation to true equality for possibly infinite and partial trees. In this paper, we investigate the first-order theory of strict equality, arriving to remarkable and not obvious results: the theory is again decidable and admits a complete axiomatization, not requiring predicate symbols other than strict equality itself. Besides, the results stem from an effective —taking into account the intrinsic complexity of the problem— decision procedure that can be seen as a constraint solver for general strict equality constraints.

1 Introduction

Computing with data values that are —or can be interpreted as— some kind of trees is at the core of declarative programming, either logic, functional or functional-logic programming. The family of trees may vary from finite trees, for the case of standard logic programming, infinite rational trees, for the case of Prolog-II and variants, or infinite trees (that correspond to data values in constructor data-types) for the case of functional or functional-logic programming that allow non-terminating programs by following a lazy evaluation regime.

Understanding trees, in particular the logical principles governing tree equality, is a fundamental question with impact in different aspects of declarative

* This work has been partially supported by the Spanish projects TIN2005-09207-C03-03, TIN2008-06622-C03-01, S-0505/TIC/0407, UCM-BSCH-GR58/08-910502, TIN2007-66523 and GIU07/35.

programming languages. For instance, adding constructive negation abilities to a logic language requires solving complex Herbrand constraints over finite trees.

The theory of *true* equality $\approx$[1] over finite or infinite trees is quite well known. In a seminal paper [12], Maher proved its decidability and gave a complete axiomatization for the cases of finite and infinite trees, and finite and infinite signatures. In another influential paper [4], the authors provided more effective decision procedures, based on reduction to solved forms by quantifier elimination.

In functional or functional-logic languages like Haskell, Curry or Toy [13, 7, 9], the universe of interest is that of (possibly) *infinite partial trees*, because non-terminating programs handled with lazy evaluation lead to that kind of trees as denotations of expressions. *Partiality* means that some of the tree components may be undefined. For instance, with the definitions $loop = loop$ and $repeat(x) = [x|repeat(x)]$, $repeat(0)$ denotes the infinite list $[0, 0, 0, \ldots]$, $repeat(loop)$ denotes the infinite partial list $[\bot, \bot, \bot, \ldots]$, and $[0|loop]$ denotes the finite partial list $[0|\bot]$. In those languages, true equality is not the sensible notion to consider, because true equality over partial trees is not Scott continuous (hence not computable). It is then replaced (see e.g. [7]) by *strict* equality $==$ —the largest continuous approximation to $\approx$—, defined as the restriction of $\approx$ to finite and total trees. The theories of equality and of strict equality for infinite partial trees are far from being the same: for instance, the formulae $\forall x\ x \approx x$ and $\exists x\ x \approx s(x)$ hold, while $\forall x\ x == x$ and $\exists x\ x == s(x)$ do not.

As far as we know, a comprehensive study of the full first-order theory of strict equality has not been done before. Certainly, strict equality is incorporated as primitive in the aforementioned languages, and there are several works incorporating various Herbrand constraint systems —and corresponding solving procedures— to functional-logic languages [1, 8, 2]. But in all cases, the considered class of formulae over $==$ is only a subset of general first-order formulae. Besides, the works that study true equality cannot be easily extended to handle strict equality. For example, in [6] the authors propose to extend the theory of true equality with the predicate $finite_{/1}$, that only holds for finite trees. Coping with strict equality would require an additional predicate $total_{/1}$ (to characterize those trees not having $\bot$ as component). Comparing that hypothetical approach with our proposal of directly considering the first-order theory of $==$, we see several disadvantages: first, the axiomatization of $\approx + finite + total$ would be larger than ours, leading also to larger proofs of decidability and completeness; second, the theory would be less directly connected to the mentioned languages (Haskell, Curry, Toy) because programs in those languages use $==$ but not $finite$ or $total$.

Our aim is precisely to investigate the full first-order theory of strict equality over the algebra $\mathcal{IT}$ of possibly infinite partial trees. Note that decidability and existence of complete axiomatization for $\approx$ says nothing about the same problems for $==$, even although $==$ is a strict subset of $\approx$ (i.e., $\forall x \cdot y\ (\ x == y \rightarrow x \approx y\ )$ is valid in $\mathcal{IT}$). These are indeed the main questions tackled in this paper:

– Does the theory of $==$ over $\mathcal{IT}$ admit a complete recursive axiomatization?

[1] By *true* equality we mean $t_1 \approx t_2$ iff t_1 and t_2 are the same tree.

– In the affirmative case, is it possible that the axioms use only the symbol ==? We cannot discard *a priori* the possibility of explicitly connecting == to $\approx$, although the resulting set of axioms and transformations rules would be more complicated since we would need to add connection axioms (like the formula stated above) to the axiomatization of $\approx$ in [12].
– A complete recursive axiomatization of a theory implies its decidability (at least a brute force decision procedure exists). Can we give a more practical decision procedure, in the style of [4]? As a matter of fact, such a procedure —if existing— will be itself a proof of completeness for the theory.

We obtain affirmative answers to these questions, both in the cases of infinite and finite signatures. Our paper does not look for immediate applications, keeping in a theoretical realm and trying to achieve fundamental and not obvious results about strict equality that could be a basis for potential applications: the design of constraint systems more expressive than existing ones or the development of reasoning frameworks for functional-logic programs with built-in equality.

Outline of the paper. In the next section, we provide preliminary definitions and notation. In Section 3, we give an axiomatization for strict equality. Next, in Section 4, we first introduce some transformation rules and then provide decision methods for strict equality, distinguishing the cases of infinite and finite signatures. Finally, in Section 5, we discuss complexity issues and future work.

2 Preliminaries

Let $\mathcal{V}$ be a set of countable variables and $\Sigma = \mathcal{P}_\Sigma \cup \mathcal{F}_\Sigma$ a signature of predicate and function symbols where each symbol s has an associated arity n, denoted by $s_{/n}$, and $\mathcal{P}_\Sigma$ exclusively consists of the symbol $==_{/2}$, known as *strict equality*. For technical convenience, we assume that $\mathcal{F}_\Sigma$ contains at least a 0-ary function symbol (constant), an n-ary function symbol with $n > 0$ and a distinguished 0-ary function symbol $\bot$ known as *bottom*. If Σ contains a finite number of function symbols, then Σ is said to be *finite*. Otherwise, Σ is infinite. By using the name *function*, we follow the tradition of first-order logic, but note that the notion of function corresponds to the notion of free constructor in functional/functional-logic programming and not to defined function, which plays no role in this paper.

We consider the classical definitions of finite and infinite ground trees. The interested reader is referred to [3] for an exhaustive definition. A tree is said to be *partial* if it contains $\bot$ at some node. Otherwise, the tree is *total*. The algebra of finite and infinite trees are respectively denoted by $\mathcal{FT}$ and $\mathcal{IT}$. Besides, we also refer to [4] for the definitions that do not appear in this paper.

A *term* (or *constructor term*) is either a variable $v \in \mathcal{V}$ or an expression $f(t_1, \ldots, t_n)$ where $f_{/n} \in \mathcal{F}_\Sigma$ and $t_1, \ldots, t_n$ are terms. For any terms t and s, the expression $t[s]$ denotes that s occurs in t. For any $n > 0$, an n-tuple of terms is denoted by $\langle t_1, \ldots, t_n \rangle$ and abbreviated by $\overline{t}$. When convenient, we also treat $\overline{t}$ as the set of its components. As for the case of trees, a term t is said to be *partial* if $t = s[\bot]$, and t is *total* otherwise. We denote by $\mathtt{Var}(t)$ the set of variables

occurring in t. Besides, a term is said to be *ground* iff it is variable-free. The *size* of a term t is the number of function symbols occurring in t.

A *sentence* ϕ is an arbitrary first-order formula built with Σ. In our case, the only predicate symbol is $==$. Thus, *atomic formulas* are $true$, $false$, *strict equations* $t_1 == t_2$ or *negated equations* $\neg t_1 == t_2$. Being $\overline{r} = \langle r_1, \ldots, r_n \rangle$ and $\overline{s} = \langle s_1, \ldots, s_n \rangle$, $\overline{r} == \overline{s}$ (resp. $\neg \overline{r} == \overline{s}$) abbreviates $r_1 == s_1 \wedge \ldots \wedge r_n == s_n$ (resp. $\neg r_1 == s_1 \vee \ldots \vee \neg r_n == s_n$). Sentences may use propositional connectives $(\neg, \wedge, \vee, \rightarrow, \leftrightarrow)$ and quantifiers $(\exists, \forall)$. Q stands for both kinds of quantifiers. $\texttt{Free}(\phi)$ denotes the set of *free* variables of ϕ. If $\texttt{Free}(\phi) = \emptyset$, then ϕ is *closed*. ϕ^Q denotes the Q-closure of ϕ and $\phi^{Q \setminus \overline{w}}$ denotes $Q\overline{v}\ \phi$, where $\overline{v} = \texttt{Free}(\phi) \setminus \overline{w}$.

Now we recall some semantics of first-order logic. An *interpretation* $\mathcal{A}$ is a carrier set A together with interpretations $f^{\mathcal{M}}$, $p^{\mathcal{M}}$ for the symbols in Σ. Given $\mathcal{A}$, an *assignment* σ maps variables to values in A; if $\phi\sigma$ is true (according to standard rules for truth-valuation) in $\mathcal{A}$, we say that σ is a *solution* (in $\mathcal{A}$) of ϕ. $\mathcal{A}$ *models* ϕ, written $\mathcal{A} \models \phi$, if all assignments are solutions in $\mathcal{A}$ of ϕ. Notice that, for given $\mathcal{A}$, ϕ and σ, σ must be a solution in $\mathcal{A}$ of either ϕ or $\neg\phi$; moreover, if ϕ is closed, either $\mathcal{A} \models \phi$ or $\mathcal{A} \models \neg\phi$ (the latter being equivalent to $\mathcal{A} \not\models \phi$).

A *theory* $\mathcal{T}$ is a set of closed sentences. $\mathcal{A}$ is a *model* of $\mathcal{T}$, written $\mathcal{A} \models \mathcal{T}$, if $\mathcal{A} \models \phi$ for each $\phi \in \mathcal{T}$. A formula ϕ is a *logical consequence* of $\mathcal{T}$, written $\mathcal{T} \models \phi$, if $\mathcal{A} \models \phi$ whenever $\mathcal{A} \models \mathcal{T}$. This notation extends naturally to sets Φ of formulas. A sentence ϕ is *satisfiable* (or *solvable*) in $\mathcal{T}$, if $\mathcal{T} \models \phi^{\exists}$. Two sentences ϕ_1 and ϕ_2 are *(logically) equivalent in* $\mathcal{T}$, denoted by $\phi_1 \equiv \phi_2$, if $\mathcal{T} \models \phi_1 \leftrightarrow \phi_2$. A theory $\mathcal{T}$ is *complete* iff for any closed sentence ϕ either $\mathcal{T} \models \phi$ or $\mathcal{T} \models \neg\phi$ holds. The *theory* $\mathcal{T}_{\mathcal{A}}$ of $\mathcal{A}$ is the set of all closed ϕ such that $\mathcal{A} \models \phi$. Note that $\mathcal{T}_{\mathcal{A}}$ is always complete. $\mathcal{A}_1$ and $\mathcal{A}_2$ are *elementarily equivalent* if $\mathcal{T}_{\mathcal{A}_1} = \mathcal{T}_{\mathcal{A}_2}$. A complete axiomatization of $\mathcal{A}$ is a theory $\mathcal{S} \subseteq \mathcal{T}_{\mathcal{A}}$ such that $\mathcal{S} \models \mathcal{T}_{\mathcal{A}}$ (or, equivalently, $\mathcal{S}$ is a complete theory and $\mathcal{A} \models \mathcal{S}$). Usually one is interested in *recursive* axiomatizations where the property '$\phi \in \mathcal{S}$' is decidable.

Given two sentences ϕ_1 and ϕ_2, a *transformation rule* $\phi_1 \mapsto \phi_2$ replaces any occurrence of ϕ_1 in a formula (module variable renaming) with ϕ_2. The application of a transformation rule R to ϕ_1 yielding ϕ_2 is denoted by $\phi_1 \rightsquigarrow_R \phi_2$. A transformation rule R is said to be *correct* in a theory $\mathcal{T}$ iff for any two formulas ϕ_1 and ϕ_2 such that $\phi_1 \rightsquigarrow_R \phi_2$ we have that $\phi_1 \equiv \phi_2$.

3 An Axiomatization of Strict Equality

Strict equality is a particular case of classical equality where, besides being syntactically equal, two terms have to be finite and total to be strictly equal.

Definition 1 (Strict equality). *Two trees t_1 and t_2 are* strictly equal*, denoted by $t_1 == t_2$, iff t_1 and t_2 are the same finite and total tree.* □

Strict equality allows us to characterize the subset of $\mathcal{IT}$ consisting of finite and total trees: x is a finite and total tree $\iff x == x$.

In Figure 1, we propose an axiomatization of infinite trees with strict equality, which is similar, but not equal, to the one of finite trees with equality given

$(\mathbf{A_1})$ For every $f_{/n} \in \mathcal{F}_\Sigma$ such that $f \neq \bot$

$$\forall \overline{x}\ \forall \overline{y}\ (\ f(\overline{x}) == f(\overline{y})\ \leftrightarrow\ \overline{x} == \overline{y}\ )$$

$(\mathbf{A_2})$ For every $f_{/n}, g_{/m} \in \mathcal{F}_\Sigma$ such that $f \neq g$

$$\forall \overline{x}\ \forall \overline{y}\ \neg f(\overline{x}) == g(\overline{y})$$

$(\mathbf{A_3})$ For every term $t[x]$ except x such that $\overline{y} = \mathtt{Var}(t[x]) \setminus \{x\}$

$$\forall x\ \forall \overline{y}\ \neg x == t[x]$$

$(\mathbf{A_4})$ Bottom: $\forall x\ \neg x == \bot$

$(\mathbf{A_5})$ Symmetry: $\forall x\ \forall y\ (\ x == y\ \rightarrow\ y == x\ )$

$(\mathbf{A_6})$ Transitivity: $\forall x\ \forall y\ \forall z\ (\ \neg x == y\ \vee\ \neg y == z\ \vee\ x == z\ )$

$(\mathbf{A_7})$ Domain Closure Axiom or DCA: *% only for finite signatures*

$$\forall x\ (\ \neg x == x\ \vee\ \bigvee\nolimits_{f_{/n} \in \mathcal{F}_\Sigma} \exists \overline{w}\ x == f(\overline{w})\ )$$

Fig. 1. Axiomatization of Infinite Trees with Strict Equality

in [12]. The main difference comes from the fact that strict equality is not reflexive: because of $\mathbf{A_3}$ and $\mathbf{A_4}$, non-finite/non-total trees are not strictly equal to themselves. Due to this property, $\bot$ and the remaining functions in $\mathcal{F}_\Sigma$ have a different treatment. We distinguish two cases, depending on whether Σ is either finite or infinite. In the case of infinite signatures, the axiomatization of strict equality over $\mathcal{IT}$ consists of $\mathbf{A_1} - \mathbf{A_6}$[2] and is denoted by $\mathcal{E}_{inf}$. For finite signatures, the axiomatization also includes $\mathbf{A_7}$ and is denoted by $\mathcal{E}_{fin}$. Axiom $\mathbf{A_7}$ is an adaptation of the *Domain Closure Axiom* introduced in [14] to the case of $==$, which prevents the existence of isolated finite and total trees in the algebra. Note that $\mathbf{A_7}$ does not provide any information about non-finite/non-total trees.

To simplify statements and reasonings, we will frequently use $\mathcal{E}$ to refer indistinctly to $\mathcal{E}_{inf}$ and $\mathcal{E}_{fin}$ for the respective cases of infinite and finite signatures.

We will also abuse of notation $\mathcal{IT}$ to refer either to the set of infinite trees or the interpretation with $\mathcal{IT}$ as carrier and symbols in Σ interpreted in the natural way (symbols in $\mathcal{F}_\Sigma$ as free constructors and $==$ as strict equality).

Three basic questions about $\mathcal{E}$ arise: Are the axioms of $\mathcal{E}$ correct for $\mathcal{IT}$? Are there enough axioms as to characterize $==$? Are there too many? The first and third questions are addressed in the next proposition. The second one concerns completeness of $\mathcal{E}$, and is far from being a trivial question. It will be proved by means of a decision procedure based on some equivalences under $\mathcal{E}$ used as transformation rules for quantifier elimination.

[2] To be more precise, $\mathbf{A_1} - \mathbf{A_3}$ are axiom schemes where $\mathbf{A_3}$ embodies an infinite number of instances (also $\mathbf{A_1}$ and $\mathbf{A_2}$ in the case of infinite signatures). To simplify notation, $\mathbf{A_4} - \mathbf{A_7}$ can be also taken as axiom schemes with a single instance.

Theorem 1 (Correctness and minimality of $\mathcal{E}$).

(i) $\mathcal{IT} \models \mathcal{E}$.
(ii) $\mathcal{E} \setminus \mathbf{A_i}$ *is not a complete theory for any (axiom scheme)* $\mathbf{A_i}$ *in* $\mathcal{E}$.

Sketch of the proof. A direct inspection of $\mathcal{E}$ proves *(i)*. For *(ii)*, it is enough to prove that $\mathcal{E} \setminus \mathbf{A_i}$ is not complete for each axiom $\mathbf{A_i}$ taken from $\mathcal{E}$. This is proved by giving a model $\mathcal{M}$ of $\mathcal{E} \setminus \mathbf{A_i}$ and a closed formula φ that is valid in $\mathcal{M}$ but not in $\mathcal{IT}$ (therefore $\neg\varphi$ is valid in $\mathcal{IT}$). Since $\mathcal{M}$ and $\mathcal{IT}$ are both models of $\mathcal{E} \setminus \mathbf{A_i}$, neither φ nor $\neg\varphi$ can be deduced from $\mathcal{E} \setminus \mathbf{A_i}$, which therefore is not complete. Let us examine $\mathcal{E} \setminus \mathbf{A_i}$ for a couple of interesting cases. We assume a constant a and a function symbol $f_{/n}$ ($n > 0$, say $n = 1$ for simplicity) in $\mathcal{F}_\Sigma$.

($\mathbf{A_1}$) We build a model $\mathcal{M}$ of $\mathcal{E} \setminus \mathbf{A_1}$ as follows: the carrier and the interpretation of symbols are as in the standard interpretation $\mathcal{IT}$, with the exception that $f^{\mathcal{M}}(\bot)$ is defined to be the tree $f(a)$ (instead of being the tree $f(\bot)$, as in the standard $\mathcal{IT}$). It is not difficult to see that $\mathcal{M}$ is indeed a model of $\mathcal{E} \setminus \mathbf{A_1}$. Now we can take $\varphi = f(\bot) == f(a)$ as the desired formula such that $\mathcal{M} \models \varphi$ and $\mathcal{IT} \models \neg\varphi$.

($\mathbf{A_3}$) In this case, the carrier of $\mathcal{M}$ is the set of possibly partial infinite trees built with $\mathcal{F}_\Sigma \cup \{b_{/0}\}$, where b is a new constant. All symbols in Σ are interpreted as usual, except that $f^{\mathcal{M}}(b)$ is defined to be the tree b instead of being the tree $f(b)$ (that exists also in the carrier). It can be seen that $\mathcal{M}$ is a model of $\mathcal{E} \setminus \mathbf{A_3}$. Note that $b =={}^{\mathcal{M}} f^{\mathcal{M}}(b)$ holds in the carrier. This does not contradict the axiom scheme $\mathbf{A_2}$ (because $\forall x \neg b == f(x)$ is not an instance of $\mathbf{A_2}$, as b is not in the original Σ), but serves to show that $\mathcal{M} \models \varphi$ where $\varphi = \exists x\ x == f(x)$. However $\mathcal{IT} \models \neg\varphi$, since $\neg\varphi$ is equivalent to $\forall x\ \neg x == f(x)$, which is an instance of $\mathbf{A_3}$. □

Remark. $\mathcal{IT} \models \mathcal{E}$ is proved by direct inspection. In particular, $\mathbf{A_3}$ is correct since, by definition, infinite trees are not strictly equal. Regarding minimality, we cannot replace *(ii)* by the stronger result "no stricter subset of $\mathcal{E}$ is a complete theory". The reason is that some instances of $\mathbf{A_3}$ can be skipped from $\mathcal{E}$ without losing completeness. For example, as discussed also in [11] for true equality, the formula $\forall x\ \neg x == f(x)$ follows from $\forall x\ \neg x == f(f(x))$ (and $\mathbf{A_1}$, $\mathbf{A_6}$). □

Finally, we show that $==$ satisfies the following weak version of reflexivity.

Proposition 1. $\mathcal{E} \models \forall x\ (\ x == x\ \leftrightarrow\ \exists y\ x == y\ )$

Proof. One implication is trivial. The reverse implication is proved by symmetry and transitivity. □

4 A Decision Method for Strict Equality

In this section, we prove that the theory of strict equality is decidable by providing an algorithm that transforms any initial constraint into an equivalent disjunction of formulas in solved form. This algorithm is based on the well-known

Bottom

$$(\mathbf{B_1})\ \ x == t[\bot] \ \mapsto false$$
$$(\mathbf{B_2})\ \neg x == t[\bot] \mapsto \ true$$

Non-finite trees

$$(\mathbf{NFT_1})\ \neg x == x \wedge \neg r == s[x] \mapsto \neg x == x$$
$$(\mathbf{NFT_2})\ \ \neg x == x \wedge r == s[x] \ \mapsto \ \ false$$
$$(\mathbf{NFT_3})\qquad \forall y\ \neg x == y \qquad \mapsto \neg x == x$$

Finite trees

$$(\mathbf{FT})\ \ x == x \wedge r == s[x] \ \ \mapsto \ \ r == s[x]$$

Decomposition

$$(\mathbf{D_1})\ \ f(r_1, \ldots, r_n) == f(s_1, \ldots, s_n) \ \mapsto \ \ r_1 == s_1 \wedge \ldots \wedge r_n == s_n$$
$$(\mathbf{D_2})\ \neg f(r_1, \ldots, r_n) == f(s_1, \ldots, s_n) \mapsto \neg r_1 == s_1 \vee \ldots \vee \neg r_n == s_n$$

Clash

$$(\mathbf{C_1})\ \ f(r_1, \ldots, r_m) == g(s_1, \ldots, s_n) \ \mapsto false \ \text{ if } f \neq g$$
$$(\mathbf{C_2})\ \neg f(r_1, \ldots, r_m) == g(s_1, \ldots, s_n) \mapsto \ true \ \text{ if } f \neq g$$

Occur-check

$$(\mathbf{O_1})\ \ x == t[x] \ \mapsto false \ \text{ if } x \neq t[x]$$
$$(\mathbf{O_2})\ \neg x == t[x] \mapsto \ true \ \text{ if } x \neq t[x]$$

Replacement

$$(\mathbf{R})\ \ x == t \wedge \varphi[x] \ \ \mapsto \ \ x == t \wedge \varphi[x \leftarrow t] \ \text{ if } t \text{ is total and } x \notin \mathtt{Var}(t)$$

Existential quantification elimination

$$(\mathbf{EE_1})\ \ \exists w\ (\ w == w \wedge \varphi\ ) \ \mapsto \qquad \varphi \qquad \text{if } w \notin \mathtt{Var}(\varphi)$$
$$(\mathbf{EE_2})\ \ \exists w\ (\ w == t \wedge \varphi\ ) \ \mapsto \overline{x} == \overline{x} \wedge \varphi \ \text{ if } t \text{ is total, } \overline{x} = \mathtt{Var}(t) \text{ and } w \notin \mathtt{Var}(t) \cup \mathtt{Var}(\varphi)$$
$$(\mathbf{EE_3})\ \exists w\ (\ \neg w == w \wedge \varphi\ ) \mapsto \qquad \varphi \qquad \text{if } w \notin \mathtt{Var}(\varphi)$$

Existential quantification introduction

$$(\mathbf{EI})\ \ r == s[x] \ \ \mapsto \ \ \exists w\ (\ x == w \wedge r == s[x \leftarrow w]\ )$$

Universal quantification elimination

$$(\mathbf{UE})\ \forall y\ (\neg y == t \vee \varphi) \mapsto \neg \overline{x} == \overline{x} \vee \varphi[y \leftarrow t] \ \text{ if } t \text{ is total, } \overline{x} = \mathtt{Var}(t) \text{ and } y \notin \mathtt{Var}(t)$$

Tautology

$$(\mathbf{T})\ \ \varphi \ \ \mapsto \ \ \varphi \wedge (\ x == x \vee \neg x == x\ )$$

Split

$$(\mathbf{S})\ \neg \exists \overline{w} \exists \overline{z}\ (\ x == t[\overline{w}] \wedge \varphi[\overline{w} \cdot \overline{z}]\ ) \mapsto \neg \exists \overline{w}\ (\ x == t[\overline{w}]\ ) \vee \exists \overline{w}\ (\ x == t[\overline{w}] \wedge \neg \exists \overline{z}\ \varphi[\overline{w} \cdot \overline{z}]\ )$$

Fig. 2. Transformation Rules

technique of quantifier elimination, as the algorithms proposed in [4, 12] for the equality theory. As in the above cited works, we distinguish two cases depending on whether the signature is finite or infinite. In the next subsections, we first provide a decision algorithm for the case of infinite signatures and then adapt that algorithm for finite ones. Those decision methods use the transformation rules introduced in Figure 2. Note that some conditions in rules, like those of **R**, are not necessary for correctness. Instead, they serve to discard the application of some rules when there exist more suitable ones. Some other basic transformations that are trivially correct in first-order logic, such as De Morgan's laws or double negation elimination, are also implicitly used in the decision methods.

Next, we prove that the transformation rules in Figure 2 are correct.

Theorem 2. *The transformation rules in Figure 2 are correct in* $\mathcal{E}$.

Proof. See Appendix. □

4.1 Infinite Signatures

In order to provide a decision algorithm, we first define a solved form for infinite signatures, called *basic formula*. Then, we show that two basic Boolean operations —conjunction and negation— can be performed on basic formulas. And, finally, we describe the decision algorithm.

Definition 2. *A* basic formula *for the variables* $\overline{x}$ *is either true, false* (closed basic formulas) *or a constraint* $\exists \overline{w}\ c(\overline{x}, \overline{w})$ *such that*

$$c(\overline{x}, \overline{w}) \;=\; \bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \;\wedge\; \overline{x}^2 == \overline{t} \;\wedge\; \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} (\neg w_i == s_{ij})^{\forall \setminus \overline{w}}$$

where $-$ $\overline{x} = \overline{x}^1 \cup \overline{x}^2$ *and* $\overline{x}^1 \cap \overline{x}^2 = \emptyset$,
$-$ $\overline{w} = \mathtt{Var}(\overline{t})$ *and* $\overline{x} \cap \overline{w} = \emptyset$,
$-$ *if* s_{ij} *is a variable, then* $s_{ij} \in \overline{w}$, *otherwise* s_{ij} *is total,* $w_i \notin \mathtt{Var}(s_{ij})$ *and* $\mathtt{Var}(s_{ij}) \cap \overline{x} = \emptyset$ *for every* $w_i \in \overline{w}$ *and* $1 \leq j \leq n_i$.

A formula is in basic normal form *(or BNF) if it is of the form* $Q\overline{y}\ \varphi[\overline{x} \cdot \overline{y}]$ *where* φ *is a disjunction of basic formulas for* $\overline{x} \cdot \overline{y}$. □

Example 1. Let $\{a_{/0}, g_{/1}, f_{/2}\} \subset \mathcal{F}_\Sigma$ and $\overline{x} = \{x_1, x_2, x_3\} \subset \mathcal{V}$. The sentences $\exists \overline{w}\ (\ \neg x_1 == x_1 \wedge x_2 = g(w_1) \wedge x_3 == g(w_2) \wedge \neg w_1 == w_2 \wedge \forall v\ \neg w_2 == f(a, v)\ )$, $(\ \neg x_1 == x_1 \wedge \neg x_2 == x_2 \wedge \neg x_3 == x_3\ )$ and *true* are basic formulas for $\overline{x}$. □

First, we will show that the notion of basic formula is a solved form.

Theorem 3. *Any basic formula different from* $false$ *is satisfiable in* $\mathcal{E}_{inf}$.

Proof. The constraint *true* is trivially satisfiable. Thus, let us consider a basic formula for the variables $\overline{x}$ of the form

$$\exists \overline{w}\ (\ \bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \;\wedge\; \overline{x}^2 == \overline{t} \;\wedge\; \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} (\neg w_i == s_{ij})^{\forall \setminus \overline{w}}\ ).$$

The conjunction $\bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1$ is trivially satisfiable since the variables $\overline{x}^1$ do not occur in the rest of the basic formula. Hence, each variable in $\overline{x}^1$ can be assigned to a non-finite/non-total term without restrictions. The conjunction $\overline{x}^2 == \overline{t}$ is also satisfiable since it is an idempotent substitution. Therefore, we focus on $\varphi = \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} (\neg w_i == s_{ij})^{\forall \setminus \overline{w}}$. Being a finite conjunction, the number of function symbols occurring in some negated equation is necessarily finite. Thus, there exists an assignment to the variables $\overline{w}$ consisting of distinct finite and total trees, each one of them starting by a function symbol not occurring in φ. Hence, the basic formula is satisfiable. □

Second, we describe the transformation of any universally quantified disjunction of negated equations into an equivalent disjunction of basic formulas.

Proposition 2. *Any universally quantified disjunction of negated equations*

$$\forall \overline{v} \; (\; \neg w_1 == t_1 \; \vee \; \neg w_2 == t_2 \; \vee \; \ldots \; \vee \; \neg w_n == t_n \;) \tag{1}$$

where $w_i \notin \overline{v}$ for each $1 \leq i \leq n$ can be transformed into an equivalent disjunction of basic formulas for the variables $\overline{x} = \mathtt{Var}(t_1, \ldots, t_n) \setminus \overline{v}$.

Proof. If $n = 0$, then the proof is trivial. Assuming that $n > 0$, formula (1) is transformed into

$$\forall \overline{v}^1 \; (\neg w_1 == t_1) \; \vee \; \exists \overline{v}^1 \; (w_1 == t_1 \wedge \underbrace{\forall \overline{v}^2 \; [\neg w_2 == t_2 \vee \ldots \vee \neg w_n == t_n]}_{\varphi})$$

using **S**, where $\overline{v}^1 = \mathtt{Var}(t_1) \cap \overline{v}$ and $\overline{v}^2 = \overline{v} \setminus \overline{v}^1$. The formula $\forall \overline{v}^1 \; (\neg w_1 == t_1)$ can be transformed into a disjunction of basic formulas for $\overline{x}$ by applying the rule **T** on each variable from $\overline{x}$ and the rules $\mathbf{NFT_1}$, **EI** and **R** on w_1. Using the same transformation, φ can be transformed into a disjunction of basic formulas for the variables $\overline{x} \setminus w_1$ which, in conjunction with $w_1 == t_1$, is a disjunction of basic formulas for $\overline{x}$. □

Then, we describe the basic Boolean operations on basic formulas.

Proposition 3. *A conjunction of disjunctions of basic formulas for $\overline{x}$ can be transformed into an equivalent disjunction of basic formulas for $\overline{x}$.*

Proof. It suffices to show that a conjunction of two basic formulas for the variables $\overline{x}$ can be transformed into an equivalent disjunction of basic formulas for $\overline{x}$. The extension to the general case is trivial. Let us consider the following two constraints:

$$c_1(\overline{x}, \overline{w}^1) = \bigwedge_{x_1 \in \overline{x}^{11}} \neg x_1 == x_1 \wedge \overline{x}^{21} == \overline{t}^1 \wedge \bigwedge_{w_i^1 \in \overline{w}^1} \bigwedge_{j=1}^{n_i^1} (\neg w_i^1 == s_{ij}^1)^{\forall \setminus \overline{w}^1}$$

$$c_2(\overline{x}, \overline{w}^2) = \bigwedge_{x_1 \in \overline{x}^{12}} \neg x_1 == x_1 \wedge \overline{x}^{22} == \overline{t}^2 \wedge \bigwedge_{w_i^2 \in \overline{w}^2} \bigwedge_{j=1}^{n_i^2} (\neg w_i^2 == s_{ij}^2)^{\forall \setminus \overline{w}^2}$$

If either $\overline{x}^{11} \neq \overline{x}^{12}$ or $\overline{t}^1$ and $\overline{t}^2$ do not unify, then

$$\exists \overline{w}^1 \; c_1(\overline{x}, \overline{w}^1) \wedge \exists \overline{w}^2 \; c_2(\overline{x}, \overline{w}^2) \; \equiv \; false$$

(by rules $\mathbf{NFT_2}$ and $\mathbf{C_1}$ respectively). Otherwise, $\overline{x}^{11} = \overline{x}^{12}$ (thus, $\overline{x}^{21} = \overline{x}^{22}$), $\sigma = \mathtt{mgu}(\overline{t}^1, \overline{t}^2)$ and the conjunction $c_1(\overline{x}, \overline{w}^1) \wedge c_2(\overline{x}, \overline{w}^2)$ is equivalent (by rules $\mathbf{R}$, $\mathbf{D_1}$, $\mathbf{EE_2}$ and $\mathbf{FT/EE_1}$) to

$$\bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \wedge \overline{x}^2 == \overline{t} \wedge \bigwedge_{w_i^1 \in \overline{w}^1} \bigwedge_{j=1}^{n_i^1} (\neg w_i^1 \sigma == s_{ij}^1 \sigma)^{\forall \setminus \overline{w}} \wedge \bigwedge_{w_i^2 \in \overline{w}^2} \bigwedge_{j=1}^{n_i^2} (\neg w_i^2 \sigma == s_{ij}^2 \sigma)^{\forall \setminus \overline{w}}$$

where $\overline{x}^1 = \overline{x}^{11} = \overline{x}^{12}$, $\overline{x}^2 = \overline{x}^{21} = \overline{x}^{22}$, $\overline{t} = \mathtt{mgi}(\overline{t}^1, \overline{t}^2)$ and $\overline{w} = \mathtt{Var}(\overline{t})$. Then, using $\mathbf{D_2}$, $\mathbf{C_2}$ and $\mathbf{UE}$, each negated equation $(\neg w_i^k \sigma == s_{ij}^k \sigma)^{\forall \setminus \overline{w}}$ for $1 \leq j \leq n_i^k$, $w_i^k \in \overline{w}^k$ and $k \in \{1, 2\}$ is transformed into $true$, $false$ or an equivalent disjunction of the form

$$\begin{array}{l} (\; \neg \overline{w}' == \overline{w}' \;) \; \vee \\ \forall \overline{v} \; (\; \neg w_1 == s_1' \; \vee \; \neg w_2 == s_2' \; \vee \; \ldots \; \vee \; \neg w_m == s_m' \;) \end{array} \qquad (2)$$

where $\overline{w}' \subseteq \overline{w}$, $w_i \in \overline{w}$ for each $1 \leq i \leq m$. If $\overline{v} = \emptyset$, then (2) is transformed is transformed into a disjunction of basic formulas for $\overline{w}$ (as proposed in Proposition 2). Then, the whole formula is simplified by distribution. Note that a conjunction of $\overline{x}^2 == \overline{t}[w] \wedge \neg w == w$ is easily reduced to $false$ by rule $\mathbf{NFT_2}$. For the remaining cases, we use rules $\mathbf{R}$, $\mathbf{EE_2}$ and $\mathbf{FT/EE_1}$ on the variables $\overline{w}$. After using the rule $\mathbf{R}$, we may have to simplify the conjunction of negated equations as above. However, this process clearly ends since the depth of universal variables strictly decreases at each iteration. □

In the next example, we show the transformation of a conjunction of two basic formulas into an equivalent disjunction of basic formulas for the same variables.

Example 2. Let $\{a_{/0}, g_{/1}, f_{/2}\} \subset \mathcal{F}_\Sigma$ and $\overline{x} = \{x_1, x_2, x_3\} \subset \mathcal{V}$. The conjunction of basic formulas for $\overline{x}$ $\varphi_1 = \exists \overline{w}^1 \; c_1(\overline{x}, \overline{w}^1) \wedge \exists \overline{w}^2 \; c_2(\overline{x}, \overline{w}^2)$ where

$$\begin{array}{l} c_1(\overline{x}, \overline{w}^1) = \neg x_1 == x_1 \wedge x_2 == w_1^1 \wedge x_3 == g(w_2^1) \wedge \forall v \; \neg w_1^1 == f(a, v) \\ c_2(\overline{x}, \overline{w}^2) = \neg x_1 == x_1 \wedge \neg x_2 == x_2 \wedge x_3 == f(w_1^2, w_2^2) \wedge \neg w_1^2 == w_2^2 \end{array}$$

is unsatisfiable since $x_2 == w_1^1 \wedge \neg x_2 == x_2$ is reduced to $false$ by $\mathbf{NFT_2}$. On the contrary, the conjunction $\varphi_2 = \exists \overline{w}^1 \; c_1(\overline{x}, \overline{w}^1) \; \wedge \; \exists \overline{w}^3 \; c_3(\overline{x}, \overline{w}^3)$ where

$$c_3(\overline{x}, \overline{w}^3) \; = \; \neg x_1 == x_1 \wedge x_2 == f(w_1^3, w_2^3) \wedge x_3 == w_1^3 \wedge \forall v \; \neg w_2^3 == g(v)$$

is transformed in the following way. Since $\sigma = \mathtt{mgu}(w_1^1 \cdot g(w_2^1), f(w_1^3, w_2^3) \cdot w_1^3) = \{w_1^1 \leftarrow f(g(w_2^1), w_2^3),\ w_1^3 \leftarrow g(w_2^1)\}$, φ_2 is transformed into

$$\exists w_2^1 \cdot w_2^3\ (\ \neg x_1 == x_1 \wedge x_2 == f(g(w_2^1), w_2^3) \wedge x_3 == g(w_2^1) \wedge \\ \forall v\ \neg f(g(w_2^1), w_2^3) == f(a, v) \wedge \forall v\ \neg w_2^3 == g(v)\ ).$$

Then, the negated equation $\forall v\ \neg f(g(w_2^1), w_2^3) == f(a, v)$ is reduced to *true* using rules $\mathbf{D_2}$ and $\mathbf{C_2}$. Thus, the resulting basic formula is

$$\exists w_2^1 \cdot w_2^3\ (\neg x_1 == x_1 \wedge x_2 == f(g(w_2^1), w_2^3) \wedge x_3 == g(w_2^1) \wedge \forall v\ \neg w_2^3 == g(v)).\ \square$$

Proposition 4. *A negated disjunction of basic formulas for the variables $\overline{x}$ can be transformed into an equivalent disjunction of basic formulas for $\overline{x}$.*

Proof. It suffices to show that a negated basic formula for the variables $\overline{x}$ can be transformed into an equivalent disjunction basic formulas for $\overline{x}$. By Proposition 3, the extension to the general case is straightforward.

Let us consider the following negated basic formula for the variables $\overline{x}$:

$$\neg\exists\overline{w}\ (\ \bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1\ \wedge\ \overline{x}^2 == \overline{t}\ \wedge\ \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} \neg w_i == s_{ij}\ )$$

By distribution, the above formula is transformed into

$$\bigvee_{x_1 \in \overline{x}^1} x_1 == x_1\ \vee\ \neg\exists\overline{w}\ (\ \overline{x}^2 == \overline{t}\ \wedge\ \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} \neg w_i == s_{ij}\ )$$

where each $x_1 == x_1$ from the first subformula is transformed into a disjunction of basic formulas for $\overline{x}$ by applying the rule $\mathbf{T}$ on each variable from $\overline{x} \setminus x_1$. Using the rule $\mathbf{S}$, the second subformula is transformed into

$$\neg\exists\overline{w}\ (\ \overline{x}^2 == \overline{t}\ )\ \vee\ \exists\overline{w}\ (\ \overline{x}^2 == \overline{t}\ \wedge\ \neg \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} \neg w_i == s_{ij}\ ).$$

By Proposition 2, $\neg\exists\overline{w}\ (\ \overline{x}^2 == \overline{t}\ )$ is transformed into a disjunction of basic formulas for $\overline{x}^2$. Besides, after simplification and double negation elimination, the remaining subformula is transformed into a disjunction of basic formulas for $\overline{x}^2$ using rules $\mathbf{R}$, $\mathbf{EE_2}$, $\mathbf{FT}$ and $\mathbf{EE_1}$ on $w_i == s_{ij}$ for each $w_i \in \overline{w}$ and $1 \leq j \leq n_i$. Finally, basic formulas for $\overline{x}^2$ are transformed into basic formulas for $\overline{x}$ using rules $\mathbf{T}$ and $\mathbf{EI}$ on each variable from $\overline{x}^1$. $\square$

Example 3. Let $\{a_{/0}, f_{/2}\} \subset \mathcal{F}_\Sigma$ and $\overline{x} = \{x_1, x_2\} \subset \mathcal{V}$. The negated basic formula for the variables $\overline{x}$

$$\varphi\ =\ \neg\exists w\ (\ \neg x_1 == x_1 \wedge x_2 == f(w, a) \wedge \forall v\ \neg w == f(a, v)\ )$$

$$(\mathbf{EE_4})\ \ \exists \overline{w}\ (\ \overline{v} == \overline{v} \wedge \bigwedge_{i=1}^{n} (\neg s_i == t_i)^{\forall \setminus \overline{w}} \wedge \varphi\ )\ \ \mapsto\ \ \varphi$$

if $\overline{w} \cap \mathtt{Var}(\varphi) = \emptyset$, $\overline{v} \subseteq \overline{w}$, $s_i \neq t_i$, $\overline{w} \cap \mathtt{Var}(s_i, t_i) \neq \emptyset$ and either
s_i (resp. t_i) is not a variable or $s_i \in \overline{w}$ (resp. $t_i \in \overline{w}$) for each $1 \leq i \leq n$

Fig. 3. Existential Quantification Elimination: Infinite Signatures

is transformed as follows. First, φ is trivially equivalent to $(\ x_1 == x_1\ ) \vee \neg\exists w\ (\ x_2 == f(w,a) \wedge \forall v\ \neg w == f(a,v)\ )$, where $(\ x_1 == x_1\ )$ is transformed into the following basic formulas for $\overline{x}$

$$\exists w_1\ (\ \neg x_2 == x_2 \wedge x_1 == w_1\ ) \vee \exists w_2 \cdot w_3\ (\ x_1 == w_2 \wedge x_2 == w_3\ )$$

using **T**, **EI**, **R** and **FT**. Besides, the remaining subformula is transformed into $\neg\exists w\ (\ x_2 == f(w,a)\ ) \vee \exists w\ (\ x_2 == f(w,a) \wedge \neg\forall v\ \neg w == f(a,v)\ )$ using the rule **S**. The constraint $\neg\exists w\ (\ x_2 == f(w,a)\ )$ is transformed into

$$\begin{array}{l}(\ \neg x_1 == x_1 \wedge \neg x_2 == x_2\ ) \vee \exists w_4\ (\ \neg x_2 == x_2 \wedge x_1 == w_4\ ) \vee \\ \exists w_5\ (\ \neg x_1 == x_1 \wedge x_2 == w_5 \wedge \forall v\ \neg w_5 == f(v,a)\ ) \vee \\ \exists w_6 \cdot w_7\ (\ x_1 == w_6 \wedge x_2 == w_7 \wedge \forall v\ \neg w_7 == f(v,a)\ )\end{array}$$

using **T**, **EI**, **R** and $\mathbf{NFT_1}$. Finally, $\exists w\ (\ x_2 == f(w,a) \wedge \neg\forall v\ \neg w == f(a,v)\ ) \equiv \exists w\ (\ x_2 == f(w,a) \wedge \exists v\ w == f(a,v)\ )$ is transformed into

$$\begin{array}{l}\exists w_8\ (\ \neg x_1 == x_1 \wedge x_2 == f(f(a,w_8),a)\ ) \vee \\ \exists w_9 \cdot w_{10}\ (\ x_1 == w_9 \wedge x_2 == f(f(a,w_{10}),a)\ )\end{array}$$

using rules **R**, $\mathbf{EE_2}$ and **FT** on w, and **T** and **EI** on x_1. □

Next, we show that the elimination of the innermost block of quantifiers is correct in $\mathcal{E}_{inf}$ when it is existential. For this purpose, we introduce the transformation rule $\mathbf{EE_4}$ (see Figure 3), which allows to eliminate existential variables only occurring in a conjunction of (universally quantified) negated equations.

Proposition 5. *The transformation rule* $\mathbf{EE_4}$ *is correct in* $\mathcal{E}_{inf}$.

Proof. We have to prove that

$$\mathcal{E}_{inf} \models\ [\ \exists \overline{w}\ (\ \overline{v} == \overline{v} \wedge \bigwedge_{i=1}^{n} (\neg s_i == t_i)^{\forall \setminus \overline{w}} \wedge \varphi\ )\ \leftrightarrow\ \varphi\ ].$$

One implication is trivial. For the reverse one, it suffices to show the existence of a witness for $\overline{w}$. Since $\psi = \bigwedge_{i=1}^{n} (\neg s_i == t_i)^{\forall \setminus \overline{w}}$ is a finite conjunction, we can use as witness any ground term such that its root function does not occur in ψ. □

Given any constraint φ_0 with free variables $\overline{x}^0$:

(Step 1) Transform φ_0 into a prenex DNF formula $\varphi_1 = Q_1\overline{x}^1 \ldots Q_n\overline{x}^n \bigvee_{i=1}^{m} \psi_i$

(Step 2) For each $1 \leq i \leq m$, transform ψ_i into a disjunction of basic formulas for the variables $\overline{x} = \overline{x}^0 \cdot \overline{x}^1 \cdot \ldots \cdot \overline{x}^n$ as follows:

(a) Apply rules $\mathbf{B_1}$, $\mathbf{B_2}$, $\mathbf{NFT_1}$, $\mathbf{NFT_2}$, $\mathbf{NFT_3}$, $\mathbf{FT}$, $\mathbf{D_1}$, $\mathbf{D_2}$, $\mathbf{C_1}$, $\mathbf{C_2}$, $\mathbf{O_1}$ and $\mathbf{O_2}$. When none of the previous rules applies, it remains a disjunction of constraints of the form $\psi_i' = \bigwedge_{j=1}^{o_1} v_j == r_j \wedge \bigwedge_{j=o_1+1}^{o_2} \neg v_j == r_j$ where v_i is a variable, r_j is total and $v_j \notin \mathtt{Var}(r_j)$ for each $1 \leq j \leq o_2$.

(b) For each conjunct ψ_i' that results from (a) and each variable $x \in \overline{x}$:
- If $x = v_j$ for some $1 \leq j \leq o_1$, then apply $\mathbf{R}$ on x.
- If $x \neq v_k$ for every $1 \leq j \leq o_1$ and $x \in \mathtt{Var}(r_k)$ for some $1 \leq k \leq o_1$, then apply $\mathbf{EI}$ and $\mathbf{R}$ on x.
- Otherwise, apply $\mathbf{T}$ on x and goto (a).

The resulting formula $\varphi_2 = Q_1\overline{x}^1 \ldots Q_n\overline{x}^n \bigvee_{i=1}^{m'} \exists \overline{w}^i \; a_i(\overline{x}, \overline{w}^i)$ is in BNF

(Step 3) Iteratively eliminate the innermost block of consecutive existential/universal quantifiers $Q_n\overline{x}^n$ in φ_2:

(i) If $Q_n = \exists$, by Theorem 4 (Theorem 7 for finite signatures) the formula φ_2 is equivalent to $Q_1\overline{x}^1 \ldots Q_{n-1}\overline{x}^{n-1} \bigvee_{i=1}^{m'} \exists \overline{w}'^i \; a_i'(\overline{x}', \overline{w}'^i)$

(ii) If $Q_n = \forall$, then apply (i) using double negation as follows $Q_1\overline{x}^1 \ldots \neg\exists\overline{x}^n \neg \bigvee_{i=1}^{m'} \exists \overline{w}^i \; a_i(\overline{x}, \overline{w}^i)$. Negation on basic formulas is therefore used before and after applying (i).

Fig. 4. A Decision Method for Strict Equality

Finally, the elimination of the innermost block of existential quantifiers is used in the decision algorithm given in Figure 4.

Theorem 4. *Let $\exists\overline{w}\; a(\overline{x} \cdot \overline{y}, \overline{w} \cdot \overline{z})$ be a basic formula for $\overline{x} \cdot \overline{y}$ of the form*

$$\exists\overline{w} \cdot \overline{z} \; (\bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \wedge \bigwedge_{y_1 \in \overline{y}^1} \neg y_1 == y_1 \wedge \overline{x}^2 == \overline{t} \wedge \overline{y}^2 == \overline{r} \wedge \varphi \wedge \psi \;)$$

where $-$ $\overline{w} = \mathtt{Var}(\overline{t})$ *and* $\overline{z} = \mathtt{Var}(\overline{r}) \setminus \overline{w}$,
$-$ φ *is a finite conjunction of negated equations such that* $\mathtt{Free}(\varphi) \subseteq \overline{w}$,
$-$ $\psi = \bigwedge_{i=1}^{n} (\neg v_i == s_i)^{\forall \setminus \overline{w} \cdot \overline{z}}$ *and* $(v_i \cup \mathtt{Var}(s_i)) \cap \overline{z} \neq \emptyset$ *for* $1 \leq i \leq n$.

The formulas $\exists\overline{y} \; [\; \exists\overline{w} \; a(\overline{x} \cdot \overline{y}, \overline{w} \cdot \overline{z}) \;]$ *and* $\exists\overline{w} \; (\bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \wedge \overline{x}^2 == \overline{t} \wedge \varphi \;)$ *are equivalent in* $\mathcal{E}_{inf}$.

Proof. It follows from rules $\mathbf{EE_1}$, $\mathbf{EE_2}$, $\mathbf{EE_3}$ and $\mathbf{EE_4}$. □

Example 4. Let $\{a_{/0}, g_{/1}, f_{/2}\} \subset \mathcal{F}_\Sigma$. The formulas

$$\exists y \; [\; \exists w_1 \cdot w_2 \; (\; x == g(w_1) \wedge y == f(w_2, a) \wedge \neg w_1 == a \wedge \neg w_1 == w_2 \wedge \forall v \; \neg w_2 == f(a, v) \;) \;]$$

and $\exists w_1 \; (\; x == g(w_1) \wedge \neg w_1 == a \;)$ are equivalent in $\mathcal{E}_{inf}$. □

The algorithm described in Figure 4 is illustrated in the next example. Roughly speaking, we first transform the input constraint φ into an equivalent formula in basic normal form. Then, we proceed to iteratively eliminate the innermost block of quantifiers $Q_i\overline{x}^i$. By Theorem 4, the elimination of $Q_i\overline{x}^i$ is trivial when Q_i is existential. However, when Q_i is universal, we have to use double negation to turn Q_i into existential. This process requires to negate the matrix of the formula and to transform it into an equivalent disjunction of basic formulas for the same variables before and after the elimination of Q_i. In both cases, the length of the block of consecutive quantifiers strictly decreases at each elimination step because no new variable is introduced in the prefix. Hence, since the length of the prefix is finite, the algorithm always terminates and transforms φ into an equivalent disjunction of basic formulas for its free variables.

Example 5. Let $\{a_{/0}, g_{/1}, f_{/2}\} \subset \mathcal{F}_\Sigma$ and $\overline{x} = \{x_1, x_2\} \subset \mathcal{V}$. The constraint

$$\forall\overline{x}\,[(\ f(x_1,a) == f(g(x_2),x_2) \wedge \neg g(x_2) == g(g(x_1))\ )\ \vee\ f(x_1,x_2) == f(x_2,x_1)]$$

is already in prenex disjunctive normal form, thus **Step 1** is not applicable. In **Step 2**, the formula is first transformed into

$$\forall\overline{x}\,[\ (\ x_1 == g(x_2) \wedge x_2 == a \wedge \neg x_2 == g(x_1)\ )\ \vee\ x_1 == x_2\ ]$$

using $\mathbf{D_1}$ and $\mathbf{D_2}$. Next, the formula is transformed into basic normal form

$$\forall\overline{x}\,[\ (\ x_1 == g(a) \wedge x_2 == a\ )\ \vee\ \exists w\ (\ x_1 == w \wedge x_2 == w\ )\ ]$$

using $\mathbf{R}$, $\mathbf{C_2}$ and $\mathbf{EI}$. Next, in **Step 3**, we proceed to eliminate $\forall\overline{x}$ (case (ii)). Using double negation, we obtain

$$\neg\exists\overline{x}\,[\ \neg(\ x_1 == g(a) \wedge x_2 == a\ )\ \wedge\ \neg\exists w\ (\ x_1 == w \wedge x_2 == w\ )\ ]$$

that is transformed into

$$\begin{aligned}\neg\exists\overline{x}\,[\ &(\ \neg x_1 == x_1 \wedge \neg x_2 == x_2\ )\ \vee\ \exists w\ (\ \neg x_1 == x_1 \wedge x_2 == w\ )\ \vee\\ &\exists w\ (\ \neg x_2 == x_2 \wedge x_1 == w \wedge \neg w == g(a)\ )\ \vee\\ &\exists\overline{w}\ (\ x_1 == w_1 \wedge x_2 == w_2 \wedge \neg w_1 == g(a) \wedge \neg w_1 == w_2\ )\ \vee\\ &(\ \neg x_2 == x_2 \wedge x_1 == g(a)\ )\ \vee\\ &\exists w\ (\ x_1 == g(a) \wedge x_2 == w \wedge \neg w == a \wedge \neg w == g(a)\ )\ ]\end{aligned}$$

by negation and conjunction of basic formulas. Then, $\exists\overline{x}$ can be eliminated and we obtain $\neg[true]$, which is trivially equivalent to $false$. □

As an easy but important consequence of having a decision method, we obtain the completeness of our axiomatization $\mathcal{E}_{inf}$.

Theorem 5 (Completeness of $\mathcal{E}_{inf}$). *$\mathcal{E}_{inf}$ is a complete theory.*

Existential Quantification Elimination

$$(\mathbf{EE_5})\ \ \exists \overline{w}\ (\ \overline{v} == \overline{v} \wedge \bigwedge_{i=1}^{n} \neg s_i == t_i \wedge \varphi\ )\ \ \mapsto\ \ \varphi$$

if $\overline{w} \cap \mathtt{Var}(\varphi) = \emptyset$, $\overline{v} \subseteq \overline{w}$, $s_i \neq t_i$ and $\overline{w} \cap \mathtt{Var}(s_i, t_i) = \emptyset$ for $1 \leq i \leq n$

Explosion

$$(\mathbf{E})\ \varphi[x]\ \mapsto\ \varphi[x] \wedge [\ \neg x == x \vee \bigvee_{f \in \mathcal{F}_\Sigma} \exists \overline{w}\ x == f(\overline{w})\ ]$$

Fig. 5. Transformation Rules: Finite Signatures

Proof. Consider any closed formula ϕ. By using the decision method of Fig. 4 we obtain a formula ψ in BNF such that $\mathcal{E}_{inf} \models \phi \leftrightarrow \psi$. Now, ψ must be also closed (since the transformation rules do not introduce new free variables in a formula), and therefore ψ is a disjunction made of the atoms $true$ or $false$ (which are the only closed basic formulas). Hence, ψ is equivalent to $true$ or $false$. In the first case, we have $\mathcal{E}_{inf} \models \phi \leftrightarrow true$, which implies $\mathcal{E}_{inf} \models \phi$. In the second case, we have $\mathcal{E}_{inf} \models \phi \leftrightarrow false$, and then $\mathcal{E}_{inf} \models \neg\phi$. Therefore, $\mathcal{E}_{inf}$ is complete. □

4.2 Finite Signatures

In the case of finite signatures, the normal form provided in Definition 2 is not solved. This arises from the fact that a finite conjunction of universally quantified negated equations on a variable w may be unsatisfiable if only finite and total trees can be assigned to w. For example, being $\mathcal{F}_\Sigma = \{a_{/0}, g_{/1}\}$, the constraint

$$\exists w\ (\ x == w \wedge \neg w == a \wedge \forall v\ \neg w == g(v)\ )$$

is unsatisfiable although $\exists w\ (\ \neg w == a \wedge \forall v\ \neg w == g(v)\ )$ is satisfiable. Roughly speaking, the question is that all the function symbols of the signature can be used in a constraint.

Next, we show that $\mathcal{E}_{fin}$ is a decidable theory. For this purpose, we adapt all the definitions and results in Subsection 4.1 to the case of finite signatures. Besides, we add two new transformation rules **E** and $\mathbf{EE_5}$ (see Figure 5). Rule **E**, whose correctness directly follows from Axiom $\mathbf{A_7}$, allows for the elimination of universal quantification whereas $\mathbf{EE_5}$, which is the adaptation of $\mathbf{EE_4}$ to the case of finite signatures, makes possible to eliminate the innermost block of existential quantifiers. Next, we show that both rules are correct in $\mathcal{E}_{fin}$.

Proposition 6. *The transformation rules* $\mathbf{EE_5}$ *and* **E** *are correct in* $\mathcal{E}_{fin}$.

Proof. The correctness of **E** is straightforward due to $\mathbf{A_7}$. Regarding to $\mathbf{EE_5}$, we have to prove that

$$\mathcal{E}_{fin} \models\ [\ \exists \overline{w}\ (\ \overline{v} == \overline{v} \wedge \bigwedge_{i=1}^{n} \neg s_i == t_i \wedge \varphi\ )\ \leftrightarrow\ \varphi\ ].$$

One implication is trivial. For the reverse one, it suffices to show the existence of a witness for $\overline{w}$. Assuming that m is the maximum size of any term occurring in $\bigwedge_{i=1}^{n} \neg s_i == t_i$, we can use as witness a different ground term of size $m+1$ for each variable in $\overline{w}$. □

The use of **E** for eliminating universal quantification is necessary because our notion of solved form for finite signatures is free of universal variables.

Definition 3. *A* basic formula *for the variables $\overline{x}$ is either true, false* (closed basic formulas) *or a constraint $\exists \overline{w}\ c(\overline{x}, \overline{w})$ such that*

$$c(\overline{x}, \overline{w}) \ = \bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \ \wedge\ \overline{x}^2 == \overline{t} \ \wedge \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} \neg w_i == s_{ij}$$

where – $\overline{x} = \overline{x}^1 \cup \overline{x}^2$ *and* $\overline{x}^1 \cap \overline{x}^2 = \emptyset$,
– $\overline{w} = \mathtt{Var}(\overline{t})$ *and* $\overline{x} \cap \overline{w} = \emptyset$,
– *if s_{ij} is a variable, then $s_{ij} \neq w_i$, otherwise s_{ij} is total,* $\mathtt{Var}(s_{ij}) \subseteq \overline{w}$
– *and $w_i \notin \mathtt{Var}(s_{ij})$ for every $w_i \in \overline{w}$ and $1 \leq j \leq n_i$.*
A formula is in basic normal form *(or BNF) if it is of the form $Q\overline{y}\ \varphi[\overline{x} \cdot \overline{y}]$ where φ is a disjunction of basic formulas for $\overline{x} \cdot \overline{y}$.* □

The notion of basic formula for finite signatures is also a solved form.

Theorem 6. *Any basic formula different from $false$ is satisfiable in $\mathcal{E}_{fin}$.*

Proof. Obviously, the constraint *true* is satisfiable. Thus, let us consider a basic formula for the variables $\overline{x}$ of the form

$$\exists \overline{w}\ (\ \bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \ \wedge\ \overline{x}^2 == \overline{t} \ \wedge \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} \neg w_i == s_{ij}\ ).$$

As in Theorem 3, $\bigwedge_{x_1 \in \overline{x}^1} \neg x_1 == x_1 \wedge \overline{x}^2 == \overline{t}$ is trivially satisfiable. Regarding $\varphi = \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} \neg w_i == s_{ij}$, each negated equation eliminates either one possible assignment value to w_i if s_{ij} is ground or an assignment value to $\overline{w}' = \mathtt{Var}(s_{ij})$ for each assignment value to w_i if s_{ij} is not ground. The number of assignment values to each variable is infinite, thus φ is necessarily satisfiable. □

Note that the syntactical form provided in Def. 3 is a particular case of the one in Def. 2. The only difference is that universal quantification is not allowed in the case of finite signatures. Further, there exists a very simple transformation using **E** from formulas as defined in Def. 2 into formulas as defined above.

Proposition 7. *Any constraint of the form $\varphi = \bigwedge_{w_i \in \overline{w}} \bigwedge_{j=1}^{n_i} (\neg w_i == s_{ij})^{\forall \setminus \overline{w}}$ can be transformed into an equivalent disjunction of basic formulas for $\overline{w}$.*

Proof. If φ is free of universal quantification, then it is transformed into an equivalent disjunction of basic formulas for the variables $\overline{w}$ as shown in **Step 2** (Figure 4). Otherwise, there exists some $(\neg w_i == s_{ij})^{\forall \setminus \overline{w}}$ such that $\mathtt{Var}(s_{ij}) \not\subseteq$

$\overline{w}$. In order to eliminate universal quantification, we apply **E** on w_i and, after distributing the new introduced disjunction, rules $\mathbf{NFT_1}$, **R**, $\mathbf{D_2}$ and $\mathbf{C_2}$. In the resulting formula, the depth of the universal variables that come negated equations of the form $(\neg w_i == s_{ij})^{\forall \setminus \overline{w}}$ is strictly smaller than the depth of the universal variables in the original formula. Besides, the negated equations of the form $\forall v\ \neg w_i == v$ (that is, when universal variables occur at depth 0), can be replaced with $\neg w_i == w_i$ by rule $\mathbf{NFT_3}$. Then, this last negated equation is removed using either $\mathbf{NFT_2}$ or $\mathbf{EE_3}$. Thus, we can eliminate all the universal variables by repeating the above described process. □

Being $\exists \overline{w}\ c(\overline{x}, \overline{w})$ a formula as described in Definition 2, the conjunction of negated equations in $c(\overline{x}, \overline{w})$ is transformed into a disjunction of basic formulas $\overline{w}$ as shown in Proposition 7. Then, the whole formula is transformed into an equivalent disjunction of basic formulas for $\overline{x}$ using **R**, $\mathbf{EE_2}$ and **FT**. This result allows us to easily adapt Propositions 3 and 4 to the case of finite signatures.

Example 6. Let $\mathcal{F}_\Sigma = \{a_{/0}, g_{/1}, f_{/2}\}$ and $\overline{x} = \{x_1, x_2\} \subset \mathcal{V}$. The constraint

$$\exists w\ (\ \neg x_1 == x_1 \wedge x_2 == f(w, a) \wedge \forall v\ \neg w == f(a, v)\ )$$

is transformed into a disjunction of basic formulas for $\overline{x}$ as follows. First, we transform $\forall v\ \neg w == f(a, v)$ into a disjunction of basic formulas for w using **E**:

$$\begin{aligned} &\forall v\ \neg w == f(a,v)\ \wedge\ [\ \neg w == w\ \vee\ w == a\ \vee\ \exists z\ w == g(z)\ \vee \\ &\qquad\qquad \exists \overline{z}\ w == f(z_1, z_2)\ ] \\ \equiv\ &\neg w == w\ \vee\ w == a\ \vee\ \exists z\ w == g(z)\ \vee \\ &\exists \overline{z}\ (\ w == f(z_1, z_2) \wedge \forall v\ \neg w == f(a, v)\ ) \end{aligned} \tag{3}$$

The first three subformulas are already basic formulas for w. Regarding the last one, it is transformed using rules **R** and $\mathbf{D_2}$ as follows

$$\begin{aligned} &\exists \overline{z}\ (\ w == f(z_1, z_2) \wedge [\ \neg z_1 == a\ \vee\ \forall v\ \neg z_2 == v\ ]\ ) \\ \equiv\ &\exists \overline{z}\ (w == f(z_1, z_2) \wedge \neg z_1 == a) \vee \exists \overline{z}\ (w == f(z_1, z_2) \wedge \forall v\ \neg z_2 == v) \end{aligned} \tag{4}$$

where the second subformula is equivalent to $false$ by rules $\mathbf{NFT_3}$ and $\mathbf{NFT_2}$. Thus, $\forall v\ \neg w == f(a, v)$ has been transformed into the disjunction of basic formulas for w in (3, 4). Finally, the conjunction of the above disjunction and $\neg x_1 == x_1 \wedge x_2 == f(w, a)$ is transformed into

$$\begin{aligned} &(\ \neg x_1 == x_1 \wedge x_2 == f(a, a)\ )\ \vee\ \exists z\ (\ \neg x_1 == x_1 \wedge x_2 == f(g(z), a)\ )\ \vee \\ &\exists \overline{z}\ (\ \neg x_1 == x_1 \wedge x_2 == f(f(z_1, z_2), a) \wedge \neg z_1 == a\ ) \end{aligned}$$

using rules **R**, $\mathbf{EE_1}$ and $\mathbf{NFT_2}$. □

Next, in order to be able to apply the algorithm in Figure 4 to the case of finite signatures, we adapt the result in Theorem 4.

Theorem 7. *Let* $\exists\overline{w}\ a(\overline{x}\cdot\overline{y},\overline{w}\cdot\overline{z})$ *be a basic formula for* $\overline{x}\cdot\overline{y}$ *of the form*

$$\exists\overline{w}\cdot\overline{z}\ (\bigwedge_{x_1\in\overline{x}^1}\neg x_1 == x_1 \wedge \bigwedge_{y_1\in\overline{y}^1}\neg y_1 == y_1 \wedge \overline{x}^2 == \overline{t} \wedge \overline{y}^2 == \overline{r} \wedge \varphi \wedge \psi\ )$$

where $-\ \overline{w} = \mathtt{Var}(\overline{t})$ *and* $\overline{z} = \mathtt{Var}(\overline{r}) \setminus \overline{w}$,
$-\ \varphi$ *is a finite conjunction of negated equations such that* $\mathtt{Var}(\varphi) \subseteq \overline{w}$,
$-\ \psi = \bigwedge_{i=1}^{n} \neg v_i == s_i$ *and* $(v_i \cup \mathtt{Var}(s_i)) \cap \overline{z} \neq \emptyset$ *for each* $1 \leq i \leq n$.
The formulas $\exists\overline{y}\ [\ \exists\overline{w}\ a(\overline{x}\cdot\overline{y},\overline{w}\cdot\overline{z})\ ]$ *and* $\exists\overline{w}\ (\ \bigwedge_{x_1\in\overline{x}^1}\neg x_1 == x_1 \wedge \overline{x}^2 == \overline{t} \wedge \varphi\ )$ *are equivalent in* $\mathcal{E}_{fin}$.

Proof. It follows from rules $\mathbf{EE_1}$, $\mathbf{EE_2}$, $\mathbf{EE_3}$ and $\mathbf{EE_5}$. □

Finally, and similarly to the case of infinite signatures (Th. 5), we obtain:

Theorem 8 (Completeness of $\mathcal{E}_{fin}$). $\mathcal{E}_{fin}$ *is a complete theory.* □

5 Conclusions and Future Work

We have given an axiomatization $\mathcal{E}$ of the theory of strict equality over $\mathcal{IT}$, the algebra of possibly infinite and partial trees, both for the cases of infinite and finite signatures. The notion of strict equality over that kind of trees is of particular interest for functional and functional-logic programming. Besides, we have provided a decision algorithm —which proves that the axiomatization is complete— based on the use of solved forms and quantifier elimination. Further, it is easy to see that the problem of deciding first-order equality constraints of finite trees can be reduced to the decision problem of the theory of infinite trees with strict equality: it suffices to restrict the value of every variable x in any formula to be a finite and total tree by assertions of the form $x == x$. Thus, it follows from the results in [5, 15] that the decision problem of the theory of infinite trees with strict equality is non-elementary (as lower bound).

In this paper, we have focused on the algebra $\mathcal{IT}$ of possibly infinite and partial trees. However, as a side product of our results, we can derive interesting consequences also for the algebra $\mathcal{FT}$ of finite and possibly partial trees. In particular, it is easy to see that $\mathcal{FT} \models \mathcal{E}$. Since $\mathcal{E}$ is a complete theory, it follows that $\mathcal{E}$ is also a complete axiomatization of $==$ over $\mathcal{FT}$ and, therefore, we conclude that $\mathcal{IT}$ and $\mathcal{FT}$ are elementarily equivalent (when the language of $==$ is considered). We remark that this does not happen for infinite and finite trees when true equality ($\approx$) is considered.

Although direct applications of our results have been left out of the focus of the paper, we foresee some potential uses that will be subject of future work: Herbrand constraint solvers present in existing functional-logic languages, essentially corresponding to existential constraints, could be enhanced to deal with more general formulas. Constructive failure [10, 8], the natural counterpart of constructive negation in the functional logic field, could also take profit of our methods, specially for the case of programs with extra variables, not considered

in the mentioned papers. For these envisaged continuations of our work it could be convenient to extend the theory and methods of this paper by adding two additional predicate symbols: strict disequality (a computable approximation of negation of strict equality) and true equality.

References

1. P. Arenas-Sánchez, A. Gil-Luezas, and F. J. López-Fraguas. Combining lazy narrowing with disequality constraints. In M. V. Hermenegildo and J. Penjam, editors, *PLILP'94*, volume 884 of *LNCS*, pages 385–399. Springer-Verlag, 1994.
2. E. J. G. Arias, J. Mariño-Carballo, and J. M. R. Poza. A proposal for disequality constraints in curry. *Electr. Notes Theor. Comput. Sci.*, 177:269–285, 2007.
3. A. Colmerauer. Equations and inequations on finite and infinite trees. In K. L. Clark and S. A. Tärnlund, editors, *FGCS'84*, pages 85–99, 1984.
4. H. Comon and P. Lescanne. Equational problems and disunification. *Journal of Symbolic Computation*, 7(3/4):371–425, 1989.
5. K. J. Compton and C. W. Henson. A uniform method for proving lower bounds on the computational complexity of logical theories. *Annals of Pure and Applied Logic*, 48(1):1–79, 1990.
6. K. Djelloul, T.-B.-H. Dao, and T. W. Frühwirth. Theory of finite or infinite trees revisited. *TPLP*, 8(4):431–489, 2008.
7. M. Hanus (ed.). Curry: An integrated functional logic language. Available at `http://www.informatik.uni-kiel.de/~curry/report.html`, March 2006.
8. F. J. López-Fraguas and J. Sánchez-Hernández. Failure and equality in functional logic programming. *Electr. Notes Theor. Comput. Sci.*, 86(3), 2003.
9. F. López-Fraguas and J. Sánchez-Hernández. $\mathcal{TOY}$: A multiparadigm declarative system. In *Proc. Rewriting Techniques and Applications (RTA'99)*, pages 244–247. Springer LNCS 1631, 1999.
10. F. López-Fraguas and J. Sánchez-Hernández. A proof theoretic approach to failure in functional logic programming. *TPLP*, 4(1&2):41–74, 2004.
11. M. Maher. Complete axiomatizations of the algebras of finite, rational and infinite trees. Technical report, IBM T.J. Watson Research Center, 1988. Available at `http://www.cse.unsw.edu.au/~mmaher/pubs/trees/axiomatizations.pdf`.
12. M. J. Maher. Complete axiomatizations of the algebras of finite, rational and infinite trees. In *LICS'1988*, pages 348–357. IEEE Computer Society, 1988.
13. S. e. Peyton Jones. *Haskell 98 Language and Libraries. The Revised Report.* Cambridge Univ. Press, 2003.
14. R. Reiter. On closed world data bases. *Logic and Data Bases*, pages 55–76, 1978.
15. S. G. Vorobyov. An improved lower bound for the elementary theories of trees. In M. A. McRobbie and J. K. Slaney, editors, *CADE-13*, volume 1104 of *LNCS*, pages 275–287. Springer-Verlag, 1996.

Appendix

In order to prove Theorem 2, we first prove the following auxiliary results.

Proposition 8. Replacement (**R**) *is correct in* $\mathcal{E}$.

Proof. We have to prove that

$$\mathcal{E} \models [\, (\, x == t \wedge \varphi[x] \,) \leftrightarrow (\, x == t \wedge \varphi[x \leftarrow t] \,) \,]^{\forall}.$$

The proof is made by structural induction on φ.

The base case, that is $\varphi = r == s[x]$, is proved by induction on the size n of $s[x]$. In the base case $n = 0$, we have that $s[x] = x$ and

$$\mathcal{E} \models [\, (\, x == t \wedge r == x \,) \leftrightarrow (\, x == t \wedge r == t \,) \,]^{\forall}$$

directly follows from transitivity. Assuming as induction hypothesis that

$$\mathcal{E} \models [\, (\, x == t \wedge r == s[x] \,) \leftrightarrow (\, x == t \wedge r == s[x \leftarrow t] \,) \,]^{\forall} \tag{5}$$

holds for any term $s[x]$ of size equal to or smaller than n, we have to prove that (5) holds for any term $s[x]$ of size $n + 1$. By Proposition 1, we know that $x == t \wedge r == s[x] \equiv x == t \wedge r == s[x] \wedge s[x] == s[x]$. Besides, since $s[x] = f(\overline{u})$ for some function $f \in \mathcal{F}_{\Sigma}$ and some tuple of terms $\overline{u}$, we have that $x == t \wedge r == s[x] \wedge s[x] == s[x] \equiv x == t \wedge r == s[x] \wedge \overline{u} == \overline{u}$ by Axiom $\mathbf{A_1}$, where the size of each $u \in \overline{u}$ is equal to or smaller than n. By the induction hypothesis, $x == t \wedge r == s[x] \wedge \overline{u} == \overline{u} \equiv x == t \wedge r == s[x] \wedge \overline{u} == \overline{u}[x \leftarrow t]$. Thus, also by Axiom $\mathbf{A_1}$, we have that $x == t \wedge r == s[x] \wedge \overline{u} == \overline{u}[x \leftarrow t] \equiv x == t \wedge r == s[x] \wedge s[x] == s[x \leftarrow t]$. Finally, by transitivity, we know that $x == t \wedge r == s[x] \wedge s[x] == s[x \leftarrow t] \equiv x == t \wedge r == s[x \leftarrow t]$ and, hence, (5) holds for any term $s[x]$ of size $n + 1$.

For the inductive case, we just prove the case for connectives $\neg, \wedge$ and quantifier $\exists$, since the proof for connectives $\vee, \rightarrow, \leftrightarrow$ and quantifier $\forall$ directly follows from it.

If $\varphi = \neg\psi$, then we have to prove that

$$\mathcal{E} \models [\, (\, x == t \wedge \neg\psi[x] \,) \leftrightarrow (\, x == t \wedge \neg\psi[x \leftarrow t] \,) \,]^{\forall}$$

assuming that $\mathcal{E} \models [\, (\, x == t \wedge \psi[x] \,) \leftrightarrow (\, x == t \wedge \psi[x \leftarrow t] \,) \,]^{\forall}$. For any solution σ of $x == t$, if σ is a solution of $\neg\psi[x]$, then it cannot be a solution of $\psi[x]$. Hence, by induction hypothesis, σ is not a solution of $\psi[x \leftarrow t]$ and, therefore, σ is a solution of $\neg\psi[x \leftarrow t]$.

If $\varphi = \psi_1 \wedge \psi_2$, then we have to prove that

$$\mathcal{E} \models [\, (\, x == t \wedge \psi_1[x] \wedge \psi_2[x] \,) \leftrightarrow (\, x == t \wedge \psi_1[x \leftarrow t] \wedge \psi_2[x \leftarrow t] \,) \,]^{\forall}$$

assuming that $\mathcal{E} \models [\, (\, x == t \wedge \psi_1[x] \,) \leftrightarrow (\, x == t \wedge \psi_1[x \leftarrow t] \,) \,]^{\forall}$ and $\mathcal{E} \models [\, (\, x == t \wedge \psi_2[x] \,) \leftrightarrow (\, x == t \wedge \psi_2[x \leftarrow t] \,) \,]^{\forall}$. For any solution σ of $x == t \wedge \psi_1[x] \wedge \psi_2[x]$, σ is also solution of $x == t \wedge \psi_1[x]$ and $x == t \wedge \psi_2[x]$.

Hence, by induction hypothesis, σ is solution of both $x == t \wedge \psi_1[x \leftarrow t]$ and $x == t \wedge \psi_2[x \leftarrow t]$ and, therefore, the substitution σ is also a solution of $x == t \wedge \psi_1[x \leftarrow t] \wedge \psi_2[x \leftarrow t]$.

Finally, if $\varphi = \exists w\ \psi$, then we have to prove that

$$\mathcal{E} \models\ [\ (\ x == t \wedge \exists w\ \psi[x]\ )\ \leftrightarrow\ (\ x == t \wedge \exists w\ \psi[x \leftarrow t]\ )\ ]^{\forall}$$

assuming that $\mathcal{E} \models\ [\ (\ x == t \wedge \psi[x]\ )\ \leftrightarrow\ (\ x == t \wedge \psi[x \leftarrow t]\ )\ ]^{\forall}$. In this case, the proof directly follows from the induction hypothesis since any solution of $x == t \wedge \psi[x]$ (resp. $x == t \wedge \psi[x \leftarrow t]$) is also solution of $x == t \wedge \exists w\ \psi[x]$ (resp. $x == t \wedge \exists w\ \psi[x \leftarrow t]$). □

Proposition 9. *The transformation rules* Bottom (**B**$_1$, **B**$_2$), Non-finite trees (**NFT**$_1$, **NFT**$_2$, **NFT**$_3$), Finite trees (**FT**), Decomposition (**D**$_1$, **D**$_2$), Clash (**C**$_1$, **C**$_2$), Occur-check (**O**$_1$, **O**$_2$) *and* Tautology (**T**) *are correct in* $\mathcal{E}$.

Proof. **T** is trivially correct. The correctness of **D**$_1$ and **D**$_2$ is straightforward due to **A**$_1$. **C**$_1$ and **C**$_2$ are correct due to **A**$_2$. The correctness of **O**$_1$ and **O**$_2$ is straightforward due to **A**$_3$. **B**$_1$ and **B**$_2$ are correct due to **A**$_4$. **NFT**$_1$, **NFT**$_2$, **NFT**$_3$ and **FT** are proved to be correct using symmetry, transitivity, **D**$_1$ and Proposition 1. □

Proposition 10. Existential elimination (**EE**$_1$, **EE**$_2$, **EE**$_3$), Existential introduction (**EI**) *and* Universal elimination (**UE**) *are correct in* $\mathcal{E}$.

Proof. Regarding **EE**$_1$, we have to prove that

$$\mathcal{E} \models\ [\ \exists w\ (\ w == w \wedge \varphi\ )\ \leftrightarrow\ \varphi\ ]^{\forall}.$$

One implication is trivial. For the reverse one, it suffices to show the existence of a witness. We know that there exists at least one constant function symbol $a_{/0} \in \mathcal{F}_{\Sigma}$. Thus, the term a is a witness. The proof for **EE**$_3$ is almost identical. In this case, we can use $\bot$ as witness.

With respect to **EE**$_2$, we have to prove that

$$\mathcal{E} \models\ [\ \exists w\ (\ w == t \wedge \varphi\ )\ \leftrightarrow\ (\ \overline{x} == \overline{x} \wedge \varphi\ )\ ]^{\forall}.$$

The first implication is proved by Proposition 1 and **D**$_1$. The reverse implication follows from the existence of a witness (the term t itself).

The correctness of **EI** is straightforward due to rules **R**, **EE**$_2$ and **EE**$_1$.

UE is correct since it results from rules **R** and **EE**$_2$ via negation. □

Proposition 11. *The transformation rule* Split (**S**) *is correct in* $\mathcal{E}$.

Proof. We have to prove that

$$\mathcal{E} \models\ [\ \neg\exists\overline{w}\exists\overline{z}\ (\ x == t[\overline{w}] \wedge \varphi[\overline{w} \cdot \overline{z}]\ )\ \leftrightarrow\ \neg\exists\overline{w}\ (\ x == t[\overline{w}]\ ) \vee \exists\overline{w}(x == t[w] \wedge \neg\exists\overline{z}\ \varphi[\overline{w} \cdot \overline{z}]\ )\ ]^{\forall}. \quad (6)$$

For this purpose, we start from the formula

$$\neg\exists\overline{w}\exists\overline{z}\ (\ x == t[\overline{w}] \wedge \varphi[\overline{w}\cdot\overline{z}]\ )\ \wedge\ (\ \neg\exists\overline{v}\ x == t[\overline{v}] \vee \exists\overline{v}\ x == t[\overline{v}]\ )$$

which is trivially equivalent to $\neg\exists\overline{w}\exists\overline{z}\ (\ x == t[\overline{w}] \wedge \varphi[\overline{w}\cdot\overline{z}]\ )$. By distribution, the above formula is transformed into a disjunction of two conjuncts where the first one

$$\neg\exists\overline{w}\exists\overline{z}\ (\ x == t[\overline{w}] \wedge \varphi[\overline{w}\cdot\overline{z}]\ )\ \wedge\ \neg\exists\overline{v}\ x == t[\overline{v}]$$

is equivalent to $\neg\exists\overline{w}\ x == t[\overline{w}]$ (that is, the first conjunct in the right-hand subformula of (6)). From the second conjunct

$$\neg\exists\overline{w}\exists\overline{z}\ (\ x == t[\overline{w}] \wedge \varphi[\overline{w}\cdot\overline{z}]\ )\ \wedge\ \exists\overline{v}\ x == t[\overline{v}]$$

and using the transformation rule **R**, we obtain the formula

$$\exists\overline{v}\ (\ \neg\exists\overline{w}\exists\overline{z}\ (\ t[\overline{v}] == t[\overline{w}] \wedge \varphi[\overline{w}\cdot\overline{z}]\ )\ \wedge\ x == t[\overline{v}]\ ).$$

By simplification, this formula is transformed into

$$\exists\overline{v}\ (x == t[\overline{v}] \wedge \forall\overline{w}\forall\overline{z}\ (\ \neg\overline{v} == \overline{w} \vee \neg\varphi[\overline{w}\cdot\overline{z}]\ )\ )$$

which is equivalent to

$$\exists\overline{v}\ (x == t[\overline{v}] \wedge (\ \neg\overline{v} == \overline{v} \vee \forall\overline{z}\ \neg\varphi[\overline{v}\cdot\overline{z}]\ )\ )$$

using the rule **UE**. By distribution, we obtain a disjunction of two formulas

$$\exists\overline{v}\ (x == t[\overline{v}] \wedge \neg\overline{v} == \overline{v}\ )\ \vee\ \exists\overline{v}\ (x == t[\overline{v}] \wedge \forall\overline{z}\ \neg\varphi[\overline{v}\cdot\overline{z}]\ )$$

where $\exists\overline{v}\ (x == t[\overline{v}] \wedge \neg\overline{v} == \overline{v}\ )$ is equivalent to $false$ by the rule $\mathbf{NFT_2}$ and $\exists\overline{v}\ (x == t[\overline{v}] \wedge \forall\overline{z}\ \neg\varphi[\overline{v}\cdot\overline{z}]\ )$ is equivalent to the second conjunct in the right-hand subformula of (6). Thus, formula (6) holds. □

Theorem 2. *The transformation rules in Figure 2 are correct in $\mathcal{E}$.*

Proof. It directly follows from Propositions 8, 9, 10 and 11. □