Finite-Tree Analysis for Constraint Logic-Based Languages*

Roberto Bagnara[1], Roberta Gori[2], Patricia M. Hill[3], and Enea Zaffanella[1]

[1] Department of Mathematics, University of Parma, Italy.
{bagnara,zaffanella}@cs.unipr.it
[2] Department of Computer Science, University of Pisa, Italy.
gori@di.unipi.it
[3] School of Computing, University of Leeds, U. K.
hill@comp.leeds.ac.uk

Abstract. Logic languages based on the theory of rational, possibly infinite, trees have much appeal in that rational trees allow for faster unification (due to the omission of the occurs-check) and increased expressivity. Note that cyclic terms can provide a very efficient representation of grammars and other useful objects. Unfortunately, the use of infinite rational trees has problems. For instance, many of the built-in and library predicates are ill-defined for such trees and need to be supplemented by run-time checks whose cost may be significant. Moreover, some widely-used program analysis and manipulation techniques are only correct for those parts of programs working over finite trees. It is thus important to obtain, automatically, a knowledge of those program variables (the *finite variables*) that, at the program points of interest, will always be bound to finite terms. For these reasons, we propose here a new data-flow analysis that captures such information. We present a parametric domain where a simple component for recording finite variables is coupled with a generic domain (the parameter of the construction) providing sharing information. The sharing domain is abstractly specified so as to guarantee the correctness of the combined domain and the generality of the approach.

1 Introduction

The intended computation domain of most logic-based languages[1] includes the algebra (or structure) of *finite trees*. Other (constraint) logic-based languages, such as Prolog II and its successors [9, 11], SICStus Prolog [34], and Oz [32], refer to a computation domain of *rational trees*. A rational tree is a possibly infinite tree with a finite number of distinct subtrees and, as is the case for finite trees,

* This work has been partly supported by MURST project "Certificazione automatica di programmi mediante interpretazione astratta." Some of this work was done during visits of the fourth author to Leeds, funded by EPSRC under grant M05645.

[1] That is, ordinary logic languages, (concurrent) constraint logic languages, functional logic languages and variations of the above.

where each node has a finite number of immediate descendants. These properties will ensure that rational trees, even though infinite in the sense that they admit paths of infinite length, can be finitely represented. One possible representation makes use of connected, rooted, directed and possibly cyclic graphs where nodes are labeled with variable and function symbols as is the case of finite trees.

Applications of rational trees in logic programming include graphics [16], parser generation and grammar manipulation [9, 19], and computing with finite-state automata [9]. Other applications are described in [18] and [21]. Going from Prolog to CLP, [29] combines constraints on rational trees and record structures, while the logic-based language *Oz* allows constraints over rational and feature trees [32]. The expressive power of rational trees is put to use, for instance, in several areas of natural language processing. Rational trees are used in implementations of the HPSG formalism (Head-driven Phrase Structure Grammar) [30], in the ALE system (Attribute Logic Engine) [7], and in the ProFIT system (Prolog with Features, Inheritance and Templates) [17].

While rational trees allow for increased expressivity, they also come equipped with a surprising number of problems. As we will see, some of these problems are so serious that rational trees must be used in a very controlled way, disallowing them in any context where they are "dangerous". This, in turn, causes a secondary problem: in order to disallow rational trees in selected contexts one must first detect them, an operation that may be expensive.

The first thing to be aware of is that almost any semantics-based program manipulation technique developed in the field of logic programming —whether it be an analysis, a transformation, or an optimization— assumes a computation domain of *finite trees*. Some of these techniques might work with the rational trees but their correctness has only been proved in the case of finite trees. Others are clearly inapplicable. Let us consider a very simple Prolog program:

```
list([]).
list([_|T]) :- list(T).
```

Most automatic and semi-automatic tools for proving program termination and for complexity analysis agree on the fact that `list/1` will terminate when invoked with a ground argument. Consider now the query

```
?- X = [a|X], list(X).
```

and note that, after the execution of the first rational unification, the variable `X` will be bound to a rational term containing no variables, i.e., the predicate `list/1` will be invoked with `X` ground. However, if such a query is given to, say, SICStus Prolog, then the only way to get the prompt back is by pressing `^C`. The problem stems from the fact that the analysis techniques employed by these tools are only sound for finite trees: as soon as they are applied to a system where the creation of cyclic terms is possible, their results are inapplicable. The situation can be improved by combining these termination and/or complexity analyses by a finiteness analysis providing the precondition for the applicability of the other techniques.

The implementation of built-in predicates is another problematic issue. Indeed, it is widely acknowledged that, for the implementation of a system that provides real support for the rational trees, the biggest effort concerns proper handling of built-ins. Of course, the meaning of 'proper' depends on the actual built-in. Built-ins such as `copy_term/2` and `==/2` maintain a clear semantics when passing from finite to rational trees. For others, like `sort/2`, the extension can be questionable:[2] both raising an exception and answering `Y = [a]` can be argued to be "the right reaction" to the query

```
?- X = [a|X], sort(X, Y).
```

Other built-ins do not tolerate infinite trees in some argument positions. A good implementation should check for finiteness of the corresponding arguments and make sure "the right thing" —failing or raising an appropriate exception— always happens. However, such behavior appears to be uncommon. A small experiment we conducted on six Prolog implementations with queries like

```
?- X = 1+X, Y is X.
?- X = [97|X], name(Y, X).
?- X = [X|X], Y =.. [f|X].
```

resulted in infinite loops, memory exhaustion and/or system thrashing, segmentation faults or other fatal errors. One of the implementations tested, SICStus Prolog, is a professional one and implements run-time checks to avoid most cases where built-ins can have catastrophic effects.[3] The remaining systems are a bit more than research prototypes, but will clearly have to do the same if they evolve to the stage of production tools. Again, a data-flow analysis aimed at the detection of those variables that are definitely bound to finite terms would allow to avoid a (possibly significant) fraction of the useless run-time checks. Note that what has been said for built-in predicates applies to libraries as well. Even though it may be argued that it is enough for programmers to know that they should not use a particular library predicate with infinite terms, it is clear that the use of a "safe" library, including automatic checks which ensure that such predicates are never called with an illegal argument, will result in more robust systems. With the appropriate data-flow analyses, safe libraries do not have to be inefficient libraries.

Another serious problem is the following: the ISO Prolog standard term ordering cannot be extended to rational trees [M. Carlsson, Personal communication, October 2000]. Consider the rational trees defined by `A = f(B, a)` and `B = f(A, b)`. Clearly, `A == B` does not hold. Since the standard term ordering is total, we must have either `A @< B` or `B @< A`. Assume `A @< B`. Then `f(A, b) @< f(B, a)`, since the ordering of terms having the same principal functor is inherited by the ordering of subterms considered in a left-to-right fashion. Thus `B @< A` must hold, which is a contradiction. A dual contradiction

[2] Even though `sort/2` is not required to be a built-in by the standard, it is offered as such by several implementations.

[3] SICStus 3.8.5 still loops on `?- X = [97|X], name(Y, X).`

is obtained by assuming `B @< A`. As a consequence, applying one of the Prolog term-ordering predicates to one or two infinite terms may cause inconsistent results, giving rise to bugs that are exceptionally difficult to diagnose. For this reason, any system that extends ISO Prolog with rational trees ought to detect such situations and make sure they are not ignored (e.g., by throwing an exception or aborting execution with a meaningful message). However, predicates such as the term-ordering ones are likely to be called a significant number of times, since they are often used to maintain structures implementing ordered collections of terms. This is another instance of the efficiency issue mentioned above.

In this paper, we present a parametric abstract domain for finite-tree analysis, denoted by $H \times P$. This domain combines a simple component H (the *finiteness* component), recording the set of definitely finite variables, with a generic domain P (the parameter of the construction), providing sharing information. The term "sharing information" is to be understood in its broader meaning, which includes variable aliasing, groundness, linearity, freeness and any other kind of information that can improve the precision on these components, such as explicit structural information. Several domain combinations and abstract operators, characterized by different precision/complexity trade-offs, have been proposed to capture these properties (see [4] for an account of some of them). By giving a generic specification for this parameter component, it is possible to define and establish the correctness of the abstract operators on the finite-tree domain independently from any particular domain for sharing analysis.

The paper is structured as follows. The required notations and preliminary concepts are given in Section 2. The finite-tree domain is then introduced in Section 3: Section 3.1 provides the specification of the parameter domain P; Section 3.2 defines the abstraction function for the finiteness component H; Section 3.3 defines the abstract unification operator for $H \times P$. A brief description of some ongoing work on the subject is given in Section 4. We conclude in Section 5.

2 Preliminaries

2.1 Infinite Terms and Substitutions

For a set S, $\wp(S)$ is the powerset of S, whereas $\wp_{\mathrm{f}}(S)$ is the set of all the *finite* subsets of S. Let *Sig* denote a possibly infinite set of function symbols, ranked over the set of natural numbers. It is assumed that *Sig* contains at least two distinct function symbols, one having rank 0 (so that there exist finite ground terms) and one having rank greater than 0 (so that there exist infinite terms). Let *Vars* denote a denumerable set of variables, disjoint from *Sig*. Then *Terms* denotes the free algebra of all (possibly infinite) terms in the signature *Sig* having variables in *Vars*. Thus a term can be seen as an ordered labeled tree, possibly having some infinite paths and possibly containing variables: every inner node is labeled with a function symbol in *Sig* with a rank matching the number of the

node's immediate descendants, whereas every leaf is labeled by either a variable in *Vars* or a function symbol in *Sig* having rank 0 (a constant).

If $t \in \mathit{Terms}$ then $\mathrm{vars}(t)$ and $\mathrm{mvars}(t)$ denote the set and the multiset of variables occurring in t, respectively. We will also write $\mathrm{vars}(o)$ to denote the set of variables occurring in an arbitrary syntactic object o. If a occurs more than once in a multiset M we write $a \Subset M$.

Suppose $s, t \in \mathit{Terms}$: s and t are *independent* if $\mathrm{vars}(s) \cap \mathrm{vars}(t) = \varnothing$; if $y \in \mathrm{vars}(t)$ and $\neg\big(y \Subset \mathrm{mvars}(t)\big)$ we say that variable y *occurs linearly in* t, more briefly written using the predication $\mathrm{occ\_lin}(y, t)$; t is said to be *ground* if $\mathrm{vars}(t) = \varnothing$; t is *free* if $t \in \mathit{Vars}$; t is *linear* if, for all $y \in \mathrm{vars}(t)$, we have $\mathrm{occ\_lin}(y, t)$; finally, t is a *finite term* (or *Herbrand term*) if it contains a finite number of occurrences of function symbols. The sets of all ground, linear and finite terms are denoted by *GTerms*, *LTerms* and *HTerms*, respectively.

A *substitution* is a total function $\sigma\colon \mathit{Vars} \to \mathit{HTerms}$ that is the identity almost everywhere; in other words, the *domain* of σ,

$$\mathrm{dom}(\sigma) \stackrel{\text{def}}{=} \big\{\, x \in \mathit{Vars} \;\big|\; \sigma(x) \neq x \,\big\},$$

is finite. Given a substitution $\sigma\colon \mathit{Vars} \to \mathit{HTerms}$, we overload the symbol 'σ' so as to denote also the function $\sigma\colon \mathit{HTerms} \to \mathit{HTerms}$ defined as follows, for each term $t \in \mathit{HTerms}$:

$$\sigma(t) \stackrel{\text{def}}{=} \begin{cases} t, & \text{if } t \text{ is a constant symbol;} \\ \sigma(t), & \text{if } t \in \mathit{Vars}; \\ f\big(\sigma(t_1), \ldots, \sigma(t_n)\big), & \text{if } t = f(t_1, \ldots, t_n). \end{cases}$$

If $x \in \mathit{Vars}$ and $t \in \mathit{HTerms} \setminus \{x\}$, then $x \mapsto t$ is called a *binding*. The set of all bindings is denoted by *Bind*. Substitutions are denoted by the set of their bindings, thus a substitution σ is identified with the (finite) set

$$\big\{\, x \mapsto \sigma(x) \;\big|\; x \in \mathrm{dom}(\sigma) \,\big\}.$$

We denote by $\mathrm{vars}(\sigma)$ the set of variables occurring in the bindings of σ.

A substitution is said to be *circular* if, for $n > 1$, it has the form

$$\{x_1 \mapsto x_2, \ldots, x_{n-1} \mapsto x_n, x_n \mapsto x_1\},$$

where $x_1, \ldots, x_n$ are distinct variables. A substitution is in *rational solved form* if it has no circular subset. The set of all substitutions in rational solved form is denoted by *RSubst*.

If $t \in \mathit{HTerms}$, we write $t\sigma$ to denote $\sigma(t)$ and $t[x/s]$ to denote $t\{x \mapsto s\}$.

The composition of substitutions is defined in the usual way. Thus $\tau \circ \sigma$ is the substitution such that, for all terms $t \in \mathit{HTerms}$,

$$(\tau \circ \sigma)(t) = \tau\big(\sigma(t)\big)$$

and has the formulation

$$\tau \circ \sigma = \big\{\, x \mapsto x\sigma\tau \;\big|\; x \in \mathrm{dom}(\sigma), x \neq x\sigma\tau \,\big\} \cup \big\{\, x \mapsto x\tau \;\big|\; x \in \mathrm{dom}(\tau) \setminus \mathrm{dom}(\sigma) \,\big\}.$$

As usual, σ^0 denotes the identity function (i.e., the empty substitution) and, when $i > 0$, σ^i denotes the substitution $(\sigma \circ \sigma^{i-1})$.

For each $\sigma \in \mathit{RSubst}$, $s \in \mathit{HTerms}$, the sequence of finite terms

$$\sigma^0(s), \sigma^1(s), \sigma^2(s), \ldots$$

converges to a (possibly infinite) term, denoted $\sigma^\infty(s)$ [23, 27]. Therefore, the function $\mathrm{rt}\colon \mathit{HTerms} \times \mathit{RSubst} \to \mathit{Terms}$ such that

$$\mathrm{rt}(s, \sigma) \stackrel{\mathrm{def}}{=} \sigma^\infty(s)$$

is well defined. Note that, in general, this function is not a substitution: while having a finite domain, its "bindings" $x \mapsto t$ can map a domain variable x into a term $t \in \mathit{Terms} \setminus \mathit{HTerms}$.

2.2 Equations

An *equation* is of the form $s = t$ where $s, t \in \mathit{HTerms}$. *Eqs* denotes the set of all equations. A substitution σ may be regarded as a finite set of equations, that is, as the set $\{\, x = t \mid x \mapsto t \in \sigma \,\}$. We say that a set of equations e is in *rational solved form* if $\left\{\, s \mapsto t \;\middle|\; (s = t) \in e \,\right\} \in \mathit{RSubst}$. In the rest of the paper, we will often write a substitution $\sigma \in \mathit{RSubst}$ to denote a set of equations in rational solved form (and vice versa).

Some logic-based languages, such as Prolog II, SICStus and Oz, are based on $\mathcal{RT}$, the theory of rational trees [9, 10]. This is a syntactic equality theory (i.e., a theory where the function symbols are uninterpreted), augmented with a *uniqueness axiom* for each substitution in rational solved form. Informally speaking these axioms state that, after assigning a ground rational tree to each parameter variable, the substitution uniquely defines a ground rational tree for each of its domain variables. Thus, any set of equations in rational solved form is, by definition, satisfiable in $\mathcal{RT}$. Note that being in rational solved form is a very weak property. Indeed, unification algorithms returning a set of equations in rational solved form are allowed to be much more "lazy" than one would usually expect. We refer the interested reader to [25, 26, 28] for details on the subject.

Given a set of equations $e \in \wp_{\mathrm{f}}(\mathit{Eqs})$ that is satisfiable in $\mathcal{RT}$, a substitution $\sigma \in \mathit{RSubst}$ is called a *solution for e in $\mathcal{RT}$* if $\mathcal{RT} \vdash \forall(\sigma \to e)$. If in addition $\mathrm{vars}(\sigma) \subseteq \mathrm{vars}(e)$, then σ is said to be a *relevant* solution for e. Finally, σ is a *most general solution for e in $\mathcal{RT}$* if $\mathcal{RT} \vdash \forall(\sigma \leftrightarrow e)$. In this paper, the set of all the relevant most general solution for e in $\mathcal{RT}$ will be denoted by $\mathrm{mgs}(e)$.

2.3 The Concrete Domain

Throughout the paper, we assume a knowledge of the basic concepts of abstract interpretation theory [13, 14].

For the purpose of this paper, we assume a concrete domain constituted by pairs of the form (Σ, V), where V is a finite set of *variables of interest* and Σ is a (possibly infinite) set of substitutions in rational solved form.

Definition 1. (The concrete domain.) *Let* $\mathcal{D}^\flat \stackrel{\text{def}}{=} \wp(RSubst) \times \wp_{\mathrm{f}}(Vars)$. *If* $(\Sigma, V) \in \mathcal{D}^\flat$, *then* (Σ, V) *represents the (possibly infinite) set of first-order formulas* $\{\, \exists \Delta \,.\, \sigma \mid \sigma \in \Sigma, \Delta = \mathrm{vars}(\sigma) \setminus V \,\}$ *where* σ *is interpreted as the logical conjunction of the equations corresponding to its bindings.*

Concrete domains for constraint languages would be similar. If the analyzed language allows the use of constraints on various domains to restrict the values of the variable leaves of rational trees, the corresponding concrete domain would have one or more extra components to account for the constraints (see [2] for an example).

The concrete element $\big(\{\{x \mapsto f(y)\}\}, \{x, y\}\big)$ expresses a dependency between x and y. In contrast, $\big(\{\{x \mapsto f(y)\}\}, \{x\}\big)$ only constrains x. The same concept can be expressed by saying that in the first case the variable name 'y' matters, but it does not in the second case. Thus, the set of variables of interest is crucial for defining the meaning of the concrete and abstract descriptions. Despite this, always specifying the set of variables of interest would significantly clutter the presentation. Moreover, most of the needed functions on concrete and abstract descriptions, preserve the set of variables of interest. For these reasons, we assume the existence of a set $VI \in \wp_{\mathrm{f}}(Vars)$ that contains, at each stage of the analysis, the current variables of interest.[4] As a consequence, when the context makes it clear that $\Sigma \in \wp(RSubst)$, we will write $\Sigma \in \mathcal{D}^\flat$ as a shorthand for $(\Sigma, VI) \in \mathcal{D}^\flat$.

3 An Abstract Domain for Finiteness Analysis

Finite-tree analysis applies to logic-based languages computing over a domain of rational trees where cyclic structures are allowed. In contrast, analyses aimed at occurs-check reduction [15, 33] apply to programs that are meant to compute on a domain of finite trees only, but have to be executed over systems that are either designed for rational trees or intended just for the finite trees but omit the occurs-check for efficiency reasons. Despite their different objectives, finite-tree and occurs-check analyses have much in common: in both cases, it is important to detect all program points where cyclic structures can be generated.

Note however that, when performing occurs-check reduction, one can take advantage of the following invariant: all data structures generated so far are finite. This property is maintained by transforming the program so as to force finiteness whenever it is possible that a cyclic structure could have been built.[5]

[4] This parallels what happens in the efficient implementation of data-flow analyzers. In fact, almost all the abstract domains currently in use do not need to represent explicitly the set of variables of interest. In contrast, this set is maintained externally and in a unique copy, typically by the fixpoint computation engine.

[5] Such a requirement is typically obtained by replacing the unification with a call to `unify_with_occur_check/2`. As an alternative, in some systems based on rational trees it is possible to insert, after each problematic unification, a finiteness test for the generated term.

In contrast, a finite-tree analysis has to deal with the more general case when some of the data structures computed so far may be cyclic. It is therefore natural to consider an abstract domain made up of two components. The first one simply represents the set of variables that are guaranteed not to be bound to infinite terms. We will denote this *finiteness component* by H (from *Herbrand*).

Definition 2. (The finiteness component.) *The* finiteness component *is the set* $H \stackrel{\text{def}}{=} \wp(VI)$ *partially ordered by reverse subset inclusion.*

The second component of the finite-tree domain should maintain any kind of information that may be useful for computing and possibly propagating finiteness information.

It is well-known that sharing information as a whole, therefore including possible variable aliasing, definite linearity, and definite freeness, has a crucial role in occurs-check reduction so that, as observed before, it can be exploited for finite-tree analysis too. Thus, a first choice for the second component of the finite-tree domain would be to consider one of the standard combinations of sharing, freeness and linearity as defined, e.g., in [4, 5, 20]. However, this would tie our specification to a particular sharing analysis domain, whereas the overall approach seems to be inherently more general. For this reason, we will define a finite-term analysis based on the abstract domain schema $H \times P$, where the generic *sharing component* P is a parameter of the abstract domain construction.

3.1 The parameter component P

Elements of P can encode any kind of information. We only require that substitutions that are equivalent in the theory $\mathcal{RT}$ are identified in P.

Definition 3. (The parameter component.) *The parameter component* P *is an abstract domain related to the concrete domain* $\mathcal{D}^\flat$ *by means of the* concretization function $\gamma_P \colon P \to \wp(RSubst)$ *such that, for all* $p \in P$,

$$\Big(\sigma \in \gamma_P(p) \wedge \big(\mathcal{RT} \vdash \forall(\sigma \leftrightarrow \tau)\big)\Big) \implies \tau \in \gamma_P(p).$$

The interface between H and P is provided by a set of predicates and functions that satisfy suitable correctness criteria. Note that, for space limitations, we will only specify those abstract operations that are useful to define abstract unification on the combined domain $H \times P$. The other operations needed for a full description of the analysis, such as renamings, upper bound operators and projections, are very simple and, as usual, do not pose any problems.

Definition 4. (Abstract operators on P.) *Let* $s, t \in HTerms$ *be finite terms. For each* $p \in P$, *we define the following predicates:*
s *and* t *are* independent in p *if and only if* $\mathrm{ind}_p \colon HTerms^2 \to Bool$ *holds for* (s, t), *where*

$$\mathrm{ind}_p(s,t) \implies \forall \sigma \in \gamma_P(p) : \mathrm{vars}\big(\mathrm{rt}(s,\sigma)\big) \cap \mathrm{vars}\big(\mathrm{rt}(t,\sigma)\big) = \varnothing;$$

s and t share linearly *in p if and only if* $\mathrm{share\_lin}_p\colon \mathit{HTerms}^2 \to \mathit{Bool}$ *holds for* (s,t)*, where*

$$\mathrm{share\_lin}_p(s,t) \implies \forall \sigma \in \gamma_P(p) : \forall y \in \mathrm{vars}\big(\mathrm{rt}(s,\sigma)\big) \cap \mathrm{vars}\big(\mathrm{rt}(t,\sigma)\big) : \mathrm{occ\_lin}\big(y, \mathrm{rt}(s,\sigma)\big) \wedge \mathrm{occ\_lin}\big(y, \mathrm{rt}(t,\sigma)\big);$$

t is ground *in p if and only if* $\mathrm{ground}_p\colon \mathit{HTerms} \to \mathit{Bool}$ *holds for t, where*

$$\mathrm{ground}_p(t) \implies \forall \sigma \in \gamma_P(p) : \mathrm{rt}(t,\sigma) \in \mathit{GTerms};$$

t is ground-or-free *in p if and only if* $\mathrm{gfree}_p\colon \mathit{HTerms} \to \mathit{Bool}$ *holds for t, where*

$$\mathrm{gfree}_p(t) \implies \forall \sigma \in \gamma_P(p) : \mathrm{rt}(t,\sigma) \in \mathit{GTerms} \vee \mathrm{rt}(t,\sigma) \in \mathit{Vars};$$

s and t are or-linear *in p if and only if* $\mathrm{or\_lin}_p\colon \mathit{HTerms}^2 \to \mathit{Bool}$ *holds for* (s,t)*, where*

$$\mathrm{or\_lin}_p(s,t) \implies \forall \sigma \in \gamma_P(p) : \mathrm{rt}(s,\sigma) \in \mathit{LTerms} \vee \mathrm{rt}(t,\sigma) \in \mathit{LTerms};$$

s is linear *in p if and only if* $\mathrm{lin}_p\colon \mathit{HTerms} \to \mathit{Bool}$ *holds for s, where*

$$\mathrm{lin}_p(s) \stackrel{\mathrm{def}}{\iff} \mathrm{or\_lin}_p(s,s).$$

For each $p \in P$, the following functions compute subsets of the set of variables of interest:
the function $\mathrm{share\_same\_var}_p\colon \mathit{HTerms} \times \mathit{HTerms} \to \wp(\mathit{VI})$ *returns a set of variables that may share with the given terms via the same variable. For each $s,t \in \mathit{HTerms}$,*

$$\mathrm{share\_same\_var}_p(s,t) \supseteq \left\{ y \in \mathit{VI} \;\middle|\; \begin{array}{l} \exists \sigma \in \gamma_P(p)\ . \\ \quad \exists z \in \mathrm{vars}\big(\mathrm{rt}(y,\sigma)\big)\ . \\ \qquad z \in \mathrm{vars}\big(\mathrm{rt}(s,\sigma)\big) \cap \mathrm{vars}\big(\mathrm{rt}(t,\sigma)\big) \end{array} \right\};$$

the function $\mathrm{share\_with}_p\colon \mathit{HTerms} \to \wp(\mathit{VI})$ *yields a set of variables that may share with the given term. For each $t \in \mathit{HTerms}$,*

$$\mathrm{share\_with}_p(t) \stackrel{\mathrm{def}}{=} \big\{\, y \in \mathit{VI} \;\big|\; y \in \mathrm{share\_same\_var}_p(y,t) \,\big\}.$$

The function $\mathrm{amgu}_P\colon P \times \mathit{Bind} \to P$ *correctly captures the effects of a binding on an element of P. For each $(x \mapsto t) \in \mathit{Bind}$ and $p \in P$, let*

$$p' \stackrel{\mathrm{def}}{=} \mathrm{amgu}_P\big(p, x \mapsto t\big).$$

For all $\sigma \in \gamma_P(p)$, if $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$, then $\tau \in \gamma_P(p')$.

Some of these generic operators can be directly mapped into the corresponding abstract operators defined for well-known sharing analysis domains (e.g., see those defined in [5]). However, the specification given in Definition 4, besides being more general than a particular implementation, also allows for a modular approach when proving correctness results.

3.2 The abstraction function for H

When the concrete domain is based on the theory of finite trees, idempotent substitutions provide a finitely computable *strong normal form* for domain elements, meaning that different substitutions describe different sets of finite trees.[6] In contrast, when working on a concrete domain based on the theory of rational trees, substitutions in rational solved form, while being finitely computable, no longer satisfy this property: there can be an infinite set of substitutions in rational solved form all describing the same set of rational trees (i.e., the same element in the "intended" semantics). For instance, the substitutions

$$\sigma_n = \{x \mapsto \overbrace{f(\cdots f(}^{n} x) \cdots)\}$$

for $n = 1, 2, \ldots$, all map the variable x into the same rational tree (which is usually denoted by f^ω).

Ideally, a strong normal form for the set of rational trees described by a substitution $\sigma \in \mathit{RSubst}$ can be obtained by computing the limit σ^∞. The problem is that we may end up with $\sigma^\infty \notin \mathit{RSubst}$, as σ^∞ can map domain variables to infinite rational terms.

This poses a non-trivial problem when trying to define a "good" abstraction function, since it would be really desirable for this function to map any two equivalent concrete elements to the same abstract element. As shown in [22], the classical abstraction function for set-sharing analysis [12, 24], which was defined for idempotent substitutions only, does not enjoy this property when applied, as it is, to arbitrary substitutions in rational solved form. A possibility is to look for a more general abstraction function that allows to obtain the desired property. For example, in [22] the sharing-group operator sg of [24] is replaced by an occurrence operator, occ, defined by means of a fixpoint computation. We now provide a similar fixpoint construction defining the finiteness operator.

Definition 5. (Finiteness functions.) *For each $n \in \mathbb{N}$, the* finiteness function $\mathrm{hvars}_n \colon \mathit{RSubst} \to \wp(\mathit{Vars})$ *is defined, for each $\sigma \in \mathit{RSubst}$, by*

$$\mathrm{hvars}_0(\sigma) \stackrel{\mathrm{def}}{=} \mathit{Vars} \setminus \mathrm{dom}(\sigma)$$

and, for $n > 0$, by

$$\mathrm{hvars}_n(\sigma) \stackrel{\mathrm{def}}{=} \mathrm{hvars}_{n-1}(\sigma) \cup \big\{\, y \in \mathrm{dom}(\sigma) \;\big|\; \mathrm{vars}(y\sigma) \subseteq \mathrm{hvars}_{n-1}(\sigma) \,\big\}.$$

[6] As usual, this is to be intended modulo the possible renaming of variables.

For each $\sigma \in \mathit{RSubst}$ and each $i > 0$, we have $\mathrm{hvars}_i(\sigma) \subseteq \mathrm{hvars}_{i+1}(\sigma)$. Note also that $\mathit{Vars} \setminus \mathrm{hvars}_i(\sigma) \subseteq \mathrm{dom}(\sigma)$ is a finite set. By these two properties, the following fixpoint computation is well defined and finitely computable.

Definition 6. (Finiteness operator.) *For each $\sigma \in \mathit{RSubst}$, the* finiteness operator $\mathrm{hvars}\colon \mathit{RSubst} \to \wp(\mathit{Vars})$ *is given by*

$$\mathrm{hvars}(\sigma) \stackrel{\mathrm{def}}{=} \mathrm{hvars}_\ell(\sigma)$$

where $\ell \stackrel{\mathrm{def}}{=} \ell(\sigma) \in \mathbb{N}$ is such that $\mathrm{hvars}_\ell(\sigma) = \mathrm{hvars}_n(\sigma)$ for all $n \geq \ell$.

The following proposition shows that the hvars operator precisely captures the intended property.

Proposition 1. *If $\sigma \in \mathit{RSubst}$ and $x \in \mathit{Vars}$ then*

$$x \in \mathrm{hvars}(\sigma) \iff \mathrm{rt}(x, \sigma) \in \mathit{HTerms}.$$

Example 1. Consider $\sigma \in \mathit{RSubst}$, where

$$\sigma = \big\{x_1 \mapsto f(x_2), x_2 \mapsto g(x_5), x_3 \mapsto f(x_4), x_4 \mapsto g(x_3)\big\}.$$

Then,

$$\begin{aligned}
\mathrm{hvars}_0(\sigma) &= \mathit{Vars} \setminus \{x_1, x_2, x_3, x_4\},\\
\mathrm{hvars}_1(\sigma) &= \mathit{Vars} \setminus \{x_1, x_3, x_4\},\\
\mathrm{hvars}_2(\sigma) &= \mathit{Vars} \setminus \{x_3, x_4\}\\
&= \mathrm{hvars}(\sigma).
\end{aligned}$$

Thus, $x_1 \in \mathrm{hvars}(\sigma)$, even if $\mathrm{vars}(x_1\sigma) \subseteq \mathrm{dom}(\sigma)$.

The abstraction function for H can then be defined in the obvious way.

Definition 7. (The abstraction function for H.) *The abstraction function $\alpha_H\colon \mathit{RSubst} \to H$ is defined, for each $\sigma \in \mathit{RSubst}$, by*

$$\alpha_H(\sigma) \stackrel{\mathrm{def}}{=} \mathit{VI} \cap \mathrm{hvars}(\sigma).$$

The concrete domain $\mathcal{D}^\flat$ is related to H by means of the abstraction function $\alpha_H\colon \mathcal{D}^\flat \to H$ *such that, for each $\Sigma \in \wp(\mathit{RSubst})$,*

$$\alpha_H(\Sigma) \stackrel{\mathrm{def}}{=} \bigcap \big\{\, \alpha_H(\sigma) \;\big|\; \sigma \in \Sigma \,\big\}.$$

Since the abstraction function α_H is additive, the concretization function is given by its adjoint [13]*:*

$$\gamma_H(h) \stackrel{\mathrm{def}}{=} \big\{\, \sigma \in \mathit{RSubst} \;\big|\; \alpha_H(\sigma) \supseteq h \,\big\}.$$

With these definitions, we have the desired result: equivalent substitutions in rational solved form have the same finiteness abstraction.

Theorem 1. *If $\sigma, \sigma' \in \mathit{RSubst}$ and $\mathcal{RT} \vdash \forall(\sigma \leftrightarrow \sigma')$, then $\alpha_H(\sigma) = \alpha_H(\sigma')$.*

3.3 Abstract unification on $H \times P$

The abstract unification for the combined domain $H \times P$ is defined by using the abstract predicates and functions as specified for P as well as a new finiteness predicate for the domain H.

Definition 8. (Abstract unification on $H \times P$.) *A term $t \in HTerms$ is a* finite tree in h *if and only if the predicate* $\mathrm{hterm}_h \colon HTerms \to Bool$ *holds for t, where* $\mathrm{hterm}_h(t) \stackrel{\text{def}}{=} \mathrm{vars}(t) \subseteq h$.

The function $\mathrm{amgu}_H \colon (H \times P) \times Bind \to H$ *captures the effects of a binding on an H element. Let $\langle h, p\rangle \in H \times P$ and $(x \mapsto t) \in Bind$. Then*

$$\mathrm{amgu}_H\big(\langle h, p\rangle, x \mapsto t\big) \stackrel{\text{def}}{=} h',$$

where

$$h' \stackrel{\text{def}}{=} \begin{cases} h \cup \mathrm{vars}(t), & \textit{if } \mathrm{hterm}_h(x) \wedge \mathrm{ground}_p(x); \\ h \cup \{x\}, & \textit{if } \mathrm{hterm}_h(t) \wedge \mathrm{ground}_p(t); \\ h, & \textit{if } \mathrm{hterm}_h(x) \wedge \mathrm{hterm}_h(t) \\ & \quad \wedge \mathrm{ind}_p(x,t) \wedge \mathrm{or\_lin}_p(x,t); \\ h, & \textit{if } \mathrm{hterm}_h(x) \wedge \mathrm{hterm}_h(t) \\ & \quad \wedge \mathrm{gfree}_p(x) \wedge \mathrm{gfree}_p(t); \\ h \setminus \mathrm{share\_same\_var}_p(x,t), & \textit{if } \mathrm{hterm}_h(x) \wedge \mathrm{hterm}_h(t) \\ & \quad \wedge \mathrm{share\_lin}_p(x,t) \\ & \quad \wedge \mathrm{or\_lin}_p(x,t); \\ h \setminus \mathrm{share\_with}_p(x), & \textit{if } \mathrm{hterm}_h(x) \wedge \mathrm{lin}_p(x); \\ h \setminus \mathrm{share\_with}_p(t), & \textit{if } \mathrm{hterm}_h(t) \wedge \mathrm{lin}_p(t); \\ h \setminus \big(\mathrm{share\_with}_p(x) \cup \mathrm{share\_with}_p(t)\big), & \textit{otherwise.} \end{cases}$$

The abstract unification function $\mathrm{amgu} \colon (H \times P) \times Bind \to H \times P$, *for any $\langle h, p\rangle \in H \times P$ and $(x \mapsto t) \in Bind$, is given by*

$$\mathrm{amgu}\big(\langle h, p\rangle, x \mapsto t\big) \stackrel{\text{def}}{=} \Big\langle \mathrm{amgu}_H\big(\langle h, p\rangle, x \mapsto t\big), \mathrm{amgu}_P(p, x \mapsto t)\Big\rangle.$$

In the computation of h' (the new finiteness component resulting from the abstract evaluation of a binding) there are eight cases based on properties holding for the concrete terms described by x and t.

1. In the first case, the concrete term described by x is both finite and ground. Thus, after a successful execution of the binding, any concrete term described by t will be finite. Note that t could have contained possibly cyclic variables just before the execution of the binding.
2. The second case is symmetric to the first one. Note that these are the only cases when a "positive" propagation of finiteness information is correct. In contrast, in all the remaining cases, the goal is to limit as much as possible the propagation of "negative" information, i.e., the possible cyclicity of terms.

3. The third case exploits the classical results proved in research work on occurs-check reduction [15, 33]. Accordingly, it is required that both x and t describe finite terms. The use of the implicitly disjunctive predicate $\mathrm{or\_lin}_p$ allows for the application of this case even when neither x nor t are known to be definitely linear. For instance, as observed in [15], this may happen when the component P embeds the domain *Pos* for groundness analysis.[7]
4. The fourth case exploits the observation that cyclic terms cannot be created when unifying two finite terms that are either ground or free. Ground-or-freeness has been identified in [4] as a safe and inexpensive replacement for the classical freeness property when combining sharing analysis domains.
5. The fifth case applies when unifying a linear and finite term with another finite term possibly sharing with it, provided they can only share linearly (namely, all the shared variables occur linearly in the considered terms). In such a context, cycles could have been introduced by means of the shared variables only. This case shows that, when observing the term finiteness property, set-sharing is strictly more precise than pair-sharing, since a set-sharing domain can be strictly more precise when computing the functions $\mathrm{share\_same\_var}_p$ and $\mathrm{share\_lin}_p$.[8] This is true regardless of the pair-sharing variant considered, including ASub [33, 8], *PSD* [1, 3] and $\mathsf{Sh}^{\mathsf{PSh}}$ [31].
6. In the sixth case, we drop the assumption about the finiteness of the term described by t. As a consequence, all variables sharing with x become possibly cyclic. However, provided x describes a finite and linear term, all finite variables independent from x preserve their finiteness.
7. The seventh case is symmetric to the sixth one.
8. The last case states that term finiteness is preserved for all variables that are independent from both x and t. Note that this case is only used when none of the other cases apply.

The following result, together with the assumption on amgu_P as specified in Definition 4, ensures that abstract unification on the combined domain $H \times P$ is correct.

Theorem 2. *Let* $\langle h, p\rangle \in H \times P$ *and* $(x \mapsto t) \in \mathit{Bind}$*, where* $\{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$*. Let also* $\sigma \in \gamma_H(h) \cap \gamma_P(p)$ *and* $h' = \mathrm{amgu}_H\big(\langle h, p\rangle, x \mapsto t\big)$*. Then*

$$\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big) \implies \tau \in \gamma_H(h').$$

[7] Take $t = y$. From the *Pos* formula $\phi \stackrel{\mathrm{def}}{=} (x \vee y)$, both $\mathrm{ind}_\phi(x, y)$ and $\mathrm{or\_lin}_\phi(x, y)$ can be correctly inferred. Note that from ϕ we cannot infer that x is definitely linear and neither that y is definitely linear.

[8] For the expert: on the classical set-sharing domain by Jacobs and Langen [24], we can define $\mathrm{share\_same\_var}_{sh}(x, t) \stackrel{\mathrm{def}}{=} \mathrm{vars}\Big(\mathrm{rel}\big(\{x\}, sh\big) \cap \mathrm{rel}\big(\mathrm{vars}(t), sh\big)\Big)$. For instance, taking $t = y$ and $sh = \big\{\{x, y\}, \{x, z\}, \{y, z\}\big\}$, we have $z \notin \mathrm{share\_same\_var}_{sh}(x, y)$. In contrast, when using a pair-sharing domain such as *PSD*, the element sh is equivalent to $sh' = sh \cup \big\{\{x, y, z\}\big\}$, so that we have $z \in \mathrm{share\_same\_var}_{sh'}(x, y)$. The point is that, in sh, the information provided by the lack of the sharing group $\{x, y, z\}$ is redundant when observing pair-sharing and groundness, but it is not redundant when observing term finiteness.

4 Ongoing and Further Work

The parametric domain $H \times P$ basically captures the negative aspect of term-finiteness, that is, the circumstances under which finiteness can be lost. When a binding has the potential for creating one or more rational terms, the operator amgu_H removes from h all the variables that may be bound to non-finite terms. However, term-finiteness has also a positive aspect: there are cases where a variable is guaranteed to be bound to a finite term and this knowledge can be propagated to other variables. Guarantees of finiteness are provided by several built-ins like `unify_with_occurs_check/2`, `var/1`, `name/2`, all the arithmetic predicates, and so forth. SICStus Prolog also provides an explicit `acyclic/1` predicate.

The term-finiteness information provided by the h component of $H \times P$ does not capture the information concerning how finiteness of one variable affects the finiteness of other variables. This kind of information, usually termed *relational information*, is very important as it allows the propagation of positive finiteness information. An important source of relational information comes from *dependencies*. Consider the terms $t_1 \stackrel{\text{def}}{=} f(x)$, $t_2 \stackrel{\text{def}}{=} g(y)$, and $t_3 \stackrel{\text{def}}{=} h(x, y)$: it is clear that, for each assignment of rational terms to x and y, t_3 is finite if and only if t_1 and t_2 are so. We can capture this by the Boolean formula $t_3 \leftrightarrow (t_1 \wedge t_2)$. The reasoning is based on the following facts:

1. t_1, t_2, and t_3 are finite terms, so that the finiteness of their instances depends only on the finiteness of the terms that take the place of x and y.
2. t_3 *covers* both t_1 and t_2, that is, $\text{vars}(t_3) \supseteq \text{vars}(t_1) \cup \text{vars}(t_2)$; this means that, if an assignment to the variables of t_3 produces a finite instance of t_3, that very same assignment will necessarily result in finite instances of t_1 and t_2. Conversely, an assignment producing non-finite instances of t_1 or t_2 will forcibly result in a non-finite instance of t_3.
3. Similarly, t_1 and t_2, taken together, cover t_3.

The important point to notice is that the indicated dependency will continue to hold for any further simultaneous instantiation of t_1, t_2, and t_3. In other words, such dependencies are preserved by forward computations (since they proceed by consistently instantiating program variables).

Consider the abstract binding $x \mapsto t$ where t is a finite term such that $\text{vars}(t) = \{y_1, \ldots, y_n\}$. After this binding has been successfully performed, the destinies of x and t concerning term-finiteness are tied together forever. This tie can be described by the dependency formula

$$x \leftrightarrow (y_1 \wedge \cdots \wedge y_n), \tag{1}$$

meaning that x will be bound to a finite term if and only if, for each $i = 1, \ldots, n$, y_i is bound to a finite term. While the dependency expressed by (1) is a correct description of any computation state following the application of the binding $x \mapsto t$, it is not as precise as it could be. Suppose that x and y_k are

indeed the same variable. Then (1) is logically equivalent to

$$x \rightarrow (y_1 \wedge \cdots \wedge y_{k-1} \wedge y_{k+1} \wedge \cdots \wedge y_n). \tag{2}$$

Correct: whenever x is bound to a finite term, all the other variables will be bound to finite terms. The point is that x has just been bound to a non-finite term, irrevocably: no forward computation can change this. Thus, the implication (2) holds vacuously. The precise and correct description for the state of affairs caused by the cyclic binding is, instead, the negated atom $\neg x$, whose intuitive reading is "x is not (and never will be) finite."

Following the intuition outlined above, we are studying a domain, whose carrier is the set of all Boolean functions, for representing and propagating finiteness dependencies. We believe that coupling this new domain with $H \times P$ can greatly improve the precision of the analysis.

An implementation of the finite-tree domain, where the parametric component P will be instantiated to the sharing domain *SFL* as presented in [4], is currently under development. We also plan to work on the correctness, with respect to rational unification, of the abstract operators defined on the domain *SFL*, therefore generalizing and extending the results proved in [22] for the set-sharing domain of Jacobs and Langen.

We believe that information about the actual structure of terms can have a key role in improving the precision of finite-tree analysis, since it allows to better identify where cyclic structures may appear. As soon as the implementation of $H \times SFL$ will be finished, it will be possible to experiment with the integration of the explicit structural information provided by the generic Pattern$(\cdot)$ construction [2].

5 Conclusion

Several modern logic-based languages offer a computation domain based on rational trees. On the one hand, the use of such trees is encouraged by the possibility of using efficient and correct unification algorithms and by an increase in expressivity. On the other hand, these gains are countered by the extra problems rational trees bring with themselves and that can be summarized as follows: several built-ins, library predicates, program analysis and manipulation techniques are only well-defined for program fragments working with finite trees.

In this paper we propose an abstract-interpretation based solution to the problem of detecting program variables that can only be bound to finite terms. The rationale behind this is that applications exploiting rational trees tend to do so in a very controlled way. If the analysis we propose proves to be precise enough, then we will have a practical way of taking advantage of rational trees while minimizing the impact of their disadvantages.

References

1. R. Bagnara, P. M. Hill, and E. Zaffanella. Set-sharing is redundant for pair-sharing. In P. Van Hentenryck, editor, *Static Analysis: Proceedings of the 4th International*

Symposium, volume 1302 of *Lecture Notes in Computer Science*, pages 53–67, Paris, France, 1997. Springer-Verlag, Berlin.

2. R. Bagnara, P. M. Hill, and E. Zaffanella. Efficient structural information analysis for real CLP languages. In M. Parigot and A. Voronkov, editors, *Proceedings of the 7th International Conference on Logic for Programming and Automated Reasoning (LPAR 2000)*, volume 1955 of *Lecture Notes in Computer Science*, pages 189–206, Reunion Island, France, 2000. Springer-Verlag, Berlin.
3. R. Bagnara, P. M. Hill, and E. Zaffanella. Set-sharing is redundant for pair-sharing. *Theoretical Computer Science*, 2001. To appear.
4. R. Bagnara, E. Zaffanella, and P. M. Hill. Enhanced sharing analysis techniques: A comprehensive evaluation. In M. Gabbrielli and F. Pfenning, editors, *Proceedings of the 2nd International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming*, pages 103–114, Montreal, Canada, 2000. Association for Computing Machinery.
5. M. Bruynooghe, M. Codish, and A. Mulkers. Abstract unification for a composite domain deriving sharing and freeness properties of program variables. In F. S. de Boer and M. Gabbrielli, editors, *Verification and Analysis of Logic Languages, Proceedings of the W2 Post-Conference Workshop, International Conference on Logic Programming*, pages 213–230, Santa Margherita Ligure, Italy, 1994.
6. J. A. Campbell, editor. *Implementations of Prolog.* Ellis Horwood/Halsted Press/Wiley, 1984.
7. B. Carpenter. *The Logic of Typed Feature Structures with Applications to Unification-based Grammars, Logic Programming and Constraint Resolution*, volume 32 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, New York, 1992.
8. M. Codish, D. Dams, and E. Yardeni. Derivation and safety of an abstract unification algorithm for groundness and aliasing analysis. In K. Furukawa, editor, *Logic Programming: Proceedings of the Eighth International Conference on Logic Programming*, MIT Press Series in Logic Programming, pages 79–93, Paris, France, 1991. The MIT Press.
9. A. Colmerauer. Prolog and infinite trees. In K. L. Clark and S. Å. Tärnlund, editors, *Logic Programming, APIC Studies in Data Processing*, volume 16, pages 231–251. Academic Press, New York, 1982.
10. A. Colmerauer. Equations and inequations on finite and infinite trees. In *Proceedings of the International Conference on Fifth Generation Computer Systems (FGCS'84)*, pages 85–99, Tokyo, Japan, 1984. ICOT.
11. A. Colmerauer. An introduction to Prolog-III. *Communications of the ACM*, 33(7):69–90, 1990.
12. A. Cortesi and G. Filé. Sharing is optimal. *Journal of Logic Programming*, 38(3):371–386, 1999.
13. P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the Fourth Annual ACM Symposium on Principles of Programming Languages*, pages 238–252, 1977.
14. P. Cousot and R. Cousot. Abstract interpretation frameworks. *Journal of Logic and Computation*, 2(4):511–547, 1992.
15. L. Crnogorac, A. D. Kelly, and H. Søndergaard. A comparison of three occur-check analysers. In R. Cousot and D. A. Schmidt, editors, *Static Analysis: Proceedings of the 3rd International Symposium*, volume 1145 of *Lecture Notes in Computer Science*, pages 159–173, Aachen, Germany, 1996. Springer-Verlag, Berlin.

16. P. R. Eggert and K. P. Chow. Logic programming, graphics and infinite terms. Technical Report UCSB DoCS TR 83-02, Department of Computer Science, University of California at Santa Barbara, 1983.
17. G. Erbach. ProFIT: Prolog with Features, Inheritance and Templates. In *Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics*, pages 180–187, Dublin, Ireland, 1995.
18. M. Filgueiras. A Prolog interpreter working with infinite terms. In Campbell [6], pages 250–258.
19. F. Giannesini and J. Cohen. Parser generation and grammar manipulation using Prolog's infinite trees. *Journal of Logic Programming*, 3:253–265, 1984.
20. W. Hans and S. Winkler. Aliasing and groundness analysis of logic programs through abstract interpretation and its safety. Technical Report 92–27, Technical University of Aachen (RWTH Aachen), 1992.
21. S. Haridi and D. Sahlin. Efficient implementation of unification of cyclic structures. In Campbell [6], pages 234–249.
22. P. M. Hill, R. Bagnara, and E. Zaffanella. Soundness, idempotence and commutativity of set-sharing. *Theory and Practice of Logic Programming*, 2001. To appear.
23. B. Intrigila and M. Venturini Zilli. A remark on infinite matching vs infinite unification. *Journal of Symbolic Computation*, 21(3):2289–2292, 1996.
24. D. Jacobs and A. Langen. Accurate and efficient approximation of variable aliasing in logic programs. In E. L. Lusk and R. A. Overbeek, editors, *Logic Programming: Proceedings of the North American Conference*, MIT Press Series in Logic Programming, pages 154–165, Cleveland, Ohio, USA, 1989. The MIT Press.
25. J. Jaffar, J-L. Lassez, and M. J. Maher. Prolog-II as an instance of the logic programming scheme. In M. Wirsing, editor, *Formal Descriptions of Programming Concepts III*, pages 275–299. North-Holland, 1987.
26. T. Keisu. *Tree Constraints*. PhD thesis, The Royal Institute of Technology, Stockholm, Sweden, May 1994. Also available in the SICS Dissertation Series: SICS/D–16–SE.
27. A. King. Pair-sharing over rational trees. *Journal of Logic Programming*, 46(1–2):139–155, 2000.
28. M. J. Maher. Complete axiomatizations of the algebras of finite, rational and infinite trees. In *Proceedings, Third Annual Symposium on Logic in Computer Science*, pages 348–357, Edinburgh, Scotland, 1988. IEEE Computer Society.
29. K. Mukai. *Constraint Logic Programming and the Unification of Information*. PhD thesis, Department of Computer Science, Faculty of Engineering, Tokio Institute of Technology, 1991.
30. C. Pollard and I. A. Sag. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, 1994.
31. F. Scozzari. Abstract domains for sharing analysis by optimal semantics. In J. Palsberg, editor, *Static Analysis: 7th International Symposium, SAS 2000*, volume 1824 of *Lecture Notes in Computer Science*, pages 397–412, Santa Barbara, CA, USA, 2000. Springer-Verlag, Berlin.
32. Gert Smolka and Ralf Treinen. Records for logic programming. *Journal of Logic Programming*, 18(3):229–258, 1994.
33. H. Søndergaard. An application of abstract interpretation of logic programs: Occur check reduction. In B. Robinet and R. Wilhelm, editors, *Proceedings of the 1986 European Symposium on Programming*, volume 213 of *Lecture Notes in Computer Science*, pages 327–338. Springer-Verlag, Berlin, 1986.
34. Swedish Institute of Computer Science, Programming Systems Group. *SICStus Prolog User's Manual*, release 3 #0 edition, 1995.

Appendices

Appendix A provides a candidate implementation of the domain parameter P, showing how the functions and predicates specified in Definition 4 can be actually mapped into finitely computable operations over a well-known abstract domain for sharing analysis.

Appendix B provides the proofs of the results stated in the paper. Section B.1 introduces the notations and preliminary concepts that are subsequently used in the proofs; Section B.2 recalls a few general results holding for (syntactic) equality theories; the definition of (strongly) variable idempotent substitutions is given in Section B.3, together with some properties holding for them; these are then used in Section B.4 to prove Proposition 1 and Theorem 1; finally, in Section B.5 we provide the proof of Theorem 2.

This appendix has a separate bibliography on page 48.

A An implementation for the parameter domain *P*

As discussed in Section 3, several abstract domains for sharing analysis can be used to implement the parameter component P. As a basic implementation, one could consider the well-known set-sharing domain of Jacobs and Langen [JL89]. In such a case, all the non-trivial correctness results have already been established in [HBZ01]: in particular, the abstraction function provided in [HBZ01] satisfies the requirement of Definition 3 and the abstract unification operator has been proven correct with respect to rational-tree unification. Note however that, since no freeness and linearity information is recorded in the plain set-sharing domain, some of the predicates of Definition 4 need to be grossly approximated. For instance, the predicate gfree_p will provide useful information only when applied to an argument that is known to be definitely ground.

Therefore, in order to better highlight the generality of our specification of the sharing component P, we now instantiate it to the abstract domain *SFL* [BZH00] (see also [BCM94]). The domain *SFL* represents sharing, freeness and linearity information. Possible sharing is encoded by the set-sharing domain of Jacobs and Langen. Definite freeness and linearity information are each encoded by the set of variables of interest that enjoy the corresponding property.

Definition 9. (The set-sharing domain *SH*.) *The set SH is defined by* $SH \stackrel{\text{def}}{=} \wp(SG)$*, where the set of* sharing groups *SG is defined as*

$$SG \stackrel{\text{def}}{=} \wp(VI) \setminus \{\varnothing\}.$$

SH is ordered by subset inclusion.

Definition 10. (The domain *SFL*.) *Let* $F \stackrel{\text{def}}{=} \wp(VI)$ *and* $L \stackrel{\text{def}}{=} \wp(VI)$ *be partially ordered by reverse subset inclusion. The domain SFL is defined by the Cartesian product*

$$SFL \stackrel{\text{def}}{=} SH \times F \times L$$

ordered by '$\leq_S$', the component-wise extension of the orderings defined on the sub-domains.

A complete definition, besides explicitly dealing with the set of relevant variables VI, would require the addition of a bottom element $\bot$ representing the semantics of those program fragments that have no successful computations. We could also define an equivalence relation identifying the bottom element with all the elements in SFL corresponding to an impossible concrete computation state: for example, elements $\langle sh, f, l\rangle \in SFL$ such that $f \notin \mathrm{vars}(sh)$ (because a free variable does share with itself) or $VI \setminus \mathrm{vars}(sh) \notin l$ (because variables that cannot share are also linear). Note however that these spurious elements are never generated during the analysis process.

In the next definition we introduce a few well-known operations on the set-sharing domain SH. These will be used to define the operations on the domain SFL.

Definition 11. (Abstract operators on SH.) *For each $sh \in SH$ and each $V \subseteq VI$, the extraction of the* relevant component of sh with respect to V *is given by the function* $\mathrm{rel}\colon \wp(VI) \times SH \to SH$ *defined as*

$$\mathrm{rel}(V, sh) \stackrel{\mathrm{def}}{=} \{\, S \in sh \mid S \cap V \neq \varnothing \,\}.$$

For each $sh \in SH$ and each $V \subseteq VI$, the function $\overline{\mathrm{rel}}\colon \wp(VI) \times SH \to SH$ *gives the* irrelevant component of sh with respect to V. *It is defined as*

$$\overline{\mathrm{rel}}(V, sh) \stackrel{\mathrm{def}}{=} sh \setminus \mathrm{rel}(V, sh).$$

The function $(\cdot)^\star\colon SH \to SH$, *called* star-union, *is given, for each $sh \in SH$, by*

$$sh^\star \stackrel{\mathrm{def}}{=} \left\{ S \in SG \;\middle|\; \exists n \geq 1 \,.\, \exists T_1, \ldots, T_n \in sh \,.\, S = \bigcup_{i=1}^{n} T_i \right\}.$$

For each $sh_1, sh_2 \in SH$, the function $\mathrm{bin}\colon SH \times SH \to SH$, *called* binary union, *is given by*

$$\mathrm{bin}(sh_1, sh_2) \stackrel{\mathrm{def}}{=} \{\, S_1 \cup S_2 \mid S_1 \in sh_1, S_2 \in sh_2 \,\}.$$

It is now possible to define the implementation, on the domain SFL, of all the predicates and functions specified in Definition 4.

Definition 12. (Abstract operators on SFL.) *For each $d = \langle sh, f, l\rangle \in SFL$, for each $s, t \in HTerms$, where* $\mathrm{vars}(s) \cup \mathrm{vars}(t) \subseteq VI$, *let* $R_s = \mathrm{rel}\big(\mathrm{vars}(s), sh\big)$

and $R_t = \mathrm{rel}\big(\mathrm{vars}(t), sh\big)$. *Then*

$$
\begin{aligned}
\mathrm{ind}_d(s,t) &\stackrel{\mathrm{def}}{=} \big(R_s \cap R_t = \varnothing\big);\\
\mathrm{share\_lin}_d(s,t) &\stackrel{\mathrm{def}}{=} \forall y,z \in \mathrm{vars}(R_s \cap R_t) :\\
&\qquad (y \in l) \wedge \mathrm{occ\_lin}(y,s) \wedge \mathrm{occ\_lin}(y,t)\\
&\qquad\quad \wedge \big(y \neq z \implies \mathrm{ind}_d(y,z)\big);\\
\mathrm{ground}_d(t) &\stackrel{\mathrm{def}}{=} \big(\mathrm{vars}(t) \subseteq \mathit{VI} \setminus \mathrm{vars}(sh)\big);\\
\mathrm{free}_d(t) &\stackrel{\mathrm{def}}{=} \exists y \in \mathit{VI}\ .\ (y = t) \wedge (y \in f);\\
\mathrm{gfree}_d(t) &\stackrel{\mathrm{def}}{=} \mathrm{ground}_d(t) \vee \mathrm{free}_d(t);\\
\mathrm{lin}_d(t) &\stackrel{\mathrm{def}}{=} \forall y,z \in \mathrm{vars}(t) :\\
&\qquad (y \in l) \wedge \big(y \neq z \implies \mathrm{ind}_d(y,z)\big)\\
&\qquad\quad \wedge \big(\neg\mathrm{ground}_d(y) \implies \mathrm{occ\_lin}(y,t)\big);\\
\mathrm{or\_lin}_d(s,t) &\stackrel{\mathrm{def}}{=} \mathrm{lin}_d(s) \vee \mathrm{lin}_d(t);\\
\mathrm{share\_same\_var}_d(s,t) &\stackrel{\mathrm{def}}{=} \mathrm{vars}(R_s \cap R_t);\\
\mathrm{share\_with}_d(t) &\stackrel{\mathrm{def}}{=} \mathrm{vars}(R_t).
\end{aligned}
$$

The function $\mathrm{amgu}_S \colon \mathit{SFL} \times \mathit{Bind} \to \mathit{SFL}$ *captures the effects of a binding on an element of SFL. Let* $d = \langle sh, f, l\rangle \in \mathit{SFL}$ *and* $(x \mapsto t) \in \mathit{Bind}$, *where* $V_{xt} = \{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$. *Let* $R_x = \mathrm{rel}\big(\{x\}, sh\big)$ *and* $R_t = \mathrm{rel}\big(\mathrm{vars}(t), sh\big)$. *Let also*

$$
\begin{aligned}
sh' &\stackrel{\mathrm{def}}{=} \overline{\mathrm{rel}}(V_{xt}, sh) \cup \mathrm{bin}\big(S_x, S_t\big),\\
S_x &\stackrel{\mathrm{def}}{=} \begin{cases} R_x, & \textit{if } \mathrm{free}_d(x) \vee \mathrm{free}_d(t) \vee \big(\mathrm{lin}_d(t) \wedge \mathrm{ind}_d(x,t)\big);\\ R_x^\star, & \textit{otherwise;}\end{cases}\\
S_t &\stackrel{\mathrm{def}}{=} \begin{cases} R_t, & \textit{if } \mathrm{free}_d(x) \vee \mathrm{free}_d(t) \vee \big(\mathrm{lin}_d(x) \wedge \mathrm{ind}_d(x,t)\big);\\ R_t^\star, & \textit{otherwise;}\end{cases}\\
f' &\stackrel{\mathrm{def}}{=} \begin{cases} f, & \textit{if } \mathrm{free}_d(x) \wedge \mathrm{free}_d(t);\\ f \setminus \mathrm{vars}(R_x), & \textit{if } \mathrm{free}_d(x);\\ f \setminus \mathrm{vars}(R_t), & \textit{if } \mathrm{free}_d(t);\\ f \setminus \mathrm{vars}(R_x \cup R_t), & \textit{otherwise;}\end{cases}\\
l' &\stackrel{\mathrm{def}}{=} \big(\mathit{VI} \setminus \mathrm{vars}(sh')\big) \cup f' \cup l'';\\
l'' &\stackrel{\mathrm{def}}{=} \begin{cases} l \setminus \big(\mathrm{vars}(R_x) \cap \mathrm{vars}(R_t)\big), & \textit{if } \mathrm{lin}_d(x) \wedge \mathrm{lin}_d(t);\\ l \setminus \mathrm{vars}(R_x), & \textit{if } \mathrm{lin}_d(x);\\ l \setminus \mathrm{vars}(R_t), & \textit{if } \mathrm{lin}_d(t);\\ l \setminus \mathrm{vars}(R_x \cup R_t), & \textit{otherwise.}\end{cases}
\end{aligned}
$$

Then

$$\mathrm{amgu}_S\big(d, x \mapsto t\big) \stackrel{\text{def}}{=} \langle sh', f', l' \rangle.$$

As said in Section 4, further work includes proving that these definitions satisfy all the requirements of Definitions 3 and 4. Note however that the generic specification we have provided is not targeted at *SFL*. For instance, it can be seen that the above implementations of the predicates gfree_d and $\mathrm{or\_lin}_d$ cannot fully exploit the disjunctive nature of their specification as given in Definition 4. Thus, even more powerful sharing domains can fit in this schema, including all the enhanced combinations considered in [BZH00].

B Proofs

B.1 Notations and preliminaries for the proofs

To simplify the expressions in the paper, any variable in a formula that is not in the scope of a quantifier is assumed to be universally quantified.

The function size: $\mathit{HTerms} \to \mathbb{N}$ is defined, for each $t \in \mathit{HTerms}$, by

$$\mathrm{size}(t) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } t \in \mathit{Vars}; \\ 1 + \sum_{i=1}^{n} \mathrm{size}(t_i), & \text{if } t = f(t_1, \ldots, t_n), \text{ where } n \geq 0. \end{cases}$$

A substitution σ is *idempotent* if, for all $t \in \mathit{HTerms}$, we have $t\sigma\sigma = t\sigma$. The set of all idempotent substitutions is denoted by *ISubst* and $\mathit{ISubst} \subset \mathit{RSubst}$.

If $t \in \mathit{HTerms}$, we denote the set of variables that occur more than once in t by: $\mathrm{nlvars}(t) \stackrel{\text{def}}{=} \big\{\, y \in \mathrm{vars}(t) \;\big|\; \neg\mathrm{occ\_lin}(y, t) \,\big\}$.

If $\bar{s} = (s_1, \ldots, s_n) \in \mathit{HTerms}^n$ and $\bar{t} = (t_1, \ldots, t_n) \in \mathit{HTerms}^n$ are two tuples of finite terms, then we let $\bar{s} = \bar{t}$ denote the set of equations between corresponding components of $\bar{s}$ and $\bar{t}$. Namely,

$$(\bar{s} = \bar{t}) \stackrel{\text{def}}{=} \{s_i = t_i \mid 1 \leq i \leq n\}.$$

Moreover, we overload the functions mvars, occ_lin and nlvars to work on tuples of terms; thus, we will say that $\bar{s}$ is linear if and only if $\mathrm{nlvars}(\bar{s}) = \varnothing$.

Equality theories Let $\{s, t, s_1, \ldots, s_n, t_1, \ldots, t_m\} \subseteq \mathit{HTerms}$. We assume that any equality theory T over *Terms* includes the *congruence axioms* denoted by the following schemata:

$$s = s, \tag{3}$$

$$s = t \leftrightarrow t = s, \tag{4}$$

$$r = s \wedge s = t \rightarrow r = t, \tag{5}$$

$$s_1 = t_1 \wedge \cdots \wedge s_n = t_n \rightarrow f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n). \tag{6}$$

In logic programming and most implementations of Prolog it is usual to assume an equality theory based on syntactic identity. This consists of the congruence axioms together with the *identity axioms* denoted by the following schemata, where f and g are distinct function symbols or $n \neq m$:

$$f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n) \rightarrow s_1 = t_1 \wedge \cdots \wedge s_n = t_n, \tag{7}$$

$$\neg\big(f(s_1, \ldots, s_n) = g(t_1, \ldots, t_m)\big). \tag{8}$$

The axioms characterized by schemata (7) and (8) ensure the equality theory depends only on the syntax. The equality theory for a non-syntactic domain replaces these axioms by ones that depend instead on the semantics of the domain and, in particular, on the interpretation given to functor symbols.

The equality theory of Clark [Cla78] on which pure logic programming is based, usually called the *Herbrand* equality theory and denoted $\mathcal{FT}$, is given by the congruence axioms, the identity axioms, and the axiom schema

$$\forall z \in \mathit{Vars} : \forall t \in (\mathit{HTerms} \setminus \mathit{Vars}) : z \in \mathrm{vars}(t) \rightarrow \neg(z = t). \tag{9}$$

Axioms characterized by the schema (9) are called the *occur-check axioms* and are an essential part of the standard unification procedure in SLD-resolution.

An alternative approach used in some implementations of Prolog, does not require the occur-check axioms. This approach is based on the theory of rational trees $\mathcal{RT}$ [Col82, Col84]. It assumes the congruence axioms and the identity axioms together with a *uniqueness axiom* for each substitution in rational solved form. Informally speaking these state that, after assigning a ground rational tree to each parameter variable, the substitution uniquely defines a ground rational tree for each of its domain variables.

In the sequel we will use the expression "equality theory" to denote any consistent, decidable theory T satisfying the congruence axioms. We will also use the expression "syntactic equality theory" to denote any equality theory T also satisfying the identity axioms.[9] Note that both $\mathcal{FT}$ and $\mathcal{RT}$ are syntactic equality theories. When the equality theory T is clear from the context, it is convenient to adopt the notations $\sigma \implies \tau$ and $\sigma \iff \tau$, where σ, τ are sets of equations, to denote $T \vdash \forall(\sigma \rightarrow \tau)$ and $T \vdash \forall(\sigma \leftrightarrow \tau)$, respectively.

Given an equality theory T, and a set of equations in rational solved form σ, we say that σ is *satisfiable* in T if $T \vdash \forall\, \mathit{Vars} \setminus \mathrm{dom}(\sigma) : \exists\, \mathrm{dom}(\sigma)\ .\ \sigma$.

Given a satisfiable set of equations $e \in \wp_{\mathrm{f}}(\mathit{Eqs})$ in an equality theory T, then a substitution $\sigma \in \mathit{RSubst}$ is called a *solution for e in T* if σ is satisfiable in T and $T \vdash \forall(\sigma \rightarrow e)$. If $\mathrm{vars}(\sigma) \subseteq \mathrm{vars}(e)$, then σ is said to be a *relevant* solution for e. In addition, σ is a *most general solution for e in T* if $T \vdash \forall(\sigma \leftrightarrow e)$. In this paper, the set of all the relevant most general solution for e will be denoted by $\mathrm{mgs}(e)$.

[9] Note that, as a consequence of axiom (8) and the assumption that there are at least two distinct function symbols in the language, one of which is a constant, there exist two terms $a_1, a_2 \in \mathit{GTerms} \cap \mathit{HTerms}$ such that, for any syntactic equality theory T, we have $T \vdash a_1 \neq a_2$.

Observe that, given an arbitrary equality theory T, a set of equations in rational solved form may not be satisfiable in T. For example, $\exists x \,.\, \{x = f(x)\}$ is false in the Clark equality theory. However, by the uniqueness axioms, any set of equations in rational solved form is satisfiable in $\mathcal{RT}$.

B.2 General properties of equality theories.

Lemma 1. *Suppose T is an equality theory, $\sigma \in RSubst$ is satisfiable in T, $x \in Vars \setminus \mathrm{dom}(\sigma)$, and $t \in HTerms \cap GTerms$. Then, $\tau \stackrel{\mathrm{def}}{=} \sigma \cup \{x \mapsto t\} \in RSubst$ and τ is satisfiable in T.*

Proof. Proven in [HBZ01, Lemma 1]. □

Lemma 2. *Assume T is an equality theory and $\sigma \in RSubst$. Then, for each $t \in HTerms$,*

$$T \vdash \forall\big(\sigma \rightarrow (t = t\sigma)\big).$$

Proof. Proven in [HBZ01, Lemma 2]. □

Lemma 3. *Assume T is an equality theory and $\sigma \in RSubst$. Then, for each $s, t \in HTerms$,*

$$T \vdash \forall\big(\sigma \cup \{s = t\} \leftrightarrow \sigma \cup \{s = t\sigma\}\big).$$

Proof. First, note, using the congruence axioms (4) and (5), that, for any terms $p, q, r \in HTerms$,

$$T \vdash \forall(p = q \wedge q = r) \leftrightarrow \forall(p = r \wedge q = r). \tag{10}$$

Secondly note that, using Lemma 2, for any substitution $\tau \in RSubst$ and term $r \in HTerms$, $T \vdash \forall\big(\tau \rightarrow (r = r\tau)\big)$. Thus

$$T \vdash \forall\big(\tau \leftrightarrow \tau \cup \{r = r\tau\}\big). \tag{11}$$

Using these results,

$$\begin{aligned} T \vdash & \forall\big(\sigma \cup \{s = t\} \leftrightarrow \sigma \cup \{s = t, t = t\sigma\}\big), && \text{[by (11)]} \\ & \forall\big(\sigma \cup \{s = t\} \leftrightarrow \sigma \cup \{s = t\sigma, t = t\sigma\}\big), && \text{[by (10)]} \\ & \forall\big(\sigma \cup \{s = t\} \leftrightarrow \sigma \cup \{s = t\sigma\}\big). && \text{[by (11)]} \end{aligned}$$

□

Lemma 4. *Let T be a syntactic equality theory. Let $s \in HTerms \cap GTerms$ and $t \in HTerms$ be such that* $\mathrm{size}(t) > \mathrm{size}(s)$. *Then $T \vdash \forall(s \neq t)$.*

Proof. By induction on $m = \mathrm{size}(s)$. For the base case, when $m = 1$, we have that s is a term functor of arity 0. Since $\mathrm{size}(t) > 1$, then $t = f(t_1, \ldots, t_n)$, where $n > 0$. Then, by the identity axioms, we have $T \vdash \forall(s \neq t)$.

For the inductive case, when $m > 1$, assume that the result holds for all $m' < m$ and let $s = f(s_1, \ldots, s_n)$, where $n > 0$. Since $\mathrm{size}(t) > m$, we have $t = f'(t_1, \ldots, t_{n'})$, where $n' > 0$. If $f \neq f'$ or $n \neq n'$ then, by the identity axioms, we have $T \vdash \forall(s \neq t)$. Otherwise, let $f = f'$ and $n = n'$. Note that, for all $i \in \{1, \ldots, n\}$, we have $\mathrm{size}(s_i) < m$. Also, there exists an index $j \in \{1, \ldots, n\}$ such that $\mathrm{size}(t_j) > \mathrm{size}(s_j)$. By the inductive hypothesis, $T \vdash \forall(s_j \neq t_j)$ so that, by the identity axioms, $T \vdash \forall(s \neq t)$. □

B.3 (Strong) variable idempotence.

In [HBZ01], (weak) variable-idempotent substitutions were introduced as a subclass of substitutions in rational solved form in order to allow a more convenient reasoning about the sharing of variables for possibly non-idempotent substitutions.

Definition 13. (Variable-idempotence.) *A substitution $\sigma \in RSubst$ is said* variable-idempotent *if and only if for all $t \in HTerms$ we have*

$$\mathrm{vars}(t\sigma\sigma) \setminus \mathrm{dom}(\sigma) = \mathrm{vars}(t\sigma) \setminus \mathrm{dom}(\sigma).$$

The set of variable-idempotent substitutions is denoted VSubst.

In [HBZ98] a stronger definition was used, taking into consideration also the variables in the domain of the substitution. *Strong* variable-idempotence is a useful concept when dealing with the finiteness of a rational term and the multiplicity of variables occurring in it (i.e., its linearity).

Definition 14. (Strong variable-idempotence.) *A substitution $\sigma \in RSubst$ is* strongly variable-idempotent *if and only if for all $t \in HTerms$ we have*

$$\mathrm{vars}(t\sigma\sigma) = \mathrm{vars}(t\sigma).$$

The set of strongly variable-idempotent substitutions is denoted MSubst.

Note that we have $ISubst \subset MSubst \subset VSubst \subset RSubst$.

Definition 15. ($\mathcal{S}$-transformation.) *The relation $\overset{\mathcal{S}}{\longmapsto} \subseteq RSubst \times RSubst$, called* $\mathcal{S}$-step*, is defined by*

$$\frac{(x \mapsto t) \in \sigma \qquad (y \mapsto s) \in \sigma \qquad x \neq y}{\sigma \overset{\mathcal{S}}{\longmapsto} \big(\sigma \setminus \{y \mapsto s\}\big) \cup \{y \mapsto s[x/t]\}}.$$

If we have a finite sequence of $\mathcal{S}$-steps $\sigma_1 \overset{\mathcal{S}}{\longmapsto} \cdots \overset{\mathcal{S}}{\longmapsto} \sigma_n$ mapping σ_1 to σ_n, then we write $\sigma_1 \overset{\mathcal{S}}{\longmapsto}{}^* \sigma_n$ and say that σ_1 can be rewritten, by $\mathcal{S}$-transformation, to σ_n.

The following theorems show that considering substitutions in *MSubst* is not a restrictive hypothesis.

Theorem 3. *Suppose $\sigma \in RSubst$ and $\sigma \overset{\mathcal{S}}{\longmapsto}{}^{*} \sigma'$. Then we have $\sigma' \in RSubst$, $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$, $\mathrm{vars}(\sigma) = \mathrm{vars}(\sigma')$ and, if T is any equality theory, then $T \vdash \forall(\sigma \leftrightarrow \sigma')$.*

Proof. Proven in [HBZ01, Theorem 1]. □

Theorem 4. *Suppose $\sigma \in RSubst$. Then there exists $\sigma' \in MSubst$ such that $\sigma \overset{\mathcal{S}}{\longmapsto}{}^{*} \sigma'$ and, for all $\tau \subseteq \sigma'$, $\tau \in MSubst$.*

Proof. The proof is the same given for [HBZ01, Theorem 2], where a weaker result, using *VSubst*, was stated. □

Theorem 5. *Let T be an equality theory and $\sigma \in RSubst$. Then there exists $\sigma' \in MSubst$ such that $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$, $\mathrm{vars}(\sigma) = \mathrm{vars}(\sigma')$, $T \vdash \forall(\sigma \leftrightarrow \sigma')$ and for all $\tau \subseteq \sigma'$, $\tau \in MSubst$.*

Proof. The result easily follows from Theorems 3 and 4. □

B.4 Abstracting finiteness.

For a substitution $\sigma \in MSubst$, when computing the operator hvars the fixpoint is reached after a single iteration.

Lemma 5. *For each $\sigma \in MSubst$ we have $\mathrm{hvars}(\sigma) = \mathrm{hvars}_1(\sigma)$.*

Proof. We show that $\mathrm{hvars}_2(\sigma) \subseteq \mathrm{hvars}_1(\sigma)$. Let $y \in \mathrm{hvars}_2(\sigma)$. By Definition 5, we have two cases:

1. if $y \in \mathrm{hvars}_1(\sigma)$ then there is nothing to prove;
2. assume now $y \in \mathrm{dom}(\sigma)$ and $\mathrm{vars}(y\sigma) \subseteq \mathrm{hvars}_1(\sigma)$. By Definition 5, we have two subcases:
   (a) $\mathrm{vars}(y\sigma) \subseteq \mathit{Vars} \setminus \mathrm{dom}(\sigma)$.
   Then $\mathrm{vars}(y\sigma) \subseteq \mathrm{hvars}_0(\sigma)$, so that $y \in \mathrm{hvars}_1(\sigma)$;
   (b) $V = \mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) \neq \varnothing$ and, for all $z \in V$, $\mathrm{vars}(z\sigma) \cap V = \varnothing$.
   Let $z \in V$ so that $z \in \mathrm{vars}(y\sigma)$. By hypothesis, we have $\sigma \in MSubst$ so that $z \in \mathrm{vars}(y\sigma\sigma)$. As $z \in \mathrm{dom}(\sigma)$ and $\mathrm{vars}(z\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$, $z \notin \mathrm{vars}(z\sigma)$. This means that $z \notin \mathrm{vars}(y\sigma\sigma)$, which is a contradiction since $\sigma \in MSubst$. □

Proposition 2. *For each $\sigma \in MSubst$, we have*

$$\mathrm{hvars}(\sigma) = \big\{ y \in \mathit{Vars} \;\big|\; \mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing \big\}.$$

Proof. The result is obtained by applying Lemma 5 and then unfolding Definition 5. □

The following proposition shows that, for a substitution $\sigma \in MSubst$, the finiteness operator precisely captures the intended property.

Proposition 3. *Let $\sigma \in MSubst$ and $y \in Vars$. Then*

$$\mathrm{rt}(y,\sigma) \in HTerms \quad \Longleftrightarrow \quad y \in \mathrm{hvars}(\sigma).$$

Proof. Since $\sigma \in MSubst$, by Proposition 2 we have $y \in \mathrm{hvars}(\sigma)$ if and only if $\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$.

Let $\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$. Then, for any $i > 0$, we have $y\sigma^i = y\sigma \in HTerms$. Hence $\mathrm{rt}(y,\sigma) = y\sigma \in HTerms$.

In order to prove the other inclusion, let now $\mathrm{rt}(y,\sigma) \in HTerms$ so that, by definition, we have $\mathrm{size}\big(\mathrm{rt}(y,\sigma)\big) = n < \omega$. We will prove by contradiction that $\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$. In fact, assume that there exists a variable

$$z \in \mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma),$$

so that $(z \mapsto z\sigma) \in \sigma$. Since $\sigma \in MSubst$, we also have that $z \in \mathrm{vars}(y\sigma\sigma)$, hence $z \in \mathrm{vars}(z\sigma)$ and $\mathrm{size}(z\sigma) > 1$. As a consequence, for all $i > 0$, we have $\mathrm{size}(y\sigma^{i+1}) > \mathrm{size}(y\sigma^i)$. Therefore we obtain a contradiction, since there exists an index $j < \omega$ such that for all $i \geq j$ we have $\mathrm{size}(y\sigma^i) > n$. □

Proposition 4. *Let $\sigma \in MSubst$ and $r \in HTerms$ such that* $\mathrm{vars}(r) \subseteq \mathrm{hvars}(\sigma)$. *Then*

$$\mathrm{rt}(r,\sigma) = r\sigma,$$
$$\mathrm{vars}(r\sigma) \cap \mathrm{dom}(\sigma) = \varnothing.$$

Proof. Suppose $y \in \mathrm{vars}(r)$. Then, by Proposition 2, $\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$. Thus, for any $i > 0$, we have $y\sigma^i = y\sigma \in HTerms$. Thus $\mathrm{rt}(y,\sigma) = y\sigma$. As this holds for all $y \in \mathrm{vars}(r)$, it follows that $\mathrm{rt}(r,\sigma) = r\sigma$ and $\mathrm{vars}(r\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$. □

In order to prove Proposition 1, i.e., to show that the finiteness operator precisely captures the intended property even for arbitrary substitutions in *RSubst*, we now prove that this operator is invariant under the application of S-steps.

Lemma 6. *For each $m > 0$, we have* $\mathrm{hvars}_{m-1}(\sigma) \subseteq \mathrm{hvars}_m(\sigma)$.

Proof. Straightforward by Definition 5. □

Lemma 7. *Let $\sigma, \sigma' \in RSubst$ where $\sigma \overset{\mathcal{S}}{\longmapsto} \sigma'$. Then* $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\sigma')$.

Proof. Let $(x \mapsto t), (y \mapsto s) \in \sigma$, where $x \neq y$, such that

$$\sigma' = \big(\sigma \setminus \{y \mapsto s\}\big) \cup \big\{y \mapsto s[x/t]\big\}.$$

If $x \notin \mathrm{vars}(s)$ then we have $\sigma = \sigma'$ and the result trivially holds. Thus, we assume $x \in \mathrm{vars}(s)$. We prove the two inclusions separately.

In order to prove $\mathrm{hvars}(\sigma) \subseteq \mathrm{hvars}(\sigma')$ we show, by induction on $m \geq 0$, that we have

$$\mathrm{hvars}_m(\sigma) \subseteq \mathrm{hvars}_m(\sigma').$$

For the base case, when $m = 0$, by Theorem 3 we have $\mathrm{dom}(\sigma) = \mathrm{dom}(\sigma')$ so that

$$\begin{aligned}\mathrm{hvars}_0(\sigma) &= \mathit{Vars} \setminus \mathrm{dom}(\sigma)\\ &= \mathit{Vars} \setminus \mathrm{dom}(\sigma')\\ &= \mathrm{hvars}_0(\sigma').\end{aligned}$$

For the inductive step, when $m > 0$, assume $\mathrm{hvars}_{m-1}(\sigma) \subseteq \mathrm{hvars}_{m-1}(\sigma')$ and let $z \in \mathrm{hvars}_m(\sigma)$. By Definition 5, we have two cases: if $z \in \mathrm{hvars}_{m-1}(\sigma)$ then the result follows by a straight application of the inductive hypothesis; otherwise, we have

$$z \in \mathrm{dom}(\sigma) \wedge \mathrm{vars}(z\sigma) \subseteq \mathrm{hvars}_{m-1}(\sigma).$$

Now, if $z \neq y$ we have $z\sigma = z\sigma'$, so that, by Theorem 3 and the inductive hypothesis we have

$$z \in \mathrm{dom}(\sigma') \wedge \mathrm{vars}(z\sigma') \subseteq \mathrm{hvars}_{m-1}(\sigma'),$$

so that, by Definition 5, $z \in \mathrm{hvars}_m(\sigma')$. Otherwise, if $z = y$, then

$$\begin{aligned}\mathrm{vars}(z\sigma) &= \mathrm{vars}(s)\\ &\subseteq \mathrm{hvars}_{m-1}(\sigma).\end{aligned}$$

Since $x \in \mathrm{vars}(s)$,

$$\begin{aligned}\mathrm{vars}(z\sigma') &= \mathrm{vars}\big(s[x/t]\big)\\ &= \big(\mathrm{vars}(s) \setminus \{x\}\big) \cup \mathrm{vars}(t),\end{aligned}$$

and we need to show $\mathrm{vars}(z\sigma') \subseteq \mathrm{hvars}_{m-1}(\sigma')$. By the inductive hypothesis we have

$$\mathrm{vars}(s) \subseteq \mathrm{hvars}_{m-1}(\sigma');$$

Note that, since $x \in \mathrm{vars}(s)$, it follows $x \in \mathrm{hvars}_{m-1}(\sigma')$ so that, by Definition 5 and Lemma 6,

$$\begin{aligned}\mathrm{vars}(t) &\subseteq \mathrm{hvars}_{m-2}(\sigma')\\ &\subseteq \mathrm{hvars}_{m-1}(\sigma').\end{aligned}$$

In order to prove $\mathrm{hvars}(\sigma) \supseteq \mathrm{hvars}(\sigma')$ we show, by induction on $m \geq 0$, that we have

$$\mathrm{hvars}_{m+1}(\sigma) \supseteq \mathrm{hvars}_m(\sigma').$$

For the base case, when $m = 0$, by Lemma 6 and Theorem 3 we have

$$\begin{aligned}\mathrm{hvars}_1(\sigma) &\supseteq \mathrm{hvars}_0(\sigma)\\ &= \mathit{Vars} \setminus \mathrm{dom}(\sigma)\\ &= \mathit{Vars} \setminus \mathrm{dom}(\sigma')\\ &= \mathrm{hvars}_0(\sigma').\end{aligned}$$

For the inductive step, when $m > 0$, assume $\mathrm{hvars}_m(\sigma) \supseteq \mathrm{hvars}_{m-1}(\sigma')$ and let $z \in \mathrm{hvars}_m(\sigma')$. By Definition 5, we have two cases: if $z \in \mathrm{hvars}_{m-1}(\sigma')$ then the result follows by the inductive hypothesis and Lemma 6; otherwise, we have

$$z \in \mathrm{dom}(\sigma') \wedge \mathrm{vars}(z\sigma') \subseteq \mathrm{hvars}_{m-1}(\sigma').$$

Now, if $z \neq y$ we have $z\sigma = z\sigma'$, so that, by Theorem 3 and the inductive hypothesis we have

$$z \in \mathrm{dom}(\sigma) \wedge \mathrm{vars}(z\sigma) \subseteq \mathrm{hvars}_m(\sigma),$$

so that, by Definition 5, $z \in \mathrm{hvars}_{m+1}(\sigma)$. Otherwise, if $z = y$, by definition of σ', the inductive hypothesis and Lemma 6, we have

$$\begin{aligned}\mathrm{vars}(z\sigma') &= \mathrm{vars}\big(s[x/t]\big)\\ &= \big(\mathrm{vars}(s) \setminus \{x\}\big) \cup \mathrm{vars}(t)\\ &\subseteq \mathrm{hvars}_{m-1}(\sigma')\\ &\subseteq \mathrm{hvars}_m(\sigma)\\ &\subseteq \mathrm{hvars}_{m+1}(\sigma).\end{aligned}$$

Also note that we have

$$\begin{aligned}\mathrm{vars}(x\sigma) &= \mathrm{vars}(t)\\ &\subseteq \mathrm{hvars}_m(\sigma)\end{aligned}$$

so that, by Definition 5 we have

$$x \in \mathrm{hvars}_{m+1}(\sigma).$$

The result follows by observing that

$$\mathrm{vars}(z\sigma) = \mathrm{vars}(s) = \big(\mathrm{vars}(s) \setminus \{x\}\big) \cup \{x\}.$$

□

Lemma 8. *Let $\sigma, \sigma' \in \mathit{RSubst}$, where $\sigma \overset{\mathcal{S}}{\longmapsto}{}^{*} \sigma'$. Then* $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\sigma')$.

Proof. By induction on the length $n \geq 0$ of the derivation. For the base case, when $n = 0$, there is nothing to prove. Suppose now that

$$\sigma = \sigma_0 \overset{\mathcal{S}}{\longmapsto} \cdots \overset{\mathcal{S}}{\longmapsto} \sigma_{n-1} \overset{\mathcal{S}}{\longmapsto} \sigma_n = \sigma',$$

where $n > 1$. By the inductive hypothesis, since the derivation $\sigma \overset{\mathcal{S}}{\longmapsto}^* \sigma_{n-1}$ has length $n - 1$, we have $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\sigma_{n-1})$. Then the thesis follows by Lemma 7. □

Proof (Proof of Proposition 1.). By Theorem 5, there exists $\sigma' \in \mathit{MSubst}$ such that $\sigma \overset{\mathcal{S}}{\longmapsto}^* \sigma'$ and, for all equality theories T, $T \vdash \forall(\sigma \leftrightarrow \sigma')$. Thus, by Lemma 8, we have $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\sigma')$. The thesis then follows by applying Proposition 3. □

Lemma 9. *Let $\sigma, \tau \in \mathit{MSubst}$ be satisfiable in a syntactic equality theory T and suppose that $T \vdash \forall(\sigma \leftrightarrow \tau)$. Then* $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\tau)$.

Proof. We assume that the congruence and identity axioms hold. We will prove the inclusion $\mathrm{hvars}(\sigma) \subseteq \mathrm{hvars}(\tau)$, while the other inclusion will follow by symmetry.

Let $y \in \mathrm{hvars}(\sigma)$. Then, by Proposition 2, $\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing$. We will show that $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$ so that, again by Proposition 2, $y \in \mathrm{hvars}(\tau)$.

Take $a_1 \in \mathit{HTerms} \cap \mathit{GTerms}$ and let

$$\sigma' \overset{\text{def}}{=} \sigma \cup \big\{ z \mapsto a_1 \;\big|\; z \in \mathrm{vars}(y\sigma) \big\}.$$

Then, by Lemma 1, we have $\sigma' \in \mathit{RSubst}$ is satisfiable in T. By Lemma 2, we have $\sigma' \implies \{y = y\sigma\sigma'\}$ and, for all $i \geq 0$,

$$\sigma' \implies \sigma \implies \tau \implies \{y = y\tau^i\},$$

so that, by the congruence axioms, $\sigma' \implies \{y\sigma\sigma' = y\tau^i\}$.

Note that $y\sigma\sigma' \in \mathit{HTerms} \cap \mathit{GTerms}$, so that $\mathrm{size}(y\sigma\sigma') = n < \omega$.

Suppose that there exists $z \in \mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau)$. Then, since $\tau \in \mathit{MSubst}$, we have $z \in \mathrm{vars}(y\tau\tau)$, so that $z \in \mathrm{vars}(z\tau)$ and $\mathrm{size}(z\tau) > 1$. As a consequence, for all $i > 0$, we have $\mathrm{size}(y\tau^{i+1}) > \mathrm{size}(y\tau^i)$ and therefore there exists an index $j < \omega$ such that $\mathrm{size}(y\tau^j) > n$. By Lemma 4, $T \vdash \forall(y\sigma\sigma' \neq y\tau^j)$, therefore obtaining a contradiction. Thus $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$. □

Lemma 10. *Suppose $\sigma, \tau \in \mathit{RSubst}$ such that $T \vdash \forall(\sigma \leftrightarrow \tau)$ for any syntactic equality theory T. Then* $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\tau)$.

Proof. We assume that the congruence and the identity axioms hold. By Theorem 4 and Lemma 8, there exists $\sigma', \tau' \in \mathit{MSubst}$ such that $\sigma \iff \sigma'$, $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\sigma')$, $\tau \iff \tau'$ and $\mathrm{hvars}(\tau) = \mathrm{hvars}(\tau')$. By hypothesis, $\sigma \iff \tau$ so that $\sigma' \iff \tau'$. By Lemma 9, $\mathrm{hvars}(\sigma') = \mathrm{hvars}(\tau')$. Therefore $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\tau)$. □

Corollary 1. *Let $e \subseteq Eqs$ be satisfiable in the syntactic equality theory T. If $\sigma, \tau \in \mathrm{mgs}(e)$, then $\mathrm{hvars}(\sigma) = \mathrm{hvars}(\tau)$.*

Proof. By definition of mgs, we have $\sigma, \tau \in RSubst$ and $\sigma \iff e \iff \tau$. Thus, the result follows by Lemma 10. □

Proof (Proof of Theorem 1). By Definition 7, we have $\alpha_H(\sigma) = \mathrm{hvars}(\sigma) \cap VI$ and $\alpha_H(\sigma') = \mathrm{hvars}(\sigma') \cap VI$. Since $\mathcal{RT}$ is a syntactic equality theory and $\mathcal{RT} \vdash \forall(\sigma \leftrightarrow \sigma')$, the result is a simple consequence of Lemma 10. □

B.5 Correctness of abstract unification on $H \times P$

Lemma 11. *Let $\sigma \in MSubst$ be satisfiable in a syntactic equality theory T. Let $s \in HTerms \cap GTerms$ and $t \in HTerms$ and suppose that $T \vdash \forall(\sigma \rightarrow s = t)$. Then $s = t\sigma$.*

Proof. Since $s \in GTerms$, we must have $s = f(s_1, \ldots, s_m)$ where $m \geq 0$. Moreover, by the assumption of the existence of two different function symbols in the signature *Sig*, there exists a term $r \in HTerms \cap GTerms$ whose top-level function symbol is distinct from that in s. Thus, by the identity axioms, $T \vdash \forall(r \neq s)$. Note also that, by Lemma 2 and the congruence axioms, $T \vdash \forall(\sigma \rightarrow s = t\sigma)$.

We show that $s = t\sigma$ by induction on the size of s.

First we show that $t\sigma$ is not a variable. To prove this, we suppose that $t\sigma = y \in Vars$ and derive a contradiction. If $y \notin \mathrm{dom}(\sigma)$ then, by Lemma 1, $\sigma' = \sigma \cup \{y \mapsto r\} \in RSubst$ and σ' is satisfiable in T. Therefore, using the congruence axioms, $T \vdash \forall(\sigma' \rightarrow r = s)$, which is a contradiction. If otherwise $y \in \mathrm{dom}(\sigma)$ then, since $\sigma \in MSubst$, we have $y \in \mathrm{vars}(y\sigma)$ and $\mathrm{size}(y\sigma) > 1$, so that, for all $i \geq 0$, $\mathrm{size}(t\sigma^{i+1}) > \mathrm{size}(t\sigma^i)$. Thus, by Lemma 4, there exists an index $j > 0$ such that $T \vdash \forall(s \neq t\sigma^j)$. However, by the hypothesis and Lemma 2, we have $T \vdash \forall(\sigma \rightarrow s = t\sigma^i)$ for all $i \geq 0$, therefore obtaining a contradiction. Thus $t\sigma \notin Vars$.

Therefore, by the identity axioms, we can assume that $t\sigma = f(t_1, \ldots, t_m)$. If $\mathrm{size}(s) = 1$, then $m = 0$ so that, by the congruence axioms, $s = t\sigma$. If $\mathrm{size}(s) > 1$, then $m > 0$ and, by the identity axioms, we have, for each $i = 1, \ldots, m$, that $T \vdash \forall(\sigma \rightarrow s_i = t_i)$. Note that, for each $i = 1, \ldots, m$, we have $s_i \in HTerms \cap GTerms$, $t_i \in HTerms$ and $\mathrm{size}(s_i) < \mathrm{size}(s)$, so that we can apply the inductive hypothesis, obtaining $s_i = t_i\sigma$. Thus, by the congruence axioms, $s = t\sigma\sigma$. Thus $t\sigma\sigma \in GTerms$ so that $\mathrm{vars}(t\sigma\sigma) = \varnothing$. As $\sigma \in MSubst$, we have $\mathrm{vars}(t\sigma) = \varnothing$ so that $t\sigma\sigma = t\sigma$. Hence $s = t\sigma$. □

Lemma 12. *Let $\bar{s} = (s_1, \ldots, s_n) \in HTerms^n$ be linear, and suppose the tuple of terms $\bar{t} = (t_1, \ldots, t_n) \in HTerms^n$ is such that $\mathrm{vars}(\bar{s}) \cap \mathrm{nlvars}(\bar{t}) = \varnothing$ and $\mathrm{mgs}(\bar{s} = \bar{t}) \neq \varnothing$. Then there exists $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$ such that, for each variable $z \in \mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})\big)$, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing$.*

Proof. We assume that the congruence and identity axioms hold. The proof is by induction on the number of variables in $\mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$.

Suppose first that, for some $i = 1, \ldots, n$, we have $s_i = f(r_1, \ldots, r_m)$ and $t_i = f(u_1, \ldots, u_m)$ (with $m \geq 0$). Let

$$\bar{s}' \stackrel{\text{def}}{=} (s_1, \ldots, s_{i-1}, r_1, \ldots, r_m, s_{i+1}, \ldots, s_n),$$
$$\bar{t}' \stackrel{\text{def}}{=} (t_1, \ldots, t_{i-1}, u_1, \ldots, u_m, t_{i+1}, \ldots, t_n).$$

Then $\mathrm{mvars}(\bar{s}') = \mathrm{mvars}(\bar{s})$ and $\mathrm{mvars}(\bar{t}') = \mathrm{mvars}(\bar{t})$ so that, as $\bar{s}$ is linear, $\bar{s}'$ is linear, $\mathrm{vars}(\bar{s}') \cap \mathrm{nlvars}(\bar{t}') = \varnothing$ and $\mathrm{vars}(\bar{s}') \cap \mathrm{vars}(\bar{t}') = \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})$. Moreover, by the congruence axiom (6), $\mathrm{mgs}(\bar{s}' = \bar{t}') = \mathrm{mgs}(\bar{s} = \bar{t})$. (Note that in the case that s_i and t_i are identical constants, the equation $s_i = t_i$ is just removed.) Thus, as $\bar{s}$ and $\bar{t}$ are finite sequences of finite terms, we can assume that, for all $i = 1, \ldots, n$, either $s_i \in \mathit{Vars}$ or $t_i \in \mathit{Vars}$.

Secondly, suppose that for some $i = 1, \ldots, n$, $s_i = t_i$. By the previous paragraph, we can assume that $s_i \in \mathit{Vars}$. Let

$$\bar{s}_i \stackrel{\text{def}}{=} (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n),$$
$$\bar{t}_i \stackrel{\text{def}}{=} (t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n).$$

Then $\mathrm{mvars}(\bar{s}_i) \cup \{s_i\} = \mathrm{mvars}(\bar{s})$ and $\mathrm{mvars}(\bar{t}_i) \cup \{s_i\} = \mathrm{mvars}(\bar{t})$ so that, as $\bar{s}$ is linear, $\bar{s}_i$ is linear, $\mathrm{vars}(\bar{s}_i) \cap \mathrm{nlvars}(\bar{t}_i) = \varnothing$ and

$$\big(\mathrm{vars}(\bar{s}_i) \cap \mathrm{vars}(\bar{t}_i)\big) \cup \{s_i\} = \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t}).$$

As $\bar{s}$ is linear and $\mathrm{vars}(\bar{s}) \cap \mathrm{nlvars}(\bar{t}) = \varnothing$, $s_i \notin \mathrm{vars}(\bar{s}_i) \cup \mathrm{vars}(\bar{t}_i)$ and hence $s_i \notin \mathrm{dom}(\mu)$ for all $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$. Therefore

$$\mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})\big) = \mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(\bar{s}_i) \cap \mathrm{vars}(\bar{t}_i)\big).$$

Furthermore, by the congruence axiom (3), $\mathrm{mgs}(\bar{s}_i = \bar{t}_i) = \mathrm{mgs}(\bar{s} = \bar{t})$. Thus, as $\bar{s}$ and $\bar{t}$ are sequences of finite length n, we can assume that $s_i \neq t_i$, for all $i = 1, \ldots, n$.

Therefore, for the rest of the proof, we will assume that for each $i = 1, \ldots, n$, $s_i \neq t_i$ and either $s_i \in \mathit{Vars}$ or $t_i \in \mathit{Vars}$.

For the base case, we have $\mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t}) = \varnothing$ and the result holds.

For the inductive step, $\mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t}) \neq \varnothing$ so that $n > 0$. As the order of the equations in $\bar{s} = \bar{t}$ is not relevant to the hypothesis, we assume, without loss of generality that if, for some $i = 1, \ldots, n$, $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) = \varnothing$, then $\mathrm{vars}(s_1) \cap \mathrm{vars}(t_1) = \varnothing$. There are three cases we consider separately:

a. for all $i = 1, \ldots, n$, $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) \neq \varnothing$;
b. $s_1 \in \mathit{Vars} \setminus \mathrm{vars}(t_1)$;
c. $t_1 \in \mathit{Vars} \setminus \mathrm{vars}(s_1)$.

Case a. For all $i = 1, \ldots, n$, $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) \neq \varnothing$.

For each $i = 1, \ldots, n$, we are assuming that either $s_i \in \mathit{Vars}$ or $t_i \in \mathit{Vars}$, Therefore, for each $i = 1, \ldots, n$, $s_i \in \mathrm{vars}(t_i)$ or $t_i \in \mathrm{vars}(s_i)$ so that, without

loss of generality, we can assume, for some k where $0 \leq k \leq n$, $s_i \in \mathit{Vars}$ if $1 \leq i \leq k$ and $t_i \in \mathit{Vars}$ if $k+1 \leq i \leq n$.

Let

$$\mu \stackrel{\text{def}}{=} \{s_1 = t_1, \ldots, s_k = t_k\} \cup \{t_{k+1} = s_{k+1}, \ldots, t_n = s_n\}.$$

We now show that $\mu \subseteq \mathit{Eqs}$ is in rational solved form. As $\bar{s}$ is linear, $(s_1, \ldots, s_k)$ is linear. As $\bar{s}$ is linear and $t_i \in \mathrm{vars}(s_i)$ if $k+1 \leq i \leq n$, then $(t_{k+1}, \ldots, t_n)$ is linear and $\{s_1, \ldots, s_k\} \cap \{t_{k+1}, \ldots, t_n\} = \varnothing$. As we are assuming that, for all $i = 1, \ldots, n$, $s_i \neq t_i$ and $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) \neq \varnothing$, it follows that $t_i \notin \mathit{Vars}$ when $1 \leq i \leq k$ and $s_i \notin \mathit{Vars}$ when $k+1 \leq i \leq n$, so that each equation in μ is a binding and μ has no circular subsets. Thus $\mu \in \mathit{RSubst}$ and hence, by the congruence axiom (4), $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

As $s_i \in \mathrm{vars}(t_i)$ when $1 \leq i \leq k$ and $t_i \in \mathrm{vars}(s_i)$ when $k+1 \leq i \leq n$, $\mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})\big) = \varnothing$. Therefore the required result holds.

Case b. $s_1 \in \mathit{Vars} \setminus \mathrm{vars}(t_1)$.

Let

$$\begin{aligned} \bar{s}_1 &\stackrel{\text{def}}{=} (s_2, \ldots, s_n), \\ \bar{t}_1 &\stackrel{\text{def}}{=} \big(t_2[s_1/t_1], \ldots, t_n[s_1/t_1]\big). \end{aligned} \tag{12}$$

As $\bar{s}$ is linear, $s_1 \notin \mathrm{vars}(\bar{s}_1)$. Also, all occurrences of s_1 in $\bar{t}$ are replaced in $\bar{t}_1$ by t_1 so that, as $s_1 \notin \mathrm{vars}(t_1)$, $s_1 \notin \mathrm{vars}(\bar{t}_1)$. Thus

$$s_1 \notin \mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1). \tag{13}$$

Therefore $\mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1) \subset \mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$. Now since $\bar{s}$ is linear, $\bar{s}_1$ is linear. Thus, to apply the inductive hypothesis to $\bar{s}_1$ and $\bar{t}_1$, we have to show that

$$\mathrm{vars}(\bar{s}_1) \cap \mathrm{nlvars}(\bar{t}_1) = \varnothing. \tag{14}$$

Suppose that $u \in \mathrm{vars}(\bar{s}_1)$ so that $u \in \mathrm{vars}(\bar{s})$. Now, by hypothesis, we have $\mathrm{vars}(\bar{s}) \cap \mathrm{nlvars}(\bar{t}) = \varnothing$. Thus $s_1, u \notin \mathrm{nlvars}(\bar{t})$. If $u \in \mathrm{vars}\big((t_2, \ldots, t_n)\big)$ so that $u \notin \mathrm{vars}(t_1)$, then $u \notin \mathrm{nlvars}(\bar{t}_1)$. On the other hand, if $u \notin \mathrm{vars}\big((t_2, \ldots, t_n)\big)$, then, as $s_1 \notin \mathrm{nlvars}\big((t_2, \ldots, t_n)\big)$ and $u \notin \mathrm{nlvars}(t_1)$, $u \notin \mathrm{nlvars}(\bar{t}_1)$. Thus, for all $u \in \mathrm{vars}(\bar{s}_1)$, $u \notin \mathrm{nlvars}(\bar{t}_1)$. Hence (14) holds. It follows that the inductive hypothesis for $\bar{s}_1$ and $\bar{t}_1$ holds. Therefore there exists $\mu_1 \in \mathit{RSubst}$ where

$$\mu_1 \in \mathrm{mgs}(\bar{s}_1 = \bar{t}_1)$$

such that, for each $z \in \mathrm{dom}(\mu_1) \setminus \big(\mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)\big)$, $\mathrm{vars}(z\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$.

Let

$$\mu \stackrel{\text{def}}{=} \{s_1 = t_1\mu_1\} \cup \mu_1. \tag{15}$$

We now show that $\mu \subseteq \mathit{Eqs}$ is in $\mathrm{mgs}(\bar{s} = \bar{t})$. First we show that μ is in rational solved form. By (13),

$$s_1 \notin \mathrm{vars}(\mu_1), \tag{16}$$

and, as $s_1 \notin \mathrm{vars}(t_1)$, we have

$$s_1 \notin \mathrm{vars}(t_1\mu_1). \tag{17}$$

Thus, as $\mu_1 \in \mathit{RSubst}$, μ has no identities or circular subsets so that $\mu \in \mathit{RSubst}$. By Lemma 3, $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

Let

$$z \in \mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})\big). \tag{18}$$

Then we have to show that

$$\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing. \tag{19}$$

It follows from (15) and (18) that either $z \in \mathrm{dom}(\mu_1)$ so that $z\mu = z\mu_1$ or $z = s_1$ and $z\mu = t_1\mu_1$. We consider these two cases separately.

Suppose first that $z \in \mathrm{dom}(\mu_1)$. By (12), we have $\mathrm{vars}(\bar{s}_1) \subseteq \mathrm{vars}(\bar{s})$ and $\mathrm{vars}(\bar{t}_1) \subseteq \mathrm{vars}(\bar{t})$ so that $\mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1) \subseteq \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})$. Therefore, $z \in \mathrm{dom}(\mu_1) \setminus \big(\mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)\big)$. Thus, by the inductive hypothesis, we have $\mathrm{vars}(z\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Now, as $z \in \mathrm{dom}(\mu_1)$ and (16) holds, $s_1 \notin \mathrm{vars}(z\mu_1)$. Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{s_1\}$, $\mathrm{vars}(z\mu_1) \cap \mathrm{dom}(\mu) = \varnothing$. Hence, as $z\mu = z\mu_1$, (19) holds.

Secondly suppose that $z = s_1$. Then we have that $s_1 \notin \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})$. Hence $\bar{t}_1 = (t_2, \ldots, t_n)$. Let u be any variable in $\mathrm{vars}(t_1)$. Then we have that $u \notin \mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)$, since $\mathrm{vars}(\bar{s}) \cap \mathrm{nlvars}(\bar{t}) = \varnothing$, If $u \in \mathrm{dom}(\mu_1)$, then we can apply the inductive hypothesis to obtain $\mathrm{vars}(u\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. On the other hand, if $u \notin \mathrm{dom}(\mu_1)$, we have $u = u\mu_1$ and $\mathrm{vars}(u\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Hence $\mathrm{vars}(t_1\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{s_1\}$, by (17), $\mathrm{vars}(t_1\mu_1) \cap \mathrm{dom}(\mu) = \varnothing$. Therefore, as $z\mu = t_1\mu_1$, (19) holds.

Case c. $t_1 \in \mathit{Vars} \setminus \mathrm{vars}(s_1)$.

Let

$$\begin{aligned} \bar{s}_1 &\stackrel{\mathrm{def}}{=} \big(s_2[t_1/s_1], \ldots, s_n[t_1/s_1]\big), \\ \bar{t}_1 &\stackrel{\mathrm{def}}{=} \big(t_2[t_1/s_1], \ldots, t_n[t_1/s_1]\big). \end{aligned} \tag{20}$$

All occurrences of t_1 in $\bar{s}$ and $\bar{t}$ are replaced in $\bar{s}_1$ and $\bar{t}_1$ by s_1 so that, as $t_1 \notin \mathrm{vars}(s_1)$,

$$t_1 \notin \mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1). \tag{21}$$

Therefore $\mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1) \subset \mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$. Now, $\bar{s}_1$ is linear since $\bar{s}$ is linear. Thus, to apply the inductive hypothesis to $\bar{s}_1$ and $\bar{t}_1$, we have to show that

$$\mathrm{vars}(\bar{s}_1) \cap \mathrm{nlvars}(\bar{t}_1) = \varnothing. \tag{22}$$

Suppose u is any variable in $\mathrm{vars}(\bar{s}_1)$. Then either $u \in \mathrm{vars}\big((s_2, \ldots, s_n)\big)$ or $u \in \mathrm{vars}(s_1)$ and $t_1 \in \mathrm{vars}\big((s_2, \ldots, s_n)\big)$. By hypothesis, $\mathrm{vars}(\bar{s}) \cap \mathrm{nlvars}(\bar{t}) = \varnothing$, so that $u \notin \mathrm{nlvars}(\bar{t})$. If $u \in \mathrm{vars}\big((s_2, \ldots, s_n)\big)$, then, as $\bar{s}$ is linear, $u \notin \mathrm{vars}(s_1)$. Thus, it follows from (20) that $u \notin \mathrm{nlvars}(\bar{t}_1)$. If $t_1 \in \mathrm{vars}\big((s_2, \ldots, s_n)\big)$, then

$t_1 \notin \mathrm{vars}\big((t_2,\ldots,t_n)\big)$ so that, again by (20), $\bar{t}_1 = (t_2,\ldots,t_n)$. Thus, for all $u \in \mathrm{vars}(\bar{s}_1)$, $u \notin \mathrm{nlvars}(\bar{t}_1)$. Hence (22) holds. It follows that the inductive hypothesis for $\bar{s}_1$ and $\bar{t}_1$ holds. Therefore there exists $\mu_1 \in \mathit{RSubst}$ where

$$\mu_1 \in \mathrm{mgs}(\bar{s}_1 = \bar{t}_1)$$

such that, for each $z \in \mathrm{dom}(\mu_1) \setminus \big(\mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)\big)$, $\mathrm{vars}(z\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$.

Let

$$\mu \stackrel{\mathrm{def}}{=} \{t_1 = s_1\mu_1\} \cup \mu_1. \tag{23}$$

We now show that $\mu \subseteq \mathit{Eqs}$ is in $\mathrm{mgs}(\bar{s} = \bar{t})$. First we show that μ is in rational solved form. By (21),

$$t_1 \notin \mathrm{vars}(\mu_1), \tag{24}$$

and, as $t_1 \notin \mathrm{vars}(s_1)$, we have

$$t_1 \notin \mathrm{vars}(s_1\mu_1). \tag{25}$$

Thus, as $\mu_1 \in \mathit{RSubst}$, μ has no identities or circular subsets so that $\mu \in \mathit{RSubst}$. By Lemma 3, $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

Let

$$z \in \mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})\big). \tag{26}$$

Then we have to show that

$$\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing. \tag{27}$$

It follows from (23) and (26) that either $z \in \mathrm{dom}(\mu_1)$ so that $z\mu = z\mu_1$ or $z = t_1$ and $z\mu = s_1\mu_1$. We consider these two cases separately.

Suppose first that $z \in \mathrm{dom}(\mu_1)$. To apply the inductive hypothesis to z, we need to show that,

$$\mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1) \subseteq \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t}).$$

To see this, let $u \in \mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)$. Then, by (20), either $u \in \mathrm{vars}\big((s_2,\ldots,s_n)\big)$ or $u \in \mathrm{vars}(s_1)$ and $t_1 \in \mathrm{vars}\big((s_2,\ldots,s_n)\big)$. If $u \in \mathrm{vars}\big((s_2,\ldots,s_n)\big)$, then we have $u \in \mathrm{vars}(\bar{s})$ so that, as $\bar{s}$ is linear, we have also $u \notin \mathrm{vars}(s_1)$ and hence $u \in \mathrm{vars}\big((t_2,\ldots,t_n)\big)$. Alternatively, if $u \in \mathrm{vars}(s_1)$ and $t_1 \in \mathrm{vars}\big((s_2,\ldots,s_n)\big)$, then $u, t_1 \in \mathrm{vars}(\bar{s})$. Moreover, by hypothesis, $\mathrm{vars}(\bar{s}) \cap \mathrm{nlvars}(\bar{t}) = \varnothing$, so that $t_1 \notin \mathrm{vars}\big((t_2,\ldots,t_n)\big)$. Thus $\bar{t}_1 = (t_2,\ldots,t_n)$ and hence $u \in \mathrm{vars}(\bar{t})$. Therefore, in both cases, $u \in \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})$. It follows that $z \in \mathrm{dom}(\mu_1) \setminus \big(\mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)\big)$. Thus, by the inductive hypothesis, we have $\mathrm{vars}(z\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Now, as $z \in \mathrm{dom}(\mu_1)$ and (24) holds, $t_1 \notin \mathrm{vars}(z\mu_1)$. Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{t_1\}$, $\mathrm{vars}(z\mu_1) \cap \mathrm{dom}(\mu) = \varnothing$. Hence, as $z\mu = z\mu_1$, (27) holds.

Secondly, suppose that $z = t_1$. Then $t_1 \notin \mathrm{vars}(\bar{s}) \cap \mathrm{vars}(\bar{t})$ and, consequently, $\bar{s}_1 = (s_2,\ldots,s_n)$. Let u be any variable in $\mathrm{vars}(s_1)$. Then, as $\bar{s}$ is linear, we have $u \notin \mathrm{vars}(\bar{s}_1)$ so that $u \notin \mathrm{vars}(\bar{s}_1) \cap \mathrm{vars}(\bar{t}_1)$. Thus, if $u \in \mathrm{dom}(\mu_1)$, we

can apply the inductive hypothesis to u and obtain $\mathrm{vars}(u\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. On the other hand, if $u \notin \mathrm{dom}(\mu_1)$, $u = u\mu_1$ and $\mathrm{vars}(u\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Hence $\mathrm{vars}(s_1\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{t_1\}$, by (25), $\mathrm{vars}(s_1\mu_1) \cap \mathrm{dom}(\mu) = \varnothing$. Therefore, as $z\mu = s_1\mu_1$, (27) holds. □

Lemma 13. *Suppose that the tuple of terms $\bar{s} = (s_1, \ldots, s_n) \in HTerms^n$ is linear, $\bar{t} = (t_1, \ldots, t_n) \in HTerms^n$ and $\mathrm{mgs}(\bar{s} = \bar{t}) \neq \varnothing$. Then there exists $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$ and, for each $z \in \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s})$, the following properties hold:*

1. $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s})$;
2. $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing$.

Proof. We assume that the congruence and identity axioms hold. The proof is by induction on the number of variables in $\mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$.

Suppose first that, for some $i = 1, \ldots, n$, we have $s_i = f(r_1, \ldots, r_m)$ and $t_i = f(u_1, \ldots, u_m)$ $(m \geq 0)$. Let

$$\bar{s}' \stackrel{\mathrm{def}}{=} (s_1, \ldots, s_{i-1}, r_1, \ldots, r_m, s_{i+1}, \ldots, s_n),$$
$$\bar{t}' \stackrel{\mathrm{def}}{=} (t_1, \ldots, t_{i-1}, u_1, \ldots, u_m, t_{i+1}, \ldots, t_n).$$

Then $\mathrm{mvars}(\bar{s}') = \mathrm{mvars}(\bar{s})$ and $\mathrm{mvars}(\bar{t}') = \mathrm{mvars}(\bar{t})$ so that, as $\bar{s}$ is linear, $\bar{s}'$ is linear. Moreover, by the congruence axiom (6), $\mathrm{mgs}(\bar{s}' = \bar{t}') = \mathrm{mgs}(\bar{s} = \bar{t})$. (Note that in the case that s_i and t_i are identical constants, the equation $s_i = t_i$ is just removed.) Thus, as $\bar{s}$ and $\bar{t}$ are finite sequences of finite terms, we can assume that, for all $i = 1, \ldots, n$, either $s_i \in \mathit{Vars}$ or $t_i \in \mathit{Vars}$.

Secondly, suppose that for some $i = 1, \ldots, n$, $s_i = t_i$. By the previous paragraph, we can assume that $s_i \in \mathit{Vars}$. Let

$$\bar{s}_i \stackrel{\mathrm{def}}{=} (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n),$$
$$\bar{t}_i \stackrel{\mathrm{def}}{=} (t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n).$$

Then $\mathrm{mvars}(\bar{s}_i) \cup \{s_i\} = \mathrm{mvars}(\bar{s})$ and $\mathrm{mvars}(\bar{t}_i) \cup \{s_i\} = \mathrm{mvars}(\bar{t})$ so that, as $\bar{s}$ is linear, $\bar{s}_i$ is linear. Therefore

$$\mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s}) \subseteq \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s}_i).$$

Furthermore, by the congruence axiom (3), $\mathrm{mgs}(\bar{s}_i = \bar{t}_i) = \mathrm{mgs}(\bar{s} = \bar{t})$. Thus, as $\bar{s}$ and $\bar{t}$ are sequences of finite length n, we can assume that $s_i \neq t_i$, for all $i = 1, \ldots, n$.

Therefore, for the rest of the proof, we will assume that $s_i \neq t_i$ and either $s_i \in \mathit{Vars}$ or $t_i \in \mathit{Vars}$, for all $i = 1, \ldots, n$.

For the base case, we have $\mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t}) = \varnothing$ and the result holds.

For the inductive step, $\mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t}) \neq \varnothing$ so that $n > 0$. As the order of the equations in $\bar{s} = \bar{t}$ is not relevant to the hypothesis, we assume, without loss of generality that if, for some $i = 1, \ldots, n$, $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) = \varnothing$ then, we have $\mathrm{vars}(s_1) \cap \mathrm{vars}(t_1) = \varnothing$. There are four cases we consider separately:

a. for all $i = 1, \ldots, n$, $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) \neq \varnothing$;
b. $s_1 \in \mathit{Vars} \setminus \mathrm{vars}(t_1)$;
c. $t_1 \in \mathit{Vars} \setminus \mathrm{vars}(\bar{s})$ and $s_1 \notin \mathit{Vars}$;
d. $t_1 \in \mathrm{vars}(\bar{s}) \setminus \mathrm{vars}(s_1)$ and $s_1 \notin \mathit{Vars}$.

Case a. For all $i = 1, \ldots, n$, $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) \neq \varnothing$.

For each $i = 1, \ldots, n$, we are assuming that either $s_i \in \mathit{Vars}$ or $t_i \in \mathit{Vars}$, Therefore, for each $i = 1, \ldots, n$, $s_i \in \mathrm{vars}(t_i)$ or $t_i \in \mathrm{vars}(s_i)$ so that, without loss of generality, we can assume, for some k where $0 \leq k \leq n$, $s_i \in \mathit{Vars}$ if $1 \leq i \leq k$ and $t_i \in \mathit{Vars}$ if $k+1 \leq i \leq n$.

Let

$$\mu \stackrel{\text{def}}{=} \{s_1 = t_1, \ldots, s_k = t_k\} \cup \{t_{k+1} = s_{k+1}, \ldots, t_n = s_n\}.$$

We show that $\mu \subseteq \mathit{Eqs}$ is in $\mathrm{mgs}(\bar{s} = \bar{t})$. First we must show that $\mu \in \mathit{RSubst}$. As $\bar{s}$ is linear, $(s_1, \ldots, s_k)$ is linear. As $\bar{s}$ is linear and $t_i \in \mathrm{vars}(s_i)$ if $k+1 \leq i \leq n$, then $(t_{k+1}, \ldots, t_n)$ is linear and $\{s_1, \ldots, s_k\} \cap \{t_{k+1}, \ldots, t_n\} = \varnothing$. As we are assuming that, for all $i = 1, \ldots, n$, $s_i \neq t_i$ and $\mathrm{vars}(s_i) \cap \mathrm{vars}(t_i) \neq \varnothing$, it follows that $t_i \notin \mathit{Vars}$ when $1 \leq i \leq k$ and $s_i \notin \mathit{Vars}$ when $k+1 \leq i \leq n$, so that each equation in μ is a binding and μ has no circular subsets. Thus $\mu \in \mathit{RSubst}$ and hence, by the congruence axiom (4), $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

As $\{t_{k+1}, \ldots, t_n\} \subseteq \mathrm{vars}\big((s_{k+1}, \ldots, s_n)\big)$, $\mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s}) = \varnothing$. Therefore the required result holds.

Case b. $s_1 \in \mathit{Vars} \setminus \mathrm{vars}(t_1)$.

Let

$$\bar{s}_1 \stackrel{\text{def}}{=} (s_2, \ldots, s_n),$$
$$\bar{t}_1 \stackrel{\text{def}}{=} \big(t_2[s_1/t_1], \ldots, t_n[s_1/t_1]\big).$$

As $\bar{s}$ is linear, $\bar{s}_1$ is linear and $s_1 \notin \mathrm{vars}(\bar{s}_1)$. Also, all occurrences of s_1 in $\bar{t}$ are replaced in $\bar{t}_1$ by t_1 so that, as $s_1 \notin \mathrm{vars}(t_1)$ (by the assumption for this case), $s_1 \notin \mathrm{vars}(\bar{t}_1)$. Thus

$$s_1 \notin \mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1). \tag{28}$$

It follows that $\mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1) \subset \mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$ so that the inductive hypothesis applies to $\bar{s}_1$ and $\bar{t}_1$. Thus there exists $\mu_1 \in \mathit{RSubst}$ where

$$\mu_1 \in \mathrm{mgs}(\bar{s}_1 = \bar{t}_1)$$

such that, for each $z \in \mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1)$, properties 1 and 2 hold using μ_1 and $\bar{s}_1$.

Let

$$\mu \stackrel{\text{def}}{=} \{s_1 = t_1\mu_1\} \cup \mu_1.$$

We show that $\mu \subseteq \mathit{Eqs}$ is in $\mathrm{mgs}(\bar{s} = \bar{t})$. By (28), we have $s_1 \notin \mathrm{vars}(\mu_1)$ so that $s_1 \notin \mathrm{dom}(\mu_1)$. Also, since $\mu_1 \in \mathit{RSubst}$, μ has no identities or circular subsets. Thus we have $\mu \in \mathit{RSubst}$. By Lemma 3, $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

Suppose that $z \in \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s})$. As

$$\mathrm{vars}(\bar{s}_1) \cup \{s_1\} = \mathrm{vars}(\bar{s})$$

and

$$\mathrm{dom}(\mu_1) \cup \{s_1\} = \mathrm{dom}(\mu),$$

we have

$$\mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1) = \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s}). \tag{29}$$

Therefore $z \in \mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1)$ and $z\mu_1 = z\mu$. Thus the inductive properties 1 and 2 using μ_1 and $\bar{s}_1$ can be applied to z. We show that properties 1 and 2 using μ and $\bar{s}$ can be applied to z.

1. By property 1, $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s}_1)$ and hence, $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s})$.
2. By property 2, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu_1) = \varnothing$. Now $s_1 \notin \mathrm{vars}(z\mu)$ because $s_1 \notin \mathrm{vars}(\bar{s}_1)$ (since $\bar{s}$ is linear) and $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s}_1)$ (by property 1). Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{s_1\}$, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing$.

Case c. Assume that $t_1 \in \mathit{Vars} \setminus \mathrm{vars}(\bar{s})$ and $s_1 \notin \mathit{Vars}$.
Let

$$\bar{s}_1 \stackrel{\mathrm{def}}{=} (s_2, \ldots, s_n),$$
$$\bar{t}_1 \stackrel{\mathrm{def}}{=} \big(t_2[t_1/s_1], \ldots, t_n[t_1/s_1]\big).$$

As $\bar{s}$ is linear, $\bar{s}_1$ is linear. By the assumption for this case, $t_1 \notin \mathrm{vars}(\bar{s}_1)$. Also, all occurrences of t_1 in $\bar{t}$ are replaced in $\bar{t}_1$ by s_1 so that $t_1 \notin \mathrm{vars}(\bar{t}_1)$. Thus

$$t_1 \notin \mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1). \tag{30}$$

It follows that $\mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1) \subset \mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$ so that we can apply the inductive hypothesis to $\bar{s}_1$ and $\bar{t}_1$. Thus there exists $\mu_1 \in \mathit{RSubst}$ where

$$\mu_1 \in \mathrm{mgs}(\bar{s}_1 = \bar{t}_1)$$

such that, for each $z \in \mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1)$, properties 1 and 2 hold using μ_1 and $\bar{s}_1$. Note that, by (30), $t_1 \notin \mathrm{vars}(\mu_1)$ and, in particular, $t_1 \notin \mathrm{dom}(\mu_1)$.

Let

$$\mu \stackrel{\mathrm{def}}{=} \{t_1 = s_1\mu_1\} \cup \mu_1. \tag{31}$$

As $s_1 \notin \mathit{Vars}$ and $\mu_1 \in \mathit{RSubst}$, $\mu \in \mathit{Eqs}$ has no identities or circular subsets so that $\mu \in \mathit{RSubst}$. By Lemma 3, $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

As $t_1 \in \mathrm{dom}(\mu)$ (by (31)) and $t_1 \notin \mathrm{vars}(\bar{s})$ (by the assumption for this case), we have

$$\mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1) \cup \{t_1\} = \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s}).$$

Suppose that $z \in \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s})$. Then either $z \neq t_1$ so that $z\mu = z\mu_1$ and the inductive properties 1 and 2 using μ_1 and $\bar{s}_1$ can be applied to z or $z = t_1$ and $z\mu = s_1\mu_1$. We show that properties 1 and 2 using μ and $\bar{s}$ can be applied to z.

1. Suppose $z \neq t_1$ so that $z\mu = z\mu_1$. Using property 1, $\mathrm{vars}(z\mu_1) \subseteq \mathrm{vars}(\bar{s}_1)$. As $\mathrm{vars}(\bar{s}_1) \subseteq \mathrm{vars}(\bar{s})$, it follows that $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s})$.
   Suppose that $z = t_1$ so that $z\mu = s_1\mu_1$. Let u be any variable in s_1. As $\bar{s}$ is linear, $u \notin \mathrm{vars}(\bar{s}_1)$. Thus, if $u \in \mathrm{dom}(\mu_1)$, we can use property 1 to derive that $\mathrm{vars}(u\mu_1) \subseteq \mathrm{vars}(\bar{s}_1)$. If $u \notin \mathrm{dom}(\mu_1)$, then $u\mu_1 = u$ so that $\mathrm{vars}(u\mu_1) \subseteq \mathrm{vars}(s_1)$. Moreover $\mathrm{vars}(s_1) \cup \mathrm{vars}(\bar{s}_1) = \mathrm{vars}(\bar{s})$ so that
   $$\mathrm{vars}(s_1\mu_1) \subseteq \mathrm{vars}(\bar{s}). \tag{32}$$
   Hence $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s})$.
2. Suppose $z \neq t_1$ so that $z\mu = z\mu_1$. Then, as property 2 holds, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu_1) = \varnothing$. Now $t_1 \notin \mathrm{vars}(z\mu)$ because $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s}_1)$ (by property 1) and $t_1 \notin \mathrm{vars}(\bar{s}_1)$ (by (30)). Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{t_1\}$, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing$.
   Suppose that $z = t_1$ so that $z\mu = s_1\mu_1$. Let u be any variable in $\mathrm{vars}(s_1)$. Then, as $\bar{s}$ is linear, $u \notin \mathrm{vars}(\bar{s}_1)$. Then either $u \in \mathrm{dom}(\mu_1)$, and we can apply property 2 to u to obtain $\mathrm{vars}(u\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$, or $u = u\mu_1$, and $\mathrm{vars}(u\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Hence we have $\mathrm{vars}(s_1\mu_1) \cap \mathrm{dom}(\mu_1) = \varnothing$. Now $t_1 \notin \mathrm{vars}(s_1\mu_1)$ because $\mathrm{vars}(s_1\mu_1) \subseteq \mathrm{vars}(\bar{s})$ (by (32)) and $t_1 \notin \mathrm{vars}(\bar{s})$ (by the assumption for this case). Thus, as $\mathrm{dom}(\mu) = \mathrm{dom}(\mu_1) \cup \{t_1\}$, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing$.

Case d. Assume that $t_1 \in \mathrm{vars}(\bar{s}) \setminus \mathrm{vars}(s_1)$ and $s_1 \notin \mathit{Vars}$.
Let

$$\bar{s}_1 \stackrel{\text{def}}{=} \big(s_2[t_1/s_1], \ldots, s_n[t_1/s_1]\big),$$
$$\bar{t}_1 \stackrel{\text{def}}{=} \big(t_2[t_1/s_1], \ldots, t_n[t_1/s_1]\big).$$

As $\bar{s}$ is linear, there is only one occurrence of t_1 in $\{s_2, \ldots, s_n\}$, and, in $\bar{s}_1$, this is replaced by s_1 which is also linear. Thus $\bar{s}_1$ is linear, $\bar{s}_1 \subseteq \bar{s}$ and $t_1 \notin \mathrm{vars}(\bar{s}_1)$. Also, all occurrences of t_1 in $\bar{t}$ are replaced in $\bar{t}_1$ by s_1 so that $t_1 \notin \mathrm{vars}(\bar{t}_1)$. Thus

$$t_1 \notin \mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1). \tag{33}$$

It follows that $\mathrm{vars}(\bar{s}_1) \cup \mathrm{vars}(\bar{t}_1) \subset \mathrm{vars}(\bar{s}) \cup \mathrm{vars}(\bar{t})$ so that we can apply the inductive hypothesis to $\bar{s}_1$ and $\bar{t}_1$. Thus, there exists $\mu_1 \in \mathit{RSubst}$ where

$$\mu_1 \in \mathrm{mgs}(\bar{s}_1 = \bar{t}_1)$$

such that, for each $z \in \mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1)$, properties 1 and 2 hold using μ_1 and $\bar{s}_1$.

Let

$$\mu \stackrel{\text{def}}{=} \{t_1 = s_1\mu_1\} \cup \mu_1.$$

By (33), $t_1 \notin \mathrm{vars}(\mu_1)$. Moreover $\mu_1 \in \mathit{RSubst}$ and $s_1 \notin \mathit{Vars}$ so that $\mu \in \mathit{Eqs}$ has no identities or circular subset. Thus $\mu \in \mathit{RSubst}$. By Lemma 3, $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$.

As $\mathrm{vars}(\bar{s}_1) \cup \{t_1\} = \mathrm{vars}(\bar{s})$ and $\mathrm{dom}(\mu_1) \cup \{t_1\} = \mathrm{dom}(\mu)$, we have

$$\mathrm{dom}(\mu_1) \setminus \mathrm{vars}(\bar{s}_1) = \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s}).$$

Suppose $z \in \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s})$. Then $z \neq t_1$, $z\mu = z\mu_1$ and the inductive properties 1 and 2 using μ_1 and $\bar{s}_1$ can be applied to z. We show that the properties 1 and 2 using μ and $\bar{s}$ can be applied to z.

1. By property 1, $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s}_1)$ and hence, as $\bar{s}_1 \subseteq \bar{s}$, $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s})$.
2. By property 2, we have $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu_1) = \varnothing$. Now $t_1 \notin \mathrm{vars}(z\mu)$ because $t_1 \notin \mathrm{vars}(\bar{s}_1)$ (by (33)) and $\mathrm{vars}(z\mu) \subseteq \mathrm{vars}(\bar{s}_1)$ (by property 1). As $\mathrm{dom}(\mu_1) \cup \{t_1\} = \mathrm{dom}(\mu)$, it follows that $\mathrm{vars}(z\mu) \cap \mathrm{dom}(\mu) = \varnothing$. □

Proposition 5. *Let $p \in P$ and $(x \mapsto t) \in Bind$, where $\{x\} \cup \mathrm{vars}(t) \subseteq VI$. Let also $\sigma \in \gamma_P(p) \cap MSubst$ and suppose that $\{r, r'\} = \{x, t\}$, $\mathrm{vars}(r) \subseteq \mathrm{hvars}(\sigma)$ and $\mathrm{rt}(r, \sigma) \in GTerms$. Then, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ in a syntactic equality theory T, we have*

$$\mathrm{hvars}(\sigma) \cup \mathrm{vars}(r') \subseteq \mathrm{hvars}(\tau). \tag{34}$$

Proof. We assume that the congruence and identity axioms hold. If $\sigma \cup \{x = t\}$ is not satisfiable, the result is trivial. We therefore assume, for the rest of the proof, that $\sigma \cup \{x = t\}$ is satisfiable in T. It follows from Corollary 1 that we just have to show that

a $\mathrm{vars}(r') \subseteq \mathrm{hvars}(\tau)$, for some $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$;
b $\mathrm{hvars}(\sigma) \subseteq \mathrm{hvars}(\tau)$, for some $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$.

From these, we can then conclude that, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$, (34) holds.

Note that, in both cases, since $\sigma \in MSubst$ and $\mathrm{vars}(r) \subseteq \mathrm{hvars}(\sigma)$, by Proposition 4 we have $\mathrm{rt}(r, \sigma) = r\sigma$, so that $r\sigma \in HTerms \cap GTerms$.

Case a. We must show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that $\mathrm{vars}(r') \subseteq \mathrm{hvars}(\tau)$.

As $\mathrm{mgs}\big(\sigma \cup \{x = t\}\big) \neq \varnothing$, by Theorem 5 and the definition of mgs we can assume that there exists $\tau \in MSubst \cap \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$. Thus

$$\tau \implies \big(\sigma \cup \{r = r'\}\big).$$

By Lemma 2 and the congruence axioms, we have $\tau \implies \{r\sigma = r'\}$. Since $\tau \in MSubst$ and $r\sigma \in HTerms \cap GTerms$, Lemma 11 applies (with $s = r\sigma$) so that $r\sigma = r'\tau \in HTerms \cap GTerms$. Thus, by Proposition 2, $\mathrm{vars}(r') \subseteq \mathrm{hvars}(\tau)$.

Case b. In this case, we show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that $\mathrm{hvars}(\sigma) \subseteq \mathrm{hvars}(\tau)$.

Let

$$\{u_1, \ldots, u_l\} \stackrel{\mathrm{def}}{=} \mathrm{dom}(\sigma) \cap \mathrm{vars}(r'\sigma),$$
$$\bar{s} \stackrel{\mathrm{def}}{=} (u_1, \ldots, u_l, r\sigma),$$
$$\bar{t} \stackrel{\mathrm{def}}{=} (u_1\sigma, \ldots, u_l\sigma, r'\sigma).$$

By Lemma 3 and the congruence axioms, $\sigma \cup \{x = t\} \implies \bar{s} = \bar{t}$. Thus, as $\sigma \cup \{x = t\}$ is satisfiable, $\mathrm{mgs}(\bar{s} = \bar{t}) \neq \varnothing$. Then, by Theorem 5, there exists $\mu \in MSubst \cap \mathrm{mgs}(\bar{s} = \bar{t})$. Therefore, since $r\sigma \in HTerms \cap GTerms$

and $\mu \implies \{r\sigma = r'\sigma\}$, Lemma 11 applies (with $s = r\sigma$) so that we have $r\sigma = r'\sigma\mu \in \mathit{HTerms} \cap \mathit{GTerms}$. Hence, for all $w \in \mathrm{dom}(\mu)$,

$$\mathrm{vars}(w\mu) = \varnothing. \tag{35}$$

Let

$$\nu \stackrel{\mathrm{def}}{=} \big\{z = z\sigma\mu \big| z \in \mathrm{dom}(\sigma) \setminus \mathrm{vars}(r'\sigma)\big\},$$
$$\tau \stackrel{\mathrm{def}}{=} \nu \cup \mu.$$

Then, as $\sigma, \mu \in \mathit{RSubst}$, it follows from (35) that $\nu, \tau \in \mathit{Eqs}$ have no identities or circular subsets so that $\nu, \tau \in \mathit{RSubst}$. By Lemma 3, $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$.

Suppose that $y \in \mathrm{hvars}(\sigma)$. Then we show that $y \in \mathrm{hvars}(\tau)$. Using Proposition 4, $\mathrm{rt}(y, \sigma) = y\sigma$ and

$$\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{36}$$

We show that $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$. Now, if $y \notin \mathrm{dom}(\tau)$, the result holds trivially. Suppose that $y \in \mathrm{dom}(\nu)$, then $y\tau = y\sigma\mu$ and $y \in \mathrm{dom}(\sigma)$. Let w be any variable in $\mathrm{vars}(y\sigma)$ so that, by (36), $w \notin \mathrm{dom}(\sigma)$. If $w \notin \mathrm{dom}(\mu)$, then $w = w\mu \notin \mathrm{dom}(\tau)$. If $w \in \mathrm{dom}(\mu)$, then, by (35), $\mathrm{vars}(w\mu) = \varnothing$. Therefore, $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\tau) = \varnothing$. It follows that $\mathrm{vars}(y\nu) \cap \mathrm{dom}(\tau) = \varnothing$. Finally, suppose $y \in \mathrm{dom}(\mu)$. Then, by (35), $\mathrm{vars}(y\mu) = \varnothing$. Therefore $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\tau) = \varnothing$.

Therefore, using Definition 6, we have that $y \in \mathrm{hvars}(\tau)$ as required. □

Proposition 6. *Let $p \in P$ and $(x \mapsto t) \in \mathit{Bind}$, where $\{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$. Let also $\sigma \in \gamma_P(p) \cap \mathit{MSubst}$ and suppose that $x \in \mathrm{hvars}(\sigma)$ and $\mathrm{vars}(t) \subseteq \mathrm{hvars}(\sigma)$. Suppose also that $\mathrm{ind}_p(x, t)$ and that $\mathrm{or\_lin}_p(x, t)$ hold. Then, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ in a syntactic equality theory T, we have*

$$\mathrm{hvars}(\sigma) \subseteq \mathrm{hvars}(\tau). \tag{37}$$

Proof. We assume that the congruence and identity axioms hold. If $\sigma \cup \{x = t\}$ is not satisfiable, the result is trivial. We therefore assume, for the rest of the proof, that $\sigma \cup \{x = t\}$ is satisfiable in T. It follows from Corollary 1 that we just have to show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that (37) holds.

As $x \in \mathrm{hvars}(\sigma)$ and $\mathrm{vars}(t) \subseteq \mathrm{hvars}(\sigma)$, we have, using Proposition 4, $\mathrm{rt}(x, \sigma) = x\sigma$ and $\mathrm{rt}(t, \sigma) = t\sigma$. Also

$$\mathrm{vars}(x\sigma) \cap \mathrm{dom}(\sigma) = \varnothing, \quad \mathrm{vars}(t\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{38}$$

As $\mathrm{ind}_p(x, t)$ holds,

$$\mathrm{vars}(x\sigma) \cap \mathrm{vars}(t\sigma) = \varnothing. \tag{39}$$

By hypothesis, $\mathrm{or\_lin}(x, t)$ holds so that, by Definition 4, for some $r \in \{x, t\}$, $r\sigma$ is linear. Let $r' \stackrel{\mathrm{def}}{=} \{x, t\} \setminus \{r\}$.

By Lemma 3 and the congruence axioms, $\sigma \cup \{x = t\} \implies \{r\sigma = r'\sigma\}$. Thus, as $\sigma \cup \{x = t\}$ is satisfiable, $\mathrm{mgs}(r\sigma = r'\sigma) \neq \varnothing$. Thus we can apply Lemma 12

(where $\bar{s} = r\sigma$ and $\bar{t} = r'\sigma$) so that, using (39), there exists $\mu \in \mathrm{mgs}(x\sigma = t\sigma)$ such that, for all $w \in \mathrm{dom}(\mu)$,

$$\mathrm{vars}(w\mu) \cap \mathrm{dom}(\mu) = \varnothing. \tag{40}$$

Note that, by (38),

$$\mathrm{dom}(\sigma) \cap \mathrm{vars}(\mu) = \varnothing. \tag{41}$$

Let

$$\nu \stackrel{\mathrm{def}}{=} \big\{z = z\sigma\mu \big| z \in \mathrm{dom}(\sigma)\big\},$$
$$\tau \stackrel{\mathrm{def}}{=} \nu \cup \mu.$$

Then, as $\sigma, \mu \in \mathit{RSubst}$, it follows from (41) that $\nu, \tau \in \mathit{Eqs}$ have no identities or circular subsets so that $\nu, \tau \in \mathit{RSubst}$. By Lemma 3, $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$.

Suppose $y \in \mathrm{hvars}(\sigma)$. Then we show that $y \in \mathrm{hvars}(\tau)$. As $y \in \mathit{HTerms}$, we have, using Proposition 4, $\mathrm{rt}(y, \sigma) = y\sigma$ and

$$\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{42}$$

We show that $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$. Now, if $y \notin \mathrm{dom}(\tau)$, the result holds trivially. Suppose that $y \in \mathrm{dom}(\nu)$, then $y\tau = y\sigma\mu$. Let w be any variable in $\mathrm{vars}(y\sigma)$. Then, by (42), $w \notin \mathrm{dom}(\sigma)$. If $w \notin \mathrm{dom}(\mu)$, then $w = w\mu \notin \mathrm{dom}(\tau)$. If $w \in \mathrm{dom}(\mu)$, then $\mathrm{vars}(w\mu) \subseteq \mathrm{vars}(\mu)$ so that, by (41), $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\nu) = \varnothing$. Moreover (40) applies so that $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\mu) = \varnothing$. Therefore we have $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\tau) = \varnothing$. It follows that $\mathrm{vars}(y\nu) \cap \mathrm{dom}(\tau) = \varnothing$. Finally, suppose $y \in \mathrm{dom}(\mu)$. Then $y\tau = y\mu$ and, by (41), wehave $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\nu) = \varnothing$. Also (40) applies where w is replaced by y so that $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\mu) = \varnothing$. Thus $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\tau) = \varnothing$.

Therefore, using Definition 6, we have that $y \in \mathrm{hvars}(\tau)$ as required. □

Proposition 7. *Let $p \in P$ and $(x \mapsto t) \in \mathit{Bind}$, where $\{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$. Let also $\sigma \in \gamma_P(p) \cap \mathit{MSubst}$ and suppose that $x \in \mathrm{hvars}(\sigma)$ and $\mathrm{vars}(t) \subseteq \mathrm{hvars}(\sigma)$. Suppose also that $\mathrm{gfree}_p(x)$ and $\mathrm{gfree}_p(t)$ hold. Then, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ in a syntactic equality theory T, we have*

$$\mathrm{hvars}(\sigma) \subseteq \mathrm{hvars}(\tau). \tag{43}$$

Proof. We assume that the congruence and identity axioms hold. If $\sigma \cup \{x = t\}$ is not satisfiable, the result is trivial. We therefore assume, for the rest of the proof, that $\sigma \cup \{x = t\}$ is satisfiable in T. It follows from Corollary 1 that we just have to show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that (43) holds.

By Definition 4, $\mathrm{gfree}_p(x)$ and $\mathrm{gfree}_p(t)$ imply that $\mathrm{rt}(x, \sigma) \in \mathit{GTerms}$ or $\mathrm{rt}(x, \sigma) \in \mathit{Vars}$, and that $\mathrm{rt}(t, \sigma) \in \mathit{GTerms}$ or $\mathrm{rt}(t, \sigma) \in \mathit{Vars}$. Since we have $\mathrm{rt}(x, \sigma), \mathrm{rt}(t, \sigma) \in \mathit{HTerms}$ and $\sigma \in \mathit{MSubst}$, as a consequence of Proposition 4, we have $\mathrm{rt}(x, \sigma) = x\sigma$, $\mathrm{rt}(t, \sigma) = t\sigma$ and $x\sigma, t\sigma \notin \mathrm{dom}(\sigma)$. There are three cases:

– $\mathrm{vars}(x\sigma) = \varnothing \vee \mathrm{vars}(t\sigma) = \varnothing$. Then the result follows from Proposition 5.

– $x\sigma = t\sigma \in \mathit{Vars}$. Then letting $\tau = \sigma$ gives the required result.
– $x\sigma, t\sigma \in \mathit{Vars}$ are distinct variables. Let $\tau = \sigma \cup \{x\sigma = t\sigma\}$. Then, as $x\sigma, t\sigma \notin \mathrm{dom}(\sigma)$, $\tau \in \mathit{RSubst}$. Hence, by Lemma 3, $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$. Let y be any variable in $\mathrm{hvars}(\sigma)$. We show that $y \in \mathrm{hvars}(\tau)$.
Suppose first that $y \neq x\sigma$. Then $y\tau = y\sigma$. Thus using Proposition 4, $\mathrm{rt}(y, \sigma) = y\tau$ and $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\sigma) = \varnothing$. Thus $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) \subseteq \{r\sigma\}$. However, $r\sigma\tau = t\sigma \notin \mathrm{dom}(\tau)$ so that, by Definition 5, $\mathrm{vars}(y\tau) \subseteq \mathrm{hvars}_1(\tau)$ and hence $y \in \mathrm{hvars}_2(\tau)$. Therefore, by Lemma 6 and Definition 6, we have $y \in \mathrm{hvars}(\tau)$.
Secondly, suppose that $y = x\sigma$. Then $y\tau = t\sigma$. So that, as $t\sigma \in \mathit{Vars} \setminus \mathrm{dom}(\sigma)$ and $x\sigma \neq t\sigma$, $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$. Therefore, using Definition 6, we have that $y \in \mathrm{hvars}(\tau)$ as required. □

Proposition 8. *Let $p \in P$ and $(x \mapsto t) \in \mathit{Bind}$, where $\{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$. Let $\sigma \in \gamma_P(p) \cap \mathit{MSubst}$ and suppose that $x \in \mathrm{hvars}(\sigma)$ and $\mathrm{vars}(t) \subseteq \mathrm{hvars}(\sigma)$. Furthermore, suppose that $\mathrm{or\_lin}_p(x, t)$ and $\mathrm{share\_lin}_p(x, t)$ hold. Then, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ in a syntactic equality theory T, we have*

$$\mathrm{hvars}(\sigma) \setminus \mathrm{share\_same\_var}_p(x, t) \subseteq \mathrm{hvars}(\tau). \tag{44}$$

Proof. We assume that the congruence and identity axioms hold. If $\sigma \cup \{x = t\}$ is not satisfiable, the result is trivial. We therefore assume, for the rest of the proof, that $\sigma \cup \{x = t\}$ is satisfiable in T. It follows from Corollary1 that we just have to show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that (44) holds.

As $x \in \mathrm{hvars}(\sigma)$ and $\mathrm{vars}(t) \subseteq \mathrm{hvars}(\sigma)$, we have, using Proposition 4, $\mathrm{rt}(x, \sigma) = x\sigma$ and $\mathrm{rt}(t, \sigma) = t\sigma$. Also

$$\mathrm{vars}(x\sigma) \cap \mathrm{dom}(\sigma) = \varnothing, \quad \mathrm{vars}(t\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{45}$$

By hypothesis, $\mathrm{or\_lin}_p(x, t)$ holds so that, by Definition 4, for some $r \in \{x, t\}$, $r\sigma$ is linear. Also by hypothesis, $\mathrm{share\_lin}_p(x, t)$ holds so that, by Definition 4, if $r' = \{x, t\} \setminus \{r\}$, for all $z \in \mathrm{vars}(r\sigma) \cap \mathrm{vars}(r'\sigma)$, $\mathrm{occ\_lin}(z, r'\sigma)$ holds. Therefore,

$$\mathrm{vars}(r\sigma) \cap \mathrm{nlvars}(r'\sigma) = \varnothing. \tag{46}$$

By Lemma 3 and the congruence axioms, $\sigma \cup \{x = t\} \implies \{r\sigma = r'\sigma\}$. Thus, as $\sigma \cup \{x = t\}$ is satisfiable, $\mathrm{mgs}(r\sigma = r'\sigma) \neq \varnothing$. Thus, as $r\sigma$ is linear and (46) holds, we can apply Lemma 12 (where $\bar{s} = r\sigma$ and $\bar{t} = r'\sigma$) so that there exists $\mu \in \mathrm{mgs}(x\sigma = t\sigma)$ such that, for all $w \in \mathrm{dom}(\mu) \setminus \big(\mathrm{vars}(x\sigma) \cap \mathrm{vars}(t\sigma)\big)$,

$$\mathrm{vars}(w\mu) \cap \mathrm{dom}(\mu) = \varnothing. \tag{47}$$

Note that, by (45),

$$\mathrm{dom}(\sigma) \cap \mathrm{vars}(\mu) = \varnothing. \tag{48}$$

Let

$$\nu \stackrel{\mathrm{def}}{=} \big\{z = z\sigma\mu \big| z \in \mathrm{dom}(\sigma)\big\},$$
$$\tau \stackrel{\mathrm{def}}{=} \nu \cup \mu.$$

Then, as $\sigma, \mu \in \mathit{RSubst}$, it follows from (48) that $\nu, \tau \in \mathit{Eqs}$ have no identities or circular subsets so that $\nu, \tau \in \mathit{RSubst}$. By Lemma 3, $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$.

Suppose $y \in \mathrm{hvars}(\sigma) \setminus \mathrm{share\_same\_var}_p(x, t)$. We show that $y \in \mathrm{hvars}(\tau)$. As $y \in \mathrm{hvars}(\sigma)$, using Proposition 4, $\mathrm{rt}(y, \sigma) = y\sigma$ and

$$\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{49}$$

As $y \notin \mathrm{share\_same\_var}_p(x, t)$, by Definition 4,

$$\mathrm{vars}(y\sigma) \cap \mathrm{vars}(x\sigma) \cap \mathrm{vars}(t\sigma) = \varnothing. \tag{50}$$

Therefore, using (50) if $y \notin \mathrm{dom}(\sigma)$ and (45) if $y \in \mathrm{dom}(\sigma)$, it follows that

$$y \notin \mathrm{vars}(x\sigma) \cap \mathrm{vars}(t\sigma). \tag{51}$$

We show that $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$. Now, if $y \notin \mathrm{dom}(\tau)$, the result holds trivially. Suppose that $y \in \mathrm{dom}(\nu)$, then $y\tau = y\sigma\mu$. Let w be any variable in $\mathrm{vars}(y\sigma)$. Then, by (50), $w \notin \big(\mathrm{vars}(x\sigma) \cap \mathrm{vars}(t\sigma)\big)$ and, by (49), $w \notin \mathrm{dom}(\sigma)$. If $w \notin \mathrm{dom}(\mu)$, then $w = w\mu \notin \mathrm{dom}(\tau)$. If $w \in \mathrm{dom}(\mu)$, then $\mathrm{vars}(w\mu) \subseteq \mathrm{vars}(\mu)$ so that, by (48), we also have $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\nu) = \varnothing$. Moreover (47) applies so that $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\mu) = \varnothing$. Therefore, $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\tau) = \varnothing$. It follows that $\mathrm{vars}(y\nu) \cap \mathrm{dom}(\tau) = \varnothing$. Finally, suppose $y \in \mathrm{dom}(\mu)$. Then $y\tau = y\mu$ and, by (48), $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\nu) = \varnothing$. As (51) holds, (47) applies where w is replaced by y so that $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\mu) = \varnothing$. Thus $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\tau) = \varnothing$.

Therefore, using Definition 6, we have that $y \in \mathrm{hvars}(\tau)$ as required. □

Proposition 9. *Let $p \in P$ and $(x \mapsto t) \in \mathit{Bind}$, where $\{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$. Let also $\sigma \in \gamma_P(p) \cap \mathit{MSubst}$ and suppose that $\{r, r'\} = \{x, t\}$, $\mathrm{vars}(r) \subseteq \mathrm{hvars}(\sigma)$ and $\mathrm{lin}_p(r)$ holds. Then, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ in a syntactic equality theory T, we have*

$$\mathrm{hvars}(\sigma) \setminus \mathrm{share\_with}_p(r) \subseteq \mathrm{hvars}(\tau). \tag{52}$$

Proof. We assume that the congruence and identity axioms hold. If $\sigma \cup \{x = t\}$ is not satisfiable, the result is trivial. We therefore assume, for the rest of the proof, that $\sigma \cup \{x = t\}$ is satisfiable in T. It follows from Corollary 1 that we just have to show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that (52) holds.

By hypothesis, $\mathrm{vars}(r) \subseteq \mathrm{hvars}(\sigma)$. Hence, by Proposition 4, $\mathrm{rt}(r, \sigma) = r\sigma$ and

$$\mathrm{vars}(r\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{53}$$

By hypothesis, $\mathrm{lin}_p(r)$ holds, so that, by Definition 4, $r\sigma$ is linear.

Let

$$\{u_1, \ldots, u_l\} \stackrel{\mathrm{def}}{=} \mathrm{dom}(\sigma) \cap \big(\mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma)\big),$$
$$\bar{s} \stackrel{\mathrm{def}}{=} (u_1, \ldots, u_l, r\sigma),$$
$$\bar{t} \stackrel{\mathrm{def}}{=} (u_1\sigma, \ldots, u_l\sigma, r'\sigma).$$

Since $r\sigma$ is linear, it follows from (53) that $\bar{s}$ is linear. By Lemma 3 and the congruence axioms, $\sigma \cup \{x = t\} \implies \bar{s} = \bar{t}$. Thus, as $\sigma \cup \{x = t\}$ is satisfiable, we have $\mathrm{mgs}(\bar{s} = \bar{t}) \neq \varnothing$. Therefore, we can apply Lemma 13 so that there exists $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$ such that, for all $w \in \mathrm{dom}(\mu) \setminus \mathrm{vars}(\bar{s})$,

$$\mathrm{vars}(w\mu) \cap \mathrm{dom}(\mu) = \varnothing. \tag{54}$$

Note that, since $\sigma \in \mathit{MSubst}$, for each $i = 1, \ldots, l$, we have

$$\mathrm{vars}(u_i\sigma) \subseteq \mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma).$$

Thus

$$\mathrm{vars}(\mu) \subseteq \mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma). \tag{55}$$

Let

$$\nu \stackrel{\mathrm{def}}{=} \left\{ z = z\sigma\mu \middle| z \in \mathrm{dom}(\sigma) \setminus \big(\mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma)\big) \right\},$$
$$\tau \stackrel{\mathrm{def}}{=} \nu \cup \mu.$$

Then, as $\sigma, \mu \in \mathit{RSubst}$, it follows from (55) that $\nu, \tau \in \mathit{Eqs}$ have no identities or circular subsets so that $\nu, \tau \in \mathit{RSubst}$. By Lemma 3, $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$.

Suppose $y \in \mathrm{hvars}(\sigma) \setminus \mathrm{share\_with}_p(r)$. Then we show that $y \in \mathrm{hvars}(\tau)$. As $y \in \mathrm{hvars}(\sigma)$, by Proposition 4, $\mathrm{rt}(y, \sigma) = y\sigma$ and

$$\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{56}$$

As $y \notin \mathrm{share\_with}_p(r)$, by Definition 4, $y \notin \mathrm{share\_same\_var}_p(y, r)$ so that, using the same definition,

$$\mathrm{vars}(y\sigma) \cap \mathrm{vars}(r\sigma) = \varnothing. \tag{57}$$

Therefore using (57) if $y \notin \mathrm{dom}(\sigma)$ and (53) if $y \in \mathrm{dom}(\sigma)$, it follows that

$$y \notin \mathrm{vars}(r\sigma). \tag{58}$$

We show that $\mathrm{vars}(y\tau) \cap \mathrm{dom}(\tau) = \varnothing$. Now, if $y \notin \mathrm{dom}(\tau)$, the result holds trivially. Suppose that $y \in \mathrm{dom}(\nu)$. Then $y\tau = y\sigma\mu$ and $y \in \mathrm{dom}(\sigma)$. It follows from (56) and (57) that $\mathrm{vars}(y\sigma) \cap \mathrm{vars}(\bar{s}) = \varnothing$. Let w be any variable in $\mathrm{vars}(y\sigma)$ so that $w \notin \mathrm{vars}(\bar{s})$. By (56), we have $w \notin \mathrm{dom}(\sigma)$. If $w \notin \mathrm{dom}(\mu)$, then $w = w\mu \notin \mathrm{dom}(\tau)$. If $w \in \mathrm{dom}(\mu)$, then $\mathrm{vars}(w\mu) \subseteq \mathrm{vars}(\mu)$ so that, by (55), $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\nu) = \varnothing$. Moreover (54) applies so that $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\mu) = \varnothing$. Therefore, $\mathrm{vars}(w\mu) \cap \mathrm{dom}(\tau) = \varnothing$. It follows that $\mathrm{vars}(y\nu) \cap \mathrm{dom}(\tau) = \varnothing$. Finally, suppose $y \in \mathrm{dom}(\mu)$. Then $y\tau = y\mu$ and, by (55), $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\nu) = \varnothing$. As $\sigma \in \mathit{MSubst}$ and $y \in \mathrm{hvars}(\sigma)$, $y \notin \mathrm{dom}(\sigma) \cap \big(\mathrm{vars}(r\sigma) \cup \mathrm{vars}(r'\sigma)\big)$ and hence $y \notin \mathrm{vars}(\bar{s})$. Therefore (54) applies and $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\mu) = \varnothing$. Thus $\mathrm{vars}(y\mu) \cap \mathrm{dom}(\tau) = \varnothing$.

Therefore, using Definition 6, we have that $y \in \mathrm{hvars}(\tau)$ as required. □

Proposition 10. *Let $p \in P$ and $(x \mapsto t) \in Bind$, where $\{x\} \cup \mathrm{vars}(t) \subseteq VI$. Let also $\sigma \in \gamma_P(p) \cap MSubst$. Then, for all $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ in a syntactic equality theory T,*

$$\mathrm{hvars}(\sigma) \setminus \big(\mathrm{share\_with}_p(x) \cup \mathrm{share\_with}_p(t)\big) \subseteq \mathrm{hvars}(\tau). \tag{59}$$

Proof. We assume that the congruence and identity axioms hold. If $\sigma \cup \{x = t\}$ is not satisfiable, the result is trivial. We therefore assume, for the rest of the proof, that $\sigma \cup \{x = t\}$ is satisfiable in T. It follows from Corollary 1 that we just have to show that there exists $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$ such that (59) holds.

Let

$$\begin{aligned} \{u_1, \ldots, u_l\} &\stackrel{\mathrm{def}}{=} \mathrm{dom}(\sigma) \cap \big(\mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma)\big), \\ \bar{s} &\stackrel{\mathrm{def}}{=} (u_1, \ldots, u_l, x\sigma), \\ \bar{t} &\stackrel{\mathrm{def}}{=} (u_1\sigma, \ldots, u_l\sigma, t\sigma). \end{aligned}$$

Note that, since $\sigma \in MSubst$, for each $i = 1, \ldots, l$, we have

$$\mathrm{vars}(u_i\sigma) \subseteq \mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma).$$

Thus, for any $\mu \in \mathrm{mgs}(\bar{s} = \bar{t})$, we have

$$\mathrm{vars}(\mu) \subseteq \mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma). \tag{60}$$

Let

$$\begin{aligned} \nu &\stackrel{\mathrm{def}}{=} \left\{ z = z\sigma\mu \;\middle|\; z \in \mathrm{dom}(\sigma) \setminus \big(\mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma)\big) \right\}, \\ \tau &\stackrel{\mathrm{def}}{=} \nu \cup \mu. \end{aligned}$$

Then, as $\sigma, \mu \in RSubst$, it follows from (60) that $\nu, \tau \in Eqs$ have no identities or circular subsets so that $\nu, \tau \in RSubst$. Thus, using Lemma 3 and the assumption that $\sigma \cup \{x = t\}$ is satisfiable, $\tau \in \mathrm{mgs}\big(\sigma \cup \{x = t\}\big)$.

Suppose $y \in \mathrm{hvars}(\sigma) \setminus \big(\mathrm{share\_with}_p(x) \cup \mathrm{share\_with}_p(t)\big)$. We show that $y \in \mathrm{hvars}(\tau)$. As $y \in \mathrm{hvars}(\sigma)$, by Proposition 4, $\mathrm{rt}(y, \sigma) = y\sigma$ and

$$\mathrm{vars}(y\sigma) \cap \mathrm{dom}(\sigma) = \varnothing. \tag{61}$$

As $y \notin \mathrm{share\_with}_p(x) \cup \mathrm{share\_with}_p(t)$, it follows from Definition 4 that

$$y \notin \mathrm{share\_same\_var}_p(y, x) \cup \mathrm{share\_same\_var}_p(y, t)$$

so that, using the same definition with the result that $\mathrm{rt}(y, \sigma) = y\sigma$, we obtain

$$\mathrm{vars}(y\sigma) \cap \big(\mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma)\big) = \varnothing. \tag{62}$$

Therefore, using (62) if $y \notin \mathrm{dom}(\sigma)$ and using the fact that $\sigma \in MSubst$, if $y \in \mathrm{dom}(\sigma)$, it follows that

$$y \notin \mathrm{vars}(x\sigma) \cup \mathrm{vars}(t\sigma). \tag{63}$$

We show that $\text{vars}(y\tau) \cap \text{dom}(\tau) = \varnothing$. Now, if $y \notin \text{dom}(\tau)$, the result holds trivially. Suppose that $y \in \text{dom}(\tau)$. Then, by (60) and (63), $y \notin \text{vars}(\mu)$ so that $y \notin \text{dom}(\mu)$ and $\text{vars}(y\mu) \cap \text{dom}(\mu) = \varnothing$. Thus we must have $y \in \text{dom}(\nu)$ and $y\tau = y\sigma$. Then, by (60) and (62), $\text{vars}(y\sigma) \cap \text{dom}(\mu) = \varnothing$. Moreover, by (61), $\text{vars}(y\sigma) \cap \text{dom}(\sigma) = \varnothing$. It follows that $\text{vars}(y\sigma) \cap \text{dom}(\tau) = \varnothing$ and hence, as $y\sigma = y\tau$, $\text{vars}(y\tau) \cap \text{dom}(\tau) = \varnothing$.

Therefore, using Definition 6, we have that $y \in \text{hvars}(\tau)$ as required. □

Proof (Proof of Theorem 2).

By hypothesis, $\sigma \in \gamma_P(p)$. By Theorem 5, there exists $\sigma' \in \mathit{MSubst}$ such that $\sigma \iff \sigma'$. By Lemma 10, we have $\text{hvars}(\sigma) = \text{hvars}(\sigma')$. By Definition 3, $\sigma \in \gamma_P(p)$ if and only if $\sigma' \in \gamma_P(p)$. We therefore safely assume that $\sigma \in \mathit{MSubst}$.

By hypothesis, we have $\sigma \in \gamma_H(h)$. Therefore, it follows from Definition 7 that $h \subseteq \text{hvars}(\sigma)$. Similarly, by Definition 7, in order to prove $\tau \in \gamma_H(h')$, we just need to show that $h' \subseteq \text{hvars}(\tau)$ where h' is as defined in Definition 8. There are eight cases that have to be considered.

1. $\text{hterm}_h(x) \wedge \text{ground}_p(x)$ holds.
   As $\text{hterm}_h(x)$ holds, by Definition 8, $x \in h$. Hence, by Definition 7, we have $x \in \text{hvars}(\sigma)$. As $\text{ground}_p(x)$ holds, by Definition 4, $\text{rt}(x, \sigma) \in \mathit{GTerms}$. Therefore we can apply Proposition 5, where r is replaced by x and r' by t, to conclude that
   $$\text{hvars}(\sigma) \cup \text{vars}(t) \subseteq \text{hvars}(\tau).$$
2. $\text{hterm}_h(t) \wedge \text{ground}_p(t)$ holds.
   As $\text{hterm}_h(t)$ holds, by Definition 8, $\text{vars}(t) \subseteq h$. Hence, by Definition 7, $\text{vars}(t) \subseteq \text{hvars}(\sigma)$. As $\text{ground}_p(t)$ holds, by Definition 4, $\text{rt}(t, \sigma) \in \mathit{GTerms}$. Therefore we can apply Proposition 5, where r is replaced by t and r' by x, to conclude that
   $$\text{hvars}(\sigma) \cup \{x\} \subseteq \text{hvars}(\tau).$$
3. $\text{hterm}_h(x) \wedge \text{hterm}_h(t) \wedge \text{ind}_p(x, t) \wedge \text{or\_lin}_p(x, t)$ holds.
   As $\text{hterm}_h(x)$ and $\text{hterm}_h(t)$ hold, by Definition 8, $x \in h$ and $\text{vars}(t) \subseteq h$. Hence, by Definition 7, $x \in \text{hvars}(\sigma)$ and $\text{vars}(t) \subseteq \text{hvars}(\sigma)$. Therefore we can apply Proposition 6 to conclude that
   $$\text{hvars}(\sigma) \subseteq \text{hvars}(\tau).$$
4. $\text{hterm}_h(x) \wedge \text{hterm}_h(t) \wedge \text{gfree}_p(x) \wedge \text{gfree}_p(t)$ holds.
   As $\text{hterm}_h(x)$ and $\text{hterm}_h(t)$ hold, by Definition 8, $x \in h$ and $\text{vars}(t) \subseteq h$. Hence, by Definition 7, $x \in \text{hvars}(\sigma)$ and $\text{vars}(t) \subseteq \text{hvars}(\sigma)$. Therefore we can apply Proposition 7 to conclude that
   $$\text{hvars}(\sigma) \subseteq \text{hvars}(\tau).$$
5. $\text{hterm}_h(x) \wedge \text{hterm}_h(t) \wedge \text{share\_lin}_p(x, t) \wedge \text{or\_lin}_p(x, t)$ holds.
   As $\text{hterm}_h(x)$ and $\text{hterm}_h(t)$ hold, by Definition 8, $x \in h$ and $\text{vars}(t) \subseteq h$. Hence, by Definition 7, $x \in \text{hvars}(\sigma)$ and $\text{vars}(t) \subseteq \text{hvars}(\sigma)$. Therefore we can apply Proposition 8 to conclude that
   $$\text{hvars}(\sigma) \setminus \text{share\_same\_var}_p(x, t) \subseteq \text{hvars}(\tau).$$

6. $\mathrm{hterm}_h(x) \wedge \mathrm{lin}_p(x)$ holds.
   As $\mathrm{hterm}_h(x)$ holds, by Definition 8, $x \in h$. Hence, by Definition 7, we have $x \in \mathrm{hvars}(\sigma)$. Therefore we can apply Proposition 9 where r is replaced by x and r' by t, to conclude that

$$\mathrm{hvars}(\sigma) \setminus \mathrm{share\_with}_p(x) \subseteq \mathrm{hvars}(\tau).$$

7. $\mathrm{hterm}_h(t) \wedge \mathrm{lin}_p(t)$ holds.
   As $\mathrm{hterm}_h(t)$ holds, by Definition 8, $\mathrm{vars}(t) \subseteq h$. Hence, by Definition 7, $\mathrm{vars}(t) \subseteq \mathrm{hvars}(\sigma)$. Therefore we can apply Proposition 9 where r is replaced by t and r' by x, to conclude that

$$\mathrm{hvars}(\sigma) \setminus \mathrm{share\_with}_p(t) \subseteq \mathrm{hvars}(\tau).$$

8. For all $(x \mapsto t) \in \mathit{Bind}$ where $\{x\} \cup \mathrm{vars}(t) \subseteq \mathit{VI}$, Proposition 10 applies so that

$$\mathrm{hvars}(\sigma) \setminus \big(\mathrm{share\_with}_p(x) \cup \mathrm{share\_with}_p(t)\big) \subseteq \mathrm{hvars}(\tau).$$

□

Appendix References

[BCM94] M. Bruynooghe, M. Codish, and A. Mulkers, *Abstract unification for a composite domain deriving sharing and freeness properties of program variables*, Verification and Analysis of Logic Languages, Proceedings of the W2 Post-Conference Workshop, International Conference on Logic Programming (Santa Margherita Ligure, Italy) (F. S. de Boer and M. Gabbrielli, eds.), 1994, pp. 213–230.

[BZH00] R. Bagnara, E. Zaffanella, and P. M. Hill, *Enhanced sharing analysis techniques: A comprehensive evaluation*, Proceedings of the 2nd International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (Montreal, Canada) (M. Gabbrielli and F. Pfenning, eds.), Association for Computing Machinery, 2000, pp. 103–114.

[Cla78] K. L. Clark, *Negation as failure*, Logic and Databases (Toulouse, France) (H. Gallaire and J. Minker, eds.), Plenum Press, 1978, pp. 293–322.

[Col82] A. Colmerauer, *Prolog and infinite trees*, Logic Programming, APIC Studies in Data Processing (K. L. Clark and S. Å. Tärnlund, eds.), vol. 16, Academic Press, New York, 1982, pp. 231–251.

[Col84] A. Colmerauer, *Equations and inequations on finite and infinite trees*, Proceedings of the International Conference on Fifth Generation Computer Systems (FGCS'84) (Tokyo, Japan), ICOT, 1984, pp. 85–99.

[HBZ98] P. M. Hill, R. Bagnara, and E. Zaffanella, *The correctness of set-sharing*, Static Analysis: Proceedings of the 5th International Symposium (Pisa, Italy) (G. Levi, ed.), Lecture Notes in Computer Science, vol. 1503, Springer-Verlag, Berlin, 1998, pp. 99–114.

[HBZ01] P. M. Hill, R. Bagnara, and E. Zaffanella, *Soundness, idempotence and commutativity of set-sharing*, Theory and Practice of Logic Programming (2001), To appear. Available at `http://arXiv.org/abs/cs.PL/0102030`.

[JL89] D. Jacobs and A. Langen, *Accurate and efficient approximation of variable aliasing in logic programs*, Logic Programming: Proceedings of the North American Conference (Cleveland, Ohio, USA) (E. L. Lusk and R. A. Overbeek, eds.), MIT Press Series in Logic Programming, The MIT Press, 1989, pp. 154–165.