PAIR-SHARING OVER RATIONAL TREES

ANDY KING

▷ Sharing information is useful in specialising, optimising and parallelising logic programs and thus sharing analysis is an important topic of both abstract interpretation and logic programming. Sharing analyses infer which pairs of program variables can never be bound to terms that contain a common variable. We generalise a classic pair sharing analysis from Herbrand unification to trace sharing over rational tree constraints. This is useful for reasoning about programs written in SICStus and Prolog-III because these languages use rational tree unification as the default equation solver. ◁

1. Introduction

Sharing analyses infer which program variables can never be bound to terms that contain a common variable. Variable pairs which do not share are said to be unaliased or independent. Independence information can be used, among other things, to optimise backtracking [2]; specialise unification [17]; and eliminate runtime checks in and-parallelisation [15]. Sharing analyses often additionally trace linearity [3, 13, 16]. Linearity relates to the number of times a variable occurs in a term. A term is linear if it does not contain multiple occurrences of a variable, otherwise it is non-linear. The significance of linearity is that the unification of linear terms yields only restricted forms of sharing. Thus, with linearity information, worse case transitive sharing does not need to be assumed [3]. One key result of [3] is lemma 2.2 which details some conditions on sharing that follow from the unification of linear terms. One case of the lemma can be stated as follows:

Lemma 1.1. Suppose $\theta \in mgu(\{s = t\})$, $\chi(s) = 2$ and $\chi(t) = 1$. If $var(\theta(x)) \cap var(\theta(y)) \neq \emptyset$ then either $x \notin var(s)$ or $y \notin var(s)$. ▮

Address correspondence to Andy King, Computing Laboratory, University of Kent at Canterbury, Canterbury, CT2 7NF, UK. E-mail: a.m.king@ukc.ac.uk

THE JOURNAL OF LOGIC PROGRAMMING

655 Avenue of the Americas, New York, NY 10010 0743-1066/93/$3.50

The lemma can be interpreted as saying that if a most general unifier (mgu) θ exists for the equation $s = t$, s is a non-linear term, t is a linear term, and the variables x and y share under θ, then x and y are not both in s. Unfortunately, as has been pointed out before [7, 13], this lemma is subtly wrong as is illustrated by putting $s = f(x, x, y)$ and $t = f(x, y, z)$. One mgu is the substitution $\theta = \{y \mapsto x, z \mapsto x\}$ for which $var(\theta(x)) = \{x\} = var(\theta(y))$ but $x \in var(s)$ and $y \in var(s)$.

It is important to understand, however, that the sharing analysis algorithm of [3] is not fundamentally flawed. Indeed, Dams [7] proposes a revision of lemma 2.2, with a proof sketch, that appears to be correct. Our work extends this unpublished result to show how (a reformulation of) the sharing algorithm of [3] is safe for rational tree unification [5, 11, 12] (and Herbrand unification without the occur-check). We also make the following contributions:

- We show how the notions of groundness, sharing and linearity can be straightforwardly lifted to rational tree constraints by using substitutions over infinite trees [6, 9]. For example, to decide which variables share under $\theta = \{x \mapsto f(z, y, x), y \mapsto g(y)\}$, we examine the limit of the sequence θ, $\theta \circ \theta$, $\theta \circ \theta \circ \theta$, ... where $\circ$ denotes composition. The substitution $\{x \mapsto s, y \mapsto t\}$ is the limit where $t = g(g(g(\ldots)))$ and $s = f(z, t, f(z, t, f(z, t, \ldots)))$. Since t contains no variables, y is ground and neither x or z share with y. Furthermore, x and z share, z is linear, but x is non-linear.

- We show how sharing relates to alternating paths [16]. Specifically, we show that if a rational tree solver takes as input an equation set E and produces as output E', and if there exists an alternating path in E' between two variables x and y, then there exists an alternating path between x and y in E. This result leads to a revision and a generalisation of lemma 2.2 of [3]. Thus, alternating paths turn out to be an important device for establishing the correctness of pair-sharing.

- We generalise the concretisation map for pair sharing [3] to substitutions in rational solved form. Since correctness is defined by concretisation, it is imperative that the map puts a sensible interpretation on sharing abstractions. In particular it must coincide with the classic map for idempotent substitutions. We show this is so. Correctness is then established for pair sharing and its product with a (parameterised) groundness analysis.

Generalising pair sharing to rational trees is more than an exercise in aesthetics because Prolog-III and SICStus Prolog use rational tree unification as the default solver. Specifically, our work enables programs that manipulate infinite trees to be safely analysed for, say, parallelisation. (In fact, the was the main motivation for the work.) More generally, without a deep knowledge of a Prolog-III or SICStus program, it is difficult to determine whether or not it uses rational trees. Hence, for safety, a sharing analysis must be conservative in the sense that it assumes that rational trees may be used.

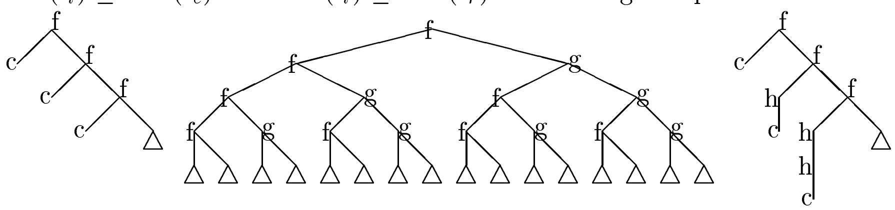
The exposition is structured as follows. Section 2 describes the notation and preliminary definitions which will be used throughout. It also introduces the idea of using limits and establishes some of its properties. Linearity is formally introduced in section 3 and its relationship with the alternating paths [16] is explained. In section 4, the focus is first on abstracting data and in particular the concretisation map. Secondly, an abstract analog for unification is defined and proved correct. Finally, sections 5 and 6 present the related work and the conclusions.

2. Preliminaries

2.1. Finite, infinite and rational trees

In the sequel, wherever possible, we follow the notation of [6]. Let the pair $\langle F, \varrho \rangle$ be a ranked alphabet consisting of a set F and a map $\varrho : F \to \mathbb{N}$ which defines the rank of any symbol $f \in F$. A tree over $\langle F, \varrho \rangle$ is a partial map $t : \mathbb{N}^* \to F$ such that its domain is non-empty and prefix-closed, that is, $dom(t) \neq \emptyset$ and if $\alpha, \beta \in \mathbb{N}^*$ and $\alpha.\beta \in dom(t)$ then $\alpha \in dom(t)$ where . denotes concatenation. The empty sequence is denoted ε and $|\alpha|$ denotes the length of the sequence α, for example, $|\varepsilon| = 0$ and $|1.2.3| = 3$. Furthermore, we require the following condition on t and ϱ: if $t(\alpha) = f$ and $i \in \{1, \ldots, \varrho(f)\}$ then $\alpha.i \in dom(t)$. Let $M^\infty(F)$ denote the set of trees over F. We assume that F includes a constant c, that is, $\varrho(c) = 0$. A tree $t \in M^\infty(F)$ is finite iff $dom(t)$ is finite, otherwise it is infinite. We denote the set of finite trees by $M(F)$. Let $occ(f, t)$ denote the set of occurrences of f in t, that is, $occ(f, t) = \{\alpha \in dom(t) \mid t(\alpha) = f\}$. The set of sub-trees of t, $sub(t)$, is defined by $sub(t) = \{\lambda\beta.t(\alpha.\beta) \mid \alpha \in dom(t)\}$. A tree t is rational iff $sub(t)$ is finite.

Example 2.1. The leftmost, centre and rightmost trees, denoted t_l, t_c and t_r respectively, are all infinite since $dom(t_l) = \{\varepsilon, 1, 2, 2.1, 2.2, \ldots\}$ is infinite and $dom(t_l) \subseteq dom(t_c)$ and $dom(t_l) \subseteq dom(t_r)$. The triangles represent sub-trees.



The trees t_l and t_c are rational since $sub(t_l) = \{\lambda\varepsilon.a,\ t_l\}$ and $sub(t_c) = \{\lambda\alpha.t_c(2.\alpha),\ t_c\}$ whereas t_r is not because $sub(t_r) = \{t_r,\ \lambda\alpha.t_r(2.\alpha),\ \lambda\alpha.t_r(2.2.\alpha), \ldots\}$. ∎

2.2. Substitutions over trees

Let X denote a (denumerable) set of symbols of rank 0 such that $F \cap X = \emptyset$. X is interpreted as a universe of variables. Let $M^\infty(F, X) = M^\infty(F \cup X)$ and $M(F, X) = M(F \cup X)$. If $t \in M^\infty(F, X)$ we define $var(t) = \{v \in X \mid occ(v, t) \neq \emptyset\}$. The size of a finite tree $t \in M(F, X)$ is defined by: $size(x) = 1$ if $x \in X$, $size(c) = 1$ if $\varrho(c) = 0$, and $size(f(t_1, \ldots, t_n)) = 1 + \sum_{i=1}^{n} size(t_i)$ if $\varrho(f) = n \in \mathbb{N}$.

A substitution is a (total) map $\theta : X \to M^\infty(F, X)$ such that $dom(\theta) = \{x \in X \mid \theta(x) \neq x\}$ is finite. We define $cod(\theta) = \cup\{var(\theta(x)) \mid x \in dom(\theta)\}$. A substitution θ can be represented as a finite set of pairs $\{x \mapsto \theta(x) \mid x \in dom(\theta)\}$. The set of substitutions is denoted Sub and the identity ϵ. If $\theta \in Sub$ and $t \in M^\infty(F, X)$, then $\theta(t)$ is the tree defined by:

$$\theta(t)(\alpha) = \begin{cases} \theta(x)(\alpha') \text{ if } x \in X & \wedge\ \beta \in occ(x,t) \\ & \wedge\ \alpha = \beta.\alpha' \wedge \alpha' \in dom(\theta(x)) \\ t(\alpha) \quad\ \text{else if } \alpha \in dom(t) & \end{cases}$$

For brevity, we write $\theta(x, \alpha)$ for $t(\alpha)$ where $\theta(x) = t$. An equation e is a pair $(s = t)$ where $s, t \in M(F, X)$. A finite set of equations is denoted E and Eqn denotes the set of finite sets of equations. We also define $\theta(E) = \{\theta(s) = \theta(t) \mid (s = t) \in E\}$. The map $eqn : Sub \to Eqn$ is defined by: $eqn(\theta) = \{x = t \mid (x \mapsto t) \in \theta\}$.

If $Y \subseteq X$, then projection onto Y, $\overline{\exists} Y.\theta$, is defined by: $\overline{\exists} Y.\theta = \exists(X \setminus Y).\theta$ where $(\exists Y.\theta)(z) = z$ if $z \in Y$ and $(\exists Y.\theta)(z) = \theta(z)$ otherwise. Composition $\theta \circ \vartheta$ of two substitutions is defined so that: $(\theta \circ \vartheta)(x) = \theta(\vartheta(x))$ for all $x \in X$. Composition induces the (more general than) relation $\leq$ defined by: $\theta \leq \vartheta$ if there exists $\delta \in Sub$ such that $\vartheta = \delta \circ \theta$. A renaming is a substitution $\rho \in Sub$ that has an inverse, that is, there exists $\rho^{-1} \in Sub$ such that $\rho^{-1} \circ \rho = \epsilon$. The set of renamings is denoted *Rename*. A substitution θ is idempotent if $\theta \circ \theta = \theta$; is circular if it has the form $\{x_1 \mapsto x_2, \ldots, x_n \mapsto x_1\}$ where $n \geq 2$; and is in rational solved form if it has no circular subset. A substitution θ is stable [9] iff for all $x \in X$ there exists $m \in \mathbb{N}$ such that either $\theta^m(x) \notin X$ or $\theta^m(x) = \theta^{m+1}(x)$. The subset of Sub in rational solved form is denoted $RSub$. An equation set E is in rational solved form iff $E = eqn(\theta)$ and $\theta \in RSub$. The following lemma shows that $RSub$ coincides with the set of stable substitutions.

Lemma 2.1. θ is stable iff θ is in rational solved form. ∎

PROOF. If θ is not in rational solved form, then θ includes $\{x_1 \mapsto x_2, \ldots, x_n \mapsto x_1\}$ so that $\theta^m(x_1) \in X$ and $\theta^m(x_1) \neq \theta^{m+1}(x_1)$ for all $m \in \mathbb{N}$. For the other direction, suppose $\theta = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ is in rational solved form. Observe that if $\theta^n(x) \in X$ then $\theta^n(x) = \theta^{n+1}(x)$. Thus θ is stable. ∎

The set of unifiers of E, $unify(E)$, is defined by: $unify(E) = \{\theta \in Sub \mid \forall (s = t) \in E.\theta(s) = \theta(t)\}$. The set of mgu's, $mgu(E)$, is defined by: $mgu(E) = \{\theta \in unify(E) \mid \forall \vartheta \in unify(E).\theta \leq \vartheta\}$. Courcelle [6, Theorem 4.9.2] shows that mgu's are unique up to renaming, that is, if $\theta, \vartheta \in mgu(E)$ then $\theta = \rho \circ \vartheta$ where $\rho \in Rename$, $dom(\rho) \subseteq cod(\vartheta)$ and $cod(\rho) \subseteq cod(\theta)$. The mapping *solve* specifies a simple rational tree unification algorithm [5, 12].

Definition 2.1. The mapping $solve : Eqn \to \wp(Eqn)$ is defined by: $solve(E) = \{E' \mid E \rightsquigarrow^\star E' \wedge E' \neq fail \wedge E' \not\rightsquigarrow E''\}$ where $\star$ is transitive closure and the relation $Eqn \rightsquigarrow Eqn \cup \{fail\}$ is the least binary relation defined by:

1. $\{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)\} \cup E \rightsquigarrow \{s_1 = t_1, \ldots, s_n = t_n\} \cup E$;
2. $\{f(s_1, \ldots, s_m) = g(t_1, \ldots, t_n)\} \cup E \rightsquigarrow fail$ if $f \neq g$;
3. $\{x = x\} \cup E \rightsquigarrow E$;
4. $\{t = x\} \cup E \rightsquigarrow \{x = t\} \cup E$ if $t \notin X$;
5. $\{x = y\} \cup E \rightsquigarrow \{x = y\} \cup \rho(E)$ if $x \in var(E)$ and $\rho = \{x \mapsto y\}$;

6. $\{x = s, x = t\} \cup E \rightsquigarrow \{x = s, s = t\} \cup E$ if $x \neq s$ and $size(s) \leq size(t)$. ▮

Example 2.2. In the following example, for clarity, we annotate $\rightsquigarrow$ with the transformation rule numbers of definition 2.1: $\{x = y,\ f(f(x)) = x,\ y = f(f(f(y)))\}$ $\rightsquigarrow^4$ $\{x = y,\ x = f(f(x)),\ y = f(f(f(y)))\}$ $\rightsquigarrow^5$ $\{x = y,\ y = f(f(y)),\ y = f(f(f(y)))\}$ $\rightsquigarrow^6$ $\{x = y,\ y = f(f(y)),\ f(f(y)) = f(f(f(y)))\}$ $\rightsquigarrow^1$ $\{x = y,\ y = f(f(y)),\ f(y) = f(f(y))\}$ $\rightsquigarrow^1$ $\{x = y,\ y = f(f(y)),\ y = f(y)\}$ $\rightsquigarrow^6$ $\{x = y,\ f(y) = f(f(y)),\ y = f(y)\}$ $\rightsquigarrow^1$ $\{x = y,\ y = f(y)\}$. ▮

Notice that *solve* operates on sets of equations rather than multisets. (This simplifies lemma 3.1.) Observe also that if $E' \in solve(E)$ then there exists $\theta \in RSub$ such that $eqn(\theta) = E'$. The transformation rules of definition 2.1 terminate, preserve equivalence and also return a rational solved form:

Theorem 2.1. [5, 12] $mgu(E) = mgu(eqn(\theta))$ and $\theta \in RSub$ if $eqn(\theta) \in solve(E)$. ▮

To build towards defining the concepts of groundness, sharing and linearity for substitutions in rational solved form, we introduce limits:

Definition 2.2. Let $\{t_n \mid n \in \mathbb{N}\} \subseteq M^\infty(F, X)$. Then $t = \lim_{n\to\infty} t_n$ iff for all $k \in \mathbb{N}$ there exists $l \in \mathbb{N}$ such that $t(\alpha) = t_m(\alpha)$ for all $m \geq l$ and $|\alpha| \leq k$. Furthermore, if $\{\theta_n \mid n \in \mathbb{N}\} \subseteq Sub$ then $\lim_{n\to\infty} \theta_n = \lambda x. \lim_{n\to\infty} \theta_n(x)$. ▮

Note that $\lim_{n\to\infty} \theta_n$ does not necessarily exist even for an increasing chain $\theta_i \leq \theta_{i+1}$ as is illustrated by putting $\theta_{2i-1} = \{x \mapsto y\}$ and $\theta_{2i} = \{y \mapsto x\}$ where $i \in \mathbb{N}$. However, $\lim_{n\to\infty} \theta^n$ exists iff θ is stable [9]. The following lemma establishes that $\lim_{n\to\infty} \theta^n$ is a mgu of $eqn(\theta)$ and follows from a result by Intrigila and Venturini Zilli [9]. Henceforth θ^∞ abbreviates $\lim_{n\to\infty} \theta^n$.

Lemma 2.2. $\theta^\infty \in mgu(eqn(\theta))$ if $\theta \in RSub$. ▮

PROOF. Let $\theta \in RSub$ where $\theta = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$. Then $\theta(x_i) = t_i$ (so θ is a matching [9]) and θ is stable. Put $\tau = \epsilon$ in the proof of Theorem 2.1 of [9] and then $\theta^\infty = \lim_{n\to\infty}(\tau \circ \theta)^n \in mgu(eqn(\vartheta))$. ▮

Example 2.3. Observe that if $\theta_1 = \{x \mapsto f(y, z), y \mapsto c\}$, $\theta_2 = \{x \mapsto f(x, y)\}$, $\theta_3 = \{x \mapsto f(x, y), y \mapsto g(y)\}$, then $\theta_1, \theta_2, \theta_3 \in RSub$ and

$$\theta_1^\infty = \{x \mapsto f(c, z), y \mapsto c\} \quad \theta_2^\infty = \{x \mapsto f(f(f(\ldots, y), y), y)\}$$

$$\theta_3^\infty = \{x \mapsto f(f(\ldots, g(g(\ldots))), g(g(\ldots))), y \mapsto g(g(\ldots))\}$$

Note that ${\theta_i}^\infty \in mgu(eqn(\theta_i))$ and that ${\theta_i}^\infty$ are idempotent. ▮

The following lemma shows that idempotency is no coincidence.

Lemma 2.3. θ^∞ is idempotent if $\theta \in RSub$. ▮

PROOF. For a contradiction let $x \in dom(\theta^\infty) \cap cod(\theta^\infty)$. Then $x \in dom(\theta)$. Also there exists $y \in dom(\theta^\infty)$ and $\alpha \in dom(\theta^\infty(y))$ such that $\theta^\infty(y, \alpha) = x$. Hence there exists $l \in \mathbb{N}$ such that $\theta^m(y, \alpha) = x$ for all $l \leq m$. This contradicts $x \in dom(\theta)$, hence θ^∞ is idempotent. ▮

3. Alternating paths and Linearity

Søndergaard [16] first connected alternating paths with linearity, and there are echos of his alternating paths approach in the abstract unification algorithm of [3]. Furthermore, Dams [7] used alternating paths to repair lemma 2.2 of [3]. We develop this work (by not requiring that unifiers are computed by Herbrand unification) and show how alternating paths can be used to reason about the restricted forms of sharing that arise in rational tree constraint solving.

An alternating path is defined over an equation set E. Distinct occurrences of variables in E are interpreted as the nodes of the graph. The graph has an edge (of type one) between two variables occurrences if the variables are on opposite sides of an equation in E. The graph has an edge (of type two) linking two distinct variable occurrences if the variables coincide. An alternating path is a sequence of edges of alternating type. This idea is formalised below:

Definition 3.1. If $E \in Eqn$, then the binary relations $-_E, \sim_E \subseteq (E \times \mathbb{N}^*)^2$ are the least symmetric relations such that:

- $\langle e, 1.\alpha_1\rangle -_E \langle e, 2.\alpha_2\rangle$ if $e \in E$, $e(1.\alpha_1) \in X$ and $e(2.\alpha_2) \in X$
- $\langle e_1, \alpha_1\rangle \sim_E \langle e_2, \alpha_2\rangle$ if $e_1, e_2 \in E$, $e_1(\alpha_1) = e_2(\alpha_2) \in X$ and ($e_1 \neq e_2$ or $\alpha_1 \neq \alpha_2$).

where $e(1.\alpha) = s(\alpha)$ and $e(2.\alpha) = t(\alpha)$ if $e = (s = t)$. A sequence $\langle e_1, \alpha_1\rangle.\langle e_1, \beta_1\rangle \ldots \langle e_n, \alpha_n\rangle.\langle e_n, \beta_n\rangle \in APath_E$ iff $\langle e_i, \alpha_i\rangle -_E \langle e_i, \beta_i\rangle$ for all $1 \leq i \leq n$ and $\langle e_i, \beta_i\rangle \sim_E \langle e_{i+1}, \alpha_{i+1}\rangle$ for all $1 \leq i < n$ where $n \in \mathbb{N}$. ▮

Example 3.1. Let $E_1 = \{e\}$ where $e = (f(x, y, z) = f(u, u, v))$ and $E_2 = \{e_1, e_2, e_3\}$ where $e_1 = (x = u), e_2 = (y = u)$ and $e_3 = (z = v)$. Observe $\langle e, 1.1\rangle -_{E_1} \langle e, 2.2\rangle$, $\langle e, 2.2\rangle \sim_{E_1} \langle e, 2.1\rangle$ and $\langle e, 2.1\rangle -_{E_1} \langle e, 1.2\rangle$ and thus $\langle e, 1.1\rangle.\langle e, 2.2\rangle.\langle e, 2.1\rangle.\langle e, 1.2\rangle \in APath_{E_1}$ where $e(1.1) = x$ and $e(1.2) = y$ so there exists an alternating path between x and y in E_1. Note also that $\langle e_1, 1\rangle.\langle e_1, 2\rangle.\langle e_2, 2\rangle.\langle e_2, 1\rangle \in APath_{E_2}$ with $e_1(1) = x$ and $e_2(1) = y$ so the same is true for E_2. Note that $E_1 \leadsto E_2$. ▮

One key result on alternating paths is that the iterative process of transforming an equation set E into a rational solved form E', cannot create new alternating paths. Specifically, if there exists an alternating path in E' whose ends points connect the variables x and y, then the same must be true of E. This is illustrated in example 3.1 and formalised in lemma 3.1.

Lemma 3.1. If $E \leadsto E'$ and $\langle e_1', \alpha_1'\rangle \ldots \langle e_m', \beta_m'\rangle \in APath_{E'}$, then there exists $\langle e_1, \alpha_1\rangle \ldots \langle e_n, \beta_n\rangle \in APath_E$ with $e_1(\alpha_1) = e_1'(\alpha_1')$ and $e_n(\beta_n) = e_m'(\beta_m')$. ▮

To aid the proof of correctness, we introduce the concept of variable multiplicity:

Definition 3.2. The (multiplicity) map $\chi : M^\infty(F, V) \to \{0, 1, 2\}$ is defined by: $\chi(t) = \max(\{0\} \cup \{\chi(x, t) \mid x \in var(t)\})$ where $\chi(x, t) = \min(\{2, |occ(x, t)|\})$. ▮

If $\chi(t) = 0$, t is ground; if $\chi(t) = 1$, t is linear; and if $\chi(t) = 2$, t is non-linear. The singleton set $\{0\}$ simply ensures that $\chi(t)$ is well-defined when t is ground.

PROOF. (for lemma 3.1) Suppose $E \rightsquigarrow E'$ and $a' = \langle e_1', \alpha_1' \rangle \ldots \langle e_m', \beta_m' \rangle \in APath_{E'}$. Since $E' \neq fail$, rules 1 and 3–6 of definition 2.1 only need be considered:

1. Suppose $E = \{e\} \cup E''$ and $E' = \{s_1 = t_1, \ldots, s_k = t_k\} \cup E''$ where $e = (f(s_1, \ldots, s_k) = f(t_1, \ldots, t_k))$. Construct a from a' by replacing each pair of the form $\langle s_i = t_i, j.\alpha \rangle$ with $\langle e, j.i.\alpha \rangle$ where $j \in \{1, 2\}$. Observe that $a \in APath_E$ and that a satisfies the two end point properties.

3. Suppose $E = \{x = x\} \cup E'$. Immediate since $APath_{E'} \subseteq APath_E$.

4. Suppose $E = \{t = x\} \cup E''$ and $E' = \{x = t\} \cup E''$. Construct a from a' by replacing pairs of the form $\langle t = x, 1.\alpha \rangle$ and $\langle t = x, 2.\alpha \rangle$ with $\langle x = t, 2.\alpha \rangle$ and $\langle x = t, 1.\alpha \rangle$ respectively. Note that $a \in APath_E$ and that a satisfies the desired properties.

5. Suppose $E = \{x = y\} \cup E''$ and $E' = \{x = y\} \cup \rho(E'')$ where $\rho = \{x \mapsto y\}$. Construct $a = \langle e_1, \alpha_1 \rangle \ldots \langle e_n, \beta_n \rangle$ from a' by replacing the pairs $\langle \rho(e), \alpha \rangle$ with $\langle e, \alpha \rangle$ where $e \in E''$. Observe that adjacent pairs in a of the form $\langle e_i, \alpha_i \rangle.\langle e_i, \beta_i \rangle$ satisfy $\langle e_i, \alpha_i \rangle -_E \langle e_i, \beta_i \rangle$. If $e_1 = (x = y)$ or ($e_1 \in E''$ and $e_1(\alpha_1) \neq x$), then $e_1(\alpha_1) = e_1'(\alpha_1')$. Otherwise, if $e_1 \in E''$ and $e_1(\alpha_1) = x$, then replace a with $\langle x = y, 2 \rangle.\langle x = y, 1 \rangle.a$ so that $e_1(\alpha_1) = e_1'(\alpha_1')$. An analogous construction can be used to obtain $e_n(\alpha_n) = e_m'(\alpha_m') = y$. Now consider the adjacent pairs $\langle e_i, \beta_i \rangle.\langle e_{i+1}, \alpha_{i+1} \rangle$. If $e_i = (x = y) = e_{i+1}$, $\langle x = y, \beta_i \rangle \sim_{E'} \langle x = y, \alpha_{i+1} \rangle$ then $\beta_i \neq \alpha_{i+1}$, which contradicts $\chi(x = y) = 1$. Thus there are three cases to consider:

   - Now consider $e_i = (x = y)$ and $e_{i+1} \in E''$. Since $x \notin var(\rho(e_{i+1}))$, $\beta_i = 2$ and either $e_{i+1}(\alpha_{i+1}) = x$ or $e_{i+1}(\alpha_{i+1}) = y$.
     - Suppose $e_{i+1}(\alpha_{i+1}) = y$. Then $\langle x = y, 2 \rangle \sim_E \langle e_{i+1}, \alpha_{i+1} \rangle$.
     - Suppose $e_{i+1}(\alpha_{i+1}) = x$. Observe $a' = \ldots \langle x = y, 1 \rangle.\langle x = y, 2 \rangle.\langle \rho(e_{i+1}), \alpha_{i+1} \rangle \ldots$. Assume that $a' = \ldots \langle e_{i-1}', \beta_{i-1}' \rangle.\langle x = y, 1 \rangle.\langle x = y, 2 \rangle.\langle \rho(e_{i+1}), \alpha_{i+1} \rangle \ldots$. Since $\chi(x = y) = 1$, $e_{i-1}' \neq (x = y)$ and thus $e_{i-1}' = \rho(e)$ where $e \in E''$. Hence $x \notin var(e_{i-1}')$ which contradicts $\langle e_{i-1}', \beta_{i-1}' \rangle \sim_{E'} \langle x = y, 1 \rangle$. Thus $i = 1$ and hence $a' = \langle x = y, 1 \rangle.\langle x = y, 2 \rangle.\langle \rho(e_2), \alpha_2 \rangle.\langle \rho(e_2), \beta_2 \rangle \ldots$. Remove the first two pairs from a to give $a = \langle e_2, \alpha_2 \rangle.\langle e_2, \beta_2 \rangle \ldots$ to ensure that the end point property $(x = y)(1) = e_2(\alpha_2)$ holds.
   - Now consider $e_i \in E''$ and $e_{i+1} = (x = y)$. Analogous to the previous case.
   - Now consider $e_i, e_{i+1} \in E''$.
     - Suppose $e_i(\beta_i) = e_{i+1}(\alpha_{i+1})$. Since $\langle \rho(e_i), \beta_i \rangle \sim_{E'} \langle \rho(e_{i+1}), \alpha_{i+1} \rangle$, $\rho(e_i) \neq \rho(e_{i+1})$ or $\beta_i \neq \alpha_{i+1}$. Thus $e_i \neq e_{i+1}$ or $\beta_i \neq \alpha_{i+1}$ and hence $\langle e_i, \beta_i \rangle \sim_E \langle e_{i+1}, \alpha_{i+1} \rangle$.
     - Suppose $e_i(\beta_i) = x$ and $e_{i+1}(\alpha_{i+1}) = y$. Observe that $\langle e_i, \beta_i \rangle \sim_E \langle x = y, 1 \rangle -_E \langle x = y, 2 \rangle \sim_E \langle e_{i+1}, \alpha_{i+1} \rangle$ and therefore insert $\langle x = y, 1 \rangle.\langle x = y, 2 \rangle$ between $\langle e_i, \beta_i \rangle$ and $\langle e_{i+1}, \alpha_{i+1} \rangle$ in a.
     - Suppose $e_i(\beta_i) = y$ and $e_{i+1}(\alpha_{i+1}) = x$. Analogous to the previous case.

6. Suppose $E = \{x = s, x = t\} \cup E''$ and $E' = \{x = s, s = t\} \cup E''$. Without loss of generality $(x = t) \notin E''$. Construct a from a' by replacing:

   (a) $\langle x = s, \alpha_i\rangle.\langle x = s, 2.\alpha\rangle.\langle s = t, 1.\alpha\rangle.\langle s = t, \beta_{i+1}\rangle$ with $\langle x = t, \alpha_i\rangle.\langle x = t, \beta_{i+1}\rangle$;
   (b) $\langle s = t, \alpha_i\rangle.\langle s = t, 1.\alpha\rangle.\langle x = s, 2.\alpha\rangle.\langle x = s, \beta_{i+1}\rangle$ with $\langle x = t, \alpha_i\rangle.\langle x = t, \beta_{i+1}\rangle$;

   and, if (a) and (b) are not applicable, then replacing:

   (c) $\langle s = t, \alpha_i\rangle.\langle s = t, \beta_i\rangle$ with $\langle x = t, 2.\alpha\rangle.\langle x = t, 1\rangle.\langle x = s, 1\rangle.\langle x = s, 2.\beta\rangle$ where $\alpha_i = 2.\alpha$ and $\beta_i = 1.\beta$;
   (d) $\langle s = t, \alpha_i\rangle.\langle s = t, \beta_i\rangle$ with $\langle x = s, 2.\alpha\rangle.\langle x = s, 1\rangle.\langle x = t, 1\rangle.\langle x = t, 2.\beta\rangle$ where $\alpha_i = 1.\alpha$ and $\beta_i = 2.\beta$.

   In (a), if $\langle e_{i-1}, \beta_{i-1}\rangle \sim_{E'} \langle x = s, \alpha_i\rangle$, then $\alpha_i = 1$ necessarily and $e_{i-1} \neq (x = t)$ since $(x = t) \notin E''$ so that $\langle e_{i-1}, \beta_{i-1}\rangle \sim_E \langle x = t, \alpha_i\rangle$. Similarly $\langle x = t, \beta_{i+1}\rangle \sim_E \langle e_{i+2}, \alpha_{i+2}\rangle$ if $\langle s = t, \beta_{i+1}\rangle \sim_{E'} \langle e_{i+2}, \alpha_{i+2}\rangle$. An analogous argument can be applied for substitution (b). In (c), if $\langle e_{i-1}, \beta_{i-1}\rangle \sim_{E'} \langle s = t, \alpha_i\rangle$, then $\alpha_i = 2.\alpha$ necessarily and $e_{i-1} \neq (x = t)$ since $(x = t) \notin E''$ so that $\langle e_{i-1}, \beta_{i-1}\rangle \sim_E \langle x = t, \alpha_i\rangle$. Furthermore, if $\langle s = t, \beta_i\rangle \sim_{E'} \langle e_{i+1}, \alpha_{i+1}\rangle$, then since (c) is applied rather than (b), it follows that $e_{i+1} \neq (x = s)$ or $\alpha_{i+1} \neq 2.\beta$ and hence $\langle x = s, 2.\beta\rangle \sim_E \langle e_{i+1}, \alpha_{i+1}\rangle$. An analogous argument can be applied for substitution (d). Thus all the adjacent pairs $\langle e_i, \beta_i\rangle.\langle e_{i+1}, \alpha_{i+1}\rangle$ of a satisfy $\langle e_i, \beta_i\rangle \sim_E \langle e_{i+1}, \alpha_{i+1}\rangle$. Observe also that $\langle e_i, \alpha_i\rangle -_E \langle e_i, \beta_i\rangle$ for pairs $\langle e_i, \alpha_i\rangle.\langle e_i, \beta_i\rangle$ of a. ∎

Lemma 3.2 explains where the end points of an alternating path can occur for simple equation sets of the form $\{s = t\}$. It is prelude to the main result in this section, proposition 3.1.

Lemma 3.2. If $\langle e, \alpha_1\rangle \ldots \langle e, \beta_n\rangle \in APath_{\{e\}}$ where $e = (s = t)$ then:

$$\begin{array}{llll} \alpha_1 = 1.\alpha \text{ and } \beta_n = 2.\beta & \text{or} & \alpha_1 = 2.\alpha,\ \beta_n = 2.\beta \text{ and } \chi(s) = 2 & \text{or} \\ \alpha_1 = 2.\alpha \text{ and } \beta_n = 1.\beta & \text{or} & \alpha_1 = 1.\alpha,\ \beta_n = 1.\beta \text{ and } \chi(t) = 2 & \end{array}$$

∎

Proof. By induction on n. The base case $\langle e, \alpha_1\rangle.\langle e, \beta_1\rangle \in APath_{\{e\}}$ is immediate so let $\langle e, \alpha_1\rangle \ldots \langle e, \beta_n\rangle.\langle e, \alpha_{n+1}\rangle.\langle e, \beta_{n+1}\rangle \in APath_{\{e\}}$ where $e = (s = t)$. By the inductive hypothesis there are 4 cases to consider:

- Suppose $\alpha_1 = 1.\alpha$ and $\beta_n = 2.\beta$. If $\alpha_{n+1} = 1.\alpha'$ then $\beta_{n+1} = 2.\beta'$ and so the result follows. Otherwise $\alpha_{n+1} = 2.\alpha'$ and by the definition of $\sim_{\{e\}}$, $\alpha' \neq \beta$ so that $\chi(t) = 2$. Note that $\beta_{n+1} = 1.\beta'$ so again the result follows.
- Suppose $\alpha_1 = 2.\alpha$ and $\beta_n = 1.\beta$. Similar to the previous case.
- Suppose $\alpha_1 = 2.\alpha$, $\beta_n = 2.\beta$ and $\chi(s) = 2$. If $\alpha_{n+1} = 1.\alpha'$ then $\beta_{n+1} = 2.\beta'$ and the result follows since $\chi(s) = 2$. Else $\alpha_{n+1} = 2.\alpha'$ and thus $\beta_{n+1} = 1.\beta'$.
- Suppose $\alpha_1 = 1.\alpha$, $\beta_n = 1.\beta$ and $\chi(t) = 2$. Similar to the previous case. ∎

Example 3.2. Recall $E_1 = \{e\}$ of example 3.1 where $e = (f(x, y, z) = f(u, u, v))$. Since $\chi(f(x, y, z)) = 1$, then by lemma 3.2 no alternating paths exists between

u and v in E_1. A corollary of this and lemma 3.1 is that no alternating paths exist between u and v in E_2 of example 3.1 since $E_1 \rightsquigarrow E_2$. ∎

Proposition 3.1 details the forms of sharing that can arise in $\theta \in mgu(\{s = t\})$. Although the lemma is similar to another stated in [7], our proof does not require that θ is computed by Herbrand unification. This is a crucial difference.

Proposition 3.1. If $\theta \in mgu(\{s = t\})$, $x \neq y$ and $var(\theta(x)) \cap var(\theta(y)) \neq \emptyset$ then:

$$\begin{array}{lcl} x \in var(s) \text{ and } y \in var(t) & \text{or} & x, y \in var(t) \text{ and } \chi(s) = 2 \quad \text{or} \\ x \in var(t) \text{ and } y \in var(s) & \text{or} & x, y \in var(s) \text{ and } \chi(t) = 2 \end{array}$$

∎

PROOF. Let $\theta \in mgu(\{s = t\})$ such that $var(\theta(x)) \cap var(\theta(y)) \neq \emptyset$. Let $E = eqn(\vartheta) \in solve(\{s = t\})$. By lemma 2.2, $\vartheta^\infty \in mgu(E)$ and by theorem 2.1, $mgu(\{s = t\}) = mgu(E)$ so that $\theta \in mgu(E)$ and thus there exists $\rho \in Rename$ such that $\rho \circ \theta = \vartheta^\infty$. Thus $var(\vartheta^\infty(x)) \cap var(\vartheta^\infty(y)) \neq \emptyset$.

- Suppose $x \in dom(\vartheta^\infty)$ and $y \notin dom(\vartheta^\infty)$. Thus there exists $\alpha \in dom(\vartheta^\infty(x))$ such that $\vartheta^\infty(x, \alpha) = y$. Hence there exist a minimum $m \in \mathbb{N}$ such that $\vartheta^m(x, \alpha) = y$. Therefore there exists $e_1{}' = (x = t_1)$, $e_2{}' = (x_2 = t_2)$, $\ldots$, $e_m{}' = (x_m = t_m) \in eqn(\vartheta) = E$ where $t_1(\beta_1) = x_2$, $\ldots$, $t_{m-1}(\beta_{m-1}) = x_m$, $t_m(\beta_m) = y$. Hence $\langle e_1{}', 1\rangle.\langle e_1{}', 2.\beta_1\rangle \ldots \langle e_m{}', 1\rangle.\langle e_m{}', 2.\beta_m\rangle \in APath_E$ where $e_1{}'(1) = x$ and $e_m{}'(2.\beta_m) = y$. Thus, by lemma 3.1, there exists $\langle e_1, \alpha_1\rangle \ldots \langle e_n, \beta_n\rangle \in APath_{\{s=t\}}$ with $e_1(\alpha_1) = x$ and $e_n(\beta_n) = y$. The result then follows by lemma 3.2.
- Suppose $x \notin dom(\vartheta^\infty)$ and $y \in dom(\vartheta^\infty)$. Analogous to the previous case.
- Suppose $x \in dom(\vartheta^\infty)$ and $y \in dom(\vartheta^\infty)$. A minimal $\langle m, n\rangle \in \mathbb{N}^2$ exists (in the lexicographical ordering) with $z \in var(\vartheta^m(x)) \cap var(\vartheta^n(y))$ since $var(\vartheta^\infty(x)) \cap var(\vartheta^\infty(y)) \neq \emptyset$. Thus there exists $e_1{}' = (x = t_1)$, $e_2{}' = (x_2 = t_2), \ldots, e_m{}' = (x_m = t_m)$, $e_1{}'' = (y = s_1)$, $e_2{}'' = (y_2 = s_2), \ldots, e_n{}'' = (y_n = s_n) \in eqn(\vartheta) = E$ with $t_1(\beta_1) = x_2$, $\ldots$, $t_{m-1}(\beta_{m-1}) = x_m$, $t_m(\beta_m) = z$, $s_1(\beta_1{}') = y_2$, $\ldots$, $s_{n-1}(\beta_{n-1}{}') = y_n$, $s_n(\beta_n{}') = z$. Since $\langle m, n\rangle$ is minimal, $e_m{}' \neq e_n{}''$ and thus $\langle e_1{}', 1\rangle \ldots \langle e_m{}', 2.\beta_m\rangle.\langle e_n{}'', 2.\beta_n{}'\rangle \ldots \langle e_1{}'', 1\rangle \in APath_E$ where $e_1{}'(1) = x$ and $e_1{}''(1) = y$. Continue with lemmas 3.1 and 3.2 as above. ∎

It is important to realise that proposition 3.1 does not make any statement about how the mgu is computed: it might be computed by Herbrand unification (with or without the occur-check), or a version of *solve* adapted to multisets, or even a nearly-linear, memoising rational tree algorithm [11]. In fact, Herbrand unification can just be regarded as an incomplete implementation of rational tree unification: it computes an mgu iff it terminates. By building on this result we establish correctness for pair-sharing across a range of Prolog implementations. The following example shows how the proposition can be used to reason about the absence of sharing. This is the main application of the result.

Example 3.3. Let $\theta_1 \in mgu(\{f(x, y) = f(u, v)\})$, $\theta_2 \in mgu(\{x = f(x, y, z)\})$ and $\theta_3 \in mgu(\{f(x, x, y) = f(x, y, z)\})$. Since $\chi(f(x, y)) = \chi(f(u, v)) = 1$, proposition 3.1 ensures $var(\theta_1(x)) \cap var(\theta_1(y)) = \emptyset = var(\theta_1(u)) \cap var(\theta_1(v))$. It also predicts that $var(\theta_2(y)) \cap var(\theta_2(z)) = \emptyset$. Observe that $var(\theta_3(x)) \cap var(\theta_3(y)) \neq \emptyset$, but unlike lemma 2.2 of [3], the proposition does not predict that $x \notin var(f(x,y,z))$ or $y \notin var(f(x,y,z))$. ▌

4. Pair-sharing

Analyses can be used in isolation, but an increasing trend is to combine domains to improve accuracy [4]. In our treatment, we assume that pair-sharing will be used with a groundness domain, say Def or Pos [1]. Thus, unlike the Søndergaard domain [3, 16], our pair-sharing domain, PS, does not capture or propagate groundness. We simply assume that a rich groundness domain can be projected onto a simple groundness domain, G (that is isomorphic to Con [14]). PS and G are defined below in terms of a finite set of program variables $V \subseteq X$.

Definition 4.1. $G_V = \wp(V)$, $PS_V = \wp(\{\{x,y\} \mid x,y \in V\})$. ▌

$\langle G_V, \subseteq, \cap, \cup\rangle$ and $\langle PS_V, \subseteq, \cap, \cup\rangle$ are finite lattices which respectively have maximal ascending chains of length $n+1$ and $\frac{1}{2}(n^2+n+2)$ where $|V| = n$. For example, if $V = \{x,y\}$ and $n = 2$, then PS_V contains the maximal chain $\emptyset$, $\{\{x\}\}$, $\{\{x\},\{y\}\}$, $\{\{x\},\{x,y\},\{y\}\}$ of length $\frac{1}{2}(2^2+2+2) = 4$. Concretisation maps are introduced to explain how groundness and sharing descriptions can be interpreted as sets of substitutions in rational solved form.

Definition 4.2. The concretisation maps $\gamma_V^G : G_V \to \wp(RSub)$ and $\gamma_V^{PS} : PS_V \to \wp(RSub)$ are defined by: $\gamma_V^G(U) = \{\theta \in RSub \mid \forall u \in U.\, var(\theta^\infty(u)) = \emptyset\}$ and $\gamma_V^{PS}(\pi) = \{\theta \in RSub \mid \alpha_V(\theta^\infty) \subseteq \pi\}$ where

$$\alpha_V(\theta) = \left\{ \{u,v\} \subseteq V \;\middle|\; \begin{array}{l} (u \neq v \wedge var(\theta(u)) \cap var(\theta(v)) \neq \emptyset) \vee \\ (u = v \wedge \chi(\theta(u)) = 2) \end{array} \right\}$$

▌

Abstraction maps $\alpha_V^G : \wp(RSub) \to G_V$ and $\alpha_V^{PS} : \wp(RSub) \to PS_V$ can be induced from the concretisation maps $\gamma_V^G : G_V \to \wp(RSub)$ and $\gamma_V^{PS} : PS_V \to \wp(RSub)$ in the usual way. If θ is idempotent and $\theta \in \gamma_V^G(U) \cap \gamma_V^{PS}(\pi)$ then $var(\theta(u)) = \emptyset$ for all $u \in U$ and $\alpha_V(\theta) \subseteq \pi$. Thus γ_V^G and γ_V^{PS} are backward compatible in the sense that they coincide with the classic concretisation maps [3] for idempotent substitutions.

Example 4.1. Let $V = \{x,y,z\}$ and consider again $\theta_1, \theta_2, \theta_3 \in RSub$ of example 2.3. Then $\alpha_V(\theta_1^\infty) = \{\{x,z\}\}$, $\alpha_V(\theta_2^\infty) = \{\{x,y\},\{x\}\}$ and $\alpha_V(\theta_3^\infty) = \emptyset$ so that $\theta_1, \theta_2, \theta_3 \in \gamma_V^{PS}(\{\{x,y\},\{x,z\},\{y\}\})$. Note that $\theta_1, \theta_3 \in \gamma_V^G(\{y\})$, $\theta_3 \in \gamma_V^G(\{x,y\})$ and $\theta_1, \theta_2, \theta_3 \in \gamma_{\{x,y\}}^{PS}(\{\{x,y\}, \{y\}\})$. ▌

To avoid making worst-case assumptions about aliasing, we need to recognise when linear terms participate in abstract unification. Thus, to conservatively calculate the variable multiplicity of a term t in the context of a set of substitutions represented by π, we introduce an abstract multiplicity map χ.

Definition 4.3. The map $\chi : M(F,V) \times PS_V \to \{0,1,2\}$ is defined by:

$$\chi(t,\pi) = \begin{cases} 0 & \text{if } \chi(t) = 0 \\ 2 & \text{else if } \chi(t) = 2 \\ 2 & \text{else if } \exists u, v \in var(t).\{u,v\} \in \pi \\ 1 & \text{otherwise} \end{cases}$$

The following lemma explains how t and π can be inspected to make an inference about the linearity of $\theta^\infty(t)$. Note how κ is used to ground those variables of t that U records as ground.

Lemma 4.1. $\chi(\theta^\infty(t)) \leq \chi(\kappa(t),\pi)$ if $\theta \in \gamma_V^G(U) \cap \gamma_V^{PS}(\pi)$ and $\kappa = \{u \mapsto c \mid u \in U\}$.

PROOF. Let $\theta \in \gamma_V^G(U) \cap \gamma_V^{PS}(\pi)$ and put $\kappa = \{u \mapsto c \mid u \in U\}$. Since $\theta \in \gamma_V^G(U)$, $\theta \in RSub$ and thus, by lemma 2.2, θ^∞ exists.

- Suppose $\chi(\kappa(t),\pi) = 0$. Since $var(\kappa(t)) = \emptyset$, $var(t) \subseteq U$ and therefore $\forall u \in var(t).var(\theta^\infty(u)) = \emptyset$ so that $\chi(\theta^\infty(t)) = 0$.
- Suppose $\chi(\kappa(t),\pi) = 1$ and, for a contradiction, that $\chi(\theta^\infty(t)) = 2$.
  - Suppose there exists $u \in var(t)$ such that $var(\theta^\infty(u)) \neq \emptyset$ and $\chi(u,t) = 2$. Thus $u \notin U$ so that $u \notin dom(\kappa)$ so that $\chi(\kappa(t)) = 2$ which contradicts $\chi(\kappa(t),\pi) = 1$.
  - Suppose there exist $u, v \in var(t)$ such that $var(\theta^\infty(u)) \neq \emptyset$, $var(\theta^\infty(v)) \neq \emptyset$ and $u \neq v$. Therefore $\{u,v\} \in \pi$ and since $u \notin U$ and $v \notin U$, $u, v \in var(\kappa(t))$ which is a contradiction.
  - Suppose there exists $u \in var(t)$ such that $\chi(\theta^\infty(u)) = 2$. Thus $\{u\} \in \pi$ and $u \in var(\kappa(t))$ which is a contradiction.
- Suppose $\chi(\kappa(t),\pi) = 2$. The result is immediate.

The operator $mgu_V(s,t,\pi)$, defined below, constitutes the basis for our sharing analysis. It solves the equation $s = t$ in the presence of the abstract substitution π returning the composition of the unifier with π. Since s, t and π are finite, $mgu_V(s,t,\pi)$ is finitely computable, and thus the definition can be interpreted as a sharing analysis algorithm.

Definition 4.4.

$$mgu_V(s,t,\pi) = \pi \cup \left\{ \{u,v\} \subseteq V \;\middle|\; \begin{array}{l} (x \in var(s) \wedge x \simeq u \wedge v \simeq y \wedge\ y \in var(t)) \vee \\ (x, y \in var(s) \wedge x \simeq u \wedge v \simeq y \wedge \chi(t,\pi) = 2) \vee \\ (x, y \in var(t) \wedge x \simeq u \wedge v \simeq y \wedge \chi(s,\pi) = 2) \end{array} \right\}$$

where $u \simeq v$ iff $u = v$ or $\{u,v\} \in \pi$.

Abstract unification algorithms usually operate on simple equations/bindings of the form $x = t$. In our presentation, however, s can be a non-variable term. This simplifies the analysis of some builtins. For example, to trace the effect of the call `sort([X, Y, Z], [U, U, V])` in the context of a description π, we just calculate $mgu_V(s,t,\pi)$ where s = `[X, Y, Z]` and t = `[U, U, V]`. This is because the call will reduce to the one of the unifications: `[X, Y, Z]` $= t$, `[X, Z, Y]` $= t, \ldots,$ `[Z, X, Y]` $= t$, `[Z, Y, X]` $= t$ and all of these behaviours are traced by $mgu_V(s,t,\pi)$. For an equation $s = t$, it can be more precise to iterate mgu_V over the equations of $solve(\{s = t\})$ rather than compute $mgu_V(s,t,\pi)$ directly. Note also that our $mgu_V(s,t,\pi)$ is basically a composition of the $\circ$ and $soln$ maps of [3]. (This reduces the number of operators that need to be implemented.)

To establish the correctness of abstract unification we state and prove two lemmas. Lemma 4.2 explains how sharing and composition of substitutions interact. Note that δ is not necessarily idempotent. Lemma 4.3 details how mgu's for the equation set $E \cup eqn(\theta)$ relate to those of $\theta^\infty(E)$ (assuming the limit exists).

Lemma 4.2. If $var(\delta \circ \theta(u)) \cap var(\delta \circ \theta(v)) \neq \emptyset$ and θ is idempotent, then either:
- $var(\theta(u)) \cap var(\theta(v)) \neq \emptyset$ or
- there exist $x, y \in var(\theta(u)) \cup var(\theta(v))$ such that $x \neq y$, $var(\theta(u)) \cap var(\theta(x)) \neq \emptyset$, $var(\delta(x)) \cap var(\delta(y)) \neq \emptyset$ and $var(\theta(y)) \cap var(\theta(v)) \neq \emptyset$. ▮

PROOF. Suppose $var(\delta \circ \theta(u)) \cap var(\delta \circ \theta(v)) \neq \emptyset$ and $var(\theta(u)) \cap var(\theta(v)) = \emptyset$. Note there exist $x \in var(\theta(u))$, $y \in var(\theta(v))$ such that $var(\delta(x)) \cap var(\delta(y)) \neq \emptyset$ and $x \neq y$. We need to show $var(\theta(u)) \cap var(\theta(x)) \neq \emptyset$.

- Suppose $x \in cod(\theta)$.
  - Suppose $x = u$. Then $var(\theta(u)) \cap var(\theta(x)) = var(\theta(u))$. The result follows because $x \in var(\theta(u))$.
  - Suppose $x \neq u$. Since θ is idempotent and $x \in cod(\theta)$, then $x \notin dom(\theta)$. Thus $\theta(x) = x$ and as $x \in var(\theta(u))$, it follows that $var(\theta(u)) \cap var(\theta(x)) \neq \emptyset$.
- Suppose $x \notin cod(\theta)$. Because $x \in var(\theta(u))$, $u = x$ and thus $x \in var(\theta(x))$. Hence $var(\theta(u)) \cap var(\theta(x)) \neq \emptyset$.

It similarly follows that $var(\theta(y)) \cap var(\theta(v)) \neq \emptyset$. ▮
The following example illustrates that two conditions of lemma 4.2 do not necessarily follow if the idempotency of θ is relaxed.

Example 4.2. Suppose $\theta = \{x \mapsto f, u \mapsto x\}$ and $\delta = \{v \mapsto x\}$. Then $var(\delta \circ \theta(u)) \cap var(\delta \circ \theta(v)) \neq \emptyset$ but $var(\theta(u)) \cap var(\theta(v)) = \emptyset$ and $var(\theta(u)) \cap var(\theta(x)) = \emptyset$. ▮

Lemma 4.3. $\delta \circ \theta^\infty \in mgu(E \cup eqn(\theta))$ if $\delta \in mgu(\theta^\infty(E))$. ▮

PROOF. Let $\delta \in mgu(\theta^\infty(E))$. Thus $\delta \circ \theta^\infty \in unify(E)$. By lemma 2.2 $\theta^\infty \in mgu(eqn(\theta))$ so that $\delta \circ \theta^\infty \in unify(eqn(\theta))$ and thus $\delta \circ \theta^\infty \in unify(E \cup eqn(\theta))$. Let $\vartheta \in unify(E \cup eqn(\theta))$. Thus $\vartheta \in unify(eqn(\theta))$ and hence there exists $\phi \in Sub$ such that $\vartheta = \phi \circ \theta^\infty$. But $\phi \in unify(\theta^\infty(E) \cup \theta^\infty(eqn(\theta))) = unify(\theta^\infty(E))$ so it follows that $\delta \leq \phi$ and thus $\delta \circ \theta^\infty \leq \phi \circ \theta^\infty = \vartheta$ as required. ▮

The following theorem establishes the correctness of abstract unification. Note that U describes the state θ immediately prior to solving the equation $s = t$.

Theorem 4.1. $mgu(\{s = t\} \cup eqn(\theta)) \subseteq \gamma_V^{PS}(mgu_V(\kappa(s), \kappa(t), \pi))$ if $\theta \in \gamma_V^G(U) \cap \gamma_V^{PS}(\pi)$ and $\kappa = \{u \mapsto c \mid u \in U\}$. ∎

PROOF. Suppose $\varphi \in mgu(\{s = t\} \cup eqn(\theta))$ and $\theta \in \gamma_V^G(U) \cap \gamma_V^{PS}(\pi)$. Since $\theta \in RSub$, θ^∞ exists and let $\delta \in mgu(\{\theta^\infty(s) = \theta^\infty(t)\})$. By lemma 4.3, $\delta \circ \theta^\infty \in mgu(\{s = t\} \cup eqn(\theta))$ and there exists $\rho \in Rename$ such that $\varphi = \rho \circ \delta \circ \theta^\infty$.

- Suppose $var(\rho \circ \delta \circ \theta^\infty(u)) \cap var(\rho \circ \delta \circ \theta^\infty(v)) \neq \emptyset$ where $u \neq v$. We need to show $\{u, v\} \in mgu_V(\kappa(s), \kappa(t), \pi)$. Since θ^∞ is idempotent then by lemma 4.2 there are 2 cases:
  - Suppose $var(\theta^\infty(u)) \cap var(\theta^\infty(v)) \neq \emptyset$. Since $\theta \in \gamma_V^{PS}(\pi)$, $\{u, v\} \in \pi$ and thus $\{u, v\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.
  - Suppose there exist $x', y' \in var(\theta^\infty(u)) \cup var(\theta^\infty(v))$ with $x' \neq y'$, $var(\theta^\infty(u)) \cap var(\theta^\infty(x')) \neq \emptyset$, $var(\rho \circ \delta(x')) \cap var(\rho \circ \delta(y')) \neq \emptyset$ and $var(\theta^\infty(y')) \cap var(\theta^\infty(v)) \neq \emptyset$. Because $\rho \in Rename$, ρ is injective and therefore $var(\delta(x')) \cap var(\delta(y')) \neq \emptyset$. But $\delta \in mgu(\{\theta^\infty(s) = \theta^\infty(t)\})$ and so by proposition 3.1 there are 4 cases:
    - Suppose $x' \in var(\theta^\infty(s))$ and $y' \in var(\theta^\infty(t))$. Since $var(\theta^\infty(u)) \cap var(\theta^\infty(x')) \neq \emptyset$, there exists $x \in var(s)$ such that $var(\theta^\infty(u)) \cap var(\theta^\infty \circ \theta^\infty(x)) \neq \emptyset$ and since θ^∞ is idempotent, $var(\theta^\infty(u)) \cap var(\theta^\infty(x)) \neq \emptyset$. Similarly there exists $y \in var(t)$ such that $var(\theta^\infty(y)) \cap var(\theta^\infty(v)) \neq \emptyset$. Because $\theta \in \gamma_V^{PS}(\pi)$, $x \simeq u$ and $v \simeq y$, since $\theta \in \gamma_V^G(U)$, $x, y \notin U$ and thus $x \in var(\kappa(s))$ and $y \in var(\kappa(t))$. Hence $\{u, v\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.
    - Suppose $y' \in var(\theta^\infty(s))$ and $x' \in var(\theta^\infty(t))$. Analogous to the previous case.
    - Suppose $x', y' \in var(\theta^\infty(s))$ and $\chi(\theta^\infty(t)) = 2$. As in the previous case but one, there exist $x, y \in var(s)$ such that $var(\theta^\infty(u)) \cap var(\theta^\infty(x)) \neq \emptyset$ and $var(\theta^\infty(y)) \cap var(\theta^\infty(v)) \neq \emptyset$. Thus $x \simeq u$, $v \simeq y$ and $x, y \notin U$ and therefore $x, y \in var(\kappa(s))$. By lemma 4.1, $2 = \chi(t, \theta^\infty) \leq \chi(\kappa(t), \pi) \leq 2$ and thus $\{u, v\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.
    - Suppose $x', y' \in var(\theta^\infty(t))$ and $\chi(\theta^\infty(s)) = 2$. Analogous to the previous case.
- Suppose $\chi(\rho \circ \delta \circ \theta^\infty(u)) = 2$. We need to show $\{u\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.
  - Suppose $\chi(\theta^\infty(u)) = 2$. Since $\theta \in \gamma_V^{PS}(\pi)$, $\{u\} \in \pi$ and therefore $\{u\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.
  - Suppose $\chi(\theta^\infty(u)) \neq 2$. Then there exist $x', y' \in var(\theta^\infty(u))$ such that $x' \neq y'$ and $var(\rho \circ \delta(x')) \cap var(\rho \circ \delta(y')) \neq \emptyset$ so that $var(\delta(x')) \cap var(\delta(y')) \neq \emptyset$ since $\rho \in Rename$. Since $\delta \in mgu(\{\theta^\infty(s) = \theta^\infty(t)\})$, then by proposition 3.1 there are 4 cases:
    - Suppose $x' \in var(\theta^\infty(s))$ and $y' \in var(\theta^\infty(t))$. Thus there exists $x \in var(s)$ such that $x' \in var(\theta^\infty(x))$ and thus $var(\theta^\infty(u)) \cap var(\theta^\infty(x)) \neq \emptyset$. Similarly there exists $y \in var(t)$ with $var(\theta^\infty(y)) \cap var(\theta^\infty(u)) \neq \emptyset$. Hence $x \simeq u$ and $u \simeq y$ and since $x, y \notin U$, it follows that $x \in var(\kappa(s))$ and $y \in var(\kappa(t))$ so that $\{u\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.

- $*$ Suppose $y' \in var(\theta^\infty(s))$ and $x' \in var(\theta^\infty(t))$. Analogous to the previous case.
- $*$ Suppose $x', y' \in var(\theta^\infty(s))$ and $\chi(\theta^\infty(t)) = 2$. As in the previous case but one, there exist $x, y \in var(\kappa(s))$ such that $var(\theta^\infty(u)) \cap var(\theta^\infty(x)) \neq \emptyset$ and $var(\theta^\infty(y)) \cap var(\theta^\infty(u)) \neq \emptyset$. As before, $x \simeq u$ and $u \simeq y$ and $\chi(\kappa(t), \pi) = 2$, and thus $\{u\} \in mgu_V(\kappa(s), \kappa(t), \pi)$.
- $*$ Suppose $x', y' \in var(\theta^\infty(t))$ and $\chi(\theta^\infty(s)) = 2$. Analogous to the previous case. ▌

Rather than apply abstract unification directly to a equation $s = t$, one tactic for improving precision is to apply the abstract unification repeatedly to simpler equations $e_1, \ldots, e_n$ where $\{e_1, \ldots, e_n\} \in solve(\{s = t\})$. We thus lift the abstract unification to equation sets as follows:

Definition 4.5. The mapping $mgu_V : Eqn \times PS_V \to \wp(PS_V)$ is defined by: $mgu_V(E, \pi) = \{\omega \mid \langle E, \pi\rangle \leadsto^\star \langle \emptyset, \omega\rangle\}$ where $\leadsto \subseteq (Eqn \times PS_V)^2$ is the least binary relation such that: $\langle \{s = t\} \cup E, \pi\rangle \leadsto \langle E, mgu_V(s, t, \pi)\rangle$. ▌

We conjecture that $\omega = \omega'$ if $\omega, \omega' \in mgu_V(E, \pi)$. The crucial point is that any $\omega \in mgu_V(E, \pi)$ is safe and this is asserted below. Note again that U describes the state θ prior to solving E.

Corollary 4.1. $mgu(E \cup eqn(\theta)) \subseteq \gamma_V^{PS}(\omega)$ if $\omega \in mgu_V(\kappa(E), \pi)$, $\theta \in \gamma_V^G(U) \cap \gamma_V^{PS}(\pi)$ and $\kappa = \{u \mapsto c \mid u \in U\}$. ▌

PROOF. By induction on the equations of E so let $E_k = \{s_1 = t_1, \ldots, s_k = t_k\}$ for $0 \leq k \leq n$, put $\omega_0 = \pi$ and $\omega_k = mgu_V(\kappa(s_k), \kappa(t_k), \omega_{k-1})$ for $1 \leq k \leq n$. The inductive hypothesis is that $mgu(E_k \cup eqn(\theta)) \subseteq \gamma_V^{PS}(\omega_k)$. Observe that $mgu(E_0 \cup eqn(\theta)) = mgu(eqn(\theta)) \subseteq \gamma_V^{PS}(\omega_0)$. Now suppose $mgu(E_k \cup eqn(\theta)) \subseteq \gamma_V^{PS}(\omega_k)$ for $0 \leq k < n$. We have to show $mgu(E_{k+1} \cup eqn(\theta)) \subseteq \gamma_V^{PS}(\omega_{k+1})$. If $mgu(E_k \cup eqn(\theta)) = \emptyset$ then $mgu(E_{k+1} \cup eqn(\theta)) = \emptyset$, and so the result follows. Otherwise, let $\theta' \in mgu(E_k \cup eqn(\theta))$. Note that $\theta' \in \gamma_V^G(U)$. By theorem 4.1, $mgu(\{s_{k+1} = t_{k+1}\} \cup eqn(\theta')) \subseteq \gamma_V^{PS}(mgu_V(\kappa(s_{k+1}), \kappa(t_{k+1}), \omega_k)) = \gamma_V^{PS}(\omega_{k+1})$. Thus $mgu(E_{k+1} \cup eqn(\theta)) = mgu(\{s_{k+1} = t_{k+1}\} \cup eqn(\theta')) \subseteq \gamma_V^{PS}(\omega_{k+1})$. ▌

So far, abstract unification has only used groundness information for the program state immediately before E. A better tactic for both precision and efficiency is to completely trace the grounding behaviour of E with Pos_V or Def_V, say, and then feed this information into the sharing analysis. To achieve a degree of domain independence, we assume that this grounding effect of E is summarised with a G_V abstraction. This information can then be used to prune sharing abstractions and simplify E by grounding variables. Theorem 4.2 formalises this tactic. Before we reach the theorem, however, we introduce a lemma that is needed in the proof of the theorem.

Lemma 4.4. $\exists(dom(\theta) \setminus cod(\theta)).\delta \in mgu(\theta(E))$ if $\delta \circ \theta \in mgu(E)$. ▌

PROOF. Since $\delta \circ \theta \in mgu(E)$, $(\delta \circ \theta)(s) = (\delta \circ \theta)(t)$ for all $(s = t) \in E$ and thus $\delta(s') = \delta(t')$ for all $(s' = t') \in \theta(E)$ so that $\delta \in unify(\theta(E))$. But if $x \in dom(\theta) \setminus cod(\theta)$ then $x \notin var(\theta(E))$ so that $\exists(dom(\theta) \setminus cod(\theta)).\delta \in unify(\theta(E))$. Now let $\zeta \in$

$unify(\theta(E))$ and $eqn(\eta) \in solve(\theta(E))$. By lemma 2.2, $\eta^\infty \in mgu(eqn(\eta))$ and by theorem 2.1, $mgu(\theta(E)) = mgu(eqn(\eta))$ so that $\eta^\infty \circ \theta \in unify(E)$ and since $\delta \circ \theta \in mgu(E)$ it follows that $\delta \circ \theta \leq \eta^\infty \circ \theta$. Thus $\exists(dom(\theta) \setminus cod(\theta)).\delta \leq \exists(dom(\theta) \setminus cod(\theta)).\eta^\infty$. By lemma 2.3, η^∞ is idempotent so that $\exists(dom(\theta) \setminus cod(\theta)).\eta^\infty \leq \eta^\infty$ and thus it follows that $\exists(dom(\theta) \setminus cod(\theta)).\delta \leq \eta^\infty \leq \zeta$. ▌

Theorem 4.2. $mgu(E \cup eqn(\theta)) \subseteq \gamma_V^{PS}(\omega)$ if $\omega \in mgu_V(\kappa(E), \varpi)$, $mgu(E \cup eqn(\theta)) \subseteq \gamma_V^G(U)$, $\theta \in \gamma_V^{PS}(\pi)$, $\varpi = \{\{u, v\} \in \pi \mid \{u, v\} \cap U \neq \emptyset\}$ and $\kappa = \{u \mapsto c \mid u \in U\}$. ▌

Proof. Let $\vartheta \in mgu(E \cup eqn(\theta))$, $\vartheta \in \gamma_V^G(U)$, $\theta \in \gamma_V^{PS}(\pi)$ and $\varpi = \{\{u, v\} \in \pi \mid \{u, v\} \cap U \neq \emptyset\}$. Since $\theta \in \gamma_V^{PS}(\pi)$, $\theta \in RSub$ and thus θ^∞ exists. Since $\vartheta \in unify(eqn(\theta))$ and, by lemma 2.2, $\theta^\infty \in mgu(eqn(\theta))$ then $\theta^\infty \leq \vartheta$. Thus there exists $\zeta \in Sub$ such that $\zeta \circ \theta^\infty = \vartheta$. Because $\vartheta \in unify(E)$, $\zeta \in unify(\theta^\infty(E))$ so that $mgu(\theta^\infty(E)) \neq \emptyset$. Thus let $\delta \in mgu(\theta^\infty(E))$ and put $Y = \{y \in dom(\delta) \mid var(\delta(y)) = \emptyset\}$, $\phi = \overline{\exists} Y.\delta$ and $\varphi = \exists Y.\delta$. Observe that $\delta = \varphi \circ \phi$ since $cod(\phi) = \emptyset$ and $dom(\phi) \cap dom(\varphi) = \emptyset$. Thus $\varphi \circ \phi \in mgu(\theta^\infty(E))$ and so by lemma 4.4, $\exists(dom(\phi) \setminus cod(\phi)).\varphi \in mgu(\{\phi \circ \theta^\infty(E)\})$. Furthermore $\exists(dom(\phi) \setminus cod(\phi)).\varphi = \varphi$ since $cod(\phi) = \emptyset$ and $dom(\phi) \cap dom(\varphi) = \emptyset$. Since $\phi \circ \theta^\infty$ is idempotent, $\varphi \in mgu((\phi \circ \theta^\infty)^\infty(E))$ and so by lemma 4.3, $\varphi \circ \phi \circ \theta^\infty = \varphi \circ (\phi \circ \theta^\infty)^\infty \in mgu(E \cup eqn(\phi \circ \theta^\infty))$.

Let $\{u, v\} \in \alpha_V((\phi \circ \theta^\infty)^\infty)$ where $u \neq v$. Because $\phi \circ \theta^\infty$ is idempotent, $var(\phi \circ \theta^\infty(u)) \cap var(\phi \circ \theta^\infty(v)) \neq \emptyset$. As $cod(\phi) = \emptyset$ it follows that $var(\theta^\infty(u)) \cap var(\theta^\infty(v)) \neq \emptyset$ and hence $\{u, v\} \in \alpha_V(\theta^\infty)$. Similarly, if $\{u\} \in \alpha_V((\phi \circ \theta^\infty)^\infty)$ then $\{u\} \in \alpha_V(\theta^\infty)$. Thus $\phi \circ \theta^\infty \in \gamma_V^{PS}(\pi)$. By lemma 4.3, $\delta \circ \theta^\infty \in mgu(E \cup eqn(\theta))$ so there exists $\rho \in Rename$ such that $\delta \circ \theta^\infty = \rho \circ \vartheta$. Let $v \in V$. Observe that $var(\vartheta(v)) = \emptyset$ iff $var(\rho \circ \vartheta(v)) = \emptyset$ iff $var(\delta \circ \theta^\infty(v)) = \emptyset$ iff $var(\phi \circ \theta^\infty(v)) = \emptyset$. Thus $\phi \circ \theta^\infty \in \gamma_V^G(U)$ and $\phi \circ \theta^\infty \in \gamma_V^{PS}(\varpi)$. Hence, by corollary 4.1, $mgu(E \cup eqn(\phi \circ \theta^\infty)) \subseteq \gamma_V^{PS}(\omega)$ where $\omega \in mgu_V(\kappa(E), \varpi)$ and $\kappa = \{u \mapsto c \mid u \in U\}$. Thus $\varphi \circ \phi \circ \theta^\infty \in \gamma_V^{PS}(\omega)$. However, $\varphi \circ \phi \circ \theta^\infty = \delta \circ \theta^\infty = \rho \circ \vartheta$ and since $\rho \in Rename$, $\vartheta \in \gamma_V^{PS}(\omega)$ as required. ▌

Finally, we give a series of examples that illustrate, among other things, how theorem 4.2 is applied, the value of linearity information, and the importance of propagating groundness before tracing sharing. The final example shows that pair sharing can infer useful (linearity) information even in the presence of infinite trees.

Example 4.3. Let $V = \{u, v, x, y\}$ and consider the sharing at point ② of the query `?- ① x = f(y, y), f(u, v) = x ②`. The substitution at ① is $\theta_1 = \epsilon$ so that $U_1 = \emptyset$ and $\pi_1 = \emptyset$. $U_2 = \emptyset$ describes the groundness at ②. To calculate the sharing at ②, let $\pi_2 \in mgu_V(\kappa(E), \pi_1)$ where $\kappa = \epsilon$ and $E = \{x = f(y, y), f(u, v) = x\}$. Thus put $\pi_2 = mgu_V(x, f(y, y), mgu_V(f(u, v), x, \pi_1)) = \{\{u\}, \{u, v\}, \{u, y\}, \{u, x\}, \{v\}, \{v, x\}, \{v, y\}, \{x\}, \{x, y\}\}$. Note that $\pi_2 = mgu_V(f(u, v), x, mgu_V(x, f(y, y), \pi_1))$ and $\theta_2 = \{x \mapsto f(y, y), u \mapsto y, v \mapsto y\} \in mgu(E \cup eqn(\theta_1)) \subseteq \gamma_X^{PS}(\pi_2)$ as predicted by theorem 4.2. ▌

Example 4.4. The query `?- ① x = f(y, z), f(u, v) = x ②` illustrates the value of tracing linearity. With $V = \{u, v, x, y, z\}$ and $\theta_1 = \epsilon$, $U_1 = \emptyset$ and $\pi_1 = \emptyset$ and a groundness analysis will give $U_2 = \emptyset$ so that $\kappa = \epsilon$. Thus the sharing at ② is described by $\pi_2 = mgu_V(x, f(y,z), mgu_V(f(u,v), x, \pi_1)) = \{\{u,x\},\{u,y\},\{u,z\},\{v,x\},\{v,y\},\{v,z\},\{x,y\},\{x,z\}\}$. Note that $\{u,v\},\{x,y\} \notin \pi_2$ and indeed $var(\theta_2(u)) \cap var(\theta_2(v)) = \emptyset = var(\theta_2(x)) \cap var(\theta_2(y))$ for all $\theta_2 \in mgu(\{x = f(y,z), f(u,v) = x\})$. ∎

Example 4.5. The importance of tracing groundness before sharing is shown by `?-` ① `x = f(y, y, z),` ② `y = c` ③. Let $V = \{x,y,z\}$ and $\theta_1 = \epsilon$. Thus $\pi_1 = \emptyset$, $U_1 = U_2 = \emptyset$ and $U_3 = \{y\}$. If $\pi_3 \in mgu_V(\kappa(\{x = f(y,y,z), y = c\}), \pi_1)$ and $\kappa = \{y \mapsto c\}$ then $\pi_3 = \{\{x,z\}\}$. However, if the groundness and sharing analyses are interleaved, then we obtain $\pi_2{}' = mgu_V(\{x = f(y,y,z)\}, \pi_1) = \{\{x\},\{x,y\},\{x,z\}\}$. Furthermore, $\pi_3{}' = mgu_V(\kappa(\{y = c\}), \varpi)$ where $\varpi = \{\{u,v\} \in \pi_2{}' \mid \{u,v\} \cap U_3 \neq \emptyset\} = \{\{x\},\{x,z\}\}$. Thus $\pi_3{}' = \{\{x\},\{x,z\}\}$ which is strictly less precise than π_3. ∎

Example 4.6. Consider `?-` ① `x = f(x, z), f(u, v) = x` ②. If $\theta_1 = \epsilon$ then $\theta_2 = \{x \mapsto t,\ u \mapsto t,\ v \mapsto z\} \in mgu(\{x = f(x,z), f(u,v) = x\})$ where $t = f(f(f(\ldots,z),z),z)$ is infinite. Let $V = \{u,v,x,y,z\}$. $U_1 = \emptyset$, $\pi_1 = \emptyset$ and $U_2 = \emptyset$ and hence $\pi_2 = mgu_V(f(u,v), x, mgu_V(x, f(x,z), \pi_1)) = \{\{u\},\{u,v\},\{u,x\},\{u,z\},\{v\},\{v,x\},\{v,z\},\{x\},\{x,z\}\}$. Note that π_2 is safe abstraction of θ_2 since $\alpha_V(\lim_{n\to\infty} \theta_2{}^n) = \alpha_V(\theta_2) = \{\{u\},\{u,v\},\{u,x\},\{u,z\},\{v,x\},\{v,z\},\{x\},\{x,z\}\} \subseteq \pi_2$. Observe also that $\{z\} \notin \pi_2$. ∎

5. Related work

Set sharing [10] has also been proved safe for rational tree unification [8]. Among other things, the paper [8] generalises the abstraction function for *Sharing* to equation sets in rational solved form. One key idea is to replace the occurrence map of [10] with a map $occ!(\theta, x)$ that is defined as the limit of a sequence of sharing sets $occ_n(\theta, x)$. Correctness of abstract unification is established by introducing the concept of variable-idempotence. A substitution θ is said to be variable-idempotent iff $var(\theta(t)) = var(t)$ for all $x \mapsto t \in \theta$. Any substitution θ can be transformed to a variable-idempotent substitution θ' that is equivalent when θ and θ' are interpreted as equations. *Sharing* abstractions for θ and θ' coincide, so the proof of correctness focuses primarily on variable-idempotent substitutions. Our tactic for lifting pair-sharing to rational tree unification is slightly different. To abstract θ, we simply apply the map α_V to the idempotent substitution $\lim_{n\to\infty} \theta^n$.

As we have already stated, Dams [7] proposes a revision of lemma 2.2 of [3] so the pair sharing analysis of [3] for idempotent substitutions does not appear to be incorrect. We take this correctness result further and argue that pair sharing is correct for Herbrand unification without the occur-check and also for rational tree unification.

6. Conclusions

We have generalised pair sharing from Herbrand unification to constraint solving over rational trees. In doing so we have: shown how substitutions over infinite trees can be used to lift concretisation maps to substitutions in rational solved form; strengthened the connection between linearity and alternating paths; and finally proven correctness for pair sharing. Although theoretical, our work has important practical applications since Prolog-III and SICStus Prolog use rational tree unification as the default solver.

Acknowledgements

We thank Dennis Dams and Kish Shen for interesting discussions that motivated the study and Jacob Howe, Stefan Kahrs and Phil Watson for their comments and, of course, the anonymous referees. This work was funded, in part, by EPSRC project GR/MO8769.

REFERENCES

1. T. Armstrong, K. Marriott, P. Schachte, and H. Søndergaard. Two Classes of Boolean Functions for Dependency analysis. *Science of Computer Programming*, 31(1):3–45, 1998.
2. J. Chang and A. Despain. Semi-intelligent Backtracking of Prolog Based Static Data Dependency Analysis. In *International Conference and Symposium on Logic Programming*. IEEE Computer Society, 1985.
3. M. Codish, D. Dams, and E. Yardeni. Derivation and Safety of an Abstract Unification Algorithm for Groundness and Aliasing Analysis. In *Proceedings of the International Conference on Logic Programming*, pages 79–93. MIT Press, 1991. Long version available as technical report CS90-28 from the Weizmann Institute of Science, Israel.
4. M. Codish, A. Mulkers, M. Bruynooghe, M. García de la Banda, and M. Hermenegildo. Improving abstract interpretation by combining domains. *ACM Transactions on Programming Languages and Systems*, 17(1):28–44, 1995.
5. A. Colmerauer. Prolog and Infinite Trees. In K. L. Clark and S. Å. Tarnland, editors, *Logic Programming, APIC Studies in Data Processing (16)*, pages 231–251. Academic Press, 1982.
6. B. Courcelle. Fundamental Properties of Infinite Trees. *Theoretical Computer Science*, 25:95–169, 1983.
7. D. Dams. Personal communication on lemma 2.2 of [3]. July, 1993.
8. P. Hill, R. Bagnara, and E. Zaffanella. The Correctness of Set-Sharing. In *International Symposium on Static Analysis*, pages 99–114. Springer-Verlag, 1998. LNCS 1503, Long version available as University of Leeds technical report number 98.03 via http://www.scs.leeds.ac.uk/services/reports/1998.html.
9. B. Intrigila and M. Venturini Zilli. A Remark on Infinite Matching vs Infinite Unification. *Journal of Symbolic Computation*, 21(3):2289–2292, 1996.

10. D. Jacobs and A. Langen. Static Analysis of Logic Programs. *J. Logic Programming*, pages 154–314, 1992.

11. J. Jaffar. Efficient Unification over Infinite Trees. *New Generation Computing*, 2(3):207–219, 1984.

12. T. Keisu. *Tree Constraints.* PhD thesis, Department of Teleinformatics, Stockholm, Sweden, 1991. Also available as a SICS Dissertation, Series 16.

13. A. King. A synergistic analysis for sharing and groundness which traces linearity. In *Proceedings of the Fifth European Symposium on Programming*, pages 363–378. Springer-Verlag, 1994. LNCS 788.

14. C. Mellish. Abstract Interpretation of Prolog Programs. In *Third International Conference on Logic Programming*, pages 463–474. Springer, 1986. LNCS 225.

15. K. Muthukumar and M. Hermenegildo. Compile-time derivation of variable dependency through abstract interpretation. *J. of Logic Programming*, pages 315–437, 1992.

16. H. Søndergaard. An application of the abstract interpretation of logic programs: occur-check reduction. In *European Symposium on Programming*, pages 327–338. Springer-Verlag, 1986. LNCS 213.

17. A. Taylor. *High Performance Prolog Implementation.* PhD thesis, Basser Department of Computer Science, Sydney, Australia, July 1991.