TLP **9** (5): 617–689, 2009. © Cambridge University Press 2009
doi:10.1017/S1471068409990111 First published online 31 July 2009

Optimality in goal-dependent analysis of Sharing

GIANLUCA AMATO and FRANCESCA SCOZZARI
Dipartimento di Scienze, Università di Chieti–Pescara, Pescara, Italy
(*e-mail:* {amato,scozzari}@sci.unich.it)

submitted 16 March 2006; revised 19 March 2009; accepted 26 May 2009

Abstract

We face the problems of correctness, optimality, and precision for the static analysis of logic programs, using the theory of abstract interpretation. We propose a framework with a denotational, goal-dependent semantics equipped with two unification operators for forward unification (calling a procedure) and backward unification (returning from a procedure). The latter is implemented through a matching operation. Our proposal clarifies and unifies many different frameworks and ideas on static analysis of logic programming in a single, formal setting. On the abstract side, we focus on the domain Sharing by Jacobs and Langen (*The Journal of Logic Programming*, 1992, vol. 13, nos. 2–3, pp. 291–314) and provide the best correct approximation of all the primitive semantic operators, namely, projection, renaming, and forward and backward unifications. We show that the abstract unification operators are strictly more precise than those in the literature defined over the same abstract domain. In some cases, our operators are more precise than those developed for more complex domains involving linearity and freeness.

KEYWORDS: abstract interpretation, logic programming, existentially quantified substitutions, unification, matching, sharing

1 Introduction

Abstract interpretation (Cousot and Cousot 1992) is a general theory for static analysis of programs. The basic idea of abstract interpretation is to use the formal semantics of languages to analyze and verify program properties. An abstract interpretation is specified by:

- a concrete domain and a concrete semantics, inductively defined on the syntax of programs from a set of primitive concrete operators;
- an abstract domain, whose elements describe the program properties we want to observe;
- the primitive abstract operators on the abstract domain, which mimic the behavior of the corresponding concrete operators; the abstract semantics is defined from the concrete one by replacing each concrete operator with its abstract counterpart.

Abstract interpretation has been widely used to design static analysis of logic programs. In the literature, we find many proposals for the concrete domain, the concrete semantics, the abstract domain, and the abstract operators. For instance, Hans and Winkler (1992) focused on the abstract domains, Howe and King (2003) on the abstract operators, and King and Longley (1995) on improving existing analysis using a more refined concrete semantics, while Cortesi *et al.* (1996) proposed a complete framework, i.e., a combination of particular concrete semantics, and abstract domains. In many cases, the correctness of the analysis is taken for granted, since the concrete semantics is not completely specified. However, when applying several of these improvements to a single analysis framework, the improved analysis may significantly differ from the original proposal, and a new proof of correctness is needed for the overall analysis. This is especially true for logic programming, whose basic computational mechanism, unification, is intrinsically more complex than assignment or matching, used in other programming paradigms.

The aim of this paper is mainly to clarify and unify several different proposals for the goal-dependent analysis of logic programs. Inspired by the work of Cortesi *et al.* (1996), we propose a new denotational framework that combines and improves many different ideas appeared in the literature. Later, we focus on the abstract domain `Sharing` by Jacobs and Langen (1992), and we develop an analysis that is strictly more precise than the others in the literature. We formally prove correctness of the overall analysis and optimality of all the involved abstract operators.

When designing a new analysis, one needs to choose a concrete domain and semantics, an abstract domain, and abstract operators. Although these choices are related, in the following we will introduce them separately, showing available alternatives, possible improvements, and the contributions of this paper.

1.1 Concrete domain

Typically, concrete semantics of logic programs are defined over substitutions. However, substitutions are often too informative. For example, consider the one-clause program $\mathtt{p(x,x)}$ and the goal $p(x,y)$. All of $\{x/y\}$, $\{y/x\}$, $\{x/u, y/u\}$, $\{x/v, y/v\}$ are computed answers, corresponding to different choices of most general unifiers and renamed clauses. Often, especially in the case of static analysis, we are not interested in making any distinction among them. Thus, it would be more natural to adopt a domain of equivalence classes of substitutions. Many frameworks for abstract interpretation of logic programs (Jacobs and Langen 1992; Marriott *et al.* 1994; Levi and Spoto 2003) have adopted similar solutions for avoiding redundancy and causality when choosing computed answers.

Nevertheless, the standard semantics of logic programs, namely, SLD resolution, is based on substitutions and unification. Thus, any framework for logic programming should relate, in some way, to standard substitutions, in order to prove that the semantics reflects the underlying operational behavior. However, none of the above frameworks formally states the correspondence between the proposed concrete domain and standard substitutions. Although this correspondence is clear from an intuitive point of view, we think that substitutions are tricky objects, where intuition often fails.

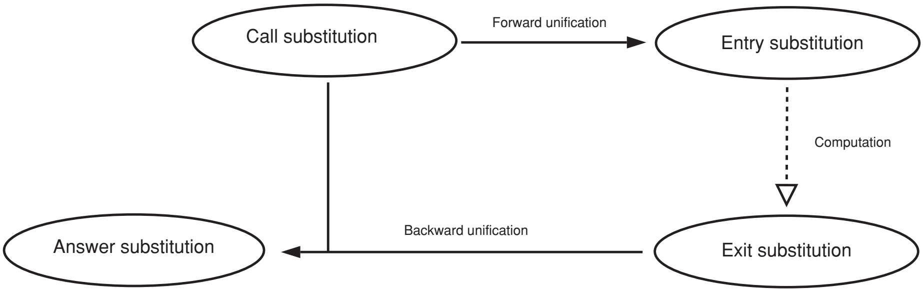
1.1.1 Our contribution

We propose a new concrete domain of classes of substitutions, called *existential substitutions*, equipped with a set of primitive operators for projection, renaming, and unification. We formally state the correspondence between substitutions and existential substitutions and, in particular, between the corresponding unification operators. Moreover, we show the relationship between our proposal and the domain *ESubst* by Jacobs and Langen (1992).

1.2 Concrete semantics

We are interested in goal-driven analysis of logic programs. Therefore we need a goal-dependent semantics that is well suited for static analysis, i.e., a collecting semantics over computed answer substitutions. Unfortunately, using a collecting goal-dependent semantics may lead to a loss of precision already at the concrete level, as shown by Marriott *et al.* (1994). Basically, in any goal-dependent semantics, the unification operator is used twice for the following reasons:

- Performing parameter passing by unifying the given goal and the *call substitution* with the head of the chosen clause. The result is a new goal and an *entry substitution*. This operation is called *forward unification.*
- Propagating back to the initial goal the *exit substitution* (that is, the result of the subcomputation), so obtaining the *answer substitution* for the initial goal. This operation is called *backward unification*[1]:



For instance, given the initial goal $p(x)$ and the call substitution $\{x/f(y)\}$, we unify with the head of the clause $\mathsf{p(z)} \leftarrow \mathsf{q(z)}$ by computing the most general unifier $\{x/f(y), z/f(y)\}$, which, projected on the variables of the clause, is simply $\{z/f(y)\}$. Projection is needed in order to avoid an unbounded growing of the set of variables in the entry substitution. This is acceptable at the concrete level but not at the

[1] We follow Cortesi *et al.* (1996) and call these operators forward and backward unifications. Bruynooghe (1991) and Hans and Winkler (1992) used *procedure entry* and *procedure exit*. Muthukumar and Hermenegildo (1991) used *call_to_entry* and *exit_to_success.*

abstract level, where it may lead to nonterminating analysis. The new goal and entry substitution become $q(z)$ and $\{z/f(y)\}$. Once we have obtained an exit substitution for the goal $q(z)$, for instance, $\{z/f(a)\}$, we have to relate this result to the original goal $p(x)$. Thus we need a so-called backward unification, which allows us to conclude that $\{x/f(a)\}$ is an answer for $p(x)$ with call substitution $\{x/f(y)\}$.

The backward unification operator introduces a loss of precision, due to the fact that we deal with a set of call substitutions, from which we possibly obtain a set of exit substitutions. Now, when we go backward to obtain the answer substitutions, we may unify a call substitution with an exit substitution that does not pertain to the same computational path (Marriott *et al.* 1994).

It is possible to reduce the impact of this problem by using two different operators for forward and backward unifications (Bruynooghe 1991; Le Charlier *et al.* 1991). In this way, backward unification can be realized using the operation of matching between substitutions.

1.2.1 Our contribution

We propose a denotational goal-dependent semantics equipped with two different forward and backward unification operators. Backward unification uses matching, exploiting the property that the exit substitution is more instantiated than the call substitution. We prove that the concrete semantics is correct and show that the new semantics is strictly more precise than semantics that do not use matching.

1.3 Abstract domain

One of the most interesting (and studied) property for logic programs is sharing. The goal of (set) sharing analysis is to detect sets of variables that share a common variable. For instance, in the substitution $\{x/f(z,a), y/g(z)\}$ the variables x and y share the common variable z. Typical applications of sharing analysis are in optimization of unification (Søndergaard 1986) and parallelization of logic programs (Hermenegildo and Rossi 1995).

The basic domain for set sharing analysis is `Sharing`, introduced in Langen (1990) and Jacobs and Langen (1992). It is widely recognized that `Sharing` is not very precise, so that it is often combined with other domains for freeness, linearity, groundness, or structural information (see Bagnara *et al.* 2005 for a comparative evaluation). Since this paper does not address the problem to find the best possible domain for set-sharing analysis, we will focus on the domain `Sharing`.

1.4 Abstract operators

Once the concrete semantics and the abstract domain have been fixed, the next step is to find suitable abstract operators that mimic the behavior of the concrete ones. The theory of abstract interpretation ensures the existence of the optimal (best correct) abstract operator for each concrete operator. Although the optimal abstract operator enjoys a constructive characterization, this is not amenable to a direct implementation. Therefore, finding an algorithm to compute optimal abstract operators is one of the main difficulties in any abstract interpretation project.

We think that there are several reasons to look for the optimal operator, instead of just using a correct one. In fact, one may argue that a well-designed correct operator may be much faster than the optimal one and does not lose much precision in real programs. Although we agree with this point, we think that knowing the optimal abstract operator, even if we do not plan to implement it, is useful to understand the potentiality and limits of the abstract domain in use and to guide the search for a more precise (or more efficient) domain. Moreover, at least in the case of sharing analysis, the more precise the abstract operators are, the smaller are the abstract objects computed during the analysis. Therefore, it may be worth spending more time in computing the abstract operators, in order to keep the abstract objects simpler (and the analysis more precise).

The primitive concrete operators used in the semantics of logic programs are renaming, projection, unification, and matching. Renaming and projections are not problematic at all: it is generally immediate to find their optimal abstract counterparts, which most of the time are also complete; i.e., they do not lose precision w.r.t. the corresponding concrete operators (Cousot and Cousot 1979; Giacobazzi *et al.* 2000).

Things are different for unification, which is a very complex operator. In fact, despite several works in this field, the best correct abstraction of unification for the domain SFL (King and Longley 1995), which combines sharing, freeness, and linearity, is still unknown. For the domain Sharing, Cortesi and Filé (1999) have shown that abstract unification defined in Jacobs and Langen (1992) is optimal. However, this result has been obtained for a concrete semantics that uses the same unification operator to compute both forward and backward unifications.

We have already said that a specialized backward unification operator may improve precision at the concrete level. In turn, the improvement in precision is reflected at the abstract level, if the abstract backward unification operator is designed to mimic matching instead of standard unification. This idea is implemented in real abstract interpreters such as GAIA (Le Charlier *et al.* 1991) and PLAI (Muthukumar and Hermenegildo 1992). However, none of the papers that are based on a specialized backward unification operator with matching (Bruynooghe 1991; Hans and Winkler 1992; Muthukumar and Hermenegildo 1992; Le Charlier and Van Hentenryck 1994; King and Longley 1995) has ever proved optimality of the proposed abstract operators. As we will show later, those abstract operators that involve set-sharing information (Hans and Winkler 1992; Muthukumar and Hermenegildo 1992; King and Longley 1995) are not optimal.

In addition, the abstract forward unification operator can be specialized in order to exploit the peculiarity of this process: the variables that occur in the clause head are always renamed apart w.r.t. the goal and the calling substitutions; hence they are free and independent. However, this idea has never been applied before in the general case but only for abstract domains that explicitly contain freeness and linearity information.

1.4.1 Our contribution

We provide abstract operators for renaming, projection, forward unification, and backward unification. We prove that all our operators are optimal and that renaming and projection are also complete. We show that abstract forward unification is able to exploit freeness and linearity information. The new backward and forward unification operators strictly improve over previous proposals for the domain Sharing.

Although freeness and linearity information are exploited by the forward abstract unification operator, this information is not encoded in the abstract domain but is just used in the internal steps of the abstract unification algorithm. This means that the algorithm cannot be immediately extended to work with more complex domains, such as SFL (King and Longley 1995), retaining optimality. Nonetheless, the abstract unification is able to exploit freeness and linearity better than other algorithms and could be used to improve the unification operation in more complex domains.

1.5 Plan of the paper

The next section recalls some basic definitions and the notations about abstract interpretation and substitutions. In Section 3 we define the domain of existentially quantified substitutions and its operators. In Sections 4 and 5 we define the concrete and abstract semantics. Finally, in Sections 6 and 7 we give the algorithms for computing the forward and backward abstract unifications and show their correctness and optimality. In Section 8 we compare our framework with related work.

The paper is a substantial expansion of Amato and Scozzari (2002), which introduces preliminary results using standard substitutions. A partial presentation of existential substitutions appeared in Amato and Scozzari (2003).

2 Notations

Given a set A, let $\wp(A)$ be the powerset of A and $\wp_f(A)$ be the set of finite subsets of A. Given two posets $(A, \leqslant_A)$ and $(B, \leqslant_B)$, we denote by $A \overset{m}{\rightarrow} B$ ($A \overset{c}{\rightarrow} B$) the space of monotonic (continuous) functions from A to B ordered pointwise. When an order for A or B is not specified, we assume the least informative order ($x \leqslant y \iff x = y$). We also use $A \uplus B$ to denote disjoint union and $|A|$ for the cardinality of the set A.

Given complete lattices A, C, a *Galois connection* (Cousot and Cousot 1979) $\langle \alpha, \gamma \rangle : C \leftrightharpoons A$ is given by a pair of maps $\alpha : C \overset{m}{\rightarrow} A$, $\gamma : A \overset{m}{\rightarrow} C$ such that $\alpha(c) \leqslant_A a \iff c \leqslant_C \gamma(a)$. A Galois connection is a *Galois insertion* when α is onto (or, equivalently, γ is injective). We say that an abstract operator $f^\alpha : A \overset{m}{\rightarrow} A$ is *correct* w.r.t. a concrete operator $f : C \overset{m}{\rightarrow} C$ when $\forall c \in C.\ (\alpha \circ f)(c) \leqslant_A (f^\alpha \circ \alpha)(c)$, which is equivalent to $\forall a \in A.\ (f \circ \gamma)(a) \leqslant_C (\gamma \circ f^\alpha)(a)$ and to $\forall a \in A.\ (\alpha \circ f \circ \gamma)(a) \leqslant_A f^\alpha(a)$. The abstract operator is *optimal* when $f^\alpha = \alpha \circ f \circ \gamma$. In this case f^α is called the *best correct approximation* of f. When $\alpha \circ f = f^\alpha \circ \alpha$, f^α is said to be *complete*, while if $f \circ \gamma = \gamma \circ f^\alpha$, then f^α is *γ-complete.*

In the following, we fix a first-order signature (Σ, Π) and an infinite set of variables $\mathcal{V}$. We assume that there are a constant symbol and a function symbol of arity at least two[2]. We use Terms and Atoms to denote the sets of terms and atomic formulas (atoms) respectively. Moreover, we call *body* or *goal* a finite sequence of atomic formulas, *clause* an object $H \leftarrow B$, where H is an atom and B is a body, and *program* a set of clauses. We use $\Box$ for the empty body, and we write H as a short form for $H \leftarrow \Box$. We denote with Bodies, $\mathsf{Clauses}$, and Progs the set of bodies, clauses, and programs respectively. Given a term t, we denote by $\mathrm{vars}(t)$ the set of variables occurring in t and by $\mathrm{uvars}(t)$ the subset of $\mathrm{vars}(t)$ whose elements appear once in t (e.g., $\mathrm{uvars}(f(x,y) = f(y,z)) = \{x,z\}$). We apply vars and uvars to any syntactic object, with the obvious meaning. We abuse the notation and write a syntactic object o instead of the set of variables $\mathrm{vars}(o)$, when it is clear from the context (e.g., if t is a term and $x \in \mathcal{V}$, then $x \in t$ should be read as $x \in \mathrm{vars}(t)$).

We denote with ϵ the empty substitution and by $\{x_1/t_1,\dots,x_n/t_n\}$ a substitution θ with $\theta(x_i) = t_i \neq x_i$. Let $\mathrm{dom}(\theta)$ be the set $\{x_1,\dots,x_n\}$ and $\mathrm{rng}(\theta)$ be the set $\mathrm{vars}(\{t_1,\dots,t_n\})$. Thus we have that $\mathrm{vars}(\theta) = \mathrm{dom}(\theta) \cup \mathrm{rng}(\theta)$. Given $U \in \wp_f(\mathcal{V})$, let $\theta_{|U}$ be the projection of θ on U, i.e., the unique substitution such that $\theta_{|U}(x) = \theta(x)$ if $x \in U$ and $\theta_{|U}(x) = x$ otherwise. We also write $\theta_{|-U}$ to denote the restriction of θ over all variables but those in U, i.e., $\theta_{|-U} = \theta_{|\mathrm{dom}(\theta)\setminus U}$. Given θ_1 and θ_2 two substitutions with disjoint domains, we denote by $\theta_1 \uplus \theta_2$ the substitution θ such that $\mathrm{dom}(\theta) = \mathrm{dom}(\theta_1) \cup \mathrm{dom}(\theta_2)$ and $\theta(x) = \theta_i(x)$ if $x \in \mathrm{dom}(\theta_i)$, for each $i \in \{1,2\}$. The application of a substitution θ to a term t is written as $t\theta$ or $\theta(t)$. Given two substitutions θ and δ, their composition, denoted by $\theta \circ \delta$, is given by $(\theta \circ \delta)(x) = \theta(\delta(x))$. A substitution ρ is called renaming if it is a bijection from $\mathcal{V}$ to $\mathcal{V}$. (This is equivalent to saying that there exists a substitution ρ^{-1} such that $\rho \circ \rho^{-1} = \rho^{-1} \circ \rho = \epsilon$). A substitution θ is idempotent when $\mathrm{dom}(\theta) \cap \mathrm{rng}(\theta) = \emptyset$. Instantiation induces a preorder on substitutions: θ is more general than δ, denoted by $\delta \leqslant \theta$, if there exists σ such that $\sigma \circ \theta = \delta$. If $\approx$ is the equivalence relation induced by $\leqslant$, we say that σ and θ are equal up to renaming when $\sigma \approx \theta$. The set of substitutions, idempotent substitutions, and renamings are denoted by *Subst*, *ISubst*, and *Ren* respectively.

Given a set of equations E, we write $\sigma = \mathrm{mgu}(E)$ to denote that σ is a most general unifier of E such that $\mathrm{vars}(\sigma) \subseteq \mathrm{vars}(E)$. Since σ is defined up to renamings, we use this notation only in cases in which the choice of the actual unifier does not matter. Any idempotent substitution σ is a most general unifier of the corresponding set of equations $\mathrm{Eq}(\sigma) = \{x = \sigma(x) \mid x \in \mathrm{dom}(\sigma)\}$. In the following, we will abuse the notation and denote by $\mathrm{mgu}(\sigma_1,\dots,\sigma_n)$, when it exists, the substitution $\mathrm{mgu}(\mathrm{Eq}(\sigma_1) \cup \ldots \cup \mathrm{Eq}(\sigma_n))$.

In the rest of the paper, we use U, V, W to denote finite sets of variables, h,k,u,v,w,x,y,z for variables, c,s,t for term symbols or terms, a,b for constants, cl for clauses, $\eta,\theta,\sigma,\delta$ for substitutions, and ρ for renamings. All these symbols can be subscripted or superscripted.

[2] Otherwise every term has at most one variable, and the structure of terms is trivial. We need this assumption in Section 8.1 and in the proofs of optimality of unification and matching.

3 Domains of existentially quantified substitutions

The first question when analyzing the behavior of logic programs is what kind of observable we are interested in. Undoubtedly, computed answers have played a prominent role, since they are the result of the process of SLD-resolution. Moreover, they have several nice properties: and-compositionality, condensing, and a bottom-up T_P-like characterization (van Emden and Kowalski 1976; Bossi *et al.* 1994). Standard semantics for logic programs, e.g., the *s*-semantics in Bossi *et al.* (1994), are defined over equivalence classes of atoms modulo renaming. For example, consider the one-clause program $\mathtt{p(x,x)}$ and the goal $p(x, y)$. All of $p(x, x)$, $p(y, y)$, $p(u, u)$, and $p(v, v)$ are computed instances, corresponding to different choices of most general unifiers and renamed clauses, but we are not interested in making any distinction among them.

However, when we consider a denotational semantics suitable for program analysis, computed answer *substitutions* are much more useful than computed instances, since most of the domains are expressed as abstraction of sets of substitutions. As before, we are not really interested in the substitutions but in their quotient-set w.r.t. a suitable equivalence relation. But in this case we cannot take renaming as the relevant equivalence relation. Let us consider the substitutions corresponding to the computed instances in the previous example: We obtain $\theta_1 = \{y/x\}$, $\theta_2 = \{x/y\}$, $\theta_3 = \{x/u, y/u\}$, and $\theta_4 = \{x/v, y/v\}$. Although θ_1 and θ_2 are equal up to renaming, the same does not hold for θ_3 and θ_4. Nonetheless, they essentially represent the same answer, since u and v are just two different variables we chose when renaming apart the clause $\mathtt{p(x,x)}$ from the goal $p(x, y)$, and therefore are not relevant. On the other side, if θ_3 and θ_4 were computed answers for the goal $q(x, y, u)$, they would correspond to computed instances $q(u, u, u)$ and $q(v, v, u)$ and therefore would be definitively different. As a consequence, the equivalence relation we need to consider must be coarser than renaming and must take into account the set of variables of interest, i.e., the set of variables which appear in the goal.

A semantics that takes into account classes of substitutions may follow three possible directions:

(1) it may compute only a subset of the computed answer substitutions, provided that the result contains at least one substitution for each equivalence class (e.g., Cortesi *et al.* 1996);
(2) it may compute all the computed answer substitutions (e.g., Le Charlier *et al.* 1991);
(3) it may be defined using a quotient domain of substitutions (e.g., Marriott *et al.* 1994).

The problem with the first two solutions is that they work by directly manipulating substitutions. It is common knowledge that this is quite tedious and error prone (Shepherdson 1994). This happens because substitutions are too much related to syntax, so that the intuition of what should happen is often betrayed by the reality, when we need to handle problems such as variable clashes and renamings. Actually, at least one framework of the first kind, namely, the widely used one in Cortesi and

Filé (1999), has a small flaw due to an unsound treatment of variable clashes (which will be discussed in details in Section 8.2).

Moreover, the first approach is generally pursued by choosing a particular most general unifier and a fixed way of renaming apart terms and substitutions. The semantics is then parametric with respect to these choices. As stated by Jacobs and Langen (1992), this makes it difficult to compare different semantics, since each of them may use different conventions for mgu and renaming. We would like to add that this also makes it difficult to state properties of a given semantics (such as compositionality properties), since they only hold up to suitable equivalence relations.

For these reasons, we think that the best solution is to move towards a domain of equivalence classes of substitutions. This does not mean we can avoid working with substitutions altogether, but all the difficulties that arise, such as renaming apart and variables clashes, may be dealt with once and for all at the domain level, reducing the opportunities for subtle mistakes to appear.

3.1 Yet another domain of existentially quantified substitutions

In the literature there are several domains of equivalence classes of substitutions: *ESubst* (Jacobs and Langen 1992), ex-equations (Marriott *et al.* 1994), and existential Herbrand constraints (Levi and Spoto 2003). For all of them, the basic idea is that some variables, in a substitution or equation, are existentially quantified, so that their names become irrelevant. However, all these proposals depart from the standard notion of substitution. As a result, the relationship between what they compute and the standard set of computed answers for a goal has never been proved. We would like to reconcile these approaches with the standard concept of substitution: in particular, we want to prove that these domains are quotient sets of substitutions, w.r.t. suitable equivalence relations.

We begin by introducing a new equivalence relation $\sim$ over substitutions, which captures the extended notion of renaming that is needed to work with computed answers. Inspired by the seminal paper of Palamidessi (1990), we introduce a new domain $Subst_{\sim}$ of classes of substitutions modulo $\sim$, which will be used in the rest of the paper[3].

Given $\theta_1, \theta_2 \in Subst$, and $U \in \wp_f(\mathscr{V})$, we define the preorder:

$$\theta_1 \preceq_U \theta_2 \iff \exists \delta \in Subst. \forall u \in U.\ \theta_1(u) = \delta(\theta_2(u)). \tag{1}$$

Intuitively, if $\theta_1 \preceq_U \theta_2$, then θ_1 is an instance of θ_2, provided we are only interested in the variables in U.

Example 3.1
It is easy to check that $\{x/a, y/u\} \preceq_{\{x,y\}} \{y/v\}$, since we may choose $\delta = \{x/a, v/u\}$ in (1). Note that the same does not happen if we consider the standard ordering on substitutions, i.e., $\{x/a, y/u\} \npreceq \{y/v\}$. Moreover, if we enlarge the set U of variables of interest, we obtain $\{x/a, y/u\} \npreceq_{\{x,y,v\}} \{y/v\}$. □

[3] In Section 8.1, we will prove that $Subst_{\sim}$ and the domain *ESubst* (Jacobs and Langen 1992) are isomorphic.

Note that in equation (1), it is important that δ is a generic substitution. If we restrict δ to be idempotent, some equivalences do not hold anymore. For example, $\{x/t(u), y/t(v)\} \preceq_{\{x,y\}} \{x/v, y/u\}$, and this is what we intuitively want, since the names of the variables u and v are not relevant. However, to prove this relation, we choose $\delta = \{u/t(v), v/t(u)\}$ in (1), and it is not an idempotent substitution.

Proposition 3.2
For any $U \in \wp_f(\mathcal{V})$, $\preceq_U$ is a preorder.

Proof
Let $U \in \wp_f(\mathcal{V})$. By definition, $\theta \preceq_U \theta \iff \exists \delta \in \mathit{Subst}. \forall v \in U.\ \theta(v) = \delta(\theta(v))$, which is a tautology by choosing as δ the empty substitution. Now assume $\theta_1 \preceq_U \theta_2$ and $\theta_2 \preceq_U \theta_3$. Therefore, there exist δ_1 and δ_2 such that $\forall v \in U$, $\theta_1(v) = \delta_1(\theta_2(v))$, and $\theta_2(v) = \delta_2(\theta_3(v))$. Therefore, $\forall v \in U$, it holds $\theta_1(v) = \delta_1(\theta_2(v)) = \delta_1(\delta_2(\theta_3(v)))$. Therefore, by choosing as δ the composition $\delta_1 \circ \delta_2$ we have that $\theta_1 \preceq_U \theta_3$. □

The next step is to define the relation

$$\theta_1 \sim_U \theta_2 \iff \exists \rho \in \mathit{Ren}. \forall v \in U.\ \theta_1(v) = \rho(\theta_2(v)), \tag{2}$$

which will be proved to be the equivalence relation induced by the preorder $\preceq_U$.

Example 3.3
It is easy to check that $\{x/v, y/u\} \sim_{\{x,y\}} \epsilon$ by choosing $\rho = \{x/v, v/x, y/u, u/y\}$. Note that $\sim_U$ is coarser than the standard equivalence relation $\approx$: there is no renaming ρ such that $\epsilon = \rho \circ \{x/v, y/u\}$. As it happens for $\preceq$, if we enlarge the set of variables of interest, not all equivalences between substitutions are preserved: for instance, $\{x/v, y/u\} \not\sim_{\{x,y,v\}} \epsilon$. □

Lemma 3.4
Let $\theta : V \to \mathcal{V}$ be an injective map of variables. Then there exists $\rho \in \mathit{Ren}$ such that $\rho(x) = \theta(x)$ for each $x \in V$ and $\mathrm{vars}(\rho) = V \cup \theta(V)$.

Proof
Since θ is injective, $|V| = |\theta(V)|$, it follows that $|V \setminus \theta(V)| = |\theta(V) \setminus V|$. Let f be any bijective map from $\theta(V) \setminus V$ to $V \setminus \theta(V)$, and let us define a substitution ρ as follows:

$$\rho(v) = \begin{cases} \theta(v) & \text{if } v \in V, \\ f(v) & \text{if } v \in \theta(V) \setminus V, \\ v & \text{otherwise.} \end{cases}$$

Note that if $x \in V$, $\rho(x) = \theta(x)$ by definition. Moreover, it is easy to check that ρ is bijective; therefore, it is a renaming. Finally, $\mathrm{vars}(\rho) = \mathrm{dom}(\rho) = V \cup (\theta(V) \setminus V) = V \cup \theta(V)$. □

Proposition 3.5
The relation $\sim_U$ is the equivalence relation induced by $\preceq_U$.

Proof
If $\theta_1 \sim_U \theta_2$ there exists $\rho \in \mathit{Ren}$ such that $\forall v \in U.\ \theta_1(v) = \rho(\theta_2(v))$. By the definition of $\preceq_U$, we have that $\theta_1 \preceq_U \theta_2$ by choosing as δ in (1) the renaming ρ. Symmetrically, by choosing as δ the renaming ρ^{-1} (the inverse of ρ), it follows that $\theta_2 \preceq_U \theta_1$.

Now assume that $\theta_1 \preceq_U \theta_2$ and $\theta_2 \preceq_U \theta_1$. Therefore there exist $\delta, \delta' \in \mathit{Subst}$ such that $\theta_2(x) = \delta'(\theta_1(x))$ and $\theta_1(x) = \delta(\theta_2(x))$; thus $\theta_2(x) = \delta'(\delta(\theta_2(x)))$ for each $x \in U$. In general, δ and δ' might not be renamings. Our goal is to build a renaming ρ, obtained by modifying δ, such that $\theta_1(x) = \rho(\theta_2(x))$, for each $x \in U$. Let $V = \mathrm{vars}(\theta_2(U))$. Since each $v \in V$ belongs to $\mathrm{vars}(\theta_2(x))$ for some $x \in U$, it follows that $(\delta' \circ \delta)(v) = v$ for all $v \in V$. Therefore, $\delta_{|V}$ may be viewed as an injective map from V to $\mathcal{V}$. By Lemma 3.4, there exists $\rho \in \mathit{Ren}$ such that $\rho_{|V} = \delta_{|V}$. Therefore, for each $x \in U$, $\rho(\theta_2(x)) = \delta(\theta_2(x)) = \theta_1(x)$; hence $\theta_1 \sim_U \theta_2$. □

It is worth noting that $\preceq_U$ is coarser than $\leqslant$ and that $\sim_U$ is coarser than renaming, as shown by the following proposition.

Proposition 3.6
Given $\theta \in \mathit{Subst}$, $\rho \in \mathit{Ren}$, and $\delta \in \mathit{Subst}$, $\rho \circ \theta \sim_U \theta$ and $\delta \circ \theta \preceq_U \theta$ for each $U \in \wp_f(\mathcal{V})$.

Proof
Simply choose ρ and δ as the relevant substitutions in (1) and (2). □

Now, let $\mathit{ISubst}_{\sim_U}$ be the quotient set of *ISubst* w.r.t. $\sim_U$. We define a new domain $\mathit{ISubst}_{\sim}$ of *existential substitutions* as the disjoint union of all the $\mathit{ISubst}_{\sim_U}$ for $U \in \wp_f(\mathcal{V})$, in formulas

$$\mathit{ISubst}_{\sim} = \biguplus_{U \in \wp_f(\mathcal{V})} \mathit{ISubst}_{\sim_U}. \tag{3}$$

In the following we write $[\theta]_U$ for the equivalence class of θ w.r.t. $\sim_U$. We call *canonical representatives* of the equivalence class $[\theta]_U \in \mathit{ISubst}_{\sim}$ the substitutions $\theta' \in \mathit{ISubst}$ such that $\theta' \sim_U \theta$ and $\mathrm{dom}(\theta') = U$. It is immediate to see that every existential substitution has a canonical representative, although it is not unique. For example, two canonical representatives of $[\{y/f(x)\}]_{x,y,z}$ are $\{y/f(h), x/h, z/k\}$ and $\{y/f(u), x/u, z/v\}$. Working with canonical representatives is of great help, especially in the proofs, since we are sure they have no variables of interest in the range.

By the definition of $\preceq_U$, when $\theta \preceq_U \theta'$, for all $W \subseteq U$ it holds that $\theta \preceq_W \theta'$. This allows us to define a partial order $\preceq$ over $\mathit{ISubst}_{\sim}$ given by

$$[\theta]_U \preceq [\theta']_V \iff U \supseteq V \wedge \theta \preceq_V \theta'. \tag{4}$$

Intuitively, $[\theta]_U \preceq [\theta']_V$ means that θ is an instance of θ' w.r.t. the variables in V, provided that they are all variables of interest of θ. It is easy to show that $\preceq$ is well defined in $\mathit{ISubst}_{\sim}$; that is, it does not depend on the choice of the representatives.

Note that although we use equivalence classes of idempotent substitutions, we could build an isomorphic domain by working with equivalence classes of the set of all the substitution. In other words, if we define $\mathit{Subst}_{\sim} = \biguplus_{U \in \wp_f(\mathcal{V})} \mathit{Subst}_{\sim_U}$, we obtain the following.

Proposition 3.7
The posets $(\mathit{Subst}_{\sim}, \preceq)$ and $(\mathit{ISubst}_{\sim}, \preceq)$ are isomorphic.

Proof
It is enough to prove that for each $U \in \wp_f(\mathcal{V})$ and $\theta \in \mathit{Subst}$, there exists $\theta' \in \mathit{ISubst}$ such that $\theta \sim_U \theta'$. Let $V = \mathrm{rng}(\theta) \cap \mathrm{dom}(\theta)$ and $W \subseteq \mathcal{V}$ such that $W \cap (U \cup \mathrm{vars}(\theta)) = \emptyset$ and $|V| = |W|$. Moreover, we take a renaming ρ such that $\mathrm{vars}(\rho) = V \cup W$ and $\rho(V) = W$. Then, we may define a substitution θ' such that

$$\theta' = (\rho \circ \theta)_{|U}.$$

Note that $\mathrm{dom}(\theta') = (\mathrm{dom}(\theta) \cup W) \cap U \subseteq \mathrm{dom}(\theta)$ and $\mathrm{rng}(\theta') \subseteq \mathrm{rng}(\theta) \setminus V \cup W$. Therefore, $\mathrm{dom}(\theta') \cap \mathrm{rng}(\theta') = \emptyset$, i.e., $\theta' \in \mathit{ISubst}$. Moreover, by definition, $\theta' \sim_U \theta$. □

The isomorphism between $\mathit{Subst}_{\sim}$ and $\mathit{ISubst}_{\sim}$ holds, since a variable in $\mathrm{rng}(\theta)$ is considered not of interest if it also occurs in $\mathrm{dom}(\theta)$. Therefore $\{x/y, y/x\} \sim_{\{x,y\}} \{x/u, y/v\}$, since y and x in the range of $\{x/y, y/x\}$ are just names for existential quantified variables. Obviously $\{x/y\} \not\sim_{\{x,y\}} \{x/u\}$, since here y only appears in the range and is therefore considered as a variable of interest.

3.2 Operations on the new domain

It is now time to define some useful operations over $\mathit{ISubst}_{\sim}$, which will be used as building blocks for the semantics to be defined further away in the paper. They will also give some more insights over the structure of $\mathit{ISubst}_{\sim}$. To ease notation, we often omit braces from the sets of variables of interest when they are given extensionally. So we write $[\theta]_{x,y}$ instead of $[\theta]_{\{x,y\}}$ and $\sim_{x,y,z}$ instead of $\sim_{\{x,y,z\}}$. When the set of variables of interest is clear from the context or is not relevant, it will be omitted. Finally, we omit the braces that enclose the bindings of a substitution when it occurs inside an equivalence class; i.e., we write $[x/y]_U$ instead of $[\{x/y\}]_U$.

3.2.1 Projection

We define an operator which projects an element of $\mathit{ISubst}_{\sim}$ on a given set of variables V, given by

$$\pi_V([\sigma]_U) = [\sigma]_{U \cap V}, \tag{5}$$

which can be easily proved to be well defined. Moreover, the following properties hold:

(1) $\pi_U \circ \pi_V = \pi_{U \cap V}$;
(2) $\pi_U([\sigma]_U) = [\sigma]_U$;
(3) π_V is monotonic w.r.t. $\preceq$.

3.2.2 Renaming

Another useful operation on classes of substitutions is renaming. We first define the application of a renaming $\rho \in \mathit{Ren}$ to a substitution $\theta \in \mathit{Subst}$ as

$$\rho(\theta) = \{\rho(x)/\rho(\theta(x)) \mid x \in \mathrm{dom}(\theta)\}. \tag{6}$$

Intuitively, we treat θ as a syntactic object and apply the renaming to both left- and right-hand sides. Note that $\rho(\theta)$ can be equivalently defined as $\rho \circ \theta \circ \rho^{-1}$.

Proposition 3.8
Given $\rho \in \mathit{Ren}$ and $\theta \in \mathit{Subst}$ it holds that $\rho(\theta) = \rho \circ \theta \circ \rho^{-1}$.

Proof
Let $\theta' = \rho(\theta)$. Since $y \neq \theta(y)$ for all $y \in \mathrm{dom}(\theta)$, $\rho(y) \neq \rho(\theta(y))$ by injectivity of ρ. It follows that $\mathrm{dom}(\theta') = \rho(\mathrm{dom}(\theta))$. We now prove that for each $x \in \mathcal{V}$, $\theta'(x) = \rho(\theta(\rho^{-1}(x)))$. We distinguish two cases:

- If $x \notin \mathrm{dom}(\theta')$, it follows that $x \notin \rho(\mathrm{dom}(\theta))$ and thus $\rho^{-1}(x) \notin \mathrm{dom}(\theta)$. As a consequence, $\rho(\theta(\rho^{-1}(x))) = \rho(\rho^{-1}(x)) = x = \theta'(x)$.
- If $x \in \mathrm{dom}(\theta')$, then $y = \rho^{-1}(x) \in \mathrm{dom}(\theta)$ and $\theta'(x) = \rho(\theta(y))$. Therefore $\rho(\theta(\rho^{-1}(x))) = \rho(\theta(y)) = \theta'(x)$. □

We may lift this definition to classes of substitutions in the standard way as follows:

$$\rho([\sigma]_U) = [\rho(\sigma)]_{\rho(U)}. \tag{7}$$

For example, let $\sigma = \{x/k, y/t(z,k)\}$, $U = \{x, y, z\}$ and consider the renaming:

$$\rho = \{x/u, u/x, y/z, z/y, k/h, h/k\}.$$

If we apply ρ to $[\sigma]_U$ we obtain $\rho([\sigma]_U) = [\{u/h, z/t(y,h)\}]_{u,y,z}$. Note that we do not need to worry about variable clashes.

Theorem 3.9
The renaming operation is well defined.

Proof
It is enough to prove monotonicity w.r.t. the preorder $\preceq_U$. Given $\theta_1, \theta_2 \in \mathit{Subst}$ such that $\theta_1 \preceq_U \theta_2$, we prove that $\rho(\theta_1) \preceq_{\rho(U)} \rho(\theta_2)$. By Proposition 3.8, we need to show that $\rho \circ \theta_1 \circ \rho^{-1} \preceq_{\rho(U)} \rho \circ \theta_2 \circ \rho^{-1}$, which is equivalent to $\theta_1 \circ \rho^{-1} \preceq_{\rho(U)} \theta_2 \circ \rho^{-1}$ thanks to Proposition 3.6. By hypothesis, there exists a substitution $\delta \in \mathit{Subst}$ such that $\theta_1(x) = \delta(\theta_2(x))$ for all $x \in U$. Therefore, for all $v \in \rho(U)$, it holds $\theta_1(\rho^{-1}(v)) = \delta(\theta_2(\rho^{-1}(v)))$, which is the thesis. □

Several properties hold for the renaming operation:

(1) $(\rho_1 \circ \rho_2)([\theta]_V) = \rho_1(\rho_2([\theta]_V))$;
(2) ρ is monotonic w.r.t. $\preceq$;
(3) $\rho(\pi_V([\theta]_U)) = \pi_{\rho(V)}(\rho([\theta]_U))$;
(4) $\rho_1([\theta]_U) = \rho_2([\theta]_U)$ if $\rho_{1|U} = \rho_{2|U}$.

We prove only the last two, since the first is trivial and the second one immediately follows from the proof of Theorem 3.9. Note that the first point implies that $\rho : \mathit{ISubst}_\sim \to \mathit{ISubst}_\sim$ is invertible.

Proposition 3.10
Renaming is a congruence w.r.t. π, i.e.,

$$\rho(\pi_V([\theta]_U)) = \pi_{\rho(V)}(\rho([\theta]_U)),$$

for $[\theta]_U \in \mathit{ISubst}_\sim$ and $\rho \in \mathit{Ren}$.

Proof
By definition $\rho(\pi_V([\theta]_U)) = \rho([\theta]_{U\cap V}) = [\rho(\theta)]_{\rho(U\cap V)}$. Since ρ is bijective, $\rho(U \cap V) = \rho(U) \cap \rho(V)$ and therefore $\rho(\pi_V([\theta]_U)) = \pi_{\rho(V)}([\rho(\theta)]_{\rho(U)}) = \pi_{\rho(V)}(\rho([\theta]_U))$, which concludes the proof. □

Proposition 3.11
Renaming only depends from the variables of interest; i.e., if $\rho_1, \rho_2 \in \mathit{Ren}$, $[\theta]_U \in \mathit{ISubst}_\sim$ and $\rho_{1|U} = \rho_{2|U}$, then $\rho_1([\theta]_U) = \rho_2([\theta]_U)$. In particular, if $\rho_1|_U = \mathit{id}$, then $\rho_1([\theta]_U) = [\theta]_U$.

Proof
Let us denote $\rho_1(U) = \rho_2(U)$ by W. We need to prove that $\rho_1(\theta) \sim_W \rho_2(\theta)$. It is obvious that $\rho_1^{-1}{}_{|W} = \rho_2^{-1}{}_{|W}$. Therefore, given $\rho = \rho_1 \circ \rho_2^{-1}$, we have that for each $x \in W$, $\rho(\rho_2(\theta)(x)) = \rho(\rho_2(\theta(\rho_2^{-1}(x)))) = \rho_1(\theta(\rho_1^{-1}(x)))$. □

3.2.3 Unification

Given $U, V \in \wp_f(\mathcal{V})$, $[\theta_1]_U, [\theta_2]_V \in \mathit{ISubst}_\sim$, we define the most general unifier between these two classes as the mgu of suitably chosen representatives, where variables not of interest are renamed apart. In formulas

$$\mathrm{mgu}([\theta_1]_U, [\theta_2]_V) = [\mathrm{mgu}(\theta_1', \theta_2')]_{U\cup V}, \tag{8}$$

where $\theta_1 \sim_U \theta_1' \in \mathit{ISubst}$, $\theta_2 \sim_V \theta_2' \in \mathit{ISubst}$, and $(U \cup \mathrm{vars}(\theta_1')) \cap (V \cup \mathrm{vars}(\theta_2')) \subseteq U \cap V$. The last condition is needed to avoid variables clashes between the chosen representatives θ_1' and θ_2'.

Example 3.12
Let $\theta_1 = \{x/a, y/t(v_1, v_1, v_2)\}$ and $\theta_2 = \{y/t(a, v_2, v_1), z/b\}$. Then

$$\mathrm{mgu}([\theta_1]_{x,y}, [\theta_2]_{y,z}) = [\{x/a, y/t(a, a, v), z/b\}]_{x,y,z}$$

by choosing $\theta_1' = \theta_1$ and $\theta_2' = \{y/t(a, w, v), z/b\}$. In this case we have

$$\{x/a, y/t(a, a, v), z/b\} \sim_{x,y,z} \mathrm{mgu}(\theta_1', \theta_2') = \{x/a, y/t(a, a, v), z/b, v_1/a, w/a, v_2/v\}. \quad \square$$

We may prove that mgu over $\mathit{ISubst}_\sim$ is well defined and that $\mathrm{mgu}([\theta_1]_U, [\theta_2]_V)$ is the greatest lower bound of $[\theta_1]_U$ and $[\theta_2]_V$ w.r.t. $\preceq$.

Theorem 3.13
The operator mgu is well defined.

Proof
We begin by proving that given $\theta_1, \theta'_1, \theta_2 \in \mathit{ISubst}$, if $\theta_1 \sim_U \theta'_1$ with $(U \cup \text{vars}(\theta_1)) \cap (V \cup \text{vars}(\theta_2)) \subseteq U \cap V$ and $(U \cup \text{vars}(\theta'_1)) \cap (V \cup \text{vars}(\theta_2)) \subseteq U \cap V$, then $\text{mgu}(\theta_1, \theta_2) \sim_{U \cup V} \text{mgu}(\theta'_1, \theta_2)$. We have the following equalities:

$$\begin{aligned} & \text{mgu}(\theta_1, \theta_2) \\ \sim_{U \cup V}\ & \text{mgu}(\theta_1, \theta_2)_{|U \cup V} \\ =\ & \text{mgu}(\theta_{1|U}, \theta_2, \theta_{1|-U})_{|U \cup V} \\ =\ & (\text{mgu}(\theta_{1|U}, \theta_2) \circ \theta_{1|-U})_{|U \cup V} \\ =\ & \text{mgu}(\theta_{1|U}, \theta_2)_{|U \cup V}. \end{aligned}$$

In the last step, we use the fact that $\text{dom}(\theta_{1|-U})$ is disjoint from $\text{vars}(\theta_{1|U})$ by idempotency of θ_1, and it is disjoint from $\text{vars}(\theta_2)$ by the assumptions $(U \cup \text{vars}(\theta_1)) \cap (V \cup \text{vars}(\theta_2)) \subseteq U \cap V$. Since $\theta_1 \sim_U \theta'_1$, there exists $\rho \in \mathit{Ren}$ such that $(\rho \circ \theta'_1)_{|U} = \theta_{1|U}$. The restriction of ρ to $\text{vars}(\theta'_{1|U})$ is an injective map of variables whose range is $\text{vars}(\theta_{1|U})$. By applying Lemma 3.4, it follows that we may choose a ρ such that $\text{vars}(\rho) \subseteq \theta_1(U) \cup \theta'_1(U) \subseteq \text{vars}(\theta_1) \cup \text{vars}(\theta'_1) \cup \text{vars}(U)$. Then $\text{vars}(\rho) \cap V \subseteq U$. We have

$$\begin{aligned} & \text{mgu}(\theta_{1|U}, \theta_2)_{|U \cup V} \\ & \quad = \text{mgu}((\rho \circ \theta'_1)_{|U}, \theta_2)_{|U \cup V} \\ & \quad = (\text{mgu}((\rho \circ \theta'_1)_{|U}, \theta_2) \circ \theta')_{|U \cup V} && \text{(for each } \theta' \text{ such that } \text{dom}(\theta') \cap (U \cup V) = \emptyset) \\ & \quad = \text{mgu}((\rho \circ \theta'_1)_{|U}, \theta_2, (\rho \circ \theta'_1)_{|-U})_{|U \cup V} && \text{(by choosing } \theta' = (\rho \circ \theta'_1)_{|-U}) \\ & \quad = \text{mgu}(\rho \circ \theta'_1, \theta_2)_{|U \cup V} \\ & \quad = (\rho' \circ \text{mgu}(\theta'_1, \theta_2))_{|U \cup V} && \text{(by Palamidessi (1990, Theorem 5.10))} \\ & \quad \sim_{U \cup V} \text{mgu}(\theta'_1, \theta_2)_{|U \cup V} && \text{(by Proposition 3.6)} \\ & \quad \sim_{U \cup V} \text{mgu}(\theta'_1, \theta_2), \end{aligned}$$

which proves the required property. Now, to prove the general theorem, assume there are $\theta_1 \sim_U \theta'_1$, $\theta_2 \sim_V \theta'_2$ with $(U \cup \text{vars}(\theta_1)) \cap (V \cup \text{vars}(\theta_2)) \subseteq U \cap V$ and $(U \cup \text{vars}(\theta'_1)) \cap (V \cup \text{vars}(\theta'_2)) \subseteq U \cap V$. Then consider a new substitution $\theta''_1 \sim_U \theta'_1$ such that $(U \cup \text{vars}(\theta''_1)) \cap (V \cup \text{vars}(\theta_2)) \subseteq U \cap V$, $(U \cup \text{vars}(\theta''_1)) \cap (V \cup \text{vars}(\theta'_2)) \subseteq U \cap V$, and we repeatedly apply the previous property, obtaining

$$\text{mgu}(\theta_1, \theta_2) \sim_{U \cup V} \text{mgu}(\theta''_1, \theta_2) \sim_{U \cup V} \text{mgu}(\theta''_1, \theta'_2) \sim_{U \cup V} \text{mgu}(\theta'_1, \theta'_2). \quad \square$$

Note that in the proof, the condition $(U \cup \text{vars}(\theta'_1)) \cap (V \cup \text{vars}(\theta'_2)) \subseteq U \cap V$ implies that $\text{vars}(\theta'_1) \cap V \subseteq U \cap V$ and $\text{vars}(\theta'_2) \cap U \subseteq U \cap V$. If we relax the condition to $\text{vars}(\theta'_1) \cap \text{vars}(\theta'_2) \subseteq U \cap V$, then this property no longer holds and mgu ceases to be well defined. This is actually the origin of the flaw in Cortesi and Filé (1999), which we will examine in Section 8.2.

Example 3.14
Consider $\theta_1 = \{x/a\}$ and $\theta_2 = \{u/b\}$. Assume we have a relaxed definition of mgu as stated above. Then, to compute $\mathrm{mgu}([\theta_1]_x, [\theta_2]_{u,v})$ we may choose $\theta'_1 = \theta_1$ and $\theta'_2 = \theta_2$ to obtain $\{x/a, u/b\}$. But with the relaxed condition we might also choose $\theta'_1 = \{x/a, v/a\}$ and $\theta'_2 = \theta_2$, since it is true that $\mathrm{vars}(\theta'_1) \cap \mathrm{vars}(\theta'_2) = \emptyset$. However $\mathrm{mgu}(\theta'_1, \theta'_2) = \{x/a, v/a, u/b\} \not\sim_{x,u,v} \{x/a, u/b\}$. □

Theorem 3.15
The operator mgu is the greatest lower bound of $(\mathit{ISubst}_\sim, \preceq)$.

Proof
If $[\theta]_{U\cup V} = \mathrm{mgu}([\theta_1]_U, [\theta_2]_V)$, we may assume, without loss of generality, that $\theta = \mathrm{mgu}(\theta_1, \theta_2)$ and θ_1, θ_2 are canonical representatives. It immediately follows that $\theta \leqslant \theta_1$ and therefore $\theta \preceq_U \theta_1$. In the same way, $\theta \preceq_V \theta_2$.

Now assume $[\eta]_{U\cup V} \preceq [\theta_1]_U$ and $[\eta]_{U\cup V} \preceq [\theta_2]_V$. We want to prove that $[\eta]_{U\cup V} \preceq [\theta]_{U\cup V}$. By the definition of $\preceq$, there is a σ_1 such that $\eta(x) = \sigma_1(\theta_1(x))$ for each $x \in U$. We may choose σ_1 such that $\mathrm{dom}(\sigma_1) \subseteq \mathrm{rng}(\theta_1)$. In the same way, there is σ_2 such that $\mathrm{dom}(\sigma_2) \subseteq \mathrm{rng}(\theta_2(x))$ and $\eta(x) = \sigma_2(\theta_2(x))$ for each $x \in V$. We may define a new substitution σ such that

$$\sigma(x) = \begin{cases} \sigma_1(\theta_1(x)) & \text{if } x \in U \cup \mathrm{dom}(\sigma_1),\\ \sigma_2(\theta_2(x)) & \text{if } x \in V \cup \mathrm{dom}(\sigma_2),\\ x & \text{otherwise.} \end{cases}$$

Note this definition is correct, since the first two cases may occur simultaneously only if $x \in U \cap V$, which implies $\sigma_1(\theta_1(x)) = \sigma_2(\theta_2(x)) = \eta(x)$. It is easy to check that $\eta \sim_{U\cup V} \sigma$ and $\sigma = \sigma \circ \theta_1 = \sigma \circ \theta_2$. Therefore

$$\eta \sim_{U\cap V} \sigma \leqslant \mathrm{mgu}(\theta_1, \theta_2) = \theta,$$

i.e., $\eta \preceq_{U\cup V} \theta$, which proves the thesis. □

We now give some properties which relate the mgu with the other operations on $\mathit{ISubst}_\sim$, namely, renaming and projection.

Proposition 3.16
With respect to unification, ρ is a congruence. In formulas, if E is a set of equations and $[\theta_1]_{U_1}, [\theta_2]_{U_2} \in \mathit{ISubst}_\sim$, then the following hold:

- $\mathrm{mgu}(\rho(E)) = \rho(\mathrm{mgu}(E))$
- $\rho(\mathrm{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2})) = \mathrm{mgu}(\rho([\theta_1]_{U_1}), \rho([\theta_2]_{U_2}))$

Proof
The first property is trivial, since the unification algorithm does not depend on the actual name of variables. Therefore, to prove the second property, we only need to check that $\mathrm{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2}) = [\mathrm{mgu}(\theta'_1, \theta'_2)]_{U_1\cup U_2}$ (according to equation (8)) implies $\mathrm{mgu}(\rho([\theta_1]_{U_1}), \rho([\theta_2]_{U_2})) = [\mathrm{mgu}(\rho(\theta'_1), \rho(\theta'_2))]_{\rho(U_1)\cup\rho(U_2)}$. First of all, since $\theta'_1 \sim_{U_1} \theta_1$, by Theorem 3.9 $\rho(\theta'_1) \sim_{\rho(U_1)} \rho(\theta_1)$. With the same reasoning, we obtain that

$\rho(\theta_2') \sim_{\rho(U_2)} \rho(\theta_2)$. Then, we prove that $(\rho(U_1) \cup \text{vars}(\rho(\theta_1'))) \cap (\rho(U_2) \cup \text{vars}(\rho(\theta_2'))) \subseteq \rho(U_1) \cap \rho(U_2)$. It is obvious that $\rho(\text{vars}(\theta)) = \text{vars}(\rho(\theta))$. Therefore, since ρ is bijective,

$$\begin{aligned}&(\rho(U_1) \cup \text{vars}(\rho(\theta_1'))) \cap (\rho(U_2) \cup \text{vars}(\rho(\theta_2')))\\ &\quad = \rho((U_1 \cup \text{vars}(\theta_1')) \cap (U_2 \cup \text{vars}(\theta_2'))) \subseteq \rho(U_1 \cap U_2) = \rho(U_1) \cap \rho(U_2). \quad \square\end{aligned}$$

Proposition 3.17
Given a set of variables V and $[\theta_1]_{U_1}, [\theta_2]_{U_2} \in \mathit{ISubst}_{\sim}$, we have that

$$\pi_V(\text{mgu}(\pi_V([\theta_1]_{U_1}), [\theta_2]_{U_2})) = \text{mgu}(\pi_V([\theta_1]_{U_1}), \pi_V([\theta_2]_{U_2})).$$

Proof
First observe that $\pi_V(\text{mgu}(\pi_V([\theta_1]_{U_1}), [\theta_2]_{U_2}))) = [\theta]_{V\cap((V\cap U_1)\cup U_2)} = [\theta]_{V\cap(U_1\cup U_2)}$, where $\theta \in \text{mgu}(\theta_1', \theta_2')$, θ_1' and θ_2' are canonical representatives of $[\theta_1]_{V\cap U_1}$ and $[\theta_2]_{U_2}$ and $\text{vars}(\theta_1') \cap \text{vars}(\theta_2') \subseteq V \cap U_1 \cap U_2$. Note that $\theta_2' \sim_{U_2} \theta_2$ and therefore $\theta_2' \sim_{V\cap U_2} \theta_2$. Moreover $(\text{vars}(\theta_1') \cup (V \cap U_1)) \cap (\text{vars}(\theta_2') \cup (V \cap U_2)) \subseteq V \cap U_1 \cap U_2$, and therefore θ_1' and θ_2' are valid representatives to compute $\text{mgu}(\pi_V([\theta_1]_{U_1}), \pi_V([\theta_2]_{U_2}))$ according to (8). Therefore $[\theta]_{V\cap(U_1\cup U_2)} = \text{mgu}(\pi_V([\theta_1]_{U_1}), \pi_V([\theta_2]_{U_2}))$, and this proves the thesis. $\square$

Thanks to the above properties, the algebraic structure of the domain $\mathit{ISubst}_{\sim}$ is very similar to (locally finite) cylindric algebras (Henkin *et al.* 1971). In particular, if the unit element is defined as $[\epsilon]_{\emptyset}$, the diagonal elements are given by the substitutions $[x/y]_{\{x,y\}}$, and cylindrification is defined as $c_x([\theta]_V) = \pi_{V\setminus\{x\}}([\theta]_V)$, then these operators satisfy the axioms defining a cylindric algebra. The fundamental difference is that the underlying set $\mathit{ISubst}_{\sim}$ is not a Boolean algebra.

It would be possible, as in Palamidessi (1990), to define a “least common anti-instance” operator that corresponds to the least upper bound in $\mathit{ISubst}_{\sim}$. However, since it is not used in the semantic framework we are going to describe, we omit to define this operator.

4 Concrete semantics

Since we are interested in goal-dependent analysis of logic programs, we need a goal-dependent semantics that is well suited for static analysis, i.e., a collecting semantics over computed answers. Unfortunately, using a collecting goal-dependent semantics may lead to a loss of precision already at the concrete level, as shown by Marriott *et al.* (1994). It is possible to reduce the impact of this problem by using two different operators for forward and backward unifications. In particular, it turns out that backward unification may be realized using the operation of matching between substitutions (Bruynooghe 1991; Le Charlier *et al.* 1991). We follow the same approach and define a new denotational framework based on existential substitutions and inspired by Cortesi *et al.* (1994).

4.1 Concrete domain

We start to define the concrete domain for the semantics. A concrete object is essentially a set of existential substitutions with a fixed set of variables of interest.

In formulas

$$\mathtt{Psub} = \{[\Theta, U] \mid \Theta \subseteq \mathit{ISubst}_{\sim_U},\ U \in \wp_f(\mathcal{V})\} \cup \{\bot_{\mathrm{Ps}}, \top_{\mathrm{Ps}}\},$$

where $\top_{\mathrm{Ps}}$ and $\bot_{\mathrm{Ps}}$ are the top and bottom elements respectively and

$$[\Theta_1, U_1] \sqsubseteq_{\mathrm{Ps}} [\Theta_2, U_2] \iff U_1 = U_2 \text{ and } \Theta_1 \subseteq \Theta_2.$$

The notation we adopt may appear clumsy, since the set of variables of interest U in $[\Theta, U]$ may be derived from Θ. However, when we move to the abstract domain, we need to explicitly keep track of this set U. By using $[\Theta, U]$ in Psub, we want to keep a consistent notation for both concrete and abstract domains.

It turns out that $(\mathtt{Psub}, \sqsubseteq_{\mathrm{Ps}})$ is a complete lattice, and we denote by $\sqcup_{\mathrm{Ps}}$ its least upper bound, which is given by

$$\begin{aligned}
\top_{\mathrm{Ps}} \sqcup_{\mathrm{Ps}} \chi &= \chi \sqcup_{\mathrm{Ps}} \top_{\mathrm{Ps}} = \top_{\mathrm{Ps}},\\
\bot_{\mathrm{Ps}} \sqcup_{\mathrm{Ps}} \chi &= \chi \sqcup_{\mathrm{Ps}} \bot_{\mathrm{Ps}} = \chi, \\
[\Theta_1, U_1] \sqcup_{\mathrm{Ps}} [\Theta_2, U_2] &= \begin{cases} [\Theta_1 \cup \Theta_2, U_1] & \text{if } U_1 = U_2,\\ \top_{\mathrm{Ps}} & \text{otherwise.}\end{cases}
\end{aligned} \tag{9}$$

We now define the main operations over Psub, that is, projection on a set of variables, unification of an object with a single substitution, and the operation for matching two objects of Psub. All the operations are strict: when one of the argument is $\bot_{\mathrm{Ps}}$ the result is $\bot_{\mathrm{Ps}}$. If no argument is $\bot_{\mathrm{Ps}}$ and at least one of the argument is $\top_{\mathrm{Ps}}$ the result is $\top_{\mathrm{Ps}}$. Therefore, in the following, we will omit the cases for the objects $\bot_{\mathrm{Ps}}$ and $\top_{\mathrm{Ps}}$.

Given $[\Theta, U] \in \mathtt{Psub}$ and $V \subseteq \mathcal{V}$, we define the projection of $[\Theta, U]$ on the set of variables V as

$$\pi_{\mathrm{Ps}}([\Theta, U], V) = [\{\pi_V([\sigma]_U) \mid [\sigma]_U \in \Theta\}, U \cap V]. \tag{10}$$

The concrete unification $\mathsf{unif}_{\mathrm{Ps}} : \mathtt{Psub} \times \mathit{ISubst} \to \mathtt{Psub}$ is given by

$$\mathsf{unif}_{\mathrm{Ps}}([\Theta, U], \delta) = [\{\mathrm{mgu}([\sigma]_U, [\delta]_{\mathrm{vars}(\delta)}) \mid [\sigma]_U \in \Theta\}, U \cup \mathrm{vars}(\delta)]. \tag{11}$$

The operations π_{Ps} and $\mathsf{unif}_{\mathrm{Ps}}$ are just the pointwise extensions of π and mgu. Note that in $\mathsf{unif}_{\mathrm{Ps}}$, the argument δ may have variables that do not appear in U. This is not always the case in literature. For example, in Cortesi and Filé (1999) and Bagnara *et al.* (2005) we find a variant of $\mathsf{unif}_{\mathrm{Ps}}$ that only consider the case when $\mathrm{vars}(\delta) \subseteq U$. When this does not happen, the same effect is obtained by first enlarging the set of variables of interest U and then applying unification. Although nothing changes at the concrete level, this gives a loss of precision when we move to the abstract side, since the composition of two optimal abstract operators is generally less precise than the optimal abstract counterpart of the whole $\mathsf{unif}_{\mathrm{Ps}}$ (see Section 6).

Finally, we define the matching operation. The idea is to design an operator which performs unification between two substitutions $[\theta_1]_{U_1}$ and $[\theta_2]_{U_2}$ only if the process of unification does not instantiate the first substitution. In other words, we require that if we compute $\mathrm{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2})$ and only observe variables in U_1, that is,

$\pi_{U_1}(\text{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2}))$, then we obtain exactly $[\theta_1]_{U_1}$. The next proposition shows this is equivalent to requiring that $\theta_1 \preceq_{U_1 \cap U_2} \theta_2$.

Proposition 4.1
Given two existential substitutions $[\theta_1]_{U_1}$ and $[\theta_2]_{U_2}$, we have that $\theta_1 \preceq_{U_1 \cap U_2} \theta_2$ iff $[\theta_1]_{U_1} = \pi_{U_1}(\text{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2}))$.

Proof
By Proposition 3.17 we obtain $\pi_{U_1}(\text{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2})) = \text{mgu}(\pi_{U_1}([\theta_1]_{U_1}), \pi_{U_1}([\theta_2]_{U_2})) = \text{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_1 \cap U_2})$. Since mgu is the greatest lower bound of $\mathit{ISubst}_{\sim}$, we have that $[\theta_1]_{U_1} = \text{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_1 \cap U_2})$ iff $[\theta_1]_{U_1} \preceq [\theta_2]_{U_1 \cap U_2}$, which, by definition, is equivalent to $\theta_1 \preceq_{U_1 \cap U_2} \theta_2$. □

We can now define the matching operator $\mathsf{match}_{\text{Ps}} : \mathtt{Psub} \times \mathtt{Psub} \rightarrow \mathtt{Psub}$ as follows:

$$\mathsf{match}_{\text{Ps}}([\Theta_1, U_1], [\Theta_2, U_2]) = [\{\text{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2}) \mid \theta_1 \preceq_{U_1 \cap U_2} \theta_2, [\theta_1]_{U_1} \in \Theta_1, [\theta_2]_{U_2} \in \Theta_2\}, U_1 \cup U_2]. \quad (12)$$

The above operator allows us to unify all the pairs of substitutions $[\theta_1]_{U_1} \in \Theta_1$ and $[\theta_2]_{U_2} \in \Theta_2$, under the condition that the common variables in U_1 and U_2 may not be further instantiated w.r.t. their values in θ_1.

Example 4.2
Let $\Theta_1 = \{[x/y]_{x,y}\}$ and $\Theta_2 = \{[u/x]_{u,x}, [x/t(u)]_{u,x}\}$. Then

$$\mathsf{match}_{\text{Ps}}([\Theta_1, \{x, y\}], [\Theta_2, \{u, x\}]) = [\{[x/y, u/y]_{x,y,u}\}, \{x, y, u\}].$$

Note that $[y/t(u), x/t(u)]_{u,x,y}$, obtained by unifying $[x/y]_{x,y}$ with $[x/t(u)]_{u,x}$, is not in the result of matching. This is because $[x/t(u)]_{u,x}$ is strictly more instantiated than $[x/y]_{x,y}$ w.r.t. the variable x, and therefore $\{x/y\} \not\preceq_x \{x/t(u)\}$. □

Proposition 4.3
The operations π_{Ps}, $\mathsf{unif}_{\text{Ps}}$, and $\mathsf{match}_{\text{Ps}}$ are continuous over $\mathtt{Psub}$.

Proof
Trivial from their definitions. If we do not consider the element $\top_{\text{Ps}}$, they are actually additive. □

4.2 Semantics

Using the operators defined so far, we introduce a denotational semantics for logic programs. It computes, for a given goal G, the set of computed answers for a program w.r.t. G modulo the equivalence relation $\sim_{\text{vars}(G)}$. It is a goal-dependent *collecting semantics* (Cousot and Cousot 1994), in that it works by computing the set of possibly entry and exit substitutions at each point in the program.

We call *denotation* an element in the set of continuous maps:

$$\mathscr{D}en = \mathsf{Atoms} \rightarrow \mathtt{Psub} \xrightarrow{c} \mathtt{Psub}. \quad (13)$$

We have the following semantic functions:

$$\begin{aligned}\mathscr{P} &: \textsf{Progs} \to \mathscr{D}en,\\ \mathscr{C} &: \textsf{Clauses} \to \mathscr{D}en \xrightarrow{c} \mathscr{D}en,\\ \mathscr{B} &: \textsf{Bodies} \to \mathscr{D}en \xrightarrow{c} \texttt{Psub} \xrightarrow{c} \texttt{Psub}.\end{aligned}$$

The corresponding definitions[4], given $d \in Den$ and $x \in \texttt{Psub}$, are

$$\begin{gathered}\mathscr{P}[\![P]\!] = lfp\ \lambda d.\left(\underset{cl\in P}{\bigsqcup\nolimits_{\mathrm{Ps}}} \mathscr{C}[\![cl]\!]d\right),\\ \mathscr{C}[\![H \leftarrow B]\!]\ d\ A\ \chi = \mathbf{U}^{b}_{\mathrm{Ps}}\big((\mathscr{B}[\![B]\!]d\mathbf{U}^{f}_{\mathrm{Ps}}(\chi, A, H)), \chi, H, A\big),\\ \mathscr{B}[\![\Box]\!]\ d\ \chi = \chi,\\ \mathscr{B}[\![A, B]\!]\ d\ \chi = \mathscr{B}[\![B]\!]d(dA\chi)\end{gathered}$$

defined by means of the following operators:

$$\begin{aligned}\mathbf{U}^{f}_{\mathrm{Ps}} &: \texttt{Psub} \times \textsf{Atoms} \times \textsf{Atoms} \to \texttt{Psub},\\ \mathbf{U}^{b}_{\mathrm{Ps}} &: \texttt{Psub} \times \texttt{Psub} \times \textsf{Atoms} \times \textsf{Atoms} \to \texttt{Psub}.\end{aligned}$$

Here $\mathbf{U}^{f}_{\mathrm{Ps}}$ and $\mathbf{U}^{b}_{\mathrm{Ps}}$ are respectively the *forward* and *backward* unifications (Muthukumar and Hermenegildo 1992). They are used according to the following pattern:

- the *forward unification*, in order to compute the set of *entry substitutions* $\mathbf{U}^{f}_{\mathrm{Ps}}(\chi, A, H)$ from the set of *call substitutions* χ;
- the *backward unification*, in order to compute the set of *answer substitutions* $\mathbf{U}^{b}_{\mathrm{Ps}}((\mathscr{B}[\![B]\!]d\mathbf{U}^{f}_{\mathrm{Ps}}(\chi, A, H)), \chi, H, A)$ starting from the set of *exit substitutions* $\mathscr{B}[\![B]\!]d\mathbf{U}^{f}_{\mathrm{Ps}}(\chi, A, H)$.

The formal definitions of $\mathbf{U}^{f}_{\mathrm{Ps}}$ and $\mathbf{U}^{b}_{\mathrm{Ps}}$ are the following:

$$\mathbf{U}^{f}_{\mathrm{Ps}}([\Theta, U], A_1, A_2) = \pi_{\mathrm{Ps}}(\textsf{unif}_{\mathrm{Ps}}(\rho([\Theta, U]), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2)), \tag{14}$$

where ρ is a renaming such that $\rho(U \cup \mathrm{vars}(A_1)) \cap \mathrm{vars}(A_2) = \emptyset$ and $\rho([\Theta, U]) = [\{\rho([\sigma]_U) \mid [\sigma]_U \in \Theta\}, \rho(U)]$ is the obvious lifting of renamings from $ISubst_{\sim}$ to $\texttt{Psub}$, and

$$\begin{aligned}&\mathbf{U}^{b}_{\mathrm{Ps}}([\Theta_1, U_1], [\Theta_2, U_2], A_1, A_2)\\ &\quad = \pi_{\mathrm{Ps}}(\textsf{match}_{\mathrm{Ps}}(\rho([\Theta_1, U_1]), \textsf{unif}_{\mathrm{Ps}}([\Theta_2, U_2], \mathrm{mgu}(\rho(A_1) = A_2))), U_2 \cup \mathrm{vars}(A_2)),\end{aligned} \tag{15}$$

where ρ is a renaming such that $\rho(U_1 \cup \mathrm{vars}(A_1)) \cap (U_2 \cup \mathrm{vars}(A_2)) = \emptyset$. If $\rho(A_1)$ and A_2 do not unify, the results for both the operations is assumed to be $\bot_{\mathrm{Ps}}$.

Example 4.4

Consider the goal $p(x, y, z)$ with $y = f(x, z)$ and the trivial program P with just one clause

```
p(u,v,w).
```

[4] Here we use the lambda notation, writing $lfp\ \lambda x.E(x)$ to denote the least fixed point of the function f given by $f(x) = E(x)$.

We first compute the concrete semantics $\mathscr{P}[\![P]\!] = lfp\ \lambda d.\mathscr{C}[\![\mathtt{p(u,v,w)} \leftarrow \square]\!]d$. According to the semantic definition, we have that

$$\mathscr{C}[\![\mathtt{p(u,v,w)} \leftarrow \square]\!]d = \lambda A.\lambda\chi.\mathbf{U}^b_{\mathrm{Ps}}\big(\big(\mathscr{B}[\![\square]\!]d\mathbf{U}^f_{\mathrm{Ps}}(\chi, A, p(u,v,w))\big)\big), \chi, p(u,v,w), A).$$

Since $\mathscr{B}[\![\square]\!]d = \lambda\chi.\chi$, this is equivalent to

$$\lambda A.\lambda\chi.\mathbf{U}^b_{\mathrm{Ps}}\big(\mathbf{U}^f_{\mathrm{Ps}}(\chi, A, p(u,v,w)), \chi, p(u,v,w), A\big),$$

from which we immediately obtain the semantics of the program P:

$$\mathscr{P}[\![P]\!] = \lambda A.\lambda\chi.\mathbf{U}^b_{\mathrm{Ps}}\big(\mathbf{U}^f_{\mathrm{Ps}}(\chi, A, p(u,v,w)), \chi, p(u,v,w), A\big).$$

We now compute the semantics of the goal $p(x,y,z)$ with $y = f(x,z)$. In order to improve readability, we will omit subscripts on classes of substitutions.

$$\begin{aligned}&\mathscr{P}[\![P]\!]p(x,y,z)[\{[y/f(x,z)]\}, \{x,y,z\}]\\ &\quad = \mathbf{U}^b_{\mathrm{Ps}}(\mathbf{U}^f_{\mathrm{Ps}}([\{[y/f(x,z)]\}, \{x,y,z\}], p(x,y,z), p(u,v,w)),\\ &\qquad [\{[y/f(x,z)]\}, \{x,y,z\}], p(u,v,w), p(x,y,z)).\end{aligned}$$

We first compute the forward unification

$$\mathbf{U}^f_{\mathrm{Ps}}([\{[y/f(x,z)]\}, \{x,y,z\}], p(x,y,z), p(u,v,w)) = [\{[u/x', v/f(x',z'), w/z']\}, \{u,v,w\}],$$

where we have renamed x and z as x' and z' to avoid ambiguities, although it is not needed. Now we can compute the semantics of the goal:

$$\begin{aligned}&\mathscr{P}[\![P]\!]p(x,y,z)[\{[y/f(x,z)]\}, \{x,y,z\}]\\ &\quad = \mathbf{U}^b_{\mathrm{Ps}}([\{[u/x', v/f(x',z'), w/z']\}, \{u,v,w\}], [\{[y/f(x,z)]\}, \{x,y,z\}],\\ &\qquad p(u,v,w), p(x,y,z))\\ &\quad = \pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}([\{[u/x', v/f(x',z'), w/z']\}, \{u,v,w\}],\\ &\qquad [\{[u/x, v/f(x,z), w/z, y/f(x,z)]\}, \{u,v,w,x,y,z\}]), \{x,y,z\})\\ &\quad = \pi_{\mathrm{Ps}}([\{[u/x, v/f(x,z), w/z, y/f(x,z)]\}, \{u,v,w,x,y,z\}], \{x,y,z\})\\ &\quad = [\{[y/f(x,z)]\}, \{x,y,z\}].\end{aligned}$$

Thus, we have only one computed answer substitution for the goal $p(x,y,z)$ with $y = f(x,z)$, which is $\{y/f(x,z)\}$. □

Theorem 4.5

The operators $\mathbf{U}^f_{\mathrm{Ps}}$ and $\mathbf{U}^b_{\mathrm{Ps}}$ are well defined, in that they are independent from the choice of ρ. Moreover, they are continuous.

Proof

Continuity is trivial from their definition; therefore we only need to prove the independence from the choice of the renaming ρ. We only consider the case when none of the arguments is $\bot_{\mathrm{Ps}}$ or $\top_{\mathrm{Ps}}$, since otherwise the result is always $\bot_{\mathrm{Ps}}$ or $\top_{\mathrm{Ps}}$. Moreover, note that given atoms A_1 and A_2, if ρ_1 and ρ_2 are renamings such that $\rho_i(\mathrm{vars}(A_1)) \cap \mathrm{vars}(A_2) = \emptyset$ for $i \in \{1,2\}$, then $\rho_1(A_1)$ and A_2 unify iff $\rho_2(A_1)$ and A_2 unify. Therefore, we can restrict ourselves to the case in which the two atoms given as arguments, appropriately renamed, do unify. Otherwise, the result is always $\bot_{\mathrm{Ps}}$.

Observe that, by Proposition 3.16, given $\rho \in \mathit{Ren}$, $[\theta_1]_{U_1}, [\theta_2]_{U_2} \in \mathit{ISubst}_{\sim}$, we have that $\rho(\mathrm{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2})) = \mathrm{mgu}(\rho([\theta_1]_{U_1}), \rho([\theta_2]_{U_2}))$. By the definition of $\mathsf{unif}_{\mathrm{Ps}}$, it follows that $\rho(\mathsf{unif}_{\mathrm{Ps}}([\Theta, U], \delta)) = \mathsf{unif}_{\mathrm{Ps}}(\rho([\Theta, U]), \rho(\delta))$, since $\mathrm{vars}(\rho(\delta)) = \rho(\mathrm{vars}(\delta))$.

Let ρ_1, ρ_2 be renamings. We first show that

$$\begin{aligned}&\pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho_1([\Theta, U]), \mathrm{mgu}(\rho_1(A_1) = A_2)), \mathrm{vars}(A_2))\\ &\quad = \pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho_2([\Theta, U]), \mathrm{mgu}(\rho_2(A_1) = A_2)), \mathrm{vars}(A_2))\end{aligned}$$

provided that $\rho_i(U \cup \mathrm{vars}(A_1)) \cap \mathrm{vars}(A_2) = \emptyset$, for $i \in \{1, 2\}$. Let $W = \rho_1(U \cup \mathrm{vars}(A_1))$ and $\delta = (\rho_2 \circ \rho_1^{-1})_{|W}$. Then δ may be viewed as an injective map from V to $\mathcal{V}$, since it is the composition of injective functions. By Lemma 3.4 there exists a renaming ρ such that $\rho_{|W} = \delta$ and $\mathrm{vars}(\rho) = \mathrm{vars}(\delta) \subseteq W \cup \mathrm{rng}(\delta) \subseteq W \cup \rho_2(U \cup \mathrm{vars}(A_1))$. Observe that $\mathrm{vars}(\rho) \cap \mathrm{vars}(A_2) = \emptyset$, since, by hypothesis, for each $i \in \{1, 2\}$ it is the case that $\rho_i(U \cup \mathrm{vars}(A_1)) \cap \mathrm{vars}(A_2) = \emptyset$. Thus the following equivalences hold:

$$\begin{aligned}&\pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho_1([\Theta, U]), \mathrm{mgu}(\rho_1(A_1) = A_2)), \mathrm{vars}(A_2))\\ &\quad = \rho(\pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho_1([\Theta, U]), \mathrm{mgu}(\rho_1(A_1) = A_2)), \mathrm{vars}(A_2)))\\ &\qquad \text{(since } \rho_{|\mathrm{vars}(A_2)} = \mathit{id} \text{ and by Proposition 3.11)}\\ &\quad = \pi_{\mathrm{Ps}}(\rho(\mathsf{unif}_{\mathrm{Ps}}(\rho_1([\Theta, U]), \mathrm{mgu}(\rho_1(A_1) = A_2))), \mathrm{vars}(A_2))\\ &\qquad \text{(since } \rho \text{ is a congruence for } \pi_{\mathrm{Ps}} \text{ by Proposition 3.10)}\\ &\quad = \pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho(\rho_1([\Theta, U])), \mathrm{mgu}(\rho(\rho_1(A_1)) = \rho(A_2))), \mathrm{vars}(A_2))\\ &\qquad \text{(since } \rho \text{ is a congruence for } \mathsf{unif}_{\mathrm{Ps}} \text{ by Proposition 3.16)}\\ &\quad = \pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho_2([\Theta, U]), \mathrm{mgu}(\rho_2(A_1)) = A_2), \mathrm{vars}(A_2))\\ &\qquad \text{(since } (\rho \circ \rho_1)_{|U \cup \mathrm{vars}(A_1)} = \rho_{2|U \cup \mathrm{vars}(A_1)} \text{ and by Proposition 3.11).}\end{aligned}$$

We now show that $\mathbf{U}^b_{\mathrm{Ps}}$ is independent from the choice of the renaming. First of all, note that by Proposition 3.16 and Theorem 3.9 the following is obtained:

$$\rho(\mathsf{match}_{\mathrm{Ps}}([\Theta_1, U_1], [\Theta_2, U_2])) = \mathsf{match}_{\mathrm{Ps}}(\rho([\Theta_1, U_1]), \rho([\Theta_2, U_2])).$$

Assume given $\rho_1, \rho_2 \in \mathit{Ren}$ such that $\rho_i(U_1 \cup \mathrm{vars}(A_1)) \cap (U_2 \cup \mathrm{vars}(A_2)) = \emptyset$, for $i \in \{1, 2\}$. Let $W = \rho_1(U_1 \cup \mathrm{vars}(A_1))$ and $\delta = (\rho_2 \circ \rho_1^{-1})_{|W}$. As shown above, there exists $\rho \in \mathit{Ren}$ such that $\rho_{|W} = \delta$ and $\mathrm{vars}(\rho) = \mathrm{vars}(\delta) \subseteq W \cup \rho_2(U_1 \cup \mathrm{vars}(A_1))$. Observe that $\delta_{|U_2 \cup \mathrm{vars}(A_2)} = \mathit{id}$. Thus the following equivalences hold, where $Z = U_2 \cup \mathrm{vars}(A_2)$:

$$\begin{aligned}&\pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}(\rho_1([\Theta_1, U_1]), \mathsf{unif}_{\mathrm{Ps}}([\Theta_2, U_2], \mathrm{mgu}(\rho_1(A_1) = A_2))), Z)\\ &\quad = \rho(\pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}(\rho_1([\Theta_1, U_1]), \mathsf{unif}_{\mathrm{Ps}}([\Theta_2, U_2], \mathrm{mgu}(\rho_1(A_1) = A_2))), Z))\\ &\quad = \pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}(\rho(\rho_1([\Theta_1, U_1])), \mathsf{unif}_{\mathrm{Ps}}(\rho([\Theta_2, U_2]), \mathrm{mgu}(\rho(\rho_1(A_1)) = \rho(A_2)))), Z)\\ &\quad = \pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}(\rho_2([\Theta_1, U_1]), \mathsf{unif}_{\mathrm{Ps}}([\Theta_2, U_2], \mathrm{mgu}(\rho_2(A_1) = A_2))), Z).\end{aligned}$$

This concludes the proof of the theorem. □

Theorem 4.6

All the semantic functions are well defined and continuous.

Proof
The proof is trivial, since the semantic functions are obtained by composition, application, projection, and tupling of continuous functions. Therefore, they are continuous and compute continuous denotations. Moreover, they do not depend on the choice of ρ in $\mathbf{U}^f_{\mathrm{Ps}}$ and $\mathbf{U}^b_{\mathrm{Ps}}$, as proved in Theorem 4.5. □

Note that several frameworks have been developed for logic programs, and not all of them use the same operators for forward and backward unifications. We will discuss the benefits of our choices later, when we introduce the abstract operators, since the relative merits of the different proposals mainly arise when speaking about abstractions.

4.3 Correctness and completeness

The semantics we have defined in this section is significant only up to the point that studying its properties, it is possible to derive some conclusions about the properties of the real operational behavior of logic programs. We said before that we considered as the relevant operational observable of our analysis the set of classes of computed answers for a goal. Therefore, the best we can expect from our collecting semantics is that it enables us to recover the set of computed answer for each goal. Our first theorem is a partial positive answer to this question.

Theorem 4.7
(Semantic correctness)
Given a program P and an goal G, if θ is a computed answer for the goal G, then $\mathscr{B}[\![G]\!](\mathscr{P}[\![P]\!])G[\{\epsilon\}, \mathrm{vars}(G)] \sqsupseteq_{\mathrm{Ps}} [\{[\theta]\}, \mathrm{vars}(G)]$.

Proof
The proof, quite long and tedious, may be found in Appendix A. □

Therefore, we know that all the computed answers may be obtained by our semantics. However, the opposite is not true: the semantics given in this paper, although more precise than a semantics that only uses unification, is not complete w.r.t. computed answers. Actually, Marriott *et al.* (1994) give an example in which a collecting goal-dependent semantics computes a substitution that is not a computed answer. When matching is used to compute the backward unification, as is the case in our framework, that example does not work anymore (see Example 7.3).

However, also with the use of matching, the collecting semantics computes substitutions that are not computed answers. Consider the program P given by the following clauses:

```
p(x,y) :- q(x).
q(x).
```

We want to compute $\mathscr{P}[\![P]\!]p(x,y)[\Theta, \{x,y\}]$, where $\Theta = \{[x/y], [x/a]\}$. It is easy to check that

$$\mathscr{P}[\![P]\!]q(x)[\Delta, \{x\}] = [\Delta, \{x\}]$$

for each $[\Delta, \{x\}] \in \mathtt{Psub}$. Therefore, this implies that

$$\mathscr{P}[\![P]\!]p(x, y)[\Theta, \{x, y\}] = [\{[x/y], [x/a], [x/a, y/a]\}, \{x, y\}].$$

The substitution $[x/a, y/a]$ arises from calling $q(x)$ with the substitution $[x/a]$ and matching the result with $[x/y]$, which is not forbidden by matching. However, there is no substitution in the class of $[\{x/a, y/a\}]_{x,y}$, which is a computed answer for the goal $p(x, y)$ in the program P with entry substitution in Θ.

This loss of precision is not relevant for downward-closed abstract domains, where goal-dependent collecting semantics are more precise than goal-independent ones. This is not the case for upward-closed abstract domains, where goal-independent semantics are more precise than goal-dependent ones. García de la Banda *et al.* (1998) dealt with this topic and showed several semantics that combine a goal-dependent and a goal-independent computation to improve precision over all the conditions.

5 Abstract domain and semantics

Several abstract domains have been used for analyses of sharing and aliasing. We use the domain Sharing (Jacobs and Langen 1992; Cortesi and Filé 1999) that computes set-sharing information:

$$\mathtt{Sharing} = \{[A, U] \mid A \subseteq \wp(U), (A \neq \emptyset \Rightarrow \emptyset \in A), U \in \wp_f(\mathscr{V})\} \cup \{\top_{\mathrm{Sh}}, \bot_{\mathrm{Sh}}\}.$$

Intuitively, an abstract object $[A, U]$ describes the relations between the variables in U: if $S \in A$, the variables in S are allowed to share a common variable. For instance, $[\{\{x, y\}, \{z\}, \emptyset\}, \{x, y, z\}]$ represents the (equivalence classes of) substitutions in which x and y may possibly share, while z is independent from both x and y: $\{x/y\}$ and ϵ are two of such substitutions while $\{x/z\}$ is not.

The domain is ordered like Psub, with $\top_{\mathrm{Sh}}$ and $\bot_{\mathrm{Sh}}$ as the greatest and least element respectively, and $[A_1, U_1] \sqsubseteq_{\mathrm{Sh}} [A_2, U_2]$ iff $U_1 = U_2$ and $A_1 \subseteq A_2$. The least upper bound satisfies the following property:

$$[A_1, U_1] \sqcup_{\mathrm{Sh}} [A_2, U_2] = \begin{cases} [A_1 \cup A_2, U_1] & \text{if } U_1 = U_2, \\ \top_{\mathrm{Sh}} & \text{otherwise.} \end{cases} \tag{16}$$

To design the abstraction from Psub to Sharing, we first define a map $\alpha_{\mathrm{Sh}} : \mathit{ISubst}_{\sim} \rightarrow \mathtt{Sharing}$ as

$$\alpha_{\mathrm{Sh}}([\sigma]_V) = [\{occ(\sigma, y) \cap V \mid y \in \mathscr{V}\}, V], \tag{17}$$

where $occ(\sigma, y) = \{z \in \mathscr{V} \mid y \in \mathrm{vars}(\sigma(z))\}$ is the set of variables z such that y occurs in $\sigma(z)$. For instance, $occ(\{x/t(y, z), x'/z, y'/z'\}, z) = \{x, x', z\}$. We call *sharing group* an element of $\wp_f(\mathscr{V})$.

We say that x is *independent* from y in $[\sigma]_V$ when, given $\alpha_{\mathrm{Sh}}([\sigma]_V) = [S, U]$, there is no $X \in S$ such that $\{x, y\} \subseteq X$. Given $U \in \wp(\mathscr{V})$, we say that x is independent from U in $[\sigma]_V$ when it is independent from y for each $y \in U$ different from x. Finally, x is independent in $[\sigma]_V$ if it is independent from V in $[\sigma]_V$.

Proposition 5.1
The map $\alpha_{\mathrm{Sh}} : \mathit{ISubst}_{\sim} \to \mathtt{Sharing}$ is well defined; i.e., it does not depend on the choice of representatives.

Proof
If $\sigma \sim_V \sigma'$, let $\rho \in \mathit{Ren}$ such that $\sigma'(x) = \rho(\sigma(x))$ for each $x \in V$. Then

$$\begin{aligned} \mathit{occ}(\sigma', \rho(y)) \cap V &= \{z \in V \mid \rho(y) \in \mathrm{vars}(\sigma'(z))\} \\ &= \{z \in V \mid y \in \rho^{-1}(\mathrm{vars}(\rho(\sigma(z))))\} \\ &= \{z \in V \mid y \in \mathrm{vars}(\sigma(z))\} \\ &= \mathit{occ}(\sigma, y) \cap V. \end{aligned}$$

Therefore, $x \in \mathit{occ}(\sigma, y) \cap V$ iff $x \in \mathit{occ}(\sigma', \rho(y)) \cap V$, which proves the thesis. □

The abstraction map may be lifted pointwise to $\alpha_{\mathrm{Sh}} : \mathtt{Psub} \to \mathtt{Sharing}$ as follows:

$$\left.\begin{aligned} \alpha_{\mathrm{Sh}}(\bot_{\mathrm{Ps}}) &= \bot_{\mathrm{Sh}}, & \alpha_{\mathrm{Sh}}(\top_{\mathrm{Ps}}) = \top_{\mathrm{Sh}}, \\ \alpha_{\mathrm{Sh}}([\Theta, U]) &= \textstyle\bigsqcup_{\mathrm{Sh}\,[\sigma]_U \in \Theta} \alpha_{\mathrm{Sh}}([\sigma]_U). & \end{aligned}\right\} \tag{18}$$

To ease the notation, often we will write a sharing group as the sequence of its elements in any order (e.g., xyz represents $\{x, y, z\}$), and we omit the empty set when clear from the context. For example,

$$\begin{gathered} \alpha_{\mathrm{Sh}}([\{[\epsilon]\}, \{x, y, z\}]) = [\{\mathrm{x}, \mathrm{y}, \mathrm{z}\}, \{x, y, z\}], \\ \alpha_{\mathrm{Sh}}([\{[x/y, z/a]\}, \{x, y, z\}]) = [\{\mathrm{xy}\}, \{x, y, z\}], \\ \alpha_{\mathrm{Sh}}([\{[\epsilon], [x/y, z/a]\}, \{x, y, z\}]) = [\{\mathrm{xy}, \mathrm{x}, \mathrm{y}, \mathrm{z}\}, \{x, y, z\}]. \end{gathered}$$

Since α_{Sh} is additive, there is an induced concretization function γ_{Sh}, the right adjoint of α_{Sh}, which maps each abstract object to the set of substitutions it represents:

$$\gamma_{\mathrm{Sh}}([S, U]) = [\{[\theta]_U \mid \alpha_{\mathrm{Sh}}([\theta]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]\}, U]. \tag{19}$$

Note that each abstract object represents the *possible* relations between variables: a substitution in which all the variables in U are ground is always in $\gamma_{\mathrm{Sh}}([A, U])$, independent from A.

Proposition 5.2
A Galois insertion is defined by $\langle \alpha_{\mathrm{Sh}}, \gamma_{\mathrm{Sh}} \rangle : \mathtt{Psub} \leftrightarrows \mathtt{Sharing}$.

Proof
That $\langle \alpha_{\mathrm{Sh}}, \gamma_{\mathrm{Sh}} \rangle$ is a Galois connection immediately follows from the fact that they are an adjoint pair. Now, we want to prove that α_{Sh} is onto. Given $[S, V] \in \mathtt{Sharing}$ and $X \in S$, consider the substitution θ_X defined as

$$\theta_X(x) = \begin{cases} w & \text{if } x \in X, \\ a & \text{if } x \in V \setminus X, \\ x & \text{otherwise,} \end{cases}$$

where w is a fresh variable not in V. It is easy to check that $\alpha_{\mathrm{Sh}}([\theta_X]_V) = [\{X\}, S]$ and therefore $\alpha_{\mathrm{Sh}}([\{[\theta_X]_V \mid X \in S\}, V]) = [S, V]$. Moreover, we have $\alpha_{\mathrm{Sh}}(\bot_{\mathrm{Ps}}) = \bot_{\mathrm{Sh}}$ and $\alpha_{\mathrm{Sh}}(\top_{\mathrm{Ps}}) = \top_{\mathrm{Sh}}$. □

5.1 The abstract semantics

The abstract semantics is obtained by replacing, in the definition of the concrete semantics in Section 4.2, the concrete domain `Psub` with the abstract domain `Sharing` and the basic operators, namely, least upper bound $\sqcup_{\mathrm{Ps}}$, forward unification $\mathbf{U}^f_{\mathrm{Ps}}$, and backward unification $\mathbf{U}^b_{\mathrm{Ps}}$ with their corresponding abstract counterparts. The abstract least upper bound $\sqcup_{\mathrm{Sh}}$ has been already defined in the previous section. We recall that on the concrete side, we have defined the forward and backward unification operators in (14) and (15) as

$$\begin{aligned}
&\mathbf{U}^f_{\mathrm{Ps}}([\Theta,U],A_1,A_2) = \pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho([\Theta,U]),\mathrm{mgu}(\rho(A_1)=A_2)),\mathrm{vars}(A_2)),\\
&\mathbf{U}^b_{\mathrm{Ps}}([\Theta_1,U_1],[\Theta_2,U_2],A_1,A_2)\\
&\qquad = \pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}(\rho([\Theta_1,U_1]),\mathsf{unif}_{\mathrm{Ps}}([\Theta_2,U_2],\mathrm{mgu}(\rho(A_1)=A_2))),U_2\cup\mathrm{vars}(A_2)).
\end{aligned}$$

The abstract forward and backward unification operators are obtained by replacing, in the above definitions, the primitive operators with their abstract counterparts, namely, abstract projection π_{Sh}, abstract renaming ρ, abstract unification $\mathsf{unif}_{\mathrm{Sh}}$, and abstract matching $\mathsf{match}_{\mathrm{Sh}}$.

The abstract operators behave exactly as the concrete ones on $\top_{\mathrm{Sh}}$ and $\bot_{\mathrm{Sh}}$. Abstract projection and renaming are defined as

$$\pi_{\mathrm{Sh}}([A_1,U_1],U_2) = [\{B\cap U_2 \mid B\in A_1\},U_1\cap U_2], \tag{20}$$
$$\rho([A,U]) = [\rho(A),\rho(U)]. \tag{21}$$

The definition of the abstract versions of matching and unification is the main argument of the rest of this paper. Here we show some properties of completeness for projection and renaming. Since the concrete and abstract operators behave in the same way on top and bottom elements, here and in the following proofs we only consider the case when all the arguments are different from $\bot_{\mathrm{Ps}}/\bot_{\mathrm{Sh}}$ and $\top_{\mathrm{Ps}}/\top_{\mathrm{Sh}}$.

Theorem 5.3

π_{Sh} is correct and complete w.r.t. π_{Ps}.

Proof

Given $[\Theta,V]\in$ `Psub`, we prove that $\alpha_{\mathrm{Sh}}(\pi_{\mathrm{Ps}}([\Theta,V],U)) = \pi_{\mathrm{Sh}}(\alpha_{\mathrm{Sh}}([\Theta,V]),U)$. We first prove that for each $[\phi]_V\in \mathit{ISubst}_{\sim}$, it holds that $\pi_{\mathrm{Sh}}(\alpha_{\mathrm{Sh}}([\phi]_V),U) = \alpha_{\mathrm{Sh}}([\phi]_{V\cap U})$. Actually

$$\begin{aligned}
\alpha_{\mathrm{Sh}}([\phi]_{V\cap U}) &= [\{occ(\phi,z)\cap V\cap U \mid z\in\mathcal{V}\},V\cap U]\\
&= \pi_{\mathrm{Sh}}([\{occ(\phi,z)\cap V \mid z\in\mathcal{V}\},V],U)\\
&= \pi_{\mathrm{Sh}}(\alpha_{\mathrm{Sh}}([\phi]_V),U).
\end{aligned}$$

The result for the lifted α_{Sh} follows trivially. □

Theorem 5.4

Abstract renaming is correct, complete, and γ-complete w.r.t. concrete renaming.

Proof

First of all, given $\rho \in Ren, y \in \mathscr{V}$ and $\phi \in Subst$, we prove that $occ(\rho(\phi), \rho(y)) = \rho(occ(\phi, y))$. Actually

$$\begin{aligned} occ(\rho(\phi), \rho(y)) &= \{z \in \mathscr{V} \mid \rho(y) \in \text{vars}(\rho(\phi(\rho^{-1}(z))))\} \\ &= \{z \in \mathscr{V} \mid y \in \text{vars}(\phi(\rho^{-1}(z)))\} \\ &= \{\rho(k) \mid k \in \mathscr{V}, y \in \text{vars}(\phi(k))\} \qquad \text{(by letting } k = \rho^{-1}(z)) \\ &= \rho(occ(\phi), y). \end{aligned}$$

Then we prove that given $[\phi]_V \in$ `Psub` and $\rho \in Ren$, $\alpha_{\text{Sh}}(\rho([\phi]_V)) = \rho(\alpha_{\text{Sh}}([\phi]_V))$. Using the fact that ρ as an operation over $ISubst_{\sim}$ is bijective, we have

$$\begin{aligned} \alpha_{\text{Sh}}(\rho([\phi]_V)) &= [\{occ(\rho(\phi), z) \cap \rho(V) \mid z \in \mathscr{V}\}, \rho(V)] \\ &= [\{\rho(occ(\phi, \rho^{-1}(z)) \cap \rho(V) \mid z \in \mathscr{V}\}, \rho(V)] \\ &= \rho([occ(\phi, k) \cap V \mid k \in \mathscr{V}\}, V]) \qquad \text{(by letting } z = \rho(k)) \\ &= \rho(\alpha_{\text{Sh}}([\phi]_V)). \end{aligned}$$

This property, lifted to `Psub`, gives the completeness of abstract renaming. Finally, we need to prove that renaming is γ-complete, i.e., $\gamma_{\text{Sh}} \circ \rho = \rho \circ \gamma_{\text{Sh}}$:

$$\begin{aligned} \gamma_{\text{Sh}}(\rho([S, V])) &= \gamma_{\text{Sh}}([\rho(S), \rho(V)]) \\ &= \left[\{[\theta]_V \mid \alpha_{\text{Sh}}([\theta]_V) \sqsubseteq_{\text{Sh}} \rho(S)\}, \rho(V)\right] \\ &= \left[\{\rho([\theta]_V) \mid \alpha_{\text{Sh}}(\rho([\theta]_V)) \sqsubseteq_{\text{Sh}} \rho(S)\}, \rho(V)\right] \\ &= \left[\{\rho([\theta]_V) \mid \rho(\alpha_{\text{Sh}}([\theta]_V)) \sqsubseteq_{\text{Sh}} \rho(S)\}, \rho(V)\right] \\ &= \left[\{\rho([\theta]_V) \mid \alpha_{\text{Sh}}([\theta]_V) \sqsubseteq_{\text{Sh}} S\}, \rho(V)\right] \\ &= \rho(\gamma_{\text{Sh}}([S, V])), \end{aligned}$$

which concludes the proof of the theorem. □

6 Forward unification

We briefly recall from Cortesi and Filé (1999) and Bagnara *et al.* (2002) the definition of the standard operator $\mathsf{unif}'_{\text{Sh}}$ for abstract unification on `Sharing`. The abstract unification is performed between a set of sharing groups A and a single substitution δ, under the assumption that $\text{vars}(\delta) \subseteq U$, and it is defined as

$$\mathsf{unif}'_{\text{Sh}}([A, U], \delta) = [\mathbf{u}_{\text{Sh}}(A, \delta), U], \tag{22}$$

where $\mathbf{u}_{\text{Sh}} : \wp(\wp_f(\mathscr{V})) \times ISubst \rightarrow \wp(\wp_f(\mathscr{V}))$ is defined by induction as follows:

$$\begin{aligned} \mathbf{u}_{\text{Sh}}(A, \epsilon) &= A, \\ \mathbf{u}_{\text{Sh}}(A, \{x/t\} \uplus \theta) &= \mathbf{u}_{\text{Sh}}(A \setminus (\mathbf{rel}(A, \{x\}) \cup \mathbf{rel}(A, \text{vars}(t))) \\ &\qquad \cup \mathbf{bin}(\mathbf{rel}(A, \{x\})^*, \mathbf{rel}(A, \text{vars}(t))^*), \theta). \end{aligned} \tag{23}$$

The auxiliary operators used in the definition of $\mathbf{u}_{\mathrm{Sh}}$ are given by:

- the *closure under union* (or *star union*) $(.)^* : \wp(\wp_f(\mathcal{V})) \to \wp(\wp_f(\mathcal{V}))$,
$$A^* = \{\bigcup T \mid \emptyset \neq T \in \wp_f(A)\}^5; \tag{24}$$
- the *extraction of relevant components* $\mathbf{rel} : \wp(\wp_f(\mathcal{V})) \times \wp_f(\mathcal{V}) \to \wp(\wp_f(\mathcal{V}))$,
$$\mathbf{rel}(A, V) = \{T \in A \mid T \cap V \neq \emptyset\}; \tag{25}$$
- the *binary union* $\mathbf{bin} : \wp(\wp_f(\mathcal{V})) \times \wp(\wp_f(\mathcal{V})) \to \wp(\wp_f(\mathcal{V}))$,
$$\mathbf{bin}(A, B) = \{T_1 \cup T_2 \mid T_1 \in A, T_2 \in B\}. \tag{26}$$

We recall that we will often abuse the notation and write $\mathbf{rel}(A, o)$ for $\mathbf{rel}(A, \mathrm{vars}(o))$ and $x \in o$ for $x \in \mathrm{vars}(o)$, where o is any syntactic object.

Example 6.1
Take $A = \{\mathrm{xy}, \mathrm{xz}, \mathrm{y}\}$, $U = \{w, x, y, z\}$, and $\delta = \{x/t(y, z), w/t(y)\}$. Note that since w does not appear in A, w is always bound to a ground term in $\gamma_{\mathrm{Sh}}([A, U])$. We have $\mathbf{rel}(A, x) = \{\mathrm{xy}, \mathrm{xz}\}$, $\mathbf{rel}(A, y) = \{\mathrm{xy}, \mathrm{y}\}$, $\mathbf{rel}(A, z) = \{\mathrm{xz}\}$, and therefore

$$\begin{aligned}\mathbf{u}_{\mathrm{Sh}}(A, \{x/t(y, z)\}) &= A \setminus \{\mathrm{xy}, \mathrm{xz}, \mathrm{y}\} \cup \mathbf{bin}(\{\mathrm{xy}, \mathrm{xz}\}^*, \{\mathrm{xy}, \mathrm{xz}, \mathrm{y}\}^*)\\ &= \mathbf{bin}(\{\mathrm{xy}, \mathrm{xz}, \mathrm{xyz}\}, \{\mathrm{xy}, \mathrm{xz}, \mathrm{xyz}, \mathrm{y}\})\\ &= \{\mathrm{xy}, \mathrm{xz}, \mathrm{xyz}\}.\end{aligned}$$

If we take $B = \{\mathrm{xy}, \mathrm{xz}, \mathrm{xyz}\}$, we obtain $\mathbf{rel}(B, w) = \emptyset$, $\mathbf{rel}(B, y) = \{\mathrm{xy}, \mathrm{xyz}\}$, and therefore

$$\begin{aligned}\mathbf{u}_{\mathrm{Sh}}(A, \delta) &= \mathbf{u}_{\mathrm{Sh}}(B, \{w/t(y)\})\\ &= B \setminus \{\mathrm{xy}, \mathrm{xyz}\} \cup \mathbf{bin}(\emptyset, \{\mathrm{xy}, \mathrm{xyz}\}^*)\\ &= B \setminus \{\mathrm{xy}, \mathrm{xyz}\}\\ &= \{\mathrm{xz}\}. \quad \square\end{aligned}$$

It is worth noting that $\mathsf{unif}'_{\mathrm{Sh}}$ is not the abstract counterpart of $\mathsf{unif}_{\mathrm{Ps}}$ because $\mathsf{unif}'_{\mathrm{Sh}}([S, U], \delta)$ is defined only under the condition that $\mathrm{vars}(\delta) \subseteq U$. Since this is not enough to define a goal-dependent semantics, when this solution is adopted, there is the need of an operator to expand the set of variables of interest in a substitution. Let us introduce the concrete operator

$$\iota_{\mathrm{Ps}}([\Theta, U], V) = [\{\mathrm{mgu}([\sigma]_U, [\epsilon]_V) \mid [\sigma]_U \in \Theta\}, U \cup V], \tag{27}$$

whose optimal abstract counterpart is simply given by

$$\iota_{\mathrm{Sh}}([\Theta, U], V) = [\Theta \cup \{\{x\} \mid x \in V \setminus U\}, U \cup V)]. \tag{28}$$

By using ι_{Ps}, the operator $\mathsf{unif}_{\mathrm{Ps}}$ can be equivalently rewritten as

$$\mathsf{unif}_{\mathrm{Ps}}([\Theta, U], \theta) = \mathsf{unif}_{\mathrm{Ps}}(\iota_{\mathrm{Ps}}([\Theta, U], \mathrm{vars}(\theta)), \theta), \tag{29}$$

[5] Note that due to the condition $T \neq \emptyset$, the notation A^+ would be more appropriate. However, we retain the notation A^* for historical reasons.

and now, on the right-hand side, $\iota_{\mathrm{Ps}}([\Theta, U], \mathrm{vars}(\theta))$ is an object of the kind $[\Delta, U \cup \mathrm{vars}(\theta)]$. Therefore, a correct abstract forward unification operator for $\mathbf{U}^f_{\mathrm{Ps}}$ may be obtained as

$$\begin{aligned}\mathbf{U}'^f_{\mathrm{Sh}}([\Theta, U], A_1, A_2) = \pi_{\mathrm{Sh}}(\mathsf{unif}'_{\mathrm{Sh}}(\iota_{\mathrm{Sh}}(\rho([\Theta, U]), \mathrm{vars}(\rho(A_1)) \cup \mathrm{vars}(A_2)),\\ \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2)),\end{aligned} \tag{30}$$

provided that ρ is a renaming such that $\rho(U \cup \mathrm{vars}(A_1)) \cap \mathrm{vars}(A_2) = \emptyset$. However, $\mathbf{U}'^f_{\mathrm{Sh}}$ is not optimal w.r.t. $\mathbf{U}^f_{\mathrm{Ps}}$.

Example 6.2
We keep on Example 4.4 and compute the abstract counterpart of the concrete forward unification,

$$\mathbf{U}^f_{\mathrm{Ps}}([\{[y/f(x,z)]\}, \{x,y,z\}], p(x,y,z), p(u,v,w)) = [\{[u/x, v/f(x,z), w/z]\}, \{u,v,w\}].$$

Since the abstraction of $[\{[y/f(x,z)]\}, \{x,y,z\}]$ is $[\{\mathtt{xy}, \mathtt{yz}\}, \{x,y,z\}]$, we compute

$$\begin{aligned}&\mathbf{U}'^f_{\mathrm{Sh}}([\{\mathtt{xy}, \mathtt{yz}\}, \{x,y,z\}], p(x,y,z), p(u,v,w))\\ &\quad = \pi_{\mathrm{Sh}}([\mathbf{u}_{\mathrm{Sh}}(\{\mathtt{xy}, \mathtt{yz}, \mathtt{u}, \mathtt{v}, \mathtt{w}\}, \{x/u, y/v, z/w\}), \{x,y,z,u,v,w\}], \{u,v,w\})\\ &\quad = \pi_{\mathrm{Sh}}([\{\mathtt{xyuv}, \mathtt{yzvw}, \mathtt{xyzuvw}\}, \{x,y,z,u,v,w\}], \{u,v,w\})\\ &\quad = [\{\mathtt{uv}, \mathtt{vw}, \mathtt{uvw}\}, \{u,v,w\}].\end{aligned}$$

There exists a sharing group $\mathtt{uvw}$ computed by the forward unification. However, when computing $\mathsf{unif}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([\{\mathtt{xy}, \mathtt{yz}\}, \{x,y,z\}]), \{x/u, y/v, z/w\})$ we know that u, v, and w are free in $\gamma_{\mathrm{Sh}}([\{\mathtt{xy}, \mathtt{yz}\}, \{x,y,z\}]$. Following Hans and Winkler (1992), we can avoid computing the star unions when considering the binding y/v in $\mathbf{u}_{\mathrm{Sh}}$, obtaining the smaller result $[\{\mathtt{xyuv}, \mathtt{yzvw}\}, \{x,y,z,u,v,w\}]$. If we now compute the projection on the variables $\{u,v,w\}$ we obtain the entry substitution $[\{\mathtt{uv}, \mathtt{vw}\}, \{u,v,w\}]$, with an obvious gain of precision. □

Example 6.3
Let us consider the following unification:

$$\begin{aligned}&\mathbf{U}'^f_{\mathrm{Sh}}([\{\mathtt{xy}, \mathtt{xz}\}, \{x,y,z\}], p(x,y,z), p(t(u,v),h,k))\\ &\qquad = \pi_{\mathrm{Sh}}([\mathbf{bin}(\{\mathtt{xyh}, \mathtt{xzk}, \mathtt{xyzhk}\}, \{\mathtt{u}, \mathtt{v}, \mathtt{uv}\}), \{x,y,z,h,k,u\}], \{u,v,h,k\}).\end{aligned}$$

Since the term $t(u,v)$ is linear and independent from x, following Hans and Winkler (1992) we can avoid to compute the star union over $\{\mathtt{xy}, \mathtt{xz}\}$, obtaining the abstract object $[\mathbf{bin}(\{\mathtt{xyh}, \mathtt{xzk}\}, \{\mathtt{u}, \mathtt{v}, \mathtt{uv}\}), \{x,y,z,h,k,u\}]$. If we project on $\{h,k,u,v\}$ we obtain $\mathbf{bin}(\{\mathtt{h}, \mathtt{k}\}, \{\mathtt{u}, \mathtt{v}, \mathtt{uv}\})$ against $\mathbf{bin}(\{\mathtt{h}, \mathtt{k}, \mathtt{hk}\}, \{\mathtt{u}, \mathtt{v}, \mathtt{uv}\})$. In this way, we are able to prove the independence of h from k. □

These examples show that when computing forward abstract unification by first enlarging the domain of variables of interest, there is a loss of precision. In fact, such a forward abstract unification operator is not optimal. We now show that it is possible to design an optimal operator for forward unification that is able to exploit linearity and freeness information that stems from the fact that variables in the third argument of $\mathbf{U}^f_{\mathrm{Ps}}$ are fresh. Note that we are not proposing to embed freeness and linearity information inside the domain but only to use all the information coming from the syntax of the clauses.

6.1 The refined forward unification

We are going to define an abstract operator $\mathsf{unif}_{\mathrm{Sh}}$ that is correct and optimal w.r.t. $\mathsf{unif}_{\mathrm{Ps}}$.

Definition 6.4
The *abstract unification* $\mathsf{unif}_{\mathrm{Sh}} : \mathtt{Sharing} \times \mathit{ISubst} \to \mathtt{Sharing}$ is defined as

$$\mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta) = [\mathbf{u}^f_{\mathrm{Sh}}(S_1 \cup \{\{x\} \mid x \in U_2\}, U_2, \theta), U_1 \cup U_2],$$

where $U_2 = \mathrm{vars}(\theta) \setminus U_1$ and $\mathbf{u}^f_{\mathrm{Sh}} : \wp(\wp_f(\mathscr{V})) \times \wp_f(\mathscr{V}) \times \mathit{ISubst} \to \wp(\wp_f(\mathscr{V}))$ is defined as

$$\begin{aligned}
\mathbf{u}^f_{\mathrm{Sh}}(S, U, \epsilon) &= S,\\
\mathbf{u}^f_{\mathrm{Sh}}(S, U, \{x/t\} \uplus \delta) &= \mathbf{u}^f_{\mathrm{Sh}}((S \setminus (\mathbf{rel}(S,t) \cup \mathbf{rel}(S,x))) \cup \\
&\quad \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,t)), U \setminus \{x\}, \delta) && \text{if } x \in U,\\
\mathbf{u}^f_{\mathrm{Sh}}(S, U, \{x/t\} \uplus \delta) &= \mathbf{u}^f_{\mathrm{Sh}}((S \setminus (\mathbf{rel}(S,t) \cup \mathbf{rel}(S,x))) \cup \\
&\quad \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,Y)^*) \cup \\
&\quad \mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*) \cup \\
&\quad \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*), \mathbf{rel}(S,Y)^*), \\
&\quad U \setminus \mathrm{vars}(\{x/t\}), \delta) && \text{if } x \notin U,
\end{aligned}$$

where $Y = \mathrm{uvars}(t) \cap U$, $Z = \mathrm{vars}(t) \setminus Y$.

The idea is simply to carry on, in the second argument of $\mathbf{u}^f_{\mathrm{Sh}}$, the set of variables that are definitively free and to apply the optimizations for the abstract unification with linear terms and free variables (Hans and Winkler 1992). Actually, while the case for $x \in U$ is standard, the case for $x \notin U$ exploits some optimizations that are not found in the literature. When $Z = \emptyset$, we obtain

$$(S \setminus (\mathbf{rel}(S,t) \cup \mathbf{rel}(S,x))) \cup \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,Y)^*),$$

which is the standard result when the term t is linear and independent from x. However, when $Z \neq \emptyset$, the standard optimizations that appear, e.g., in Hans and Winkler (1992), do not apply, since t cannot be proved to be linear and independent from x, and we should obtain the following standard result:

$$(S \setminus (\mathbf{rel}(S,t) \cup \mathbf{rel}(S,x))) \cup \mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,t)^*).$$

We are able to avoid some star unions by distinguishing the variables in t that are "linear and independent" (the set Y) from the others (the set Z) and observing that two sharing groups in $\mathbf{rel}(S,x)$ may be merged together only under the effect of the unification with some variable in Z. We will come back later to this topic.

We can now define the forward abstract unification $\mathbf{U}^f_{\mathrm{Sh}} : \mathtt{Sharing} \times \wp_f(\mathscr{V}) \times \mathsf{Atoms} \times \mathsf{Atoms} \rightarrow \mathtt{Sharing}$. We only need to introduce the necessary renamings and projections, as done for the concrete case,

$$\mathbf{U}^f_{\mathrm{Sh}}([S_1, U_1], A_1, A_2) = \pi_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Sh}}(\rho([S_1, U_1]), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2)), \tag{31}$$

with ρ a renaming such that $\rho(U_1 \cup \mathrm{vars}(A_1)) \cap \mathrm{vars}(A_2) = \emptyset$.

Example 6.5

We keep on Examples 4.4 and 6.2 and compute the abstract counterpart of the concrete forward unification,

$$\mathbf{U}^f_{\mathrm{Ps}}([\{[y/f(x,z)]\}, \{x,y,z\}], p(x,y,z), p(u,v,w)) = [\{[u/x, v/f(x,z), w/z]\}, \{u,v,w\}],$$

using our optimized forward unification operator:

$$\begin{aligned}
&\mathbf{U}^f_{\mathrm{Sh}}([\{\mathtt{xy}, \mathtt{yz}\}, \{x,y,z\}], p(x,y,z), p(u,v,w)) \\
&\quad = \pi_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Sh}}(\{\mathtt{xy}, \mathtt{yz}\}, \{x/u, y/v, z/w\}), \{u,v,w\}) \\
&\quad = \pi_{\mathrm{Sh}}([\{\mathtt{uvxy}, \mathtt{vwyz}\}, \{u,v,w,x,y,z\}], \{u,v,w\}) \\
&\quad = [\{\mathtt{uv}, \mathtt{vw}\}, \{u,v,w\}].
\end{aligned}$$

Thus the optimized operator is able to prove that u and w are independent after the unification. □

6.2 Correctness of forward unification

We prove that the unification operator $\mathsf{unif}_{\mathrm{Sh}}$ is correct w.r.t. the concrete operator $\mathsf{unif}_{\mathrm{Ps}}$. We begin to analyze the abstract behavior of unification when the second argument is a substitution with only one binding. Let σ and $\{x/t\}$ be the two substitutions we want to unify. In this simple case, the resultant sharing groups can be easily computed by exploiting the substitution $\delta = \mathrm{mgu}(x\sigma = t\sigma)$. We show that under suitable conditions, any sharing group either belongs to $\alpha_{\mathrm{Sh}}([\sigma]_U)$ or is of the form $occ(\sigma, occ(\delta, v)) \cap U$, where $v \in \mathrm{vars}(x\sigma = t\sigma)$.

Proposition 6.6

Let $[\sigma]_U \in \mathit{ISubst}_{\sim}$ and $\{x/t\} \in \mathit{ISubst}$ such that $\mathrm{vars}(\{x/t\}) \subseteq U$ and σ and $\{x/t\}$ unify. If $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$ and $\delta = \mathrm{mgu}(x\sigma = t\sigma)$, we obtain

$$\begin{aligned}
\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\mathrm{Sh}} {} & [(S \setminus (\mathbf{rel}(S,x) \cup \mathbf{rel}(S,t))) \\
& \cup \{occ(\sigma, occ(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}, U].
\end{aligned}$$

Proof

The proof can be found in Appendix B as Proposition B.3 □

This result may be refined by introducing further hypotheses. We have anticipated that our abstract algorithm takes advantage of the fact that some variables are known to be free in order to produce better results than standard abstract unification. We may be more formal.

Definition 6.7
We say that a variable $x \in \mathcal{V}$ is *free* in $[\theta]_V$ when $\theta_{|V}(x) \in \mathcal{V}$.

Note that this definition does not depend on the choice of the representative for $[\theta]_V$. Moreover, if x is free and independent from V in $[\theta]_V$, there exists a representative $\theta' \sim_V \theta$ such that $x \notin \text{vars}(\theta')$. It is enough to take $\theta' = \theta''_{|\{-x\}}$, where θ'' is a canonical representative.

Now, we consider again Proposition 6.6, but we assume x to be free and independent from U in $[\sigma]_U$. A result similar to the following proposition has been already proved in the literature (e.g., Hans and Winkler 1992). Since our treatment of substitutions is slightly different from the standard one, for the sake of completeness we present the altered proof.

Proposition 6.8
Let $[\sigma]_U \in \mathit{ISubst}_{\sim}$ and $\{x/t\} \in \mathit{ISubst}$ such that $\text{vars}(\{x/t\}) \subseteq U$ and σ and $\{x/t\}$ unify. If $\alpha_{\text{Sh}}([\sigma]_U) \sqsubseteq_{\text{Sh}} [S, U]$ and x is free and independent from U in $[\sigma]_U$, then

$$\alpha_{\text{Sh}}(\text{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\text{Sh}} [(S \setminus (\mathbf{rel}(S,x) \cup \mathbf{rel}(S,t))) \cup \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,t)), U].$$

Proof
The proof can be found in Appendix B as Proposition B.4 □

Now we analyze the case in which x is not guaranteed to be free and independent from U in $[\sigma]_U$. We show that it is possible to consider three distinct cases depending on the set of variables $Y = \{y \in \text{vars}(t) | \text{vars}(\sigma(y)) \subseteq \text{uvars}(x\sigma = t\sigma)\}$, that is, the set of variables y such that all the variables in $\text{vars}(\sigma(y))$ appear once in $x\sigma = t\sigma$. Such variables play a special role in the unification process. Generally speaking, we can form new sharing groups by merging sets from $\mathbf{rel}(S,x)$ and $\mathbf{rel}(S,t)$. Obviously, any new sharing group must be formed by choosing at least one element from $\mathbf{rel}(S,x)$ and at least one from $\mathbf{rel}(S,t)$. We show that if we do not include any variable from $\text{vars}(t) \setminus Y$, then we may avoid the inclusion of more than one sharing group from $\mathbf{rel}(S,x)$. Intuitively speaking, variables from Y do not allow the merging of different sharing groups from $\mathbf{rel}(S,x)$, since such variables appear only once and thus cannot be bound to different occurrences of x.

Example 6.9
Let $\sigma = \{x/f(u,v)\}$, $U = \{u,v,x,y,z\}$ and consider the binding $x/f(f(y,z),z)$. We have that $Y = \{y\}$, $\alpha_{\text{Sh}}([\sigma]_U) = [S, U] = [\{\texttt{ux}, \texttt{vx}, \texttt{y}, \texttt{z}\}, U]$, $\mathbf{rel}(S,x) = \{\texttt{ux}, \texttt{vx}\}$, and $\mathbf{rel}(S,t) = \{\texttt{y}, \texttt{z}\}$. In the standard definition of abstract unification, uvxy would be one of the possible resultant sharing groups. However, since uvxy is obtained by joining two sharing groups in $\mathbf{rel}(S,x)$ and does not contain any variable in $\text{vars}(t) \setminus Y$, it cannot be generated. In fact, the result of the unification is $\eta = \{x/f(f(y,z),z), u/f(y,z), v/z\}$ and $\alpha_{\text{Sh}}([\eta]_U) = [\{\texttt{uxy}, \texttt{uvxz}\}, U]$. The variables u and v occur in the same sharing group thanks to the two occurrences of z. □

Proposition 6.10
Let $[\sigma]_U \in \mathit{ISubst}_\sim$ and $\{x/t\} \in \mathit{ISubst}$ such that $\mathrm{vars}(\{x/t\}) \subseteq U$ and σ and $\{x/t\}$ unify. Given $Y \subseteq \mathrm{vars}(t)$ such that for all $y \in Y$, $\mathrm{vars}(\sigma(y)) \subseteq \mathrm{uvars}(x\sigma = t\sigma)$, if $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$, then

$$\begin{aligned}\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\mathrm{Sh}} [&(S \setminus (\mathbf{rel}(S,t) \cup \mathbf{rel}(S,x)))\\ &\cup \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,Y)^*) \cup \mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*)\\ &\cup \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*), \mathbf{rel}(S,Y)^*), U],\end{aligned}$$

where $Z = \mathrm{vars}(t) \setminus Y$.

Proof
The proof can be found in Appendix B as Proposition B.6 □

Now, by combining the results from Propositions 6.8 and 6.10 we can show the correctness of $\mathsf{unif}_{\mathrm{Sh}}$.

Theorem 6.11
(Correctness of $\mathsf{unif}_{\mathrm{Sh}}$)
The unification operator $\mathsf{unif}_{\mathrm{Sh}}$ is correct w.r.t. $\mathsf{unif}_{\mathrm{Ps}}$.

Proof
The proof can be found in Appendix B as Theorem B.8 □

6.3 Optimality of forward unification

In this section we prove that the abstract unification operator $\mathsf{unif}_{\mathrm{Sh}}$ is optimal w.r.t. the concrete operator $\mathsf{unif}_{\mathrm{Ps}}$, that is to say that given $[S_1, U_1] \in \mathtt{Sharing}$ and $\theta \in \mathit{ISubst}$, the following holds:

$$\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), \theta)) \sqsupseteq_{\mathrm{Sh}} \mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta).$$

Let $\mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta) = [S, U]$, where $U = U_1 \cup \mathrm{vars}(\theta)$. In the rest of this section, we assume fixed S, S_1, U, U_1, θ as defined above.

For each $X \in S$, we need to exhibit a substitution δ such that $\alpha_{\mathrm{Sh}}([\delta]_{U_1}) \sqsubseteq_{\mathrm{Sh}} [S_1, U_1]$ and $\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\delta]_{U_1}, [\theta]_U)) \sqsupseteq_{\mathrm{Sh}} [\{X\}, U]$. Any resultant sharing group is obtained by merging together sharing groups from S_1 and variables in $\mathrm{vars}(\theta) \setminus U_1$. We show that two sharing groups B_1 and B_2 may be joined by the abstract unification algorithm only if there are two variables $x_1 \in B_1$, $x_2 \in B_2$ such that $\theta(x_1)$ and $\theta(x_2)$ share some variable. Actually, we need to be careful when $x_1 = x_2$, since we need a variable which occurs at least twice in $\theta(x_1)$. More formally, given $X \in \wp_f(\mathcal{V})$ and $\theta \in \mathit{ISubst}$, we define a relation $\mathcal{R}_{\theta X} \subseteq S_1 \times S_1$ as follows:

$$\begin{aligned}B_1 \mathcal{R}_{\theta X} B_2 \iff \exists x_1 \in B_1\ \exists x_2 \in B_2\ \exists y.\ (y \in \mathrm{vars}(\theta(x_1)) \cap \mathrm{vars}(\theta(x_2)) \cap X) \wedge\\ (x_1 = x_2 \implies y \notin \mathrm{uvars}(\theta(x_1))).\quad (32)\end{aligned}$$

We say that X is θ-connected when there exist $B_1, \ldots, B_n \in S_1$ such that $\cup_{1 \leqslant j \leqslant n} B_j = X \cap U_1$ and $B_1 \mathcal{R}^*_{\theta X} B_2 \ldots \mathcal{R}^*_{\theta X} B_n$, where $\mathcal{R}^*_{\theta X}$ is the transitive closure of $\mathcal{R}_{\theta X}$.

Lemma 6.12
For each $X \in S$, X is θ-connected.

Proof
The proof can be found in Appendix C as Lemma C.4 □

Now we will exploit the relation $\mathscr{R}_{\theta X}$ in order to find a substitution δ such that the concrete unification of θ with δ mimics the behavior of the abstract unification of θ with $[S_1, U_1]$. We define a δ that has exactly the sharing groups $B_1, \dots, B_n$ and is obtained by instantiating θ. The idea is that if $B_1 \mathscr{R}_{\theta X} B_2$ due to $x_1 \in B_1$, $x_2 \in B_2$, and the common variable $y \in \theta(x_1) \cap \theta(x_2)$, then the occurrences of y in $\theta(x_1)$ and $\theta(x_2)$ are replaced by two suitable terms that unify and merge together the two sharing groups B_1 and B_2.

Example 6.13
Let $\theta = \{x/f(u), y/g(u)\}$ and $[S_1, U_1] = [\{\mathtt{xw}, \mathtt{yz}\}, \{w, x, y, z\}]$. Consider $B_1 = \mathtt{xw}$ and $B_2 = \mathrm{yz}$. We choose the variables $x \in B_1$ and $y \in B_2$. Since $u \in \theta(x) \cap \theta(y)$, we can choose the substitution $\delta = \{x/f(w_1), y/g(w_2), w/w_1, z/w_2\}$ obtained from θ by replacing each occurrence of u, w, z with suitable new terms. It is easy to verify that θ and δ unify and that $\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\delta]_{\{w,x,y,z\}}, [\theta]_{\{u,w,x,y,z\}})) \sqsupseteq_{\mathrm{Sh}} [\{\mathtt{uwxyz}\}, \{u, w, x, y, z\}]$. □

Example 6.14
Let $\theta = \{x/f(u,u)\}$ and $[S_1, U_1] = [\{\mathtt{xw}, \mathtt{xy}, \mathtt{xz}\}, \{w, x, y, z\}]$. Consider $B_1 = \mathtt{xw}$, $B_2 = \mathrm{xy}$, and $B_3 = \mathrm{xz}$. We choose the variable $x \in B_1 \cap B_2 \cap B_3$. Then $u \notin \mathrm{uvars}(\theta(x))$, and we can choose as δ the substitution

$$\{x/f(t(w_1, w_1), t(w_2, w_3)), w/w_1, y/w_2, z/w_3\},$$

obtained from θ by replacing each occurrence of u, w, y, z with suitable new terms. It is easy to see that θ and δ unify and that $\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\delta]_{\{w,x,y,z\}}, [\theta]_{\{u,w,x,y,z\}})) \sqsupseteq_{\mathrm{Sh}} [\{\mathtt{uwxyz}\}, \{u, w, x, y, z\}]$. □

Following this idea we can now prove that mgu and $\mathsf{unif}_{\mathrm{Sh}}$ are optimal.

Proposition 6.15
For all $X \in S$ there exists $[\delta]_{U_1} \in \mathit{ISubst}_{\sim}$ such that $\alpha_{\mathrm{Sh}}([\delta]_{U_1}) \leqslant_{\mathrm{Sh}} [S_1, U_1]$ and

$$\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\delta]_{U_1}, [\theta]_U)) \sqsupseteq_{\mathrm{Sh}} [\{X\}, U].$$

Proof
The proof can be found in Appendix C as Proposition C.6 □

The optimality result for $\mathsf{unif}_{\mathrm{Sh}}$ w.r.t. $\mathsf{unif}_{\mathrm{Ps}}$ immediately follows from the above proposition.

Theorem 6.16
(Optimality of $\mathsf{unif}_{\mathrm{Sh}}$)
$\mathsf{unif}_{\mathrm{Sh}}$ is optimal w.r.t. $\mathsf{unif}_{\mathrm{Ps}}$.

Optimality of $\mathsf{unif}_{\mathrm{Sh}}$ also implies the following corollary.

Corollary 6.17
The result of $\mathsf{unif}_{\mathrm{Sh}}$ does not depend on the order of the bindings in its second argument.

6.4 Summing up

We may put together all the results of correctness, optimality, and completeness shown so far to prove the main theorem of this section.

Theorem 6.18
$\mathbf{U}^{f}_{\mathrm{Sh}}$ is well defined, correct, and optimal w.r.t. $\mathbf{U}^{f}_{\mathrm{Ps}}$.

Proof
The proof can be found in Appendix C as Theorem C.7 □

Generally speaking, in order to obtain optimality, it is always a better choice to abstract a concrete operator "as a whole," instead of abstracting each component and then composing the abstract operators. According to this rule, we could think that a better approximation may be reached by abstracting $\mathbf{U}^{f}_{\mathrm{Ps}}$ as a whole. However, since abstract projection/renaming is complete and γ-complete, this does not happen, as shown by the previous theorem. Studying the direct abstraction of this composition would still be useful to find a direct implementation that is more efficient than computing $\mathsf{unif}_{\mathrm{Sh}}$ and projecting later, but we do not consider this problem here.

Since $\mathbf{U}^{f}_{\mathrm{Sh}}$ generates less sharing groups than $\mathbf{U}'^{f}_{\mathrm{Sh}}$ and since checking whether a variable is in U is easy, we can expect an improvement in the efficiency of the analysis by replacing $\mathbf{U}'^{f}_{\mathrm{Sh}}$ with $\mathbf{U}^{f}_{\mathrm{Sh}}$ in the computation of the entry substitution. If computing Y and Z at each step of $\mathbf{u}^{f}_{\mathrm{Sh}}$ seems difficult, it is always possible to precompute these values before the actual analysis begins, since they depend on the syntax of the program only. Moreover, in the definition of $\mathbf{u}^{f}_{\mathrm{Sh}}$, when $x \in U$ we know that $\mathbf{rel}(S,x) = \{\{x\}\}$, since θ is an idempotent substitution and $x \notin U_1$.

A further optimization is obtained by replacing $\mathbf{rel}(S,Y)$ with the set of all the sharing groups whose variables are all contained in Y. Clearly, this is a subset of $\mathbf{rel}(S,Y)$, and it is immediate to check that the result of $\mathbf{u}^{f}_{\mathrm{Sh}}$ does not change. In fact, all the sharing groups in $\mathbf{bin}(\mathbf{rel}(S,x),\mathbf{rel}(S,Y)^*)$ that are not generated anymore may be found in $\mathbf{bin}(\mathbf{rel}(S,x)^*,\mathbf{rel}(S,Z)^*)$.

We said before that this operator introduces new optimizations that, to the best of our knowledge, are not used even in more complex domains for sharing analysis, which include linearity and freeness information. We give here one example that shows their effects.

Example 6.19
Let us consider the following unification:

$$\mathbf{U}^{f}_{\mathrm{Sh}}([\{\mathtt{xw},\mathtt{xz},\mathtt{yw},\mathtt{yz}\},\{x,y,w,z\}],p(x,y,w,z),p(f(u,h),f(u,k),s,t)).$$

By applying the optimizations suggested from the unification algorithm in presence of linearity and freeness information in Hans and Winkler (1992), we may start from the abstract object $S = \{\mathtt{xw},\mathtt{xz},\mathtt{yw},\mathtt{yz},\mathtt{u},\mathtt{h},\mathtt{k},\mathtt{s},\mathtt{t}\}$ and process the bindings one at a time, keeping in mind that u, h, k, s, t are initially free. This means that in the binding $x/f(u, h)$, the term $f(u, h)$ is linear, and therefore we can avoid to compute the star union in $\mathbf{rel}(S, x)$, thus obtaining

$$\{\mathtt{k},\mathtt{s},\mathtt{t},\mathtt{yw},\mathtt{yz}\} \cup \mathbf{bin}(\{\mathtt{xw},\mathtt{xz}\},\{\mathtt{u},\mathtt{h},\mathtt{uh}\}) \\ = \{\mathtt{k},\mathtt{s},\mathtt{t},\mathtt{yw},\mathtt{yz},\mathtt{xwu},\mathtt{xwh},\mathtt{xzu},\mathtt{xzh},\mathtt{xwuh},\mathtt{xzuh}\}.$$

However, after this unification, the variable u can be bound to a nonlinear term. Therefore, when we consider the next binding $y/f(u, k)$, according to Hans and Winkler (1992), we are forced to compute all the star unions, obtaining

$$\{\mathtt{s},\mathtt{t}\} \cup \mathbf{bin}(\{\mathtt{yw},\mathtt{yz}\}^*, (\{\mathtt{k}\} \cup \mathbf{bin}(\{\mathtt{xw},\mathtt{xz}\},\{\mathtt{u},\mathtt{uh}\}))^*) \cup \{\mathtt{xwh},\mathtt{xzh}\}.$$

Finally, in the bindings w/s and z/t we may omit all the star unions, since t and s are free, and we get the final result

$$\mathbf{bin}(\{\mathtt{yws},\mathtt{yzt}\}^*, (\{\mathtt{k}\} \cup \mathbf{bin}(\{\mathtt{xws},\mathtt{xzt}\},\{\mathtt{u},\mathtt{uh}\}))^*) \cup \{\mathtt{xwsh},\mathtt{xzth}\}.$$

Observe that we obtain the sharing group ywsztk, and thus, after projecting on $\{u, h, k, s, t\}$, we obtain the sharing group stk. However, when we consider the second binding, we know that k is free and independent from y, and this is enough to apply a new optimization. In fact, k can share with more than one sharing group related to y only if k shares with u. If we compute the abstract unification with our algorithm, we obtain

$$\{\mathtt{ywsk},\mathtt{yztk}\} \cup \mathbf{bin}(\{\mathtt{yws},\mathtt{yzt}\}^*, \mathbf{bin}(\{\mathtt{xws},\mathtt{xzt}\},\{\mathtt{u},\mathtt{uh}\})^*) \\ \cup \mathbf{bin}(\mathbf{bin}(\{\mathtt{yws},\mathtt{yzt}\}^*, \mathbf{bin}(\{\mathtt{xws},\mathtt{xzt}\},\{\mathtt{u},\mathtt{uh}\})^*), \{k\}) \cup \{\mathtt{xwsh},\mathtt{xzth}\},$$

and when we project on $\{u, h, k, s, t\}$, the sharing group stk does not appear. In fact, note that any sharing group generated by

$$\mathbf{bin}(\mathbf{bin}(\{\mathtt{yws},\mathtt{yzt}\}^*, \mathbf{bin}(\{\mathtt{xws},\mathtt{xzt}\},\{\mathtt{u},\mathtt{uh}\})^*), \{k\})$$

contains the variable u. The result does not change by permuting the order of the bindings. If we consider the binding $y/f(u, k)$ before $x/f(u, h)$, with the standard operators we get

$$\mathbf{bin}(\{\mathtt{xws},\mathtt{xzt}\}^*, (\{\mathtt{h}\} \cup \mathbf{bin}(\{\mathtt{yws},\mathtt{yzt}\},\{\mathtt{u},\mathtt{uk}\}))^*) \cup \{\mathtt{ywsk},\mathtt{yztk}\},$$

and when we project on $\{u, h, k, s, t\}$, we obtain the sharing group sth, which does not appear in our result. □

7 Matching and backward unification

To the best of our knowledge, in all the collecting denotational semantics for logic programs, backward unification is performed by using unification instead of matching. This means that instead of $\mathbf{U}^b_{\mathrm{Ps}}$, the concrete semantics uses a backward unification operator that unifies two concrete objects in Psub with a substitution,

$$\mathbf{U}'^{b}_{\mathrm{Ps}}([\Delta_1, U_1], [\Delta_2, U_2], A_1, A_2)$$
$$= \pi_{\mathrm{Ps}}(\mathsf{unif}''_{\mathrm{Ps}}(\rho([\Delta_1, U_1]), [\Delta_2, U_2], \mathrm{mgu}(\rho(A_1) = A_2)), U_2 \cup \mathrm{vars}(A_2)), \quad (33)$$

where ρ is a renaming such that $\rho(U_1 \cup \mathrm{vars}(A_1)) \cap (U_2 \cup \mathrm{vars}(A_2)) = \emptyset$ and

$$\mathsf{unif}''_{\mathrm{Ps}}([\Delta_1, U_1], [\Delta_2, U_2], \delta)$$
$$= [\{\mathrm{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2}, [\delta]_{\mathrm{vars}(\delta)}) \mid [\theta_1]_{U_1} \in \Delta_1, [\theta_2]_{U_2} \in \Delta_2\}, U_1 \cup U_2] \quad (34)$$

is simply the pointwise extension of mgu over Psub. It is worth observing that $\mathsf{unif}''_{\mathrm{Ps}}(\rho([\Delta_1, U_1]), [\Delta_2, U_2], \delta)$ is a very specific kind of unification, since $\rho(U_1)$ and U_2 are disjoint. The optimal abstract operator $\mathbf{U}'^{b}_{\mathrm{Sh}}$ w.r.t. $\mathbf{U}'^{b}_{\mathrm{Ps}}$ is very similar to that proposed in Cortesi and Filé (1999) (see Section 8.2 for further details), and it is given by

$$\mathbf{U}'^{b}_{\mathrm{Sh}}([S_1, U_1], [S_2, U_2], A_1, A_2)$$
$$= \pi_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Sh}}([\rho(S_1) \cup S_2, \rho(U_1) \cup U_2], \mathrm{mgu}(\rho(A_1) = A_2)), U_2 \cup \mathrm{vars}(A_2)). \quad (35)$$

As said before, this choice results in a loss of precision already at the concrete level, which leads to a loss of precision in the abstract counterpart. When we compute $\mathbf{U}'^{b}_{\mathrm{Ps}}([\Delta_1, U_1], [\Delta_2, U_2], A_1, A_2)$, we essentially unify all pairs θ_1 and θ_2, elements of Δ_1 and Δ_2, with $\delta = \mathrm{mgu}(A_1 = A_2)$ (assuming we do not need renamings). However, it could be possible to consider only the pairs in which θ_1 is an instance of $\mathrm{mgu}(\theta_2, \delta)$ w.r.t. the variables of interest in $U_1 \cap U_2$. If this does not hold, then θ_1 cannot be a success substitution corresponding to the call substitution θ_2, and therefore we are unifying two objects that pertain to different computational paths, with an obvious loss of precision, already at the concrete level. This problem has been pointed out by Marriott *et al.* (1994).

We now want to define the optimal abstract operator $\mathbf{U}^{b}_{\mathrm{Sh}}$ corresponding to $\mathbf{U}^{b}_{\mathrm{Ps}}$. This is accomplished by composing the forward unification operator $\mathsf{unif}_{\mathrm{Sh}}$ with a new operator $\mathsf{match}_{\mathrm{Sh}}$, which is the abstract counterpart of $\mathsf{match}_{\mathrm{Ps}}$.

Definition 7.1
Given $[S_1, U_1], [S_2, U_2] \in \mathtt{Sharing}$, we define

$$\mathsf{match}_{\mathrm{Sh}}([S_1, U_1], [S_2, U_2])$$
$$= [S'_1 \cup S'_2 \cup \{X_1 \cup X_2 \mid X_1 \in S''_1, X_2 \in (S''_2)^*, X_1 \cap U_2 = X_2 \cap U_1\}, U_1 \cup U_2],$$

where $S'_1 = \{B \in S_1 \mid B \cap U_2 = \emptyset\}$ and $S''_1 = S_1 \setminus S'_1$, $S'_2 = \{B \in S_2 \mid B \cap U_1 = \emptyset\}$, and $S''_2 = S_2 \setminus S'_2$

The idea is that we may freely combine those sharing groups in S_2 that have some variable in common with U_1, i.e., $X_2 \in (S''_2)^*$, if the projection of the result on U_1 is equal to some sharing group in S_1, when projected on U_2. This means that new aliasings between variables may arise in the concrete counterpart of S_2 (the entry substitution), as long as they do not affect the variables of the exit substitution.

Definition 7.2
The abstract backward unification may be defined as

$$\mathbf{U}^b_{\mathrm{Sh}}([S_1,U_1],[S_2,U_2],A_1,A_2) = \pi_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Sh}}(\rho([S_1,U_1]),\\ \mathsf{unif}_{\mathrm{Sh}}([S_2,U_2],\mathrm{mgu}(\rho(A_1)=A_2))),U_2 \cup \mathrm{vars}(A_2)), \quad (36)$$

where ρ is a renaming such that $\rho(U_1 \cup \mathrm{vars}(A_1)) \cap (U_2 \cup \mathrm{vars}(A_2)) = \emptyset$.

Example 7.3
Let $U_1 = \{u,v,w\}$, $U_2 = \{x,y,z\}$, $\Theta_1 = \{[\{v/t(u,w,w)\}]_{U_1},[\{v/t(u,u,w)\}]_{U_1}\}$, $\Theta_2 = \{[\{y/t(x,z,z)\}]_{U_2},[\{y/t(x,x,z)\}]_{U_2}\}$, and $\rho = id$. We have

$$\mathbf{U}'^b_{\mathrm{Ps}}([\Theta_1,U_1],[\Theta_2,U_2],p(u,v,w),p(x,y,z)) = \pi_{\mathrm{Ps}}([\Theta,U_1 \cup U_2],U_2),$$

with $[\theta]_{U_1\cup U_2} = [\{y/t(x,x,x),z/x,u/x,v/t(x,x,x),w/x\}]_{U_1\cup U_2} \in \Theta$. Let $[S_1,U_1] = \alpha_{\mathrm{Sh}}([\Theta_1,U_1])$, $[S_2,U_2] = \alpha_{\mathrm{Sh}}([\Theta_2,U_2])$, $S_1 = \{\mathtt{uv},\mathtt{vw}\}$, and $S_2 = \{\mathtt{xy},\mathtt{yz}\}$. We obtain

$$\mathbf{U}'^b_{\mathrm{Sh}}([S_1,U_1],[S_2,U_2],p(u,v,w),p(x,y,z)) = \pi_{\mathrm{Sh}}([S,U_1 \cup U_2],U_2)$$

and $\mathtt{xyzuvw} \in S$. So, it seems that u,v, and w may share a common variable. Note that θ is obtained by unifying $\sigma_2 = \{y/t(x,z,z)\}$ with $\sigma_1 = \{v/t(u,u,w)\}$ but $\sigma_1(v) = t(u,u,w)$ is not an instance of $\mathrm{mgu}(\sigma_2,\mathrm{mgu}(p(x,y,z) = p(u,v,w)))(v) = t(x,z,z)$. Therefore, σ_1 and σ_2 do pertain to different computational paths. Using the backward unification with matching, we obtain

$$\begin{aligned}&\mathbf{U}^b_{\mathrm{Ps}}([\Theta_1,U_1],[\Theta_2,U_2],p(u,v,w),p(x,y,z))\\ &\quad = \pi_{\mathrm{Ps}}([\{[y/t(x,z,z),u/x,v/t(x,z,z),w/z],[y/t(x,x,z),u/x,v/t(x,x,z),w/z]\},\\ &\qquad \{x,y,z,u,v,w\}],\{u,v,w\}),\end{aligned}$$

which does not contain θ. In the abstract domain, we have

$$\mathbf{U}^b_{\mathrm{Sh}}([S_1,U_1],[S_2,U_2],p(u,v,w),p(x,y,z)) = \pi_{\mathrm{Sh}}([\{\mathtt{xyuv},\mathtt{yzvw}\},U_1 \cup U_2],U_2).$$

After the unification we know that x and z are independent. Note that the abstract matching operators defined in Hans and Winkler (1992) and King and Longley (1995) cannot establish this property. The algorithm in Muthukumar and Hermenegildo (1992) computes the same result as ours in this particular example, but since their matching is partially performed by first projecting the sharing information on the term positions of the calling atom and of the clause head, this does not hold in general. For example, their algorithm states that x and z may possibly share when the unification is performed between the calling atom $p(t(x,y,z))$ and the head $p(t(u,v,w))$, where t is a function symbol, p a unary predicate, and the call substitution is the same as before. □

7.1 Correctness and optimality

We can prove that $\mathbf{U}^b_{\mathrm{Sh}}$ is actually the best correct abstraction of the backward concrete unification $\mathbf{U}^b_{\mathrm{Ps}}$. To prove correctness we only need to show that $\mathsf{match}_{\mathrm{Sh}}$

is correct w.r.t. $\mathsf{match}_{\mathrm{Ps}}$. Correctness of $\mathbf{U}^{b}_{\mathrm{Sh}}$ will follow from the fact that $\mathbf{U}^{b}_{\mathrm{Sh}}$ is a composition of correct abstract operators.

Theorem 7.4
(Correctness of $\mathsf{match}_{\mathrm{Sh}}$)
$\mathsf{match}_{\mathrm{Sh}}$ is correct w.r.t. $\mathsf{match}_{\mathrm{Ps}}$.

Proof
The proof can be found in Appendix D as Theorem D.1. □

However, the composition of optimal operators may fail to be optimal. Therefore, optimality of $\mathsf{match}_{\mathrm{Sh}}$ does not guarantee optimality of $\mathbf{U}^{b}_{\mathrm{Sh}}$. In order to prove the optimality result, we need to establish two additional properties on the abstract operators $\mathsf{match}_{\mathrm{Sh}}$ and $\mathsf{unif}_{\mathrm{Sh}}$. The idea is that both these operators are used in a very specific way in the backward unification.

Proposition 7.5

(1) $\mathsf{match}_{\mathrm{Sh}}$ is optimal w.r.t. $\mathsf{match}_{\mathrm{Ps}}$;

(2) when $\mathsf{match}_{\mathrm{Ps}}$ is restricted to the case in which the second argument contains a single substitution, $\mathsf{match}_{\mathrm{Sh}}$ is complete w.r.t. the second argument, i.e.,

$$\mathsf{match}_{\mathrm{Sh}}([S_1, U_1], \alpha_{\mathrm{Sh}}([\{[\sigma_2]\}, U_2])) = \alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2]));$$

(3) $\mathsf{unif}_{\mathrm{Sh}}$ is optimal in a very strong way: given $[S_1, U_1] \in \mathtt{Sharing}$ and $\theta \in \mathit{ISubst}$, there exists a substitution $\delta \in \mathit{ISubst}$ such that $\alpha_{\mathrm{Sh}}([\delta]_{U_1}) \sqsubseteq_{\mathrm{Sh}} [S_1, U_1]$ and

$$\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}([\{[\delta]\}, U_1], \theta)) = \mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta).$$

Proof
Proofs of these properties can be found in Appendix D as Theorems D.2, D.3, and D.4. □

On the last point, note that the standard definition of optimality for $\mathsf{unif}_{\mathrm{Sh}}$ only assures the existence of a set of substitutions Δ such that $\alpha_{\mathrm{Sh}}([\Delta, U_1]) \sqsubseteq_{\mathrm{Sh}} [S_1, U_1]$ and $\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}([\Delta, U_1], \theta)) = \mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta)$. However, we show that any set Δ can be reduced to a singleton. This allows us to find a single substitution to be used for proving the optimality result for all the resultant sharing groups. Finally, using Theorem 7.4 and Proposition 7.5 we may prove the expected result.

Theorem 7.6
$\mathbf{U}^{b}_{\mathrm{Sh}}$ is correct and optimal w.r.t. $\mathbf{U}^{b}_{\mathrm{Ps}}$.

Proof
The proof can be found in Appendix D as Theorem D.5. □

To the best of our knowledge, this is the first abstract matching operator that is optimal for the corresponding concrete operator. We now give an example of a program in which the use of $\mathbf{U}^{f}_{\mathrm{Sh}}$ and $\mathbf{U}^{b}_{\mathrm{Sh}}$ gives better results than the standard operators $\mathbf{U}'^{f}_{\mathrm{Sh}}$ and $\mathbf{U}'^{b}_{\mathrm{Sh}}$.

Example 7.7

We keep on Examples 4.4, 6.2, and 6.5 and consider the trivial program with just one clause $\mathtt{p(u,v,w)}$ and the goal $p(x,y,z)$ with $\{\mathtt{xy},\mathtt{yz}\}$. Using our abstract operators, we obtain the entry substitution $\{\mathtt{uv},\mathtt{vw}\}$ and the success substitution $\{\mathtt{xy},\mathtt{yz}\}$ (see Examples 6.5 and 7.3), thus proving that x and z are independent.

We now compute the abstract semantics of the goal $p(x,y,z)$ with $\{\mathtt{xy},\mathtt{yz}\}$. From Example 4.4, we have that the abstract semantics of P is

$$\lambda A.\lambda\chi.\mathbf{U}^b_{\mathrm{Sh}}(\mathbf{U}^f_{\mathrm{Sh}}(\chi,A,p(u,v,w)),\chi,p(u,v,w),A).$$

Thus, in order to compute the semantics of the goal $p(x,y,z)$ with $\{\mathtt{xy},\mathtt{yz}\}$, we need to compute

$$\begin{aligned}&\mathbf{U}^b_{\mathrm{Sh}}(\mathbf{U}^f_{\mathrm{Sh}}([\{\mathtt{xy},\mathtt{yz}\},\{x,y,x\}],p(x,y,z),p(u,v,w)),\\&\qquad\qquad\qquad\qquad[\{\mathtt{xy},\mathtt{yz}\},\{x,y,x\}],p(u,v,w),p(x,y,z)).\end{aligned}$$

From Example 6.5, we know that

$$\mathbf{U}^f_{\mathrm{Sh}}([\{\mathtt{xy},\mathtt{yz}\},\{x,y,x\}],p(x,y,z),p(u,v,w)) = [\{\mathtt{uv},\mathtt{vw}\},\{u,v,w\}],$$

from which we obtain (see Example 7.3)

$$\begin{aligned}&\mathbf{U}^b_{\mathrm{Sh}}([\{\mathtt{uv},\mathtt{vw}\},\{u,v,w\}],[\{\mathtt{xy},\mathtt{yz}\},\{x,y,x\}],p(u,v,w),p(x,y,z))\\&\qquad\qquad\qquad\qquad\qquad\qquad = [\{\mathtt{xy},\mathtt{yz}\},\{x,y,z\}],\end{aligned}$$

which shows that x and y are independent.

If we replace either $\mathbf{U}^b_{\mathrm{Sh}}$ or $\mathbf{U}^f_{\mathrm{Sh}}$ with $\mathbf{U}'^f_{\mathrm{Sh}}$ or $\mathbf{U}'^b_{\mathrm{Sh}}$, then the success substitution will contain the sharing group $\mathtt{xyz}$. In fact, as shown in Example 6.2, the entry substitution in the latter case would be $[\{\mathtt{uv},\mathtt{vw},\mathtt{uvw}\},\{u,v,w\}]$. If we compute the success substitution we obtain

$$\begin{aligned}&\mathbf{U}'^b_{\mathrm{Sh}}([\{\mathtt{uv},\mathtt{vw},\mathtt{uvw}\},\{u,v,w\}],[\{\mathtt{xy},\mathtt{yz}\},\{x,y,z\}],p(u,v,w),p(x,y,z)),\{x,y,z\})\\&\qquad\qquad\qquad\qquad\qquad\qquad = [\{\mathtt{xy},\mathtt{yz},\mathtt{xyz}\},\{x,y,z\}],\end{aligned}$$

which contains the sharing group $\mathtt{xyz}$. □

7.2 Programs in head normal form

It is worth noting that the improvement in the previous example is obtained with a program in *head normal form*. Usually, when programs are in head normal form, the forward and backward unifications may be replaced by renamings, which are complete and do not cause any loss in precision. However, there is the need of an unification operator for the explicit constraints that appear in the body of the clauses. In general, the analyses we obtain in our framework are more precise than those that can be obtained by using the standard domain `Sharing` by translating the same program to the head normal form.

Example 7.8
Consider again Example 7.7 and the program $\mathtt{p(u,f(s),w)} \leftarrow$ that is not in head normal form. Using our abstract operators, we obtain the success substitution $\{\mathtt{xy},\mathtt{yz}\}$, as in Example 7.7. If we normalize the program, we obtain the clause $\mathtt{p(u,v,w)} \leftarrow \mathtt{v = f(s)}$. The entry substitution obtained from $\{\mathtt{xy},\mathtt{yz}\}$ by simply renaming the variables x, y, z to u, v, w and introducing the new variable s is $\{\mathtt{uv},\mathtt{vw},\mathtt{s}\}$. By using the standard operator for unification, when applying the binding $v/f(s)$ we obtain $\{\mathtt{uvs},\mathtt{vws},\mathtt{uvws}\}$, and thus the success substitution will contain the sharing group xyz, resulting in a loss of precision. □

It is possible to use our forward abstract unification in a normalized program by enlarging the set of variables of interest only when new variables are effectively met, instead of adding all the variables that appear in the body of a clause once for all when the entry substitution is computed. In the example above, the variable s can be introduced when unifying the abstract object $\{\mathtt{uv},\mathtt{vw}\}$ with $v/f(s)$. Since $\mathsf{unif}_{\mathrm{Sh}}([\{\mathtt{uv},\mathtt{vw}\},\{u,v,w\}],\{v/f(s)\}) = [\{\mathtt{uvs},\mathtt{vws}\},\{u,v,w,s\}]$, we still obtain as success substitution $\{\mathtt{xy},\mathtt{yz}\}$, thus proving that x and z are independent.

In the general case, translating a program in head normal form will negatively affect the precision of the analysis. To achieve the same precision in both cases, we need to add structural information to the abstract domain (Le Charlier and Van Hentenryck 1994).

8 Related works

8.1 Relationship with ESubst

The domain *ESubst* proposed by Jacobs and Langen (1992) uses a nonstandard definition of substitution. We may prove that *ESubst* is isomorphic to $\mathit{ISubst}_{\sim}$. This formalizes the intuition, which has never been proved before, that working with *ESubst* is essentially like working with substitutions. Similar proofs may be developed for ex-equations (Marriott *et al.* 1994) and existential Herbrand constraints (Levi and Spoto 2003).

We now briefly recall the definition of the domain *ESubst*. For the sake of clarity, in the following, we call E-substitution the nonstandard substitution defined in Jacobs and Langen (1992). An E-substitution σ is a mapping from a finite set of variables $\mathrm{dom}(\sigma) \subseteq \mathscr{V}$ to Terms. This approach differs from the standard definition of substitutions, which are mappings from $\mathscr{V}$ to Terms that are almost everywhere the identity. The preorder on E-substitutions is defined as follows:

$$\sigma \leqslant_E \theta \iff \mathrm{dom}(\theta) \subseteq \mathrm{dom}(\sigma) \wedge \big(\forall t \in \mathsf{Terms}.\ \mathrm{vars}(t) \subseteq \mathrm{dom}(\theta) \Rightarrow \\ \exists \delta \text{ an E-substitution s.t. } \sigma t = \delta(\theta(t))\big), \quad (37)$$

where the application of an E-substitution to a term is defined as usual.

Let $\sim_E$ be the equivalence relation on E-substitutions induced by $\leqslant_E$. The domain *ESubst* is defined as the set of equivalence classes of E-substitutions w.r.t. $\sim_E$, that is, $\mathit{ESubst} = \{[\sigma]_{\sim_E} \mid \sigma \text{ is an E-substitution}\}$. The next theorem shows that *ESubst* is isomorphic to $\mathit{Subst}_{\sim}$ that, as shown in Proposition 3.7, is isomorphic to $\mathit{ISubst}_{\sim}$.

Theorem 8.1
ESubst and $\mathit{Subst}_{\sim}$ are isomorphic posets.

Proof
To each E-substitution θ we may associate a substitution θ' such that $\theta'(x) = \theta(x)$ if $x \in \mathrm{dom}(\theta)$ and $\theta'(x) = x$ otherwise. Note that for each term t, $\theta(t) = \theta'(t)$: an E-substitution and the corresponding standard substitution behave in the same way on terms.

We may prove that if $\theta_1 \leqslant_E \theta_2$, then $\theta'_1 \preceq_{\mathrm{dom}(\theta_2)} \theta'_2$. By definition, if $\theta_1 \leqslant_E \theta_2$, then $\mathrm{dom}(\theta_2) \subseteq \mathrm{dom}(\theta_1)$ and $\forall t \in \mathsf{Terms}$ with $\mathrm{vars}(t) \subseteq \mathrm{dom}(\theta_2)$, and there exists an E-substitution δ such that $\theta_1(t) = \delta(\theta_2(t))$. Let $\mathrm{dom}(\theta_2) = \{x_1, \ldots, x_n\}$ and consider a term t such that $\mathrm{vars}(t) = \{x_1, \ldots, x_n\}$. (Note that t exists iff there is at least a term symbol of arity strictly greater than 1.) By definition, there exists an E-substitution δ such that $\theta_1(t) = \delta(\theta_2(t))$; that is, for any $v \in \mathrm{dom}(\theta_2)$, $\theta_1(v) = \delta(\theta_2(v))$ holds. This means that $\theta'_1(v) = \delta'(\theta'_2(v))$ and therefore $\theta'_1 \preceq_{\mathrm{dom}(\theta_2)} \theta'_2$.

On the converse, for each $\theta \in \mathit{Subst}$ and $U \in \wp_f(\mathcal{V})$, we associate a corresponding E-substitution θ^{*U} such that $\mathrm{dom}(\theta^{*U}) = U$ and $\theta^{*U}(v) = \theta(v)$ for each $v \in U$. As for the previous case, we have that if $\theta_1 \preceq_U \theta_2$, then $\theta_1^{*U} \leqslant_E \theta_2^{*U}$. First of all, note that $\mathrm{dom}(\theta_1^{*U}) = U = \mathrm{dom}(\theta_2^{*U})$. Moreover, by the definition of $\preceq_U$, there is $\delta \in \mathit{Subst}$ such that $\theta_1(v) = \delta(\theta_2(v))$ for each $v \in U$. Now, given a term t such that $\mathrm{vars}(t) \subseteq U$, we may check that $\theta_1^{*U}(t) = \delta^{*\mathrm{vars}(\theta_2(U))}(\theta_2^{*U}(t))$, and this proves $\theta_1^{*U} \leqslant_E \theta_2^{*U}$.

Now, we may lift these operations to equivalence classes to obtain the function $\imath : \mathit{ESubst} \rightarrow \mathit{Subst}_{\sim}$ such that

$$\imath([\theta]_{\sim_E}) = [\theta']_{\mathrm{dom}(\theta)}.$$

The map $\imath$ is well defined: if $\theta_1 \sim_E \theta_2$, then $\mathrm{dom}(\theta_1) = \mathrm{dom}(\theta_2)$ and, by the above property, $\theta'_1 \sim_{\mathrm{dom}(\theta_2)} \theta'_2$. Moreover, there is an inverse $\imath^{-1}$ given by

$$\imath^{-1}([\theta]_U) = [\theta^{*U}]_{\sim_E}.$$

It is easy to check that $\imath^{-1}$ is well defined: if $\theta_1 \preceq_U \theta_2$, then $\theta_1^{*U} \leqslant_E \theta_2^{*U}$.

It is immediate to check, given the properties above, that $\imath$ and $\imath^{-1}$ are the inverse of each other. Moreover, they are both monotonic. If $[\theta_1]_E \leqslant_E [\theta_2]_E$, then $\mathrm{dom}(\theta_2) \subseteq \mathrm{dom}(\theta_1)$ and $\theta'_1 \preceq_{\mathrm{dom}(\theta_2)} \theta'_2$, i.e., $\imath([\theta_1]_{\sim_E}) = [\theta'_1]_{\mathrm{dom}(\theta_1)} \preceq [\theta'_2]_{\mathrm{dom}(\theta_2)} = \imath([\theta_2]_{\sim_E})$. On the converse, if $[\theta_1]_U \preceq [\theta_2]_V$, then $[\theta_1]_V \preceq [\theta_2]_V$ and therefore $\imath^{-1}([\theta_1]_V) \leqslant_E \imath^{-1}([\theta_2]_V)$. We only need to prove that $\imath^{-1}([\theta_1]_U) \leqslant_E \imath^{-1}([\theta_1])_V$. This follows from that fact that given a term t with $\mathrm{vars}(t) \subseteq V$, $\theta_1^{*U}(t) = \theta_1^{*V}(t)$. □

It is worth noting that the most general unifier as defined in Jacobs and Langen (1992) corresponds to mgu in $\mathit{ISubst}_{\sim}$. In formulas, given term t_1 and t_2, we have that

$$\imath([\mathrm{mgu}(t_1, t_2)]_{\sim_E}) = [\mathrm{mgu}(\{t_1 = t_2\})]_{\mathrm{vars}(t_1 = t_2)}, \tag{38}$$

where mgu on the left-hand side is the operator in Definition 1 of Jacobs and Langen (1992) and $\imath : \mathit{ESubst} \rightarrow \mathit{ISubst}_{\sim}$ is the isomorphism defined in the proof of Theorem 8.1. To the best of our knowledge, this is the first proof of the relationship between the mgu in a domain of existential substitutions and the standard mgu for substitutions. Moreover, it is worth noting that by adding a bottom element to $ISubst_{\sim}$ and *ESubst*, they turn out to be isomorphic complete lattices.

8.2 A case study

In Section 3 we said that in order to define a good collecting semantics for correct answer substitutions, there are several possible directions. We may work with a domain of existentially quantified substitutions like $ISubst_{\sim}$, or we may work with standard substitutions, being careful to keep enough representatives for each equivalence class. We have already discussed the benefits of using equivalence classes. In order to show the kind of problems that arise from the use of domains of substitutions, without any equivalence relation, we want to show a small flaw of the semantic framework defined in Cortesi and Filé (1999) for the analysis of sharing and widely used in several other works on program analysis, such as Bagnara *et al.* (2002) and Hill *et al.* (2004).

The framework is based upon the domain $\mathtt{Rsub} = (\wp(Subst) \times \wp_f(\mathcal{V})) \cup \{\top_{\mathrm{Rs}}, \bot_{\mathrm{Rs}}\}$ that is a complete lattice, partially ordered as follows: $\top_{\mathrm{Rs}}$ is the top element; $\bot_{\mathrm{Rs}}$ is the bottom element; and $[\Theta_1, U_1] \sqsubseteq_{\mathrm{Rs}} [\Theta_2, U_2]$ if and only if $U_1 = U_2$ and $\Theta_1 \subseteq \Theta_2$. An object $[\Theta, U]$ is a set of substitution Θ, where the set of variables of interest U is explicitly provided.

The main operation in Rsub is the concrete unification $\mathbf{U}_{\mathrm{Rs}} : \mathtt{Rsub} \times \mathtt{Rsub} \times ISubst \to \mathtt{Rsub}$ such that

$$\left.\begin{aligned} \mathbf{U}_{\mathrm{Rs}}(\bot_{\mathrm{Rs}}, \xi, \delta) &= \mathbf{U}_{\mathrm{Rs}}(\xi, \bot_{\mathrm{Rs}}, \delta) = \bot_{\mathrm{Rs}}, \\ \mathbf{U}_{\mathrm{Rs}}(\xi, \top_{\mathrm{Rs}}, \delta) &= \mathbf{U}_{\mathrm{Rs}}(\top_{\mathrm{Rs}}, \xi, \delta) = \top_{\mathrm{Rs}} \qquad \text{if } \xi \neq \bot_{\mathrm{Rs}}, \\ \mathbf{U}_{\mathrm{Rs}}([\Theta_1, U_1], [\Theta_2, U_2], \delta) &= [\{\mathrm{mgu}(\sigma_1, \sigma_2, \delta) \mid \sigma_1 \in \Theta_1, \sigma_2 \in \Theta_2, \\ &\qquad \mathrm{vars}(\sigma_1) \cap \mathrm{vars}(\sigma_2) = \emptyset\}, U_1 \cup U_2]. \end{aligned}\right\} \tag{39}$$

Although it is well defined for all the values of the domain, $\mathbf{U}_{\mathrm{Rs}}([\Theta_1, U_1], [\Theta_2, U_2], \delta)$ may be restricted to those values according to which $U_1 \cap U_2 = \emptyset$ and $\mathrm{vars}(\delta) \subseteq U_1 \cup U_2$, since this is the only way $\mathbf{U}_{\mathrm{Rs}}$ is used in the semantics defined in Cortesi and Filé (1999).

The abstract domain is the same as the Sharing we use in our paper, with abstraction map $\alpha_{\mathrm{Sh}} : \mathtt{Rsub} \to \mathtt{Sharing}$ and unification $\mathbf{U}_{\mathrm{Sh}} : \mathtt{Sharing} \times \mathtt{Sharing} \times ISubst \to \mathtt{Sharing}$ defined by

$$\alpha_{\mathrm{Sh}}([\Theta, U]) = \bigsqcup\nolimits_{\mathrm{Sh}} \{\alpha_{\mathrm{Sh}}([\sigma]_U) \mid \sigma \in \Theta\}, \tag{40}$$

$$\mathbf{U}_{\mathrm{Sh}}([\Theta_1, U_1], [\Theta_2, U_2], \delta) = \mathsf{unif}_{\mathrm{Sh}}([\Theta_1 \cup \Theta_2, U_1 \cup U_2], \delta). \tag{41}$$

The domain of $\mathbf{U}_{\mathrm{Sh}}$ is restricted to the case $U_1 \cap U_2 = \emptyset$ and $\mathrm{vars}(\delta) \subseteq U_1 \cup U_2$.

By looking at the paper, we think that, in the idea of the authors, $[\Theta, U] \in \mathtt{Rsub}$ should have been treated as $[\{[\sigma]_U \mid \sigma \in \Theta\}, U] \in \mathtt{Psub}$ is in our framework. However, the condition $\mathrm{vars}(\sigma_1) \cap \mathrm{vars}(\sigma_2) = \emptyset$, introduced in $\mathbf{U}_{\mathrm{Rs}}$ in order to avoid variable clashes between the two chosen substitutions, is not enough for this purpose. Actually, $\mathbf{U}_{\mathrm{Rs}}$ only checks that σ_1 and σ_2 do not have variables in common, without considering their sets of variables of reference U_1 and U_2. This unification can lead to counterintuitive results.

Example 8.2
Consider the following concrete unification:

$$\mathbf{U}_{\mathrm{Rs}}([\{\{x/y\}\},\{x\}],[\{\epsilon\},\{y\}],\epsilon)=[\{\{x/y\}\},\{x,y\}]. \tag{42}$$

Being $\mathrm{vars}(\epsilon)=\emptyset$, the concrete unification operator allows us to unify $\{x/y\}$ with ϵ without renaming the variable y, which is not a variable of interest in the first element but is treated as if it were. This also causes the incorrectness of $\mathbf{U}_{\mathrm{Sh}}$. If we consider equation (42) and compute the result on the abstract side by using the abstract unification operator $\mathbf{U}_{\mathrm{Sh}}$, we have

$$\begin{array}{llll}\mathbf{U}_{\mathrm{Sh}}(& \alpha_{\mathrm{Sh}}([\{\{x/y\}\},\{x\}]), & \alpha_{\mathrm{Sh}}([\{\epsilon\},\{y\}]), & \epsilon) \\ =\mathbf{U}_{\mathrm{Sh}}(& [\{\mathrm{x}\},\{x\}], & [\{\mathrm{y}\},\{y\}], & \epsilon)=[\{\mathrm{x},\mathrm{y}\},\{x,y\}].\end{array}$$

This is not a correct approximation of the concrete result, since

$$\alpha_{\mathrm{Sh}}([\{\{x/y\}\},\{x,y\}])=[\{\mathrm{xy}\},\{x,y\}]\not\sqsubseteq_{\mathrm{Sh}}[\{\mathrm{x},\mathrm{y}\},\{x,y\}]. \quad\square$$

This counterexample proves that the abstract unification operator $\mathbf{U}_{\mathrm{Sh}}$ is not correct w.r.t. the concrete one $\mathbf{U}_{\mathrm{Rs}}$, invalidating Theorem 6.3 in Cortesi and Filé (1999). The problem can be solved by introducing a stronger check on variable clashes, namely, by replacing the condition $\mathrm{vars}(\sigma_1)\cap\mathrm{vars}(\sigma_2)=\emptyset$ with $(\mathrm{vars}(\sigma_1)\cup U_1)\cap(\mathrm{vars}(\sigma_2)\cup U_2)=\emptyset$ in the definition of $\mathbf{U}_{\mathrm{Rs}}$, thus obtaining the following operator:

$$\begin{aligned}\mathbf{U}^*_{\mathrm{Rs}}([\Theta_1,U_1],[\Theta_2,U_2],\delta)=[\{\mathrm{mgu}(\sigma_1,\sigma_2,\delta)\mid \sigma_1\in\Theta_1,\sigma_2\in\Theta_2,\\ (\mathrm{vars}(\sigma_1)\cup U_1)\cap(\mathrm{vars}(\sigma_2)\cup U_2)=\emptyset\},U_1\cup U_2].\end{aligned} \tag{43}$$

By using $\mathbf{U}^*_{\mathrm{Rs}}$ instead of $\mathbf{U}_{\mathrm{Rs}}$, the proof of Theorem 6.3 in Cortesi and Filé (1999) becomes valid.

Theorem 8.3
$\mathbf{U}_{\mathrm{Sh}}$ is correct w.r.t. $\mathbf{U}^*_{\mathrm{Rs}}$.

Proof
If we look at the proof of Theorem 6.3 in Cortesi and Filé (1999), it appears that the problem is in the base case of the inductive argument, when $i=0$. Here, it is stated that given $[A_1,U_1]$ and $[A_2,U_2]$ in `Sharing` with $U_1\cap U_2=\emptyset$, $\sigma_i\in\gamma_{\mathrm{Sh}}([A_i,U_i])$ for $i\in\{1,2\}$ with $\mathrm{vars}(\sigma_1)\cap\mathrm{vars}(\sigma_2)=\emptyset$, it holds that $[\{\rho_0\},U_0]\sqsubseteq_{\mathrm{Rs}}\gamma_{\mathrm{Sh}}([R_0,U_0])$, where $\rho_0=\sigma_1\uplus\sigma_2$, $U_0=U_1\cup U_2$, and $R_0=A_1\cup A_2$. However, the substitutions $\sigma_1=\{x/y\}\in\gamma_{\mathrm{Sh}}([\{\mathrm{x}\},\{x\}])$ and $\sigma_2=\epsilon\in\gamma_{\mathrm{Sh}}([\{\mathrm{y}\},\{y\}])$ of the previous example make the statement false. On the contrary, when $\mathbf{U}^*_{\mathrm{Rs}}$ is used instead of $\mathbf{U}_{\mathrm{Rs}}$, σ_1 and σ_2 are required to satisfy the condition $(\mathrm{vars}(\sigma_1)\cup U_1)\cap(\mathrm{vars}(\sigma_2)\cup U_2)=\emptyset$. From this, it truly follows that $[\{\rho_0\},U_0]=[\{\sigma_1\uplus\sigma_2\},U_0]\sqsubseteq_{\mathrm{Rs}}\gamma_{\mathrm{Sh}}([R_0,U_0])$. The inductive case for $i>0$ is identical to that in Cortesi and Filé (1999), since for any $A,B\in$ `Rsub` and $\delta\in \mathit{ISubst}$ it holds that $\mathbf{U}^*_{\mathrm{Rs}}(A,B,\delta)\sqsubseteq_{\mathrm{Rs}}\mathbf{U}_{\mathrm{Rs}}(A,B,\delta)$. $\square$

Observer that in order to define a real semantics for logic programs, a renaming operation should be introduced in the framework of Cortesi and Filé (1999). This can be done along the lines of Cortesi *et al.* (1994). Due to the kind of renamings involved, by replacing $\mathbf{U}_{\mathrm{Rs}}$ with $\mathbf{U}^{*}_{\mathrm{Rs}}$, the semantics in Cortesi *et al.* (1994) does not change. Therefore this flaw does not affect the real analysis of logic programs.

8.3 Other related works

8.3.1 Backward unification

The idea of using a refined operator for computing answer substitutions is not new and may be traced back to the frameworks in Bruynooghe (1991) and Le Charlier and Van Hentenryck (1994). The abstract domains considered in these papers contain structural information, freeness, groundness, and pair-sharing but no set-sharing information. Working within these frameworks, Hans and Winkler (1992) and King and Longley (1995) proposed correct abstract operators w.r.t. matching for the domain SFL. Muthukumar and Hermenegildo (1991, 1992) used a refined algorithm for backward unification in Sharing, although it is not presented in algebraic form. However, to the best of our knowledge, this is the first paper that formally introduces matching from the point of view of a collecting denotational semantics, deriving the abstract operator from the concrete one and proving correctness and optimality. Moreover, this is the first paper that presents optimal abstract matching for a domain for set-sharing analysis (see Example 7.3).

8.3.2 Forward/backward unification and PSD

Although the usual goal of sharing analyses is to discover the pairs of variables that may possibly share, Sharing is a domain that keeps track of set-sharing information. Bagnara *et al.* (2002) proposed a new domain, called PSD, which is the complete shell (Giacobazzi *et al.* 2000) of pair-sharing w.r.t. Sharing. They recognized that in an abstract object $[S, U]$, some sharing groups in S may be *redundant* as far as pair-sharing is concerned. Although our forward unification is more precise than the standard unification, it could be the case that they have the same precision in PSD. This would mean that $\mathbf{U}^{f}_{\mathrm{Sh}}([S_1, U_1], A_1, A_2)$ and $\mathbf{U}'^{f}_{\mathrm{Sh}}([S_1, U_1], A_1, A_2)$ only differ for redundant sharing groups. However, this is not the case, and Examples 6.2, 6.3, and 6.19 show improvements that are still significant in PSD. The same holds for backward unification in Example 7.3. It is not clear whether PSD is still complete w.r.t. pair-sharing when our specialized operators are used.

8.3.3 Domains with freeness and linearity

Although the use of freeness and linearity information has been pursued in several papers (e.g., Muthukumar and Hermenegildo 1991; Hans and Winkler 1992), optimal operators for these domains have never been developed. All the abstract unification operators for SFL (e.g., Hans and Winkler 1992; Muthukumar and Hermenegildo 1992; Hill *et al.* 2004) when unifying with a binding $\{x/t\}$, where neither x nor t

are linear, does compute all the star unions. On the contrary, in $\mathbf{u}^f_{\mathrm{Sh}}$ we apply an optimization that is able to avoid some sharing groups (see, e.g., Example 6.19). This optimization could be integrated in a domain that explicitly contains freeness and linearity information.

Actually, Hill *et al.* (2004) included some optimizations for the standard abstract unification of SFL that are similar to ours, in the case of a binding $\{x/t\}$ with x linear. In addition, in Howe and King (2003) and Hill *et al.* (2004) the authors proposed to remove the check for independence between x and t. We think it should be possible to devise an optimal abstract unification for an enhanced domain including linearity information, by combining these improvements with our results.

A first optimality result is shown in Amato and Scozzari (2003), which is based on a preliminary version of the framework we present here. The authors considered two domains for set-sharing and linearity (without freeness), namely, the standard reduced product of Sharing and linearity, and the domain proposed by King (1994). The paper presents the abstract operators for forward unification, which turn out to be optimal in the case of a single-binding substitution. These are the only operators in the literature that are strictly more precise than our optimized forward unification operator for Sharing.

8.3.4 Another optimality proof

Codish *et al.* (2000) provided an alternative approach to the analysis of sharing by using set logic programs and ACI1 unification. They defined abstract operators that are proved to be correct and optimal and examine the relationship between set substitutions and Sharing, proving that they are essentially isomorphic. However, they do not extend this correspondence to the abstract operators, so that a proof of optimality of $\mathbf{U}^f_{\mathrm{Sh}}$ w.r.t. $\mathbf{U}^f_{\mathrm{Ps}}$ starting from their results should be feasible but it is not immediate. Moreover, since they provide a goal-independent analysis, they do not have different operators for forward and backward unification.

9 Conclusions

We think that there are three major contributions in this paper.

- We integrate the framework of Cortesi *et al.* (1996) with several different proposals that have appeared in the literature for goal-dependent analysis of logic programs. We give formal proofs of the correctness of the resulting analysis and of optimality of the abstract operators. The aim is to clarify the relationships between these proposals and to provide a clear guidance for the development of static analysis for logic programs.
- We introduce a new concrete domain of equivalence classes of substitutions that address the problem of variable clashes by taking into account sets of variables of interest. This problem has been considered by many authors, but in our opinion, none of them fully developed a corresponding theory of substitutions, in the style of Palamidessi (1990).

- Our definition of abstract forward unification sheds new light on the role of freeness and linearity information, suggesting new optimizations that can also be used in more powerful domains such as `SFL`.

Although sharing analysis with more complex domains, including freeness and linearity information, will likely be more precise than the analysis performed with `Sharing` in our optimized framework, we think that this paper may be a guideline for developing new analysis for logic programs. The main ideas contained in this paper are not tied to the abstract domain in use. The framework we propose may be instantiated with more precise abstract domains to further improve the result of the abstract analysis. Moreover, the algorithm for the abstract forward unification can be easily slotted into other analysis frameworks based on different concrete semantics, including goal-independent ones.

To the best of our knowledge, this is the first work that optimizes the abstract forward unification for sharing analysis using freeness and linearity information implicitly, i.e., without using a domain that contains such information.

This is also the first work in which an abstract backward unification operator using matching is proved to be optimal. We have shown that to the best of our knowledge, all the abstract backward unification operators proposed so far for `Sharing` or more powerful domains (Hans and Winkler 1992; Muthukumar and Hermenegildo 1992; King and Longley 1995) were not optimal.

As a future work, we think that our results could be easily generalized for designing optimal unification operators for more complex domains possibly including linearity, freeness, and structural information. Preliminary results have appeared in Amato and Scozzari (2003). Moreover, the problem of efficiently implementing the refined backward unification could be addressed.

Appendix A: Correctness of the goal-dependent collecting semantics

In this appendix we provide a tedious proof that the collecting semantics we define is correct w.r.t. computed answers. We begin by formally introducing a notation for SLD-derivations, following Lloyd (1987) and Apt (1990). Given a goal $G = g_1 \dots g_k$ and a clause $cl = H \leftarrow B$ such that $\mathrm{vars}(G) \cap \mathrm{vars}(cl) = \emptyset$, we write

$$G \underset{\sigma}{\xrightarrow{cl}} (g_1 \dots g_{i-1} B g_{i+1} \dots g_k)\sigma \tag{A 1}$$

when $\sigma = \mathrm{mgu}(g_i, H)$. Given a goal G and a program P, an *SLD-derivation* of G in P is given by a sequence of clauses $cl_1, \dots, cl_n$ and idempotent substitutions $\sigma_1, \dots, \sigma_n$, such that

$$G \underset{\sigma_1}{\xrightarrow{cl_1}} G_1 \underset{\sigma_2}{\xrightarrow{cl_2}} \dots \underset{\sigma_n}{\xrightarrow{cl_n}} G_n, \tag{A 2}$$

where each cl_i is the renaming of a clause in P apart from $G, cl_1, \dots, cl_{i-1}$. The goal G_n is called the *end-goal*; n is the length of the derivation; and $(\sigma_n \circ \sigma_{n-1} \circ \dots \circ \sigma_2 \circ \sigma_1)_{|\mathrm{vars}(G)}$ is the (partial) computed answer. An SLD-refutation is an SLD-derivation with the empty end-goal (denoted by $\Box$). A *leftmost* SLD-derivation is an SLD-derivation in which we always rewrite the leftmost atom in the goal (i.e., such that $i = 1$ at every step in (A 1)).

We write $G \overset{*}{\underset{\sigma}{\rightarrow}} G'$ to denote an SLD-derivation with end-goal G' and partial computed answer σ. We also write $G \overset{\leqslant i}{\underset{\sigma}{\longrightarrow}} G'$ to denote an SLD-derivation with end-goal G', partially computed answer σ, and length less than or equal to i. A substitution σ is a *computed answer* for G in P if there is an SLD-refutation $G \overset{*}{\underset{\sigma}{\rightarrow}} \square$.

In this appendix we will prove the relationship between the set of computed answers for P and its collecting semantics $\mathscr{P}[\![P]\!]$.

A.1 Relevant denotations

We have defined a denotation as a continuous map in $\mathsf{Atoms} \rightarrow \mathsf{Psub} \xrightarrow{c} \mathsf{Psub}$. We now want to characterize the denotations that may arise as the results of our collecting semantics.

Definition A.1

A denotation $d \in \mathscr{D}en$ is said to be *relevant* when

- d is strict, i.e., $dA\bot_{\mathrm{Ps}} = \bot_{\mathrm{Ps}}$;
- $dA[\Delta, V]$ is either $\bot_{\mathrm{Ps}}$ or $[\Delta', V \cup \mathrm{vars}(A)]$ for some Δ'.

Note that the least denotation $\lambda A.\lambda[\Delta, V].\bot_{\mathrm{Ps}}$ is relevant. A relevant denotation is well behaved, in the sense that it either does not say anything or gives information for all and only the variables that occur in the atom A and the entry substitution $[\Delta, V]$.

Proposition A.2

If d is relevant, then

(1) $\mathscr{B}[\![B]\!]d\bot_{\mathrm{Ps}} = \bot_{\mathrm{Ps}}$;
(2) $\mathscr{B}[\![B]\!]d[\Delta, V]$ is either $\bot_{\mathrm{Ps}}$ or $[\Delta', V \cup \mathrm{vars}(B)]$ for some Δ';
(3) $\mathscr{C}[\![H \leftarrow B]\!]d$ is relevant;
(4) $\mathscr{P}[\![P]\!]$ is relevant.

Proof

The first two points easily follow by induction on the structure of the body B. For the third point, consider the definition of $\mathscr{C}$. Note that

$$\mathbf{U}^f_{\mathrm{Ps}}(x, A, H) = \pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho(x), \mathrm{mgu}(\rho(A) = H)), \mathrm{vars}(H)).$$

Since $\mathrm{vars}(\rho(A))$ is disjoint from H by the definition of ρ and since we consider relevant mgus, either $\mathrm{vars}(\mathrm{mgu}(\rho(A) = H)) = \mathrm{vars}(\rho(A)) \cup \mathrm{vars}(H)$ or $\mathrm{mgu}(\rho(A) = H) = \bot$. In the latter case, $\mathscr{C}[\![H \leftarrow B]\!]dA = \bot_{\mathrm{Ps}}$; otherwise $\mathbf{U}^f_{\mathrm{Ps}}(x, A, H) = [\Delta', \mathrm{vars}(H)]$ for some Δ'. By the previous point, we have that $\mathscr{B}[\![B]\!]d(\mathbf{U}^f_{\mathrm{Ps}}(x, A, H))$ is either $\bot_{\mathrm{Ps}}$ or $[\Delta'', \mathrm{vars}(H) \cup \mathrm{vars}(B)]$ for some Δ''. In the first case, $\mathscr{C}[\![H \leftarrow B]\!]dA = \bot_{\mathrm{Ps}}$; otherwise, assuming $x = [\Theta, V]$, we have

$$\begin{aligned}\mathscr{C}[\![H \leftarrow B]\!]dAx &= \mathbf{U}^b_{\mathrm{Ps}}([\Delta'', \mathrm{vars}(H) \cup \mathrm{vars}(B)], x, H, A)\\ &= \pi_{\mathrm{Ps}}\Big(\mathsf{match}_{\mathrm{Ps}}(\rho([\Delta'', \mathrm{vars}(H) \cup \mathrm{vars}(B)]),\\ &\qquad \mathsf{unif}_{\mathrm{Ps}}([\Theta, V], \mathrm{mgu}(\rho(H) = A))), V \cup \mathrm{vars}(A)\Big).\end{aligned}$$

For the same reason as explained above and since we can ignore the case in which $\rho(H)$ and A do not unify, we have that $\mathsf{unif}_{\mathrm{Ps}}([\Theta, V], \mathrm{mgu}(\rho(H) = A)) = [\Theta', V \cup \mathrm{vars}(A)]$ and therefore

$$\begin{aligned}&\pi_{\mathrm{Ps}}(\mathsf{match}_{\mathrm{Ps}}(\rho([\Delta'', \mathrm{vars}(H) \cup \mathrm{vars}(B)]), [\Theta', V \cup \mathrm{vars}(A)]), V \cup \mathrm{vars}(A))\\ &\qquad = [\Theta'', V \cup \mathrm{vars}(A)],\end{aligned}$$

which is what we wanted to prove.

The fourth point follows by the fact that given the proof of the third point, $\mathscr{C}[\![cl]\!]d$ is relevant for each clause cl and that least upper bounds of relevant denotations are easily seen to be relevant. □

A.2 Unused variables

Definition A.3
Given $[\phi]_V \in \mathit{ISubst}_\sim$ and $x \in \mathcal{V}$, we say that x is *unused* in $[\phi]_V$ when $[\phi]_V = \mathrm{mgu}\big(\pi_{V \setminus \{x\}}([\phi]_V), [\epsilon]_{\{x\}}\big)$.

First of all, note that this definition does not depend on the choice of representatives. If a variable x is unused in $[\phi]_V$, it means that $[\phi]_V$ does not constraint in any way its value. In other words, x is free and independent from all the other variables in V. This is made clear by the following characterization.

Proposition A.4
The variable $x \in V$ is unused in $[\phi]_V$ iff it is free and independent in $[\phi]_V$.

Proof
If x is free and independent in $[\phi]_V$, we may assume without loss of generality that $x \notin \mathrm{vars}(\phi)$. Let $V' = V \setminus \{x\}$. We have that

$$\mathrm{mgu}(\pi_{V'}([\phi]_V), [\epsilon]_{\{x\}}) = \mathrm{mgu}([\phi]_{V'}, [\epsilon]_{\{x\}}) = [\phi_{|V'}]_V = [\phi]_V,$$

which proves that x is unused. On the other hand, assume ϕ is a canonical representative and $\mathrm{mgu}([\phi]_{V'}, [\epsilon]_{\{x\}}) = [\phi]_V$. Then $\phi_{|V'} \sim_V \phi$. It is obvious that x is free and independent in $[\phi_{|V'}]_V = [\phi]_V$, since $x \notin \mathrm{dom}(\phi_{|V'})$ and $x \notin \mathrm{rng}(\phi)$. □

A.3 ISubst$_\sim$ and composition

The operations described in Section 3.2 are those required to provide a collecting semantics for logic programs over the domain *ISubst*$_\sim$. Note that we do not define any notion of composition, although it plays a central role with the standard substitutions. Actually, composition cannot be defined in our framework, since given any element of *ISubst*$_\sim$, variables not of interest are considered up to renaming only and therefore cannot be named. Nonetheless, in order to prove the equivalence between the standard semantics based on SLD-resolution and our collecting semantics, we will need to relate the composition of substitutions with unification in *ISubst*$_\sim$.

Lemma A.5

(Composition lemma)

Let $\sigma_1, \sigma_2, \sigma_3 \in \mathit{Subst}$, $U, V \in \wp_f(\mathscr{V})$. Then it holds that

$$\mathrm{mgu}([\sigma_3 \circ \sigma_2]_U, [\sigma_2 \circ \sigma_1]_V) = [\sigma_3 \circ \sigma_2 \circ \sigma_1]_{U \cup V}$$

provided that

- $\mathrm{dom}(\sigma_1) \cap U = \emptyset$;
- if $y \in \sigma_2(\sigma_1(V)) \setminus \sigma_2(\sigma_1(U \cap V))$, then $y \notin \mathrm{dom}(\sigma_3) \cup \sigma_3(\sigma_2(U))$.

Proof

Let $\theta \in [\sigma_3 \circ \sigma_2]_U, \eta \in [\sigma_2 \circ \sigma_1]_V$ be canonical representatives such that $(\mathrm{vars}(\theta) \cup U) \cap (\mathrm{vars}(\eta) \cup V) \subseteq U \cap V$. By definition, there exist $\rho, \rho' \in \mathit{Ren}$ such that $\theta = (\rho' \circ \sigma_3 \circ \sigma_2)_{|U}$ and $\eta = (\rho \circ \sigma_2 \circ \sigma_1)_{|V}$.

Then $\mathrm{mgu}([\sigma_3 \circ \sigma_2]_U, [\sigma_2 \circ \sigma_1]_V) = [\mathrm{mgu}(\theta, \eta)]_{U \cup V}$. It holds that $\mathrm{mgu}(\theta, \eta) = \mathrm{mgu}(\eta(\mathrm{Eq}(\theta))) \circ \eta$. It follows that $\eta(\mathrm{Eq}(\theta)) = \{\eta(x) = \eta(\theta(x)) \mid x \in U\} = \{\eta(x) = \theta(x) \mid x \in U\}$, since θ is a canonical representative. If $x \in U \cap V$, then $\eta(x) = \theta(x)$ becomes $\rho \circ \sigma_2 \circ \sigma_1(x) = \rho' \circ \sigma_3 \circ \sigma_2(x)$, that is, $\rho \circ \sigma_2(x) = \rho' \circ \sigma_3 \circ \sigma_2(x)$, since $\mathrm{dom}(\sigma_1) \cap U = \emptyset$ by hypothesis. Thus $\{\eta(x) = \theta(x) \mid x \in U \cap V\}$ and $\{\rho(y) = \rho' \circ \sigma_3(y) \mid y \in \sigma_2(U \cap V)\}$ have the same set of solutions. If $x \notin V$, then $\{\eta(x) = \theta(x) \mid x \in U \setminus V\} = \{x = \theta(x) \mid x \in U \setminus V\}$.

Now $\delta = \{\rho(y)/\rho' \circ \sigma_3(y) \mid y \in \sigma_2(U \cap V)\} \cup \{x/\theta(x) \mid x \in U \setminus V\}$ is an idempotent substitution. Actually, all the $\rho(y)$'s are distinct variables and different from $U \setminus V$; therefore δ is a substitution. Moreover, $\mathrm{dom}(\delta) \subseteq \mathrm{vars}(\eta(V)) \cup (U \setminus V)$ is disjoint from $\mathrm{rng}(\delta) = \mathrm{vars}(\theta(U))$.

Let ρ'' be the substitution:

$$\rho''(x) = \begin{cases} \rho'(x) & \text{if } x \in \sigma_3(\sigma_2(U)), \\ \rho(x) & \text{if } x \in \sigma_2(\sigma_1(V)) \setminus \sigma_2(\sigma_1(U \cap V)), \\ x & \text{otherwise.} \end{cases}$$

Note that thanks to the second hypothesis of the lemma, we are sure that the first and second cases in the definition of ρ'' may not occur together. We want to prove that $\delta(\eta(x)) = \rho''(\sigma_3(\sigma_2(\sigma_1(x))))$ for each $x \in U \cup V$. Since ρ'' restricted to $\mathrm{vars}(\sigma_3(\sigma_2(\sigma_1(U \cup V))))$ is an injective map from variables to variables, by Lemma 3.4 this implies $\delta \circ \eta \sim_{U \cup V} \sigma_3 \circ \sigma_2 \circ \sigma_1$, which is the statement of the theorem.

Thus if $x \in U \setminus V$, then $\eta(x) = x$ and $\delta(\eta(x)) = \theta(x) = \rho'(\sigma_3(\sigma_2(x))) = \rho''(\sigma_3(\sigma_2(x))) = \rho''(\sigma_3(\sigma_2(\sigma_1(x))))$ because $\mathrm{dom}(\sigma_1) \cap U = \emptyset$ and by the definition of ρ''.

If $x \in U \cap V$, then $\delta(\eta(x)) = \delta(\rho(\sigma_2(x)))$, since $\mathrm{dom}(\sigma_1) \cap U = \emptyset$, and thus $\delta(\eta(x)) = \rho'(\sigma_3(\sigma_2(x)))$, which is equal to $\rho''(\sigma_3(\sigma_2(\sigma_1(x))))$ because $\mathrm{dom}(\sigma_1) \cap U = \emptyset$ and by the definition of ρ''.

If $x \in V \setminus U$, then $\delta(\eta(x)) = \delta(\rho(\sigma_2\sigma_1(x)))$. Let $y \in \mathrm{vars}(\sigma_2(\sigma_1(x)))$. If we assume that $y \in \mathrm{vars}(\sigma_2(U \cap V))$, then $\delta(\rho(y)) = \rho'(\sigma_3(y)) = \rho''(\sigma_3(y))$ by the definition of δ and ρ''. If $y \notin \mathrm{vars}(\sigma_2(U \cap V))$, then $\delta(\rho(y)) = \rho(y) = \rho''(y) = \rho''(\sigma_3(y))$ by the definition of ρ'' and the second condition in the theorem. In both cases we obtain $\delta(\rho(y)) = \rho''(\sigma_3((y)))$ for each $y \in \mathrm{vars}(\sigma_2(\sigma_1(x)))$. Therefore, for each $x \in U \cap V$, $\delta(\eta(x)) = \delta(\rho(\sigma_2(\sigma_1(x)))) = \rho''(\sigma_3(\sigma_2(\sigma_1(x))))$, and this concludes the proof. □

A.4 Proof of correctness

Let D_P be defined as $\lambda d.\bigsqcup_{\mathrm{Ps}}\{\mathscr{C}[\![cl]\!]d \mid cl \in P\}$ and D_P^i be the ith iteration of D_P with $D_P^0 = \lambda A.\lambda x.\bot_{\mathrm{Ps}}$. Note that $D_P^\omega = \mathscr{P}[\![P]\!]$ and that D_P^i is relevant for each i.

Lemma A.6
(Correctness lemma)
Let $i \in \mathbb{N}$, $[\phi]_V \in \mathit{ISubst}_\sim$, $G \in \mathsf{Bodies}$, and $P \in \mathsf{Progs}$. If $[\phi]_{V\cup G} = \mathrm{mgu}([\phi]_V, [\epsilon]_G)$ and $G\phi \overset{*}{\underset{\sigma}{\to}} \square$ is a leftmost SLD-refutation, with at most i steps, where all clauses are renamed apart from V, G, ϕ, and the program P, then $\mathscr{B}[\![G]\!]D_P^i[\{[\phi]\}, V] \sqsupseteq_{\mathrm{Ps}} [\{[\sigma \circ \phi]\}, V \cup \mathrm{vars}(G)]$.

Remark A.7
The condition $[\phi]_{V\cup G} = \mathrm{mgu}([\phi]_V, [\epsilon]_G)$ is used to check that the chosen representative ϕ does not bind any variable in $\mathrm{vars}(G) \setminus V$. All the variables in $\mathrm{vars}(G) \setminus V$ are forced to be unused, according to Definition A.3.

Remark A.8
The theorem probably holds under weaker conditions on the variables of the SLD-resolution. However, proving the result in this case would be more difficult. Since the obtained generalization is not very interesting, we valued that it was not worth the effort.

Proof
The proof is by double induction on i and on the structure of the goal G. Assume fixed $\Phi = \{[\phi]_V\}$ such that $[\phi]_{V\cup G} = \mathrm{mgu}([\phi]_V, [\epsilon]_G)$.

We start with the case $i = 0$. The only SLD-refutation of length 0 is the SLD-derivation for the empty goal □, whose computed answer substitution is ϵ. In the collecting semantics, we have $\mathscr{B}[\![\square]\!]D_P^i[\{[\phi]\}, V] = [\{[\phi]\}, V] = [\{[\epsilon \circ \phi]\}, V]$, which is the required result.

If $i > 0$, assume the lemma holds for all $j < i$, and we prove it for i, by induction on the structure of goals. The case for the empty goal has been already examined, so we assume $G = A, G'$, where A is an atom. To ease the exposition, we first consider the atomic case in which $G' = \square$ and then analyze the general one.

Atomic goal. Given the not-empty SLD-derivation $G\phi \overset{*}{\underset{\sigma}{\to}} \square$, we may decompose it as

$$G\phi \xrightarrow[\sigma_1]{\rho(cl)} (C_1 \dots C_n)\rho\sigma_1 \overset{*}{\underset{\sigma_2}{\longrightarrow}} \square,$$

where $cl = H \leftarrow C_1 \dots C_n$ is a program clause; $\sigma_1 = \text{mgu}(G\phi, H\rho)$; and ρ is a renaming of cl apart from G, V, ϕ, and the program P. Note that this implies the standard renaming condition for SLD-resolutions; i.e., $\rho(cl)$ is renamed apart from $G\phi$. Since G is atomic,

$$\mathscr{B}[\![G]\!]D_P^i[\Phi, V] = D_P^i G[\Phi, V] \sqsupseteq_{\text{Ps}} \mathscr{C}[\![H \leftarrow C_1 \dots C_n]\!]D_P^{i-1}G[\Phi, V],$$

which, in turn, is equal to $\mathbf{U}^b_{\text{Ps}}(\mathscr{B}[\![C_1 \dots C_n]\!]D_P^{i-1}(\mathbf{U}^f_{\text{Ps}}([\Phi, V], G, H)), [\Phi, V], H, G)$. We know that

$$\mathbf{U}^f_{\text{Ps}}([\{[\phi]\}, V], G, H) = \pi_{\text{Ps}}(\text{mgu}(\rho'([\phi]_V), [\text{mgu}(\rho'(G) = H)]_{\rho'(G) \cup H}), \text{vars}(H)),$$

where ρ' is any renaming such that $\rho'(\text{vars}(G) \cup V) \cap \text{vars}(H) = \emptyset$. We can choose as ρ' the renaming ρ^{-1}, since $\rho(\text{vars}(cl)) \cap \text{vars}(G) = \emptyset$ and $\rho(\text{vars}(cl)) \cap V = \emptyset$ implies that $\rho^{-1}(\text{vars}(G) \cup V) \cap \text{vars}(H) = \emptyset$. In turn, this implies that

$$\begin{aligned}
&\text{mgu}(\rho'([\phi]_V), [\text{mgu}(\rho'(G) = H)]_{\rho'(G) \cup H}) \\
&\quad = \rho^{-1}(\text{mgu}([\phi]_V, [\text{mgu}(G = \rho(H))]_{G \cup \rho(H)}) \\
&\quad = \rho^{-1}(\text{mgu}([\phi]_V, [\text{mgu}(G = \rho(H))]_{G \cup \rho(H)}, [\epsilon]_G)) \\
&\quad = \rho^{-1}(\text{mgu}([\phi]_{V \cup G}, [\text{mgu}(G = \rho(H))]_{G \cup \rho(H)}) \\
&\quad = \rho^{-1}([\text{mgu}(\phi, \text{mgu}(G = \rho(H)))]_{V \cup G \cup \rho(H)}).
\end{aligned}$$

The last pass is only valid when $(V \cup \text{vars}(G) \cup \text{vars}(\phi)) \cap (\text{vars}(G) \cup \text{vars}(\rho(H)) \subseteq (V \cup \text{vars}(G)) \cap (\text{vars}(G) \cup \text{vars}(\rho(H))) = \text{vars}(G)$. This is the case because $\text{vars}(\phi) \cap \rho(\text{vars}(cl)) = \emptyset$, thanks to our choice of ρ.

By standard properties of substitutions, we obtain

$$\begin{aligned}
&\rho^{-1}([\text{mgu}(\phi, \text{mgu}(G = \rho(H)))]_{V \cup G \cup \rho(H)}) \\
&\quad = \rho^{-1}([\text{mgu}(G\phi = (\rho(H))\phi) \circ \phi]_{V \cup G \cup \rho(H)}) \\
&\quad = \rho^{-1}([\text{mgu}(G\phi = \rho(H)) \circ \phi]_{V \cup G \cup \rho(H)}) \\
&\quad = \rho^{-1}([\sigma_1 \circ \phi]_{V \cup G \cup \rho(H)}),
\end{aligned}$$

since $\text{vars}(\phi) \cap \text{vars}(\rho(H)) = \emptyset$. For the same reason, $\sigma_1 \circ \phi \sim_{\text{vars}(\rho(H))} \sigma_1$. It follows that

$$\rho^{-1}(\sigma_1 \circ \phi) \sim_{\text{vars}(H)} \rho^{-1}(\sigma_1) = \rho^{-1} \circ \sigma_1 \circ \rho \sim_{\text{vars}(H)} \sigma_1 \circ \rho.$$

Therefore $\mathbf{U}^f_{\text{Ps}}([\{[\phi]\}, V], G, H) = [\{[\sigma_1 \circ \rho]\}, \text{vars}(H)]$ and

$$\begin{aligned}
&\mathbf{U}^b_{\text{Ps}}(\mathscr{B}[\![C_1 \dots C_n]\!]D_P^{i-1}(\mathbf{U}^f_{\text{Ps}}([\Phi, V], G, H)), [\Phi, V], H, G) \sqsupseteq_{\text{Ps}} \\
&\qquad\qquad \mathbf{U}^b_{\text{Ps}}(\mathscr{B}[\![C_1 \dots C_n]\!]D_P^{i-1}[\{[\sigma_1 \circ \rho]\}, \text{vars}(H)], [\Phi, V], H, G).
\end{aligned}$$

Note that the SLD resolution $(C_1 \dots C_n)\rho\sigma_1 \xrightarrow[\sigma_2]{*} \Box$ can be seen as $(C_1 \dots C_n)(\sigma_1 \circ \rho) \xrightarrow[\sigma_2]{*} \Box$. In order to apply the inductive hypothesis on the latter derivation, we need to verify that $[\sigma_1 \circ \rho]_{\text{vars}(cl)} = \text{mgu}([\sigma_1 \circ \rho]_{\text{vars}(H)}, [\epsilon]_{\text{vars}(C_1 \dots C_n)})$. By definition $\sigma_1 \circ \rho = \text{mgu}(G\phi, H\rho) \circ \rho$. Moreover, since $\rho(\text{vars}(cl)) \cap \text{vars}(G\phi) = \emptyset$ and $\rho(\text{vars}(cl)) \cap \text{vars}(H\rho) = \text{vars}(H\rho)$, it follows that for all $v \in \rho(\text{vars}(cl) \setminus \text{vars}(H))$, $v \notin \text{vars}(\sigma_1)$. Hence, for each $v \in \text{vars}(cl) \setminus \text{vars}(H)$, $\sigma_1(\rho(v)) = \rho(v)$. Moreover, if $\rho(v)$ occurs in $(\sigma_1 \circ \rho)(x)$ for some x, then $\rho(v)$ occurs in $\rho(x)$, and this is only possible if $x = v$.

By Proposition A.4, this proves that $\mathrm{mgu}([\sigma_1 \circ \rho]_{\mathrm{vars}(H)}, [\epsilon]_{\mathrm{vars}(C_1 \ldots C_n)}) = [\sigma_1 \circ \rho]_{\mathrm{vars}(cl)}$. Thus, by inductive hypothesis, we have that

$$\mathbf{U}^b_{\mathrm{Ps}}(\mathscr{B}[\![C_1 \ldots C_n]\!]D_P^{i-1}[\{[\sigma_1 \circ \rho]\}, \mathrm{vars}(H)], [\Phi, V], H, G) \sqsupseteq_{\mathrm{Ps}} \mathbf{U}^b_{\mathrm{Ps}}([\{[\sigma_2 \circ \sigma_1 \circ \rho]\}, \mathrm{vars}(cl)], [\Phi, V], H, G).$$

We know that $\mathsf{unif}_{\mathrm{Ps}}([\{[\phi]\}, V], \mathrm{mgu}(\rho(H) = G)) = [\{[\sigma_1 \circ \phi]\}, V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))]$. Therefore, choosing ρ as the renaming for $\mathbf{U}^b_{\mathrm{Ps}}$, we obtain

$$\begin{aligned} &\mathsf{match}_{\mathrm{Ps}}(\rho([\{[\sigma_2 \circ \sigma_1 \circ \rho]\}, \mathrm{vars}(cl)]), [\{[\sigma_1 \circ \phi]\}, V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))]) \\ &\quad = \mathsf{match}_{\mathrm{Ps}}([\{[\rho \circ \sigma_2 \circ \sigma_1]\}, \mathrm{vars}(\rho(cl))]), [\{[\sigma_1 \circ \phi]\}, V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))]) \\ &\quad = \mathsf{match}_{\mathrm{Ps}}([\{[\sigma_2 \circ \sigma_1]\}, \mathrm{vars}(\rho(cl))]), [\{[\sigma_1 \circ \phi]\}, V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))]). \end{aligned}$$

Since $\mathrm{vars}(\rho(cl)) \cap (V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))) = \mathrm{vars}(\rho(H))$ and $\sigma_2 \circ \sigma_1 \preceq_{\mathrm{vars}(\rho(H))} \sigma_1 \circ \phi$ (being $\mathrm{vars}(\phi) \cap \mathrm{vars}(\rho(H)) = \emptyset$), the following holds:

$$\begin{aligned} &\mathsf{match}_{\mathrm{Ps}}([\{[\sigma_2 \circ \sigma_1]\}, \mathrm{vars}(\rho(cl))]), [\{[\sigma_1 \circ \phi]\}, V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))]) \\ &\qquad = [\mathrm{mgu}([\sigma_2 \circ \sigma_1]_{\rho(cl)}, [\sigma_1 \circ \phi]_{V \cup G \cup \rho(H)}), V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))]. \end{aligned}$$

We would like to apply the composition lemma (Lemma A.5) to this unification. We need to check that

- $\mathrm{dom}(\phi) \cap \rho(cl) = \emptyset$;
- $y \in \sigma_1\phi(V \cup \mathrm{vars}(G) \cup \rho(H)) \setminus \sigma_1\phi(\rho(H))$ then $y \notin \mathrm{dom}(\sigma_2) \cup \sigma_2\sigma_1(\rho(cl))$.

The first property trivially follows by the hypothesis that ρ renames cl apart from ϕ. For the second condition, note that since $\sigma_1 = \mathrm{mgu}(G\phi, H\rho)$, if $y \in \sigma_1(\phi(G))$, then $y \in \sigma_1(\rho(H)) = \sigma_1(\phi(\rho(H)))$. Therefore $y \in \sigma_1(\phi(V \cup \mathrm{vars}(G))) \setminus \sigma_1(\phi(\rho(H)))$ iff $y \in \sigma_1(\phi(V \setminus G)) = \phi(V \setminus G)$. However, since such a variable does not appear in the initial goal of the SLD-resolution $G\phi$ and since the resolution is renamed apart from ϕ, it happens that it does not appear in $\mathrm{vars}(\sigma_2)$ and thus in $\mathrm{dom}(\sigma_2)$. We now show that $y \notin \sigma_2(\sigma_1(\rho(cl)))$. By hypothesis, $y \notin \sigma_1(\phi(\rho(cl)))$, and since $\rho(cl)$ is renamed apart from ϕ, it follows that $y \notin (\sigma_1(\rho(cl)))$. Moreover, as we have seen before, $y \notin \mathrm{vars}(\sigma_2)$; hence $y \notin \mathrm{vars}(\sigma_2(\sigma_1(\rho(cl))))$.

It turns out that we may apply the composition lemma (Lemma A.5) and obtain

$$\begin{aligned} &[\mathrm{mgu}([\sigma_2 \circ \sigma_1]_{\rho(cl)}, [\sigma_1 \circ \phi]_{V \cup G \cup \rho(H)}), V \cup \mathrm{vars}(G) \cup \mathrm{vars}(\rho(H))] \\ &\qquad = [\{\sigma_2 \circ \sigma_1 \circ \phi\}, \rho(cl) \cup V \cup G]. \end{aligned}$$

By projecting on $G \cup V$ we obtain

$$\mathscr{B}[\![G]\!]D_P^i[\Phi, V] \sqsupseteq_{\mathrm{Ps}} [\{\sigma_2 \circ \sigma_1 \circ \phi]\}, V \cup \mathrm{vars}(G)],$$

which concludes the proof of the atomic case.

Nonatomic goal. In this case, decompose the (leftmost) SLD-resolution for $G = A, G'$ in the following way:

$$A\phi, G'\phi \overset{*}{\underset{\sigma_1}{\longrightarrow}} G'\phi\sigma_1 \overset{*}{\underset{\sigma_2}{\longrightarrow}} \Box, \tag{A 3}$$

where both the subderivations have length strictly less than i. Note that since the complete derivation is renamed apart from V, G, ϕ, and the program P, the same holds for the first subderivation. Moreover, since $[\phi]_{V \cup G} = \mathrm{mgu}([\phi]_V, [\epsilon]_G)$, each $v \in A$ is free and independent in $[\phi]_{V \cup G}$, i.e., $[\phi]_{V \cup A} = \mathrm{mgu}([\phi]_V, [\epsilon]_A)$. Therefore, we may apply what was proved in the atomic case above, obtaining

$$D_P^i A[\Phi, V] \sqsupseteq_{\mathrm{Ps}} [\{\sigma_1 \circ \phi\}, V \cup \mathrm{vars}(A)].$$

The second subderivation in (A 3) is renamed apart from

- V, since the complete derivation is renamed apart from V;
- A and G', since the complete derivation is renamed apart from G;
- $\sigma_1 \circ \phi$, since the complete derivation is renamed apart from ϕ and the second part is renamed apart from σ_1;
- P, since the complete derivation is renamed apart from P.

Moreover, assume $x \in \mathrm{vars}(G') \setminus \mathrm{vars}(V \cup A)$ and $x \neq y \in \mathrm{vars}(V \cup G)$. By hypothesis, $[\phi]_{V \cup G} = \mathrm{mgu}([\phi]_V, [\epsilon]_G)$, which implies that $\phi(x) \in \mathscr{V}$ and $\phi(x) \notin \mathrm{vars}(\phi(y))$. Since $\mathrm{vars}(\sigma_1) = W \cup X$, where W is a fresh set of variables disjoint from $V \cup G$ and ϕ and $X \subseteq \mathrm{vars}(A\phi)$, it happens that $\phi(x) \notin \mathrm{vars}(\sigma_1)$. Therefore $\sigma_1(\phi(x)) = \phi(x)$ and $\phi(x) \notin \mathrm{vars}(\sigma_1(\phi(y)))$. This implies that $[\sigma_1 \circ \phi]_{V \cup G} = \mathrm{mgu}([\sigma_1 \circ \phi]_{V \cup A}, [\epsilon]_{G'})$ by Proposition A.4. This means that we may apply the inductive hypothesis on the second subderivation, obtaining

$$\mathscr{B}[\![G']\!] D_P^i [\{\sigma_1 \circ \phi\}, V \cup \mathrm{vars}(A)] \sqsupseteq_{\mathrm{Ps}} [\{\sigma_2 \circ \sigma_1 \circ \phi\}, V \cup \mathrm{vars}(G)].$$

Since $\mathscr{B}[\![A, G']\!] D_P^i [\Phi, V] = \mathscr{B}[\![G']\!] D_P^i (D_P^i A[\Phi, V])$ by the above disequalities and monotonicity of $\mathscr{B}$, we obtain

$$\mathscr{B}[\![A, G']\!] D_P^i [\Phi, V] \sqsupseteq_{\mathrm{Ps}} [\{\sigma_2 \circ \sigma_1 \circ \phi\}, V \cup \mathrm{vars}(G)],$$

which concludes the proof. □

Now we may use standard properties of SLD-resolution together with Lemma A.6 to prove the required correctness theorem.

Theorem A.9

(Semantic correctness)

Given a program P and an goal G, if θ is a computed answer for the goal G, then

$$\mathscr{B}[\![G]\!](\mathscr{P}[\![P]\!])G[\{\epsilon\}, \mathrm{vars}(G)] \sqsupseteq_{\mathrm{Ps}} [\{[\theta]\}, \mathrm{vars}(G)].$$

Proof

If θ is a computed answer for a goal G and ρ is a renaming, then $\theta' = (\rho \circ \theta)_{|\mathrm{vars}(G)}$ is a computed answer too (Apt 1990) and $\theta \sim_{\mathrm{vars}(G)} \theta'$. Consider any such θ' with the property that $\mathrm{vars}(\theta') \cap \mathrm{vars}(P) = \emptyset$, and let $G \overset{*}{\underset{\theta'}{\longrightarrow}} \square$ be a leftmost SLD-resolution for θ'. Since there exists a leftmost SLD-resolution $G \overset{*}{\underset{\theta'}{\longrightarrow}} \square$ that is renamed apart from P, the thesis follows, by Lemma A.6. □

Appendix B: Correctness of forward unification

Lemma B.1
Given $\delta, \sigma \in \mathit{Subst}$, $v \in \mathscr{V}$, it is the case that $\mathit{occ}(\delta \circ \sigma, v) = \mathit{occ}(\sigma, \mathit{occ}(\delta, v))$.

Proof
By definition, $x \in \mathit{occ}(\delta \circ \sigma, v)$ iff $v \in \delta(\sigma(x))$; i.e., there exists $w \in \mathscr{V}$ such that $w \in \sigma(x)$ and $v \in \delta(w)$. In other words, $x \in \mathit{occ}(\delta \circ \sigma, v)$ iff there exists $w \in \mathscr{V}$ such that $w \in \mathit{occ}(\delta, v)$ and $x \in \mathit{occ}(\sigma, w)$ iff $x \in \mathit{occ}(\sigma, \mathit{occ}(\delta, v))$. □

Proposition B.2
Let $t \in \mathsf{Terms}$, $\sigma \in \mathit{Subst}$, and $U \in \wp_f(\mathscr{V})$ such that $\mathrm{vars}(t) \subseteq U$. Let $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$. Then the following property holds:

$$\forall v \in \mathscr{V}. v \in \mathrm{vars}(t\sigma) \iff \mathit{occ}(\sigma, v) \cap U \in \mathbf{rel}(S, t).$$

Proof
Note that $v \in \mathrm{vars}(t\sigma)$ iff $\exists u \in t$ such that $v \in \sigma(u)$. In turn, this holds iff $\exists u \in t$ such that $u \in \mathit{occ}(\sigma, v)$ iff $\mathit{occ}(\sigma, v) \cap \mathrm{vars}(t) \neq \emptyset$ iff $(\mathit{occ}(\sigma, v) \cap U) \cap \mathrm{vars}(t) \neq \emptyset$. Note that $X = \mathit{occ}(\sigma, v) \cap U \in S$ and therefore $X \cap \mathrm{vars}(t) \neq \emptyset$ iff $X \in \mathbf{rel}(S, t)$ by the definition of **rel**. □

Proposition B.3
Let $[\sigma]_U \in \mathit{ISubst}_\sim$, $\{x/t\} \in \mathit{ISubst}$ such that $\mathrm{vars}(\{x/t\}) \subseteq U$ and σ and $\{x/t\}$ unify. If $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$ and $\delta = \mathrm{mgu}(x\sigma = t\sigma)$, we obtain

$$\begin{aligned}\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\mathrm{Sh}} [(S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t)))\\ \cup \{\mathit{occ}(\sigma, \mathit{occ}(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}, U].\end{aligned}$$

Proof
Since $\mathrm{vars}(\{x/t\}) \subseteq U$, we have $\mathrm{mgu}([\sigma]_U, [x/t]_U) = [\mathrm{mgu}(\sigma, \{x/t\})]_U$. Then, by the definition of δ, it holds that $\mathrm{mgu}(\sigma, x = t) = \mathrm{mgu}(\mathrm{Eq}(\sigma) \cup x\sigma = t\sigma) = \mathrm{mgu}(x\sigma = t\sigma) \circ \sigma = \delta \circ \sigma$ (Palamidessi 1990, Proposition 6.1). Therefore, we only need to show that

$$\begin{aligned}\alpha_{\mathrm{Sh}}([\delta \circ \sigma]_U) \sqsubseteq_{\mathrm{Sh}} [(S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t)))\\ \cup \{\mathit{occ}(\sigma, \mathit{occ}(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}, U]. \qquad \text{(B 1)}\end{aligned}$$

By the definition of α_{Sh}, we have to show that for all $v \in \mathscr{V}$, $\mathit{occ}(\delta \circ \sigma, v) \cap U \in (S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t))) \cup \{\mathit{occ}(\sigma, \mathit{occ}(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}$. Let $v \in \mathscr{V}$. We have the following cases:

- $v \in \mathrm{vars}(x\sigma = t\sigma)$: by Lemma B.1, $\{\mathit{occ}(\delta \circ \sigma, v) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\} = \{\mathit{occ}(\sigma, \mathit{occ}(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}$.
- $v \notin \mathrm{vars}(x\sigma = t\sigma)$: thus $v \notin \mathrm{vars}(\delta)$ and $\mathit{occ}(\delta \circ \sigma, v) = \mathit{occ}(\sigma, v)$. We know that $\mathit{occ}(\sigma, v) \cap U \in S$, by the definition of S. Moreover, we show that $\mathit{occ}(\sigma, v) \cap U \notin \mathbf{rel}(S, x) \cup \mathbf{rel}(S, t)$. Since $v \notin \mathrm{vars}(x\sigma = t\sigma)$, we can apply Proposition B.2 twice to the terms x and t and obtain $\mathit{occ}(\sigma, v) \cap U \notin \mathbf{rel}(S, x) \cup \mathbf{rel}(S, t)$.

By collecting the results of the two cases, equation (B 1) is proved. □

Proposition B.4
Let $[\sigma]_U \in \mathit{ISubst}_{\sim}$, $\{x/t\} \in \mathit{ISubst}$ such that $\mathrm{vars}(\{x/t\}) \subseteq U$ and σ and $\{x/t\}$ unify. If $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$ and x is free and independent from U in $[\sigma]_U$, then

$$\begin{aligned}&\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U))\\ &\qquad\sqsubseteq_{\mathrm{Sh}} [(S \setminus (\mathbf{rel}(S,x) \cup \mathbf{rel}(S,t))) \cup \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,t)), U].\end{aligned}$$

Proof
First of all note that without loss of generality, we may assume $x \notin \mathrm{vars}(\sigma)$. Then, by Proposition B.3, we have that

$$\begin{aligned}\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\mathrm{Sh}} [(S \setminus (&\mathbf{rel}(S,x) \cup \mathbf{rel}(S,t)))\\ &\cup\{occ(\sigma, occ(\delta, v)) \cap U \mid\in \mathrm{vars}(x\sigma = t\sigma)\}, U],\end{aligned}$$

where $\delta = \mathrm{mgu}(x\sigma = t\sigma)$. Since $x \notin \mathrm{vars}(\sigma)$, we have that $x\sigma = t\sigma$ is equal to $x = t\sigma$. Moreover, $x \notin \mathrm{vars}(t\sigma)$, since $x \notin \mathrm{vars}(t)$ and $x \notin \mathrm{vars}(\sigma)$ by hypothesis. Thus $\delta = \mathrm{mgu}(x = t\sigma) = \{x/t\sigma\}$. It follows that $\mathrm{vars}(x\sigma = t\sigma) = \{x\} \cup \mathrm{vars}(t\sigma)$. Therefore, the following equalities hold:

$$\begin{aligned}&\{occ(\sigma, occ(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}\\ &\quad= \{occ(\sigma, occ(\delta, v)) \cap U \mid v \in \{x\} \cup \mathrm{vars}(t\sigma)\}\\ &\quad= \{occ(\sigma, occ(\delta, v)) \cap U \mid v \in \mathrm{vars}(t\sigma)\} && (\text{since } x \in \mathrm{dom}(\delta),\ occ(\delta, x) = \emptyset)\\ &\quad= \{occ(\sigma, \{x, v\}) \cap U \mid v \in \mathrm{vars}(t\sigma)\} && (\text{since } \delta = \{x/t\sigma\})\\ &\quad= \{(occ(\sigma, x) \cup occ(\sigma, v)) \cap U \mid v \in \mathrm{vars}(t\sigma)\}\\ &\quad= \{(\{x\} \cup occ(\sigma, v)) \cap U \mid v \in \mathrm{vars}(t\sigma)\} && (\text{since } x \notin \mathrm{vars}(\sigma)).\end{aligned}$$

Moreover, for each $v \in \mathrm{vars}(t\sigma)$, by Proposition B.2 it holds that $occ(\sigma, v) \cap U \in \mathbf{rel}(S, t)$. Therefore, $\{(\{x\} \cup occ(\sigma, v)) \cap U \mid v \in \mathrm{vars}(t\sigma)\} \subseteq \mathbf{bin}(\{x\}, \mathbf{rel}(S, t))$. Since $x \notin \mathrm{vars}(\sigma)$ and $x \in U$, it follows that $occ(\sigma, x) = \{x\}$ and thus $\{x\} \in \mathbf{rel}(S, x)$ being $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$. As a consequence $\mathbf{bin}(\{x\}, \mathbf{rel}(S, t)) \subseteq \mathbf{bin}(\mathbf{rel}(S, x), \mathbf{rel}(S, t))$ from which it follows that $\alpha_{\mathrm{Sh}}([\mathrm{mgu}(\mathrm{Eq}(\sigma) \cup x = t)]_U) \sqsubseteq_{\mathrm{Sh}} [(S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t))) \cup \mathbf{bin}(\mathbf{rel}(S, x), \mathbf{rel}(S, t)), U]$. □

Proposition B.5
Given $s, t \in \mathsf{Terms}$ and $W, Y \in \wp_f(\mathcal{V})$ such that s and t unify, $\mathrm{vars}(s = t) \subseteq W$ and $Y \subseteq \mathrm{uvars}(s = t)$, $\delta = \mathrm{mgu}(s = t)$ enjoys the following properties:

(1) $\forall v \in \mathrm{vars}(s).\ occ(\delta, v) \cap \mathrm{vars}(s) \neq \emptyset \Rightarrow occ(\delta, v) \cap \mathrm{vars}(t) \neq \emptyset$,
(2) $\forall v \in \mathrm{vars}(s).\ occ(\delta, v) \cap \mathrm{vars}(s) \supseteq \{x_1, x_2\} \wedge x_1 \neq x_2 \Rightarrow occ(\delta, v) \cap Z \neq \emptyset$,

where $Z = \mathrm{vars}(t) \setminus Y$.

Proof
We prove the two points separately.

(1) If $occ(\delta, v) \cap \mathrm{vars}(s) \neq \emptyset$, then $v \notin \mathrm{dom}(\delta)$ and therefore $v \in \delta(s)$. Since δ is an unifier for s and t, it should be $v \in \delta(t)$, and therefore there exists $y \in t$ such that $y \in occ(\delta, v)$.

(2) First of all, note that given two terms s and t in a given signature Σ, the result of $\mathrm{mgu}(s = t)$ does not change if we enlarge Σ with a new constant symbol. Therefore, assume without loss of generality that there is a constant symbol a in the signature. The proof proceeds by contradiction.
Assume that there exist $x_1, x_2 \in \mathrm{vars}(s)$, $v \in W$ such that $x_1, x_2 \in occ(\delta, v)$ and $occ(\delta, v) \cap Z = \emptyset$. Let $\sigma = \{x = a \mid x \in W\}$ and consider the substitution $\delta' = \{z/(\delta(z))\sigma \mid z \in Z\}$. Note that this is an idempotent substitution, since it is ground. Now consider $\delta'' = \mathrm{mgu}(\mathrm{Eq}(\delta) \cup \mathrm{Eq}(\delta'))$, which clearly exists and, by the definition of δ', is $\delta'' = \{x/a \mid x \in \mathrm{vars}(\delta(Z))\} \circ \delta$. Therefore, $occ(\delta'', v) = occ(\delta, v)$ because $v \notin \mathrm{vars}(\delta(Z))$ being $occ(\delta, v) \cap Z = \emptyset$.
Moreover, $\delta'' = \mathrm{mgu}(\mathrm{Eq}(\delta) \cup \mathrm{Eq}(\delta')) = \mathrm{mgu}(\{s = t\} \cup \mathrm{Eq}(\delta')) = \mathrm{mgu}(s\delta' = t\delta') \circ \delta' = \delta' \uplus \mathrm{mgu}(s\delta' = t\delta')$. By the definition of δ', it holds that $\mathrm{vars}(t\delta') \cap Z = \emptyset$, and thus $\mathrm{vars}(t\delta') \subseteq Y$. From the definition of Y it follows that $\mathrm{vars}(t\delta') \subseteq \mathrm{uvars}(s = t)$, and thus $\mathrm{vars}(t\delta') \subseteq \mathrm{uvars}(s\delta' = t\delta')$, since $\mathrm{rng}(\delta') = \emptyset$. Therefore the term $t\delta'$ is linear and independent from $s\delta'$ and $occ(\mathrm{mgu}(s\delta' = t\delta'), v) = occ(\mathrm{mgu}(s\delta' = t\delta') \uplus \delta', v) = occ(\delta, v)$.
If we apply the result for linear and independent terms (e.g., King 2000, Proposition 3.1), we obtain an absurd, since it is not possible that both x_1 and x_2 are elements of $occ(\mathrm{mgu}(s\delta' = t\delta'), v)$.

This concludes the proof. □

Proposition B.6

Let $[\sigma]_U \in ISubst_{\sim}$, $\{x/t\} \in ISubst$ such that $\mathrm{vars}(\{x/t\}) \subseteq U$ and σ and $\{x/t\}$ unify. Given $Y \subseteq \mathrm{vars}(t)$ such that for all $y \in Y$, $\mathrm{vars}(\sigma(y)) \subseteq \mathrm{uvars}(x\sigma = t\sigma)$, if $\alpha_{\mathrm{Sh}}([\sigma]_U) \sqsubseteq_{\mathrm{Sh}} [S, U]$, then

$$\begin{aligned}\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\mathrm{Sh}} [&(S \setminus (\mathbf{rel}(S,t) \cup \mathbf{rel}(S,x)))\\ &\cup \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,Y)^*) \cup \mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*)\\ &\cup \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*), \mathbf{rel}(S,Y)^*), U],\end{aligned}$$

where $Z = \mathrm{vars}(t) \setminus Y$.

Proof

By Proposition B.3, we have that

$$\begin{aligned}\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_U, [x/t]_U)) \sqsubseteq_{\mathrm{Sh}} [&(S \setminus (\mathbf{rel}(S,x) \cup \mathbf{rel}(S,t)))\\ &\cup \{occ(\sigma, occ(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}, U],\end{aligned}$$

where $\delta = \mathrm{mgu}(x\sigma = t\sigma)$. We show that

$$\begin{aligned}&\{occ(\sigma, occ(\delta, v)) \cap U \mid v \in \mathrm{vars}(x\sigma = t\sigma)\}\\ &\quad\subseteq \mathbf{bin}(\mathbf{rel}(S,x), \mathbf{rel}(S,Y)^*) \cup \mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*)\\ &\quad\cup \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S,x)^*, \mathbf{rel}(S,Z)^*), \mathbf{rel}(S,Y)^*) \cup \{\emptyset\},\end{aligned}$$

from which the thesis follows. The following equalities hold, for all $v \in \text{vars}(x\sigma = t\sigma)$:

$$\begin{aligned}
&occ(\sigma, occ(\delta, v)) \cap U \\
&\quad = \bigcup\{occ(\sigma, w) \cap U \mid w \in occ(\delta, v)\} \\
&\quad = \bigcup\{occ(\sigma, w) \cap U \mid w \in occ(\delta, v) \cap \text{vars}(x\sigma)\} \\
&\qquad \cup \bigcup\{occ(\sigma, w) \cap U \mid w \in occ(\delta, v) \cap \text{vars}(t\sigma)\} \\
&\qquad\qquad \text{(by partitioning the variables in } occ(\delta, v) \subseteq \text{vars}(\delta) \cup \{v\}).
\end{aligned}$$

By applying Proposition B.5 to the equation $x\sigma = t\sigma$ we get $occ(\delta, v) \cap \text{vars}(x\sigma) \neq \emptyset$ iff $occ(\delta, v) \cap \text{vars}(t\sigma) \neq \emptyset$. Since the case $occ(\delta, v) = \emptyset$ is trivial, it only remain to consider the case $occ(\delta, v) \neq \emptyset$ that implies $occ(\delta, v) \cap \text{vars}(t\sigma) \neq \emptyset \neq occ(\delta, v) \cap \text{vars}(x\sigma)$. In the following, let $A = \bigcup\{occ(\sigma, w) \cap U \mid w \in occ(\delta, v) \cap \text{vars}(x\sigma)\}$ and $B = \bigcup\{occ(\sigma, w) \cap U \mid w \in occ(\delta, v) \cap \text{vars}(t\sigma)\}$. Note that by Proposition B.2, $occ(\sigma, w) \cap U \in \mathbf{rel}(S, \{x\})$ if $w \in \text{vars}(x\sigma)$ and $x \in U$, which implies $A \in \mathbf{rel}(S, \{x\})^*$. For the same reason, $B \in \mathbf{rel}(S, \text{vars}(t))^*$, i.e.,

$$occ(\sigma, occ(\delta, v)) \cap U \in \mathbf{bin}(\mathbf{rel}(S, \{x\})^*, \mathbf{rel}(S, \text{vars}(t))^*),$$

which is the standard result for abstract unification without considering freeness or linearity. We can do better if we proceed by cases on $occ(\delta, v) \cap \text{vars}(t\sigma)$.

- $occ(\delta, v) \cap \text{vars}(t\sigma) \subseteq \text{vars}(\sigma(Y))$: Let $Z' = \text{vars}(t\sigma) \setminus \text{vars}(\sigma(Y))$ it follows that $occ(\delta, v) \cap Z' = \emptyset$. Therefore, by Proposition B.5(2) applied to the terms $x\sigma$ and $t\sigma$, we have that $\nexists x_1, x_2 \in \text{vars}(x\sigma)$ such that $x_1, x_2 \in occ(\delta, v)$. Since $occ(\delta, v) \cap \text{vars}(x\sigma) \neq \emptyset$, it follows that there exists $x' \in \text{vars}(x\sigma)$ such that $occ(\delta, v) \cap \text{vars}(x\sigma) = \{x'\}$. This implies that $A \in \mathbf{rel}(S, \{x\})$. Moreover, by Proposition B.2 applied to the set of variables Y, $B \in \mathbf{rel}(S, Y)^*$ and this proves

$$occ(\sigma, occ(\delta, v)) \cap U \in \mathbf{bin}(\mathbf{rel}(S, \{x\}), \mathbf{rel}(S, Y)^*).$$

- Otherwise: We have the case that $occ(\delta, v) \cap \text{vars}(t\sigma) \nsubseteq \text{vars}(\sigma(Y))$, i.e., $occ(\delta, v) \cap \text{vars}(\sigma(Z)) \neq \emptyset$. Therefore, there exists $w \in occ(\delta, v) \cap \text{vars}(\sigma(Z))$, and using Proposition B.2 we have that $occ(\sigma, w) \cap U \in \mathbf{rel}(S, Z)$. This implies that $B \in \{B_1 \cup \ldots B_n \cup C_1 \cup \ldots C_p \mid B_i \in \mathbf{rel}(S, Y), n \geqslant 0, C_i \in \mathbf{rel}(S, Z), p \geqslant 1\} = \mathbf{rel}(S, Z)^* \cup \mathbf{bin}(\mathbf{rel}(S, Y)^*, \mathbf{rel}(S, Z)^*)$. As a final result we have that

$$\begin{aligned}
occ(\sigma, occ(\delta, v)) \cap U &\in \mathbf{bin}(\mathbf{rel}(S, \{x\})^*, \mathbf{rel}(S, Z)^* \cup \mathbf{bin}(\mathbf{rel}(S, Y)^*, \mathbf{rel}(S, Z)^*)) \\
&= \mathbf{bin}(\mathbf{rel}(S, \{x\})^*, \mathbf{rel}(S, Z)^*) \cup \\
&\quad\ \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S, \{x\})^*, \mathbf{rel}(S, Z)^*), \mathbf{rel}(S, Y)^*),
\end{aligned}$$

  which proves the theorem. □

Lemma B.7

Let $[\sigma]_V \in ISubst_\sim$, $\theta \in ISubst$ such that $\text{vars}(\theta) \subseteq V$ and σ and θ unify. Assume given $U \subseteq V$ such that for each $x \in U$,

(1) x is free in $[\sigma]_V$;
(2) x is independent from $\text{vars}(\theta)$ in $[\sigma]_V$;
(3) if $x \in \text{dom}(\theta)$ then x is independent in $[\sigma]_V$.

If $\alpha_{\mathrm{Sh}}([\sigma]_V) \sqsubseteq_{\mathrm{Sh}} [S, V]$, then $\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_V, [\theta]_V)) \sqsubseteq_{\mathrm{Sh}} [\mathbf{u}^f_{\mathrm{Sh}}(S, U, \theta), V]$.

Proof

The proof is by induction on $|\mathrm{dom}(\theta)|$. Assume $|\mathrm{dom}(\theta)| = 0$; then $\theta = \epsilon$ and $\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_V, [\epsilon]_V)) = \alpha_{\mathrm{Sh}}([\sigma]_V) \sqsubseteq_{\mathrm{Sh}} [S, V] = [\mathbf{u}^f_{\mathrm{Sh}}(S, U, \epsilon), V]$.

Now assume that it holds for $|\mathrm{dom}(\theta)| \leqslant n$, and we show it holds for $|\mathrm{dom}(\theta)| = n + 1$, too. Let θ be $\theta' \uplus \{x/t\}$. We distinguish two cases: either $x \in U$ or $x \notin U$.

(1) ($x \in U$) By the definition of $\mathbf{u}^f_{\mathrm{Sh}}$ we have that

$$\begin{aligned}&\mathbf{u}^f_{\mathrm{Sh}}(S, U, \{x/t\} \uplus \theta')\\ &= \mathbf{u}^f_{\mathrm{Sh}}((S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t))) \cup \mathbf{bin}(\mathbf{rel}(S, x), \mathbf{rel}(S, t)), U \setminus \{x\}, \theta').\end{aligned}$$

Since $x \in U \cap \mathrm{dom}(\theta)$, by hypothesis x is free and independent in $[\sigma]_V$. Thus we can apply Proposition B.4, from which we obtain that

$$\begin{aligned}&\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_V, [x/t]_V))\\ &\quad\sqsubseteq_{\mathrm{Sh}} \quad [S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t)) \cup \mathbf{bin}(\mathbf{rel}(S, x), \mathbf{rel}(S, t)), V].\end{aligned}$$

Let $[\sigma']_V = \mathrm{mgu}([\sigma]_V, [x/t]_V)$ and $U' = U \setminus \{x\}$. We may assume without loss of generality that $\mathrm{vars}(\sigma) \cap U = \emptyset$, and we obtain $\sigma' = \mathrm{mgu}(\mathrm{Eq}(\sigma) \cup \{x = t\}) = \sigma \uplus \{x/t\sigma\}$. Given $u \in U'$, we have $\sigma'(u) = \sigma(u) = u \in \mathcal{V}$; hence u is free in $[\sigma]_V$. If $u \neq v \in \mathrm{vars}(\theta')$, then $v \neq x$ and therefore $u \notin \sigma'(v) = \sigma(v)$. Thus u is independent from $\mathrm{vars}(\theta')$ in $[\sigma']_V$. Moreover, if $u \in \mathrm{dom}(\theta')$, then $u \neq x$, $u \notin t$ and $u \notin vars(\sigma)$, and therefore $u \notin \mathrm{vars}(\sigma') \subseteq \mathrm{vars}(\sigma) \cup \mathrm{vars}(x = t)$. This means that u is independent in $[\sigma']_U$. Therefore, by inductive hypothesis,

$$\begin{aligned}&\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_V, [\theta]_V)) = \alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma']_V, [\theta']_V))\\ &\qquad\qquad\sqsubseteq_{\mathrm{Sh}} [\mathbf{u}^f_{\mathrm{Sh}}(S', U', \theta'), V] = [\mathbf{u}^f_{\mathrm{Sh}}(S, U, \theta), V],\end{aligned}$$

which concludes this part of the proof.

(2) ($x \notin U$) By the definition of $\mathbf{u}^f_{\mathrm{Sh}}$ we have that

$$\begin{aligned}&\mathbf{u}^f_{\mathrm{Sh}}(S, U, \{x/t\} \uplus \theta) = \mathbf{u}^f_{\mathrm{Sh}}((S \setminus (\mathbf{rel}(S, x) \cup \mathbf{rel}(S, t)))\\ &\quad \cup \mathbf{bin}(\mathbf{rel}(S, x), \mathbf{rel}(S, Y)^*) \cup \mathbf{bin}(\mathbf{rel}(S, x)^*, \mathbf{rel}(S, Z)^*)\\ &\quad \cup \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S, x)^*, \mathbf{rel}(S, Y)^*), \mathbf{rel}(S, Z)^*)), U \setminus \mathrm{vars}(\{x/t\}), \delta),\end{aligned}$$

where $Y = \mathrm{uvars}(t) \cap U$ and $Z = \mathrm{vars}(t) \setminus Y$. Since $Y \subseteq U$, for all $u \in Y$ and for all $v \in \mathrm{vars}(x = t)$ with $v \neq u$, it is the case that v and u do not share variables, i.e., $v \neq u \Rightarrow \sigma(u) \notin \sigma(v)$. Therefore $\sigma(u) \in \mathrm{uvars}(x\sigma = t\sigma)$. Then we can apply Proposition B.6 to obtain

$$\begin{aligned}&\alpha_{\mathrm{Sh}}([\sigma]_V, [x/t]_V]) \sqsubseteq_{\mathrm{Sh}} (S \setminus (\mathbf{rel}(S, t) \cup \mathbf{rel}(S, x)))\\ &\quad \cup \mathbf{bin}(\mathbf{rel}(S, x), \mathbf{rel}(S, Y)^*) \cup \mathbf{bin}(\mathbf{rel}(S, x)^*, \mathbf{rel}(S, Z)^*)\\ &\quad \cup \mathbf{bin}(\mathbf{bin}(\mathbf{rel}(S, x)^*, \mathbf{rel}(S, Z)^*), \mathbf{rel}(S, Y)^*), V].\end{aligned}$$

Again, assume $\mathrm{vars}(\sigma) \cap U = \emptyset$, $\sigma' = \mathrm{mgu}(\mathrm{Eq}(\sigma) \cup \{x = t\}) = \mathrm{mgu}(x\sigma = t\sigma) \circ \sigma$, and $U' = U \setminus \mathrm{vars}(\{x/t\})$. Given $u \in U'$ and $u \notin \mathrm{vars}(x = t)$ and since u by hypothesis does not share with any variable in $x = t$, we have

$u \notin \mathrm{vars}(\{x\sigma/t\sigma\})$. As a result $\sigma'(u) = \sigma(u) = u \in \mathcal{V}$. Moreover, for each variable v, $u \in \sigma'(v)$ iff $u \in \sigma(v)$. Therefore, if $v \in \mathrm{vars}(\theta')$ and $v \neq u$, v and u are independent in $[\sigma']_V$. Finally, if $u \in \mathrm{dom}(\theta')$, then $u \notin \mathrm{vars}(\sigma)$, which implies $u \notin \mathrm{vars}(\sigma')$. By inductive hypothesis we have

$$\begin{aligned}\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma]_V, [\theta]_V)) &= \alpha_{\mathrm{Sh}}(\mathrm{mgu}([\sigma']_V, [\theta']_V))\\ &\sqsubseteq_{\mathrm{Sh}} [\mathbf{u}^f_{\mathrm{Sh}}(S', U', \theta'), V] = [\mathbf{u}^f_{\mathrm{Sh}}(S, U, \theta), V],\end{aligned}$$

which proves the lemma. □

Theorem B.8
(Correctness of $\mathsf{unif}_{\mathrm{Sh}}$)
The unification operator $\mathsf{unif}_{\mathrm{Sh}}$ is correct w.r.t. $\mathsf{unif}_{\mathrm{Ps}}$.

Proof
Given $[\Delta, V] = \mathsf{unif}_{\mathrm{Ps}}([\Delta_1, V_1], \delta)$, we know that if $[\theta]_V \in \Delta$, then

$$[\theta]_V = \mathrm{mgu}([\theta_1]_{V_1}, [\delta]_{\mathrm{vars}(\delta)}) = \mathrm{mgu}([\theta_1]_{V_1}, [\epsilon]_{V_1 \cup \mathrm{vars}(\delta)}, [\delta]_{\mathrm{vars}(\delta)}).$$

Note that if $\alpha_{\mathrm{Sh}}([\theta_1]_{V_1}) \sqsubseteq_{\mathrm{Sh}} [S, V_1]$, then

$$\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\theta_1]_{V_1}, [\epsilon]_{V_1 \cup \mathrm{vars}(\delta)})) \sqsubseteq_{\mathrm{Sh}} [S \cup \{\{x\} \mid x \in \mathrm{vars}(\delta) \setminus V_1\}, V_1 \cup \mathrm{vars}(\delta)]$$

and each variable in $\mathrm{vars}(\delta) \setminus V_1$ is free and independent in $\mathrm{mgu}([\theta_1]_{V_1}, [\epsilon]_{V_1 \cup \mathrm{vars}(\delta)})$. Therefore, by applying Lemma B.7, we obtain

$$\alpha_{\mathrm{Sh}}([\theta]_V) \sqsubseteq_{\mathrm{Sh}} \mathsf{unif}_{\mathrm{Sh}}([S, V_1], \delta).$$

The theorem follows by the pointwise extension of α_{Sh} to elements of `Psub`. □

Appendix C: Optimality of forward unification

We first introduce some notations. Given $[S_1, U_1] \in$ `Sharing` and $\theta \in \mathit{ISubst}$, let $\mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta) = [S, U_1 \cup \mathrm{vars}(\theta)]$ and $X \in S$. To ease the notation, let us define $U_2 = \mathrm{vars}(\theta) \setminus U_1$, $S_2 = \{\{x\} \mid x \in U_2\}$, $U = U_1 \cup U_2$, $X_1 = X \cap U_1$, and $X_2 = X \cap U_2$.

We begin by checking some properties of the unification algorithm in $\mathbf{u}^f_{\mathrm{Sh}}$. To simplify the notation, in the rest of this section we will use a slightly modified version of the operator $\mathbf{u}^f_{\mathrm{Sh}}$ that uses the rule $\mathbf{u}^f_{\mathrm{Sh}}(T, V, \epsilon) = (T, V)$ (instead of the original rule $\mathbf{u}^f_{\mathrm{Sh}}(T, V, \epsilon) = T$). The only consequence of this modification is that the new operator returns a pair whose first argument is the same as in the original operator and whose second argument is a set of variables guaranteed to be free after the unification.

Remark C.1
Given $(T', V') = \mathbf{u}^f_{\mathrm{Sh}}(T, V, \theta)$ the following properties are easily checked from the definition:

(1) $V' \subseteq V$;
(2) if $x \in V' \cap \mathrm{rng}(\theta)$ and $x \in \theta(v)$, then $v \in V$;
(3) $\mathbf{u}^f_{\mathrm{Sh}}(T, V, \theta \uplus \theta') = \mathbf{u}^f_{\mathrm{Sh}}(T', V', \theta')$.

Let $[H, U] = \alpha_{\mathrm{Sh}}([\theta]_U)$. We want to prove that each $X \in S$ is obtained as union of a number of sharing groups in H. However, these sharing groups cannot be joined freely but only according to some conditions.

Lemma C.2
For each $X \in S$, either $X \in H$ or there are $B_1, \dots, B_k \in H$ such that $\cup_{i \leqslant k} B_i = X$ and for each $i \leqslant k$, $B_i \cap U_1 \neq \emptyset$.

Proof
The proof proceeds by induction on the number of bindings n in θ. If $n = 0$, then $\theta = \epsilon$, $S = S_1 \cup S_2$, and $H = \{\{x\} \mid x \in U_1 \cup U_2\}$. If $X \in S_2$, then $X = \{x\}$ for some $x \in U_2$, i.e., $X \in H$. Otherwise, if $X \in S_1$, then $X = \bigcup\{\{x\} \mid x \in X\}$. Since $x \in \mathrm{vars}(S_1)$ entails $x \in U_1$, we may take as B_i's the singletons $\{x\}$ for each $x \in X$, and we have the required result.

If $n \neq 0$, then $\theta = \theta' \uplus \{x/t\}$ and $\mathbf{u}^f_{\mathrm{Sh}}(S_1 \cup S_2, U_2, \theta) = \mathbf{u}^f_{\mathrm{Sh}}(T, V, \{x/t\})$, where $(T, V) = \mathbf{u}^f_{\mathrm{Sh}}(S_1 \cup S_2, U_2, \theta')$. Let $[H', U] = \alpha_{\mathrm{Sh}}([\theta']_U)$. We distinguish the cases $x \in V$ and $x \notin V$.

Assume $x \in V$. If $X \in T \setminus (\mathbf{rel}(T, t) \cup \mathbf{rel}(T, x))$, then $X \cap \mathrm{vars}(\{x/t\}) = \emptyset$. By inductive hypothesis, $X = B_1 \cup \dots \cup B_h$, where each $B_j \in H'$. Since $B_j \cap \mathrm{vars}(\{x/t\}) = \emptyset$, we have $B_j \in H$, and therefore the property is satisfied. Otherwise, $X = A_1 \cup A_2$, where $A_1 \in \mathbf{rel}(T, x)$ and $A_2 \in \mathbf{rel}(T, t)$. Note that since $x \notin \mathrm{vars}(\theta')$, $\mathbf{rel}(H', x) = \{\{x\}\}$. Since $\{x\} \cap U_1 = \emptyset$, it is not possible to join $\{x\}$ with any other sharing group in H', and therefore $\mathbf{rel}(T, x) = \{\{x\}\}$ and $A_1 = \{x\}$. Now assume, without loss of generality, $A_2 \in \mathbf{rel}(T, y)$, with $y \in \mathrm{vars}(t)$. By inductive hypothesis $A_2 = C_1 \cup \dots \cup C_h$ with each $C_j \in H'$. First of all, note that for each j, either $C_j \cap \mathrm{vars}(\{x/t\}) = \emptyset$, which entails $C_j \in H$, or $C_j = occ(\theta', w)$ for some $w \in \mathrm{vars}(t)$, which entails $\{x\} \cup C_j = occ(\theta, w) \in H$. Therefore, it is possible to take $k = h$ and B_j equal to either C_j or $C_j \cup \{x\}$ so that $B_j \in H$. Since there is at least one index l such that $y \in C_l$, $C_l = occ(\theta', y)$ and $x \in B_l$. Therefore $\cup_j B_j = X$. Moreover, either $h = 1$ or $h > 1$ and $C_j \cap U_1 \neq \emptyset$ for each $j \leqslant h$.

Now assume $x \notin V$. If $X \in T \setminus (\mathbf{rel}(T, t) \cup \mathbf{rel}(T, x))$, then $X \cap \mathrm{vars}(\{x/t\}) = \emptyset$ and everything is as for the case $x \in V$. Otherwise, the three cases in the definition of $\mathbf{u}^f_{\mathrm{Sh}}$ may be subsumed saying that $X = A_1 \cup A_2$, where $A_1 \in \mathbf{rel}(S, x)^*$ and $A_2 \in \mathbf{rel}(S, t)^*$. Assume, by inductive hypothesis, that $A_1 = C^1_1 \cup \dots \cup C^1_h$, where each $C^1_j \in H'$ and $A_2 = C^2_1 \cup \dots \cup C^2_l$, where each $C^2_j \in H'$. Since $x \notin \mathrm{vars}(\theta')$, $\mathbf{rel}(H', x) = \{\{x\}\}$. Therefore there exists C^1_j such that $C^1_j = \{x\}$. We assume without loss of generality that $C^1_1 = \{x\}$. As for the case with $x \in V$, we may define B^2_j equal to either C^2_j or $C^2_j \cup \{x\}$ so that $B^2_j \in H$. The same holds for all the elements of the kind C^1_j for $j > 1$. Moreover, there is at least one j such that $C^2_j = occ(\theta', y)$ for some $y \in \mathrm{vars}(t)$, i.e., such that $x \in B^2_j$. Then, we have a collection of elements B^1_j and B^2_j such that each $B^1_j, B^2_j \in H$ and whose union gives X. We only need to prove that $B^1_j \cap U_1 \neq \emptyset$ and $B^2_j \cap U_1 \neq \emptyset$ for each j. Note that if $C^2_j \cap U_1 \neq \emptyset$, then $B^2_j \cap U_1 \neq \emptyset$. Assume $C^2_j \cap U_1 = \emptyset$. By inductive hypothesis, this happens if $C^2_j \in \mathbf{rel}(S, t)$. (Otherwise C^2_j is obtained by joining more than one element in H', and therefore it must contains some variable in U_1.) Thus, there exists $y \in \mathrm{vars}(t)$ such that $y \in C^2_j$, and therefore $B^2_j = C^2_j \cup \{x\}$ and $B^2_j \cap U_1 \neq \emptyset$. In the same way, if $C^1_j \cap U_1 \neq \emptyset$ the same holds for B_j^1. Note that given C_j^1, by inductive hypothesis either $C_j^1 \notin \mathbf{rel}(S,x)$, and therefore $C_j^1 \cap U_1 \neq \emptyset$, or $C_j^1 \in \mathbf{rel}(S,x)$, and therefore $x \in C_j^1$, which entails again $C_j^1 \cap U_1 \neq \emptyset$. □

Corollary C.3
$X = \{x \mid \mathrm{vars}(\theta(x)) \cap X \neq \emptyset\}$.

Proof
By Lemma C.2 we know $X = B_1 \cup \ldots \cup B_N$ with $B_i \in H$. If $x \in X$, then $x \in B_i$ for some $i \leqslant N$. Assume $B_i = occ(\theta, w)$. Then $w \in B_i \subseteq X$ and $w \in \mathrm{vars}(\theta(x)) \cap X$. In the opposite direction, assume $z \in \mathrm{vars}(\theta(x)) \cap X$. Since there is only one sharing group B in H such that $z \in B$, namely, $B = occ(\theta, z)$, it must be the case that $B = B_j$ for some $j \in \{1, \ldots, N\}$ and therefore $x \in B_j \subseteq X$. □

Lemma C.4
For each $X \in S$, X is θ-connected.

Proof
First note that if X is θ-connected and $Y \subseteq U_2$, then given $\theta' = \theta \uplus \theta''$ it holds that $X \cup Y$ is θ'-connected.

The proof is by induction on the number of bindings in θ. If $\theta = \epsilon$ there is nothing to prove, since $X \in S_1 \cup S_2$, and thus $X_1 \in S_1$.

Let $\theta = \theta' \uplus \{x/t\}$, $[H', U] = \alpha_{\mathrm{Sh}}([\theta']_U)$, and $(S, V') = \mathbf{u}^f_{\mathrm{Sh}}(T, V, \{x/t\})$, where $\mathbf{u}^f_{\mathrm{Sh}}(S_1 \cup S_2, U_2, \theta') = (T, V)$.

We distinguish two cases according to the fact that $x \in V$ or not. Consider the case $x \in V$, which implies $x \in U_2$. By hypothesis $x \notin \mathrm{vars}(\theta')$; therefore, by Lemma C.2, $\mathbf{rel}(T, x) = \{\{x\}\}$. Therefore S is obtained by joining to each $Q \in \mathbf{rel}(T, t)$ the new sharing group $\{x\}$ and removing $\{x\}$ from T. It happens that each $Q \in S$ is θ-connected since (1) either $Q \in T$ (2) or $Q = Q' \cup \{x\}$ for $Q' \in T$ and $x \in U_2$. In the first case, Q is θ'-connected by inductive hypothesis; hence it is also θ connected, and the thesis follows. In the latter case, Q' is θ'-connected, and thus $Q' \cup \{x\}$ is θ-connected, since $x \in U_2$.

The other case is when $x \notin V$. If we take $Q \in S$ and assume $Q \in T \setminus (\mathbf{rel}(T,x) \cup \mathbf{rel}(T,t))$, then it is θ'-connected by inductive hypothesis, and thus it is θ-connected. Otherwise, take $Q = Q_1 \cup Q_2$ with $Q_1 \in \mathbf{rel}(T,x)$ and $Q_2 \in \mathbf{rel}(T,Y)^*$, where $Y = \mathrm{uvars}(t) \cap V$. Given $y \in Y$, since $y \in V$, for each binding x'/t' in θ', if $y \in \mathrm{vars}(t')$, then $x' \in U_2$ (see Remark C.1). Therefore $\mathbf{rel}(H, y) = \{K\}$ with $K \subseteq U_2$, and by Lemma C.2, the same holds for $\mathbf{rel}(T, y)$. This means $Q_2 \subseteq U_2$. Thus $Q \cap U_1 = Q_1 \cap U_1$. Since Q_1 is θ'-connected by inductive hypothesis, it follows that Q_1 is θ-connected.

Now, take $Q_1 \in \mathbf{rel}(T,x)^*$ and $Q_2 \in \mathbf{rel}(T,Z)^*$, where $Z = \mathrm{vars}(t) \setminus Y$. Thus $Q_1 = A_1 \cup \ldots \cup A_k$ with $A_i \in \mathbf{rel}(T,x)$. By inductive hypothesis, A_i is θ'-connected, and therefore it is θ-connected. It follows that for each $i \leqslant k$ there exist $B_1^i, \ldots, B_{k_i}^i \in S_1$ such that $\cup_{j \leqslant k_i} B_j^i = A_i \cap U_1$ and $B_{j_1}^i \mathscr{R}^*_{\theta A_i} B_{j_2}^i$ for $j_1, j_2 \leqslant k_i$. The same holds for $Q_2 = C_1 \cup \ldots \cup C_h$ with $C_i \in \mathbf{rel}(T,Z)$: for any $C_i \cap U_1 \neq \emptyset$ we have that $C_i \cap U_1 = \cup_{j \leqslant h_i} D_j^i$ with $D_{j_1}^i \mathscr{R}^*_{\theta C_i} D_{j_2}^i$ for all $j_1, j_2 \leqslant h_i$.

We need to show that given any B^i_m and D^j_n, $B^i_m \mathscr{R}^*_{\theta Q} D^j_n$. Actually, it is enough to show that for each $i \leqslant k, j \leqslant h$ such that $C_j \cap U_1 \neq \emptyset$, there are m, n such that $B^i_m \mathscr{R}_{\theta Q} D^j_n$.

Since $x \in A_i$ and $x \in U_1$, without loss of generality we may assume that $x \in B^i_1$. On the other hand, although $\text{vars}(t) \cap C_j \neq \emptyset$, we cannot infer that there exists any D^j_n such that $\text{vars}(t) \cap D^j_n \neq \emptyset$, since it may well happen that $\text{vars}(t) \cap C_j \subseteq U_2$ even though $U_1 \cap C_j \neq \emptyset$.

Assume $C_j \in \mathbf{rel}(T, z)$ for some $z \in Z \cap U_1$. Then, we may assume without loss of generality that $z \in D^j_1$, and $B^i_1 \mathscr{R}_{\theta Q} D^j_1$ follows from the definition of $\mathscr{R}_{\theta Q}$, being $z \in Q$. Otherwise, $C_j \in \mathbf{rel}(T, z)$ for some $z \in Z \cap U_2$. By applying Lemma C.2, we have $C_j = E_1 \cup \ldots \cup E_p$ with $E_i \in H'$ and $E_i \cap U_1 \neq \emptyset$. (This holds even if $p = 1$, since $C_j \cap U_1 \neq \emptyset$.) Since $\mathbf{rel}(H', z) = \{occ(\theta', z)\}$, $occ(\theta', z) \cap U_1 \neq \emptyset$; i.e., there exists $z' \in U_1$ such that $z \in \text{vars}(\theta'(z'))$. Then $z' \in C_j$, and we may assume, without loss of generality, that $z' \in D^j_1$. Again, we have $B^i_1 \mathscr{R}_{\theta Q} D^j_1$ by the definition of $\mathscr{R}_{\theta Q}$.

Observe that if $Q_2 \cap U_1 \neq \emptyset$, by symmetry and transitivity, this alone proves that $B^i_m \mathscr{R}^*_{\theta X} B^{i'}_{m'}$ and $D^j_n \mathscr{R}^*_{\theta Q} D^{j'}_{n'}$ for each i, m, i', m' and j, n, j', n'. Otherwise, there is no D^j_n, and we need to prove in other ways that $B^i_m \mathscr{R}^*_{\theta Q} B^{i'}_{m'}$. Since $Q_2 \cap U_1 = \emptyset$, $C_i \subseteq U_2$ for each i. This means $C_i = occ(\theta', y)$ for some $y \in U_2$, and since $C_i \subseteq U_2$ it follows immediately that $y \in V$. Then, since $y \in Z$, it must be the case that $y \notin \text{uvars}(t)$, and therefore $B^i_1 \mathscr{R}_{\theta Q} B^{i'}_1$ by the definition of $\mathscr{R}_{\theta Q}$.

It remains the case $Q = Q_1 \cup Q_2 \cup Q_3$ with $Q_1 \in \mathbf{rel}(T, x)^*$, $Q_2 \in \mathbf{rel}(T, Y)^*$, and $Q_3 \in \mathbf{rel}(T, Z)^*$. However, this is a trivial corollary of the previous two cases, since we know that $Q_1 \cup Q_3$ is θ-connected and $Q_2 \subseteq U_2$. □

Fixed $X \in S$, our aim is to provide a substitution δ with $\alpha_{\text{Sh}}([\delta]_{U_1}) \sqsubseteq [S_1, U_1]$ and $\alpha_{\text{Sh}}(\text{mgu}([\delta]_{U_1}, [\theta]_U)) \sqsupseteq [\{X\}, U]$. By Lemma C.4, $X_1 = B_1 \cup \ldots \cup B_n$ with $B_i \in S_1$ and $B_i \mathscr{R}^*_{\theta X} B_j$ for each $i, j \leqslant n$ (where $X_1 = X \cap U_1$). We let $K_1 = \{B_1, \ldots, B_n\}$. We now want to define a substitution δ such that $\alpha_{\text{Sh}}([\delta]_{U_1}) = [K_1, U_1]$. For each sharing group $B \in K_1$, let us consider a fresh variable w_B. Let $W = \{w_B \mid B \in K_1\}$. For each variable x, let $B_x = \{B^1_x, \ldots, B^k_x\}$ be the set $\mathbf{rel}(K_1, x)$. Let N be the maximum cardinality of all the B_x for $x \in X_1$, i.e., $N = \max_{x \in X_1} |B_x|$. For each $x \in X_1$, we define two terms:

$$s_x = t(\underbrace{c(w_{B^1_x}, w_{B^1_x}), c(w_{B^2_x}, w_{B^2_x}), \ldots, c(w_{B^k_x}, w_{B^k_x})}_{k = |B_x| \text{ times}}, \underbrace{c(w_{B^1_x}, w_{B^1_x}), \ldots, c(w_{B^1_x}, w_{B^1_x})}_{N - |B_x| \text{ times}}).$$

$$s'_x = t(\underbrace{c(w_{B^1_x}, w_{B^2_x}), c(w_{B^2_x}, w_{B^3_x}), \ldots, c(w_{B^k_x}, w_{B^1_x})}_{k = |B_x| \text{ times}}, \underbrace{c(w_{B^1_x}, w_{B^1_x}), \ldots, c(w_{B^1_x}, w_{B^1_x})}_{N - |B_x| \text{ times}}).$$

Note that if $N = 0$, then $X_1 = \emptyset$ and s_x, s'_x are undefined for any variable x.

We introduce the following notation: given a term t we distinguish different occurrences of the same variable by calling (y, n) the nth occurrence of a variable y in t, where the order is lexicographic. For instance, a term $f(x, g(y, y, x))$ can be seen as the term $f((x, 1), g((y, 1), (y, 2), (x, 2)))$. For each $y \in \text{vars}(\theta(U_1)) \cap X$, we choose a variable $x_y \in U_1$ such that $y \in \theta(x_y)$. Let a be a constant. We are now ready to define the substitution δ in the following way: for each variable $x \in U_1$, $\delta(x)$ is the same as $\theta(x)$ with the difference that each occurrence (y, i) of a variable $y \in \theta(x)$ is replaced by $t_{x,y,i}$ defined as

- $t_{x,y,i} = a$ if $y \notin X$, else
- $t_{x,y,i} = s_x$ if $x = x_y$ and $i = 1$, or
- $t_{x,y,i} = s'_x$ otherwise.

Note that by Corollary C.3, if $x \in X_1$, then $\theta(x)$ is not ground. Therefore, by construction, $\text{dom}(\delta) = U_1$ and $\text{rng}(\delta) = W$. It is easy to check that $\alpha_{\text{Sh}}([\delta]_{U_1}) = [K_1, U_1]$, since given a variable w_B, it appears in $\delta(x)$ iff $x \in B$ and therefore $occ(\delta, w_B) \cap U_1 = B$. For all the other variables $occ(\delta, v) = \emptyset$ if $v \in U_1$ and $occ(\delta, v) = \{v\} \not\subseteq U_1$ otherwise. Let us compute the value of $\text{mgu}([\delta]_{U_1}, [\theta]_U)$.

Lemma C.5

$$\text{mgu}(\delta, \theta) = \text{mgu}\{w_1 = w_2 \mid w_1, w_2 \in W\} \circ \rho \circ \theta,$$

where $\rho = \{v/s_{x_v} \mid v \in \text{vars}(\theta(U_1)) \cap X\} \cup \{v/a \mid v \in \text{vars}(\theta(U_1)) \setminus X\}$.

Proof

Since $t_{x_v,v,1} = s_{x_v}$, by using the properties of equation sets it follows that

$$\begin{aligned} \text{mgu}(\delta, \theta) &= \text{mgu}(\{v = t_{x,v,i} \mid x \in U_1, (v, i) \text{ is an occurrence of } v \text{ in } \theta(x)\}) \circ \theta \\ &= \text{mgu}(E) \circ \rho \circ \theta, \end{aligned}$$

where $E = \{t_{x_v,v,1} = t_{x',v,j} \mid x' \in U_1, (v, j) \text{ is an occurrence of } v \text{ in } \theta(x')\}$. Let us define a relation between variables:

$$v\mathscr{R}'u \iff \exists y \in \text{vars}(\theta(v)) \cap X.\ u = x_y \wedge (u = v \Rightarrow y \notin \text{uvars}(\theta(v)))\}.$$

Note that $\mathscr{R}'$ is not a symmetric relationship. Moreover, it depends on θ and X, just as $\mathscr{R}_{\theta X}$. However, since in this proof θ and X are fixed, we decided to omit the indexes in order to simplify notation. By exploiting the above definition, we can rewrite $\text{mgu}(E)$ as follows:

$$\text{mgu}(E) = \text{mgu}(\{s'_v = s_u \mid v, u \in X_1, v\mathscr{R}'u\}). \tag{C 1}$$

The above characterization shows that $\text{Eq}(\delta) \cup \text{Eq}(\theta)$ is solvable, since s_u and s'_v are terms that unify by construction. Moreover, note that

$$\text{mgu}\{s_u = s'_v\} = \text{mgu}\{w_B = w_{B'} \mid B \in B_u \wedge B' \in B_v\}.$$

We want to prove that $\text{mgu}\{s'_v = s_u \mid v, u \in X_1, v\mathscr{R}'u\} = \text{mgu}\{w_1 = w_2 \mid w_1, w_2 \in W\}$. It is obvious that $\text{mgu}\{s'_v = s_u \mid v, u \in X_1, v\mathscr{R}'u\} = \text{mgu}\{w_B = w_{B'} \mid v, u \in X_1.\ B \in B_v, B' \in B_u, v\mathscr{R}'u\} = \text{mgu}\{w_B = w_{B'} \mid B\hat{\mathscr{R}}B'\}$, where $\hat{\mathscr{R}}$ is the relation on $K_1 \times K_1$ given by

$$B\hat{\mathscr{R}}B' \iff \exists x, y \in X_1.\ B \in B_x \wedge B' \in B_y \wedge x\mathscr{R}'y.$$

Since equality is transitive and reflexive, we know that

$$\text{mgu}\{w_B = w_{B'} \mid B\hat{\mathscr{R}}B'\} = \text{mgu}\{w_B = w_{B'} \mid B\hat{\mathscr{R}}^*B'\},$$

where $\hat{\mathscr{R}}^*$ is the symmetric and transitive closure of $\hat{\mathscr{R}}$. We now prove that $\hat{\mathscr{R}} \subseteq \mathscr{R}_{\theta X} \subseteq \hat{\mathscr{R}}^*$, from which the thesis follows by Lemma C.4.

If $B\hat{\mathscr{R}}B'$ there are $x, y \in X_1$ such that $B \in B_x \wedge B' \in B_y \wedge x\mathscr{R}'y$. However $B \in B_x$ iff $x \in B \in S_1$ and $B' \in B_y$ iff $y \in B' \in S_1$. Now, assume $z \in \mathrm{vars}(\theta(x)) \cap X$ and $y = x_z$. Then $z \in \mathrm{vars}(\theta(x)) \cap \mathrm{vars}(\theta(y)) \cap X$, and this proves that $B\mathscr{R}_{\theta X}B'$. On the other side, assume $B\mathscr{R}_{\theta X}B'$; i.e., there are $x \in B$, $y \in B'$, $z \in \mathrm{vars}(\theta(x)) \cap \mathrm{vars}(\theta(y)) \cap X$ such that $x = y \implies z \notin \mathrm{uvars}(\theta(x))$. Since $x \in B$ and $y \in B'$, $B \in B_x$ and $B' \in B_y$. Since $z \in \mathrm{vars}(\theta(U_1)) \cap X$, x_z is defined and $B_{x_z} \neq \emptyset$. Assume that $x = y = x_z$. Then $z \notin \mathrm{uvars}(\theta(x))$ and thus $x\mathscr{R}'y$ and $B\hat{\mathscr{R}}B'$. Otherwise, we may assume without loss of generality that $x \neq x_z$. If $y = x_z$ then $x\mathscr{R}'y$ and thus $B\hat{\mathscr{R}}B'$. If $y \neq x_z$ we can choose any $B'' \in B_{x_z}$. We know that $x\mathscr{R}'x_z$, $y\mathscr{R}'x_z$, and thus it holds that $B\hat{\mathscr{R}}B''$ and $B'\hat{\mathscr{R}}B''$, from which $B\hat{\mathscr{R}}^*B'$ follows. The case $y \neq x_z$ is symmetric. □

Proposition C.6

$$\alpha_{\mathrm{Sh}}(\mathrm{mgu}([\delta]_{U_1}, [\theta]_U)) \sqsupseteq_{\mathrm{Sh}} [\{X\}, U].$$

Proof

First of all, note that $\mathrm{mgu}([\delta]_{U_1}, [\theta]_U) = [\mathrm{mgu}(\delta, \theta)]_U$ since $\mathrm{vars}(\theta) \subseteq U$. We proceed with two different proofs when $W = \emptyset$ and $W \neq \emptyset$. If $W \neq \emptyset$, then according to Lemma C.5, we can choose $\bar{w} \in W$ and define the substitution $\sigma = \{w'/\bar{w} \mid \bar{w} \neq w' \in W\} = \mathrm{mgu}(E)$. It only remains to prove that $occ(\sigma \circ \rho \circ \theta, \bar{w}) \cap U = X$.

It follows easily that $occ(\sigma \circ \rho \circ \theta, \bar{w}) = occ(\rho \circ \theta, W) = occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X) \cup W) = occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X) \cup W$. Since $U \cap W = \emptyset$ it follows that $occ(\sigma \circ \rho \circ \theta, \bar{w}) \cap U = occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X)$.

By definition, $occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X) = \{y \mid \mathrm{vars}(\theta(y)) \cap \mathrm{vars}(\theta(U_1)) \cap X \neq \emptyset\}$. Thus, for any of such y, we have that $\mathrm{vars}(\theta(y)) \cap X \neq \emptyset$ and thus, by Corollary C.3, $y \in X$. It follows that $occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X) \subseteq X$. For the opposite direction, by Lemma C.2 there exist $B_1, \ldots, B_k \in H$ such that $\cup B_i = X$ and $B_i \cap U_1 \neq \emptyset$ for each i. Since $B_i \in H$, there exists v such that $B_i = occ(\theta, v)$. Moreover, $v \in X$, since $v \in B_i$ by the definition of occ and $\theta(v) = v$. Since $B_i \cap U_1 \neq \emptyset$ it follows that there exists $y \in B_i \cap U_1$ such that $v \in \theta(y) \subseteq \theta(U_1)$ and thus $B_i \subseteq occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X)$. Thus $X \subseteq occ(\theta, \mathrm{vars}(\theta(U_1)) \cap X)$.

When $W = \emptyset$, $\mathrm{mgu}(E) = \epsilon$ and $X = X_2$. In this case, by Lemma C.2, $X_2 = occ(\theta, x)$ for some $x \in U_2$. Since $X_2 \cap U_1 = \emptyset$, $x \notin \mathrm{vars}(\theta(U_1))$, i.e., $x \notin \mathrm{dom}(\rho)$ and therefore $occ(\rho \circ \theta, x) = occ(\theta, x) = X_2$. □

Note that in this proof, we worked with a signature endowed with a constant a and term symbols c and t of arity two and N respectively. Actually, it is evident that the proof may be easily rewritten for the case in which the signature has a constant and a symbol of arity at least two. Given s of arity n, we may replace in δ a term $t(t_1, \ldots, t_N)$ with $c(t_1, c(t_2, c(\ldots, t_N)))$. Then, we replace $c(t_1, t_2)$ with $s(t_1, t_2, a, a, \ldots, a)$, where a is repeated $n - 2$ times.

Theorem C.7

$\mathbf{U}^f_{\mathrm{Sh}}$ is well defined, correct, and optimal w.r.t. $\mathbf{U}^f_{\mathrm{Ps}}$.

Proof
By equation (31), we need to prove that

$$\begin{aligned}&\pi_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Sh}}(\rho([S_1, U_1]), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2)) = \\ &\qquad \alpha_{\mathrm{Sh}}(\pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho(\gamma_{\mathrm{Ps}}([S_1, U_1])), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2))).\end{aligned}$$

By Theorems 5.3 and 5.4, we know that π_{Sh} is correct and complete and that abstract renaming is correct and γ-complete. Moreover, by Theorem 6.16, abstract unification $\mathsf{unif}_{\mathrm{Sh}}$ is optimal. We have the following equalities:

$$\begin{aligned}&\alpha_{\mathrm{Sh}}(\pi_{\mathrm{Ps}}(\mathsf{unif}_{\mathrm{Ps}}(\rho(\gamma_{\mathrm{Ps}}([S_1, U_1])), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2))) \\ &\quad = \pi_{\mathrm{Sh}}(\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}(\rho(\gamma_{\mathrm{Ps}}([S_1, U_1])), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2))) && \text{(by Theorem 5.3)} \\ &\quad = \pi_{\mathrm{Sh}}(\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}(\gamma_{\mathrm{Ps}}(\rho([S_1, U_1])), \mathrm{mgu}(\rho(A_1) = A_2))), \mathrm{vars}(A_2)) && \text{(by Theorem 5.4)} \\ &\quad = \pi_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Sh}}(\rho([S_1, U_1]), \mathrm{mgu}(\rho(A_1) = A_2)), \mathrm{vars}(A_2)) && \text{(by Theorem 6.16).}\end{aligned}$$

Thus $\mathbf{U}^f_{\mathrm{Sh}}$ is correct and optimal w.r.t. $\mathbf{U}^f_{\mathrm{Ps}}$. The fact that it is well defined (i.e., it does not depend on the choice of the renaming ρ) is a direct consequence of optimality. □

Appendix D: Matching

Theorem D.1
(Correctness of $\mathsf{match}_{\mathrm{Sh}}$)
$\mathsf{match}_{\mathrm{Sh}}$ is correct w.r.t. $\mathsf{match}_{\mathrm{Ps}}$.

Proof
Consider $[\Theta_i, U_i] \sqsubseteq_{\mathrm{Ps}} \gamma_{\mathrm{Sh}}([S_i, U_i])$ for $i \in \{1,2\}$ and $[\sigma]_{U_1 \cup U_2} \in \mathsf{match}_{\mathrm{Ps}}([\Theta_1, U_1], [\Theta_2, U_2])$. We need to prove that

$$\alpha_{\mathrm{Sh}}([\sigma]_{U_1 \cup U_2}) \in \mathsf{match}_{\mathrm{Sh}}([S_1, U_1], [S_2, U_2]).$$

Assume $[\sigma] = \mathrm{mgu}([\sigma_1], [\sigma_2])$ with $[\sigma_1] \in \Theta_1$ and $[\sigma_2] \in \Theta_2$. Let σ_1 and σ_2 be two canonical representatives for $[\sigma_1]$ and $[\sigma_2]$ such that $\mathrm{vars}(\sigma_1) \cap \mathrm{vars}(\sigma_2) = U_1 \cap U_2$. If $\sigma_1 \preceq_{U_1 \cap U_2} \sigma_2$, there exists $\delta \in \mathit{Subst}$ such that $\sigma_1(x) = \delta(\sigma_2(x))$ for each $x \in U_1 \cap U_2$. We may assume, without loss of generality, that $\mathrm{dom}(\delta) = \mathrm{vars}(\sigma_2(U_1 \cap U_2))$. Now, the following equalities hold:

$$\begin{aligned}\sigma &= \mathrm{mgu}(\mathrm{Eq}(\sigma_2), \mathrm{Eq}(\sigma_1)) \\ &= \mathrm{mgu}(\{\sigma_2(x) = \sigma_2(\sigma_1(x)) \mid x \in U_1\}) \circ \sigma_2 \\ &= \mathrm{mgu}(\{x = \sigma_1(x) \mid x \in U_1 \setminus U_2\} \cup \{\sigma_1(x) = \sigma_2(x) \mid x \in U_1 \cap U_2\}) \circ \sigma_2 \\ &\quad \text{(by partitioning } \mathrm{dom}(\sigma_2)\text{, since } \sigma_2(\sigma_1(x)) = \sigma_1(x) \text{ for } x \in U_1) \\ &= \mathrm{mgu}(\{x = \sigma_1(x) \mid x \in U_1 \setminus U_2\}) \circ \delta \circ \sigma_2 \\ &\quad \text{(since } \sigma_1(x) = \delta(\sigma_2(x)) \text{ and } \mathrm{dom}(\delta) = \mathrm{vars}(\sigma_2(U_1 \cap U_2))) \\ &= \sigma_{1|U_1 \setminus U_2} \circ \delta \circ \sigma_2 \\ &= \sigma_{1|U_1 \setminus U_2} \uplus (\delta \circ \sigma_2).\end{aligned} \tag{D 1}$$

Now, given a variable v, by Lemma B.1, $\mathit{occ}(\sigma, v) \cap (U_1 \cup U_2) = (\mathit{occ}(\sigma_{1|U_1 \setminus U_2}, v) \cap U_1) \cup (\mathit{occ}(\sigma_2, \mathit{occ}(\delta, v)) \cap U_2)$. We want to prove that $\mathit{occ}(\sigma, v) \cap (U_1 \cup U_2) \in \mathsf{match}_{\mathrm{Sh}}([S_1, U_1], [S_2, U_2])$.

Since $\mathrm{dom}(\sigma) = U_1 \cup U_2$, we may assume that $v \notin U_1 \cup U_2$; otherwise $occ(\sigma, v) \cap (U_1 \cup U_2) = \emptyset$. We recall that $S_1' = \{B \in S_1 \mid B \cap U_2 = \emptyset\}$ and $S_1'' = S_1 \setminus S_1'$, $S_2' = \{B \in S_2 \mid B \cap U_1 = \emptyset\}$ and $S_2'' = S_2 \setminus S_2'$, according to Definition 7.1. We distinguish two cases:

- $v \notin \mathrm{rng}(\delta)$, which implies $v \notin \mathrm{rng}(\sigma_{1|U_2})$. Note that if $v \in \mathrm{dom}(\delta)$, then $occ(\sigma_2, occ(\delta, v)) = \emptyset \in S_2'$; otherwise $occ(\sigma_2, occ(\delta, v)) = occ(\sigma_2, v) \in S_2'$. So, it always holds that $occ(\sigma_2, occ(\delta, v)) \in S_2'$. We now distinguish some subcases. If $v \in \mathrm{rng}(\sigma_1)$, then $occ(\sigma_{1|U_1 \setminus U_2}, v) = occ(\sigma_1, v)$. Moreover, since $v \in \mathrm{rng}(\sigma_1)$, $v \notin \mathrm{vars}(\sigma_2)$ and thus $occ(\sigma_2, v) = \{v\}$. We have that $occ(\sigma, v) \cap (U_1 \cup U_2) = occ(\sigma_1, v) \in S_1'$. Otherwise, if $v \in \mathrm{rng}(\sigma_2)$, then $v \notin \mathrm{vars}(\sigma_1)$ and $occ(\sigma_1, v) = \{v\}$. Therefore $occ(\sigma, v) \cap (U_1 \cup U_2) = occ(\sigma_2, occ(\delta, v)) \in S_2'$. Otherwise, if $v \notin \mathrm{rng}(\sigma_1) \cup \mathrm{rng}(\sigma_2)$, then $occ(\sigma, v) \cap (U_1 \cup U_2) = \emptyset$.
- $v \in \mathrm{rng}(\delta)$. We want to prove that $occ(\sigma, v) = X_1 \cup X_2$, where $X_1 = occ(\sigma_1, v)$ and $X_2 = occ(\sigma_2, occ(\delta, v))$ enjoy the following properties: $X_1 \in S_1''$, $X_2 \in S_2''^*$, $X_1 \cap U_2 = X_2 \cap U_1$. First of all, note that $occ(\sigma_{1|U_1 \setminus U_2}, v) \cap U_1 = X_1 \setminus U_2$. Moreover, $occ(\sigma_2, occ(\delta, v)) \cap U_1 = occ(\sigma_{2|U_1}, occ(\delta, v)) \cap U_1$, which in turn is equal to $occ(\delta \circ \sigma_{2|U_1}, v) \cap U_1 = occ(\sigma_{1|U_2}, v) \cap U_1 = occ(\sigma_1, v) \cap U_1 \cap U_2 \supseteq X_1 \cap U_2$. This proves that $occ(\sigma, v) = X_1 \cup X_2$ and $X_1 \cap U_2 = X_2 \cap U_1$.
  While it is obvious that $X_1 \in S_1$ and $X_2 \in S_2^*$, we still need to prove that $X_1 \in S_1''$ and $X_2 \in S_2''^*$. For each $y \in occ(\delta, v)$, by the definition of δ we have that $y \in \sigma_2(U_1 \cap U_2)$ and therefore $occ(\sigma_2, y) \cap U_1 \neq \emptyset$. This proves that $X_2 \in S_2''^*$. Moreover, if $v \in \mathrm{rng}(\delta)$, then $v \in \mathrm{rng}(\sigma_{1|U_2})$ and thus $occ(\sigma_1, v) \in S_1''$. □

Theorem D.2

(Weak completeness of $\mathsf{match}_{\mathrm{Sh}}$)

The operator $\mathsf{match}_{\mathrm{Sh}}$ is optimal on the first argument and complete on the second one when $\mathsf{match}_{\mathrm{Ps}}$ is restricted to the case in which the second argument contains a single substitution. In formulas,

$$\mathsf{match}_{\mathrm{Sh}}([S_1, U_1], \alpha_{\mathrm{Sh}}([\{\sigma_2\}, U_2])) = \alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2])),$$

for each $[\{[\sigma_2]\}, U_2] \in \mathtt{Psub}$ and $[S_1, U_1] \in \mathtt{Sharing}$.

Proof

Since $\mathsf{match}_{\mathrm{Sh}}$ is correct w.r.t. $\mathsf{match}_{\mathrm{Ps}}$, it follows that

$$\alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2])) \sqsubseteq_{\mathrm{Sh}} \mathsf{match}_{\mathrm{Sh}}([S_1, U_1], \alpha_{\mathrm{Sh}}([\{[\sigma_2]\}, U_2])).$$

So, we only need to prove that

$$\mathsf{match}_{\mathrm{Sh}}([S_1, U_1], \alpha_{\mathrm{Sh}}([\{[\sigma_2]\}, U_2])) \sqsubseteq_{\mathrm{Sh}} \alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2])).$$

Assume, without loss of generality, that σ_2 is a canonical representative of $[\sigma_2]_{U_2}$ and $\mathrm{rng}(\sigma_2) \cap U_1 = \emptyset$. Take $B \in S$, where $[S, U_1 \cup U_2] = \mathsf{match}_{\mathrm{Sh}}([S_1, U_1], [S_2, U_2])$, with $[S_2, U_2] = \alpha_{\mathrm{Sh}}([\{[\sigma_2]\}, U_2])$. We have three cases:

- If $B \in S_1'$, then $B \in S_1$ and $B \subseteq U_1 \setminus U_2$. Let $\delta = \{x/v \mid x \in B\} \cup \{x/a \mid x \in \text{vars}(\sigma_2(U_1 \setminus B))\}$ and $\sigma_1 = (\delta \circ \sigma_2)_{|U_1}$, where v is a fresh variable. It follows that $\text{dom}(\sigma_1) = U_1$ and $\text{rng}(\sigma_1) = \{v\}$ with $occ(\sigma_1, v) = B$; therefore $[\sigma_1, U_1] \sqsubseteq_{\text{Ps}} \gamma_{\text{Sh}}([S_1, U_1])$. Clearly $\sigma_1 \preceq_{U_1 \cap U_2} \sigma_2$, since $U_1 \cap U_2 \subseteq U_1 \setminus B$. Let $\sigma = \text{mgu}(\sigma_1, \sigma_2)$. Since $B \cap \text{dom}(\sigma_2) = \emptyset$ and v is a fresh variable, it follows that $occ(\sigma, v) = B$, and thus $B \in \alpha_{\text{Sh}}(\mathsf{match}_{\text{Ps}}(\gamma_{\text{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2]))$.
- If $B \in S_2'$, there exists $v \in \mathcal{V}$ such that $occ(\sigma_2, v) \cap U_2 = B$. Let $X = \text{vars}(\sigma_2(U_1)))$ and take $\delta = \{x/a \mid x \in X\}$. Then $\sigma_1 = (\delta \circ \sigma_2)_{|U_1}$ is such that $occ(\sigma_1, v) \cap U_1 = \emptyset$ for each $v \in \mathcal{V}$; therefore $\sigma_1 \in \gamma_{\text{Sh}}([S_1, U_1])$. Moreover $\text{mgu}(\sigma_2, \sigma_1) \in \mathsf{match}_{\text{Ps}}(\gamma_{\text{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2])$. By the proof of Theorem D.1, equation (D 1), we have $\text{mgu}(\sigma_1, \sigma_2) = \delta \circ \sigma_2$. Since $B \cap U_1 = \emptyset$, $v \notin X = \text{vars}(\delta)$, and therefore $occ(\delta \circ \sigma_2, v) \cap U_2 = occ(\sigma_2, v) \cap U_2 = B$. Hence $B \in \alpha_{\text{Sh}}(\mathsf{match}_{\text{Ps}}(\gamma_{\text{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2]))$.
- We now assume $B = X_1 \cup \bigcup X$ with $X \subseteq S_2''$, $X_1 \in S_1''$, $\bigcup X \cap U_1 = X_1 \cap U_2$. Then, for each $H \in X$, there exists $v_H \in \mathcal{V}$ such that $occ(\sigma_2, v_H) \cap U_2 = H$. Since $H \cap U_1 \neq \emptyset$ for each $H \in X$, $v_H \in Y = \text{vars}(\sigma_2(U_1))$. Consider the substitution

  $$\delta = \{v_H/v \mid H \in X\} \uplus \{w/a \mid w \in Y, \forall H \in X. w \neq v_H\}$$

  for a fresh variable v and

  $$\sigma_1 = (\delta \circ \sigma_2)_{|U_1} \uplus \{x/v \mid x \in X_1 \setminus U_2\}.$$

  We want to prove $[\{[\sigma_1]\}, U_1] \in \gamma_{\text{Sh}}([S_1, U_1])$. By the definition of σ_1 we have that $occ(\sigma_1, v) \cap U_1 = (occ(\sigma_2, \{v_H \mid H \in X\}) \cap U_1) \cup X_1 \setminus U_2 = (\bigcup X \cap U_1) \cup X_1 \setminus U_2 = X_1 \in S_1$. Otherwise, for $w \neq v$ we have that either $occ(\sigma_1, w) = \emptyset$ when $w \in U_1$ or $occ(\sigma_1, w) = occ(\sigma_2, w)$, which is disjoint from U_1. In both cases, $occ(\sigma_1, w) \cap U_1 = \emptyset \in S_1$. By the definition of σ_1, $[\text{mgu}(\sigma_1, \sigma_2)] \in \mathsf{match}_{\text{Ps}}(\gamma_{\text{Sh}}([S_1, U_1]), [\{[\sigma_2]\}, U_2])$. Moreover, we know from (D 1) that

  $$\text{mgu}(\sigma_2, \sigma_1) = \delta \circ \sigma_2 \uplus \{x/v \mid x \in X_1 \setminus U_2\}.$$

  Let $\sigma = \text{mgu}(\sigma_1, \sigma_2)$. Note that $occ(\sigma, v) \cap (U_1 \cup U_2) = X_1 \setminus U_2 \cup occ(\sigma_2, \{v_H \mid H \in X\}) \cap U_2$. By the definition of v_H, $occ(\sigma_2, v_H) \cap U_2 = H$; hence $occ(\sigma, v) \cap (U_1 \cup U_2) = (X_1 \setminus U_2) \cup \bigcup X = X_1 \cup \bigcup X = B$.

This proves the theorem. □

Theorem D.3

(Optimality of $\mathsf{match}_{\text{Sh}}$)
$\mathsf{match}_{\text{Sh}}$ is optimal.

Proof

Given $[S_1, U_1], [S_2, U_2] \in \texttt{Sharing}$, we have

$$
\begin{aligned}
&\alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), \gamma_{\mathrm{Sh}}([S_2, U_2])))\\
&\quad = \alpha_{\mathrm{Sh}}(\sqcup_{\mathrm{Ps}}\{\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), [\{[\sigma]\}, U_2]) \mid \alpha_{\mathrm{Sh}}([\sigma]_{U_2}) \sqsubseteq_{\mathrm{Sh}} [S_2, U_2]\})\\
&\qquad \text{(since } \mathsf{match}_{\mathrm{Ps}} \text{ is additive)}\\
&\quad = \sqcup_{\mathrm{Sh}}\{\mathsf{match}_{\mathrm{Sh}}([S_1, U_1], [X, U_2]) \mid X = \alpha_{\mathrm{Sh}}([\sigma]_{U_2}) \sqsubseteq_{\mathrm{Sh}} [S_2, U_2]\}\\
&\qquad \text{(by completeness of } \sqcup_{\mathrm{Sh}} \text{ and Theorem D.2)}\\
&\quad = \mathsf{match}_{\mathrm{Sh}}([S_1, U_1], \sqcup_{\mathrm{Sh}}\{[X, U_2] \mid X = \alpha_{\mathrm{Sh}}([\sigma]_{U_2}) \sqsubseteq_{\mathrm{Sh}} [S_2, U_2]\})\\
&\qquad \text{(since } \mathsf{match}_{\mathrm{Sh}} \text{ is additive).}
\end{aligned}
$$

Since α_{Sh} defines a Galois insertion, it is surjective, and therefore $\sqcup_{\mathrm{Sh}}\{[X, U_2] \mid X = \alpha_{\mathrm{Sh}}([\sigma]_{U_2}) \sqsubseteq_{\mathrm{Sh}} [S_2, U_2]\} = [S_2, U_2]$, and we obtain

$$\alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), \gamma_{\mathrm{Sh}}([S_2, U_2]))) = \mathsf{match}_{\mathrm{Sh}}([S_1, U_1], [S_2, U_2]),$$

which concludes the proof. □

Theorem D.4

(Strong optimality of $\mathsf{unif}_{\mathrm{Sh}}$)

Given $[S_1, U_1] \in \texttt{Sharing}$ and $\theta \in \mathit{ISubst}$, there exists a substitution $\delta \in \mathit{ISubst}$ such that $\alpha_{\mathrm{Sh}}([\delta]_{U_1}) \sqsubseteq_{\mathrm{Sh}} [S_1, U_1]$ and

$$\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}([\{[\delta]\}, U_1], \theta)) = \mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta).$$

Proof

The optimality result proved in Theorem 6.16 shows that there exists $[\Theta_1, U_1] \sqsubseteq_{\mathrm{Ps}} \gamma_{\mathrm{Sh}}([S_1, U_1])$ such that $\alpha_{\mathrm{Sh}}(\mathsf{unif}_{\mathrm{Ps}}([\Theta_1, U_1], \theta)) = \mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \delta)$. We need a stronger result that proves that Θ_1 can be chosen as a singleton.

Assume $\mathsf{unif}_{\mathrm{Sh}}([S_1, U_1], \theta) = [S, U_1 \cup U_2]$, where $U_2 = \mathrm{vars}(\theta) \setminus U_1$ and $S = \{X^1, \dots, X^n\}$. Following the construction in Appendix C, for each X^i let us define X_1^i, X_2^i, K^i, K_1^i, W^i, s_x^i, s'^i_x, U as in the proof of optimality for $\mathsf{unif}_{\mathrm{Sh}}$. We choose W^i, W^j such that $W^i \cap W^j = \emptyset$ if $i \neq j$, and we denote by w_B^i the elements of W^i.

For each $y \in \mathrm{vars}(\theta(U_1)) \cap (\cup_{1 \leqslant i \leqslant n} X^i)$, we choose a variable $x_y \in U_1$ such that $y \in \theta(x_y)$. Then, we define the substitution δ in the following way: for each variables $x \in U_1$, $\delta(x)$ is the same as $\theta(x)$, with the exception that each occurrence (y, j) of a variable $y \in \theta(x)$ is replaced by $t_{x,y,j} = t(t_{x,y,j}^1, \dots, t_{x,y,j}^n)$, where

- $t_{x,y,j}^i = a$ if $y \notin X^i$;
- $t_{x,y,j}^i = s_x^i$ otherwise, if $x = x_y$ and $j = 1$;
- $t_{x,y,j}^i = s'^i_x$ otherwise.

By construction $\mathrm{dom}(\delta) = U_1$ and $\mathrm{rng}(\delta) = \bigcup_{1 \leqslant i \leqslant n} W^i$. It is easy to check that $\alpha_{\mathrm{Sh}}([\{\delta\}, U_1]) = [\bigcup_{1 \leqslant i \leqslant n} K_1^i, U_1] \sqsubseteq_{\mathrm{Sh}} [S_1, U_1]$. Using the properties of the equation sets we can prove that

$$\begin{aligned}&\text{mgu}(\delta, \theta)\\ &\quad = \text{mgu}(\{v = t_{x,v,j} \mid x \in U_1, (v, j) \text{ is an occurrence of } v \text{ in } \theta(x)\}) \circ \theta\\ &\quad = \text{mgu}(E) \circ \rho \circ \theta,\end{aligned}$$

where

$$\begin{aligned}\rho &= \{v/t_{x_v,v,1} \mid v \in \text{vars}(\theta(U_1))\},\\ E &= \{t^i_{x_v,v,1} = t^i_{x',v,j} \mid i \in \{1, \dots, n\}, v \in X^i, x' \in U_1,\\ &\qquad (v, j) \text{ is an occurrence of } v \text{ in } \theta(x')\}.\end{aligned}$$

Now, each $E^i = \{t^i_{x_v,v,1} = t^i_{x',v,j} \mid x' \in U_1, (v, j)$ is an occurrence of v in $\theta(x')$, and $v \in X^i\}$ is the same equation that appears in (C 1) for $X = X^i$. Therefore, for each $i \in \{1, \dots, n\}$ such that $W^i \neq \emptyset$, we choose a single $w^i \in W^i$ and define η^i with $\text{dom}(\eta^i) = W^i \setminus \{w^i\}$ and $\eta^i(w^i_B) = w^i$ for each $w^i_B \in W^i$. If $W^i = \emptyset$, we choose $\eta^i = \epsilon$. We know from the proof of Lemma C.5 that $\eta^i = \text{mgu}(E^i)$, and $\text{mgu}(E) = \eta = \biguplus_{1 \leqslant i \leqslant n} \eta^i$, since $\text{vars}(E^i) \cap \text{vars}(E^j) = \emptyset$ for $i \neq j$. Therefore

$$\text{mgu}(\delta, \theta) = \eta \circ \rho \circ \theta.$$

We now want to prove that $\alpha_{\text{Sh}}([\eta \circ \rho \circ \theta]_{U_1 \cup U_2}) \sqsupseteq_{\text{Ps}} [\{X^i\}, U_1 \cup U_2]$ for each $i \in \{1, \dots, n\}$. If $X^i_1 \neq \emptyset$, then $W^i \neq \emptyset$, and we have $occ(\eta \circ \rho \circ \theta, w^i) = occ(\eta^i \circ \rho \circ \theta, w^i)$. Following the proof of Lemma C.5 with $X = X^i$, we have that $occ(\eta \circ \rho \circ \theta, w^i) \cap U = X^i$. When $X^i_1 = \emptyset$, we may choose $v^i \in \theta(X^i_2)$. In this case, $occ(\eta \circ \rho \circ \theta, v^i) \cap U = occ(\theta, v^i) \cap U = X^i$ as proved in Proposition C.6. □

As for Proposition C.6, in the proof of this theorem we assume that we have term symbols for each arity. However, it is possible to rewrite terms so that a constant symbol and a binary term symbol suffice.

Theorem D.5
$\mathbf{U}^b_{\text{Sh}}$ is correct and optimal w.r.t. $\mathbf{U}^b_{\text{Ps}}$.

Proof
Correctness immediately follows by the fact that $\mathbf{U}^b_{\text{Ps}}$ is obtained by tupling and composition of correct semantic functions.

By using Theorems D.2 and D.4, it is possible to prove that

$$\begin{aligned}&\mathsf{match}_{\text{Sh}}([S_1, U_1], \mathsf{unif}_{\text{Sh}}([S_2, U_2], \theta))\\ &\qquad\qquad = \alpha_{\text{Sh}}(\mathsf{match}_{\text{Ps}}(\gamma_{\text{Sh}}([S_1, U_1]), \mathsf{unif}_{\text{Sh}}(\gamma_{\text{Sh}}([S_2, U_2]), \theta)));\end{aligned}$$

i.e., that the composition of $\mathsf{match}_{\text{Sh}}$ and $\mathsf{unif}_{\text{Sh}}$, as used in $\mathbf{U}^b_{\text{Sh}}$, is optimal.

Assume given $[S_1, U_1]$ and $[S_2, U_2] \in \mathsf{Psub}$ and $\theta \in \mathit{ISubst}$. Consider $[\{[\sigma]\}, U_2] \in \gamma_{\text{Sh}}([S_2, U_2])$ obtained by Lemma D.4 such that $\mathsf{unif}_{\text{Ps}}([\{[\sigma]\}, U_2]), \theta) = [\{[\delta]\}, U_2 \cup \text{vars}(\theta)]$ and $\alpha_{\text{Sh}}([\{[\delta]\}, U_2 \cup \text{vars}(\theta)]) = \mathsf{unif}_{\text{Sh}}([S_2, U_2], \theta)$. Then, we have

$$\begin{aligned}&\mathsf{match}_{\text{Sh}}([S_1, U_1], \mathsf{unif}_{\text{Sh}}([S_2, U_2], \theta))\\ &\quad = \mathsf{match}_{\text{Sh}}([S_1, U_1], \alpha_{\text{Sh}}(\mathsf{unif}_{\text{Ps}}([\{[\sigma]\}, U_2], \theta)))\\ &\quad = \alpha_{\text{Sh}}(\mathsf{match}_{\text{Ps}}(\gamma_{\text{Sh}}([S_1, U_1]), \mathsf{unif}_{\text{Ps}}([\{[\sigma]\}, U_2], \theta)))\end{aligned}$$

by Theorem D.2, so that in general

$$\mathsf{match}_{\mathrm{Sh}}([S_1, U_1], \mathsf{unif}_{\mathrm{Sh}}([S_2, U_2], \theta)) \sqsubseteq_{\mathrm{Sh}} \\ \alpha_{\mathrm{Sh}}(\mathsf{match}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_1, U_1]), \mathsf{unif}_{\mathrm{Ps}}(\gamma_{\mathrm{Sh}}([S_2, U_2]), \theta))).$$

The proof that $\mathbf{U}^b_{\mathrm{Ps}}$ is optimal follows from this result, completeness of π_{Sh}, and γ-completeness of ρ. □

References

Amato, G. and Scozzari, F. 2002. Optimality in goal-dependent analysis of sharing. In *Proc. of the Joint Conference on Declarative Programming (AGP'02)*, J. J. Moreno-Navarro and J. Mariño-Carballo, Eds. Universidad Politécnica de Madrid, Madrid, 189–205.

Amato, G. and Scozzari, F. 2003. A general framework for variable aliasing: Towards optimal operators for sharing properties. In *Logic Based Program Synthesis and Transformation 12th International Workshop, LOPSTR 2002, Madrid, Spain, September 17–20, 2002. Revised Selected Papers*, M. Leuschel, Ed. Lecture Notes in Computer Science, vol. 2664. Springer, Berlin Heidelberg, 52–70.

Apt, K. R. 1990. Introduction to logic programming. In *Handbook of Theoretical Computer Science. Vol. B: Formal Models and Semantics*, J. van Leeuwen, Ed. Elsevier and MIT Press, 495–574.

Bagnara, R., Hill, P. M. and Zaffanella, E. 2002. Set-sharing is redundant for pair-sharing. *Theoretical Computer Science 277*, 1–2, 3–46.

Bagnara, R., Zaffanella, E. and Hill, P. M. 2005. Enhanced sharing analysis techniques: A comprehensive evaluation. *Theory and Practice of Logic Programming 5*, 1–2, 1–43.

Bossi, A., Gabbrielli, M., Levi, G. and Martelli, M. 1994. The s-semantics approach: Theory and applications. *The Journal of Logic Programming 19–20*, 149–197.

Bruynooghe, M. 1991. A practical framework for the abstract interpretation of logic programs. *The Journal of Logic Programming 10*, 1–4, 91–124.

Codish, M., Lagoon, V. and Bueno, F. 2000. An algebraic approach to sharing analysis of logic programs. *The Journal of Logic Programming 42*, 2 (Feb.), 110–149.

Cortesi, A. and Filé, G. 1999. Sharing is optimal. *The Journal of Logic Programming 38*, 3, 371–386.

Cortesi, A., Filé, G. and Winsborough, W. W. 1994. *Optimal Groundness Analysis Using Propositional Formulas*, Technical Report 94/11. Dipartimento di Matematica Pura ed Applicata, Università di Padova.

Cortesi, A., Filé, G. and Winsborough, W. W. 1996. Optimal groundness analysis using propositional logic. *The Journal of Logic Programming 27*, 2, 137–167.

Cousot, P. and Cousot, R. 1979. Systematic design of program analysis frameworks. In *Proc. of the Sixth ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages.* ACM Press, New York, 269–282.

Cousot, P. and Cousot, R. 1992. Abstract interpretation and applications to logic programs. *The Journal of Logic Programming 13*, 2–3, 103–179.

Cousot, P. and Cousot, R. 1994. Higher-order abstract interpretation (and application to comportment analysis generalizing strictness, termination, projection and PER analysis of functional languages), invited paper. In *Proc. of the 1994 International Conference on Computer Languages*, Henri E. Bal, Ed. IEEE Computer Society Press, Los Alamitos, CA, 95–112.

Furukawa, K., Ed. 1991. *Logic Programming: Proc. of the Eighth International Conference.* MIT Press, Cambridge, MA.

García de la Banda, M. J., Marriott, K., Stuckey, P. J., and Søndergaard, H. 1998. Differential methods in logic program analysis. *The Journal of Logic Programming 35*, 1 (Apr.), 1–37.

Giacobazzi, R., Ranzato, F. and Scozzari, F. 2000. Making abstract interpretations complete. *Journal of the ACM 47*, 2, 361–416.

Hans, W. and Winkler, S. 1992. *Aliasing and Groundness Analysis of Logic Programs through Abstract Interpretation and Its Safety*, Technical Report 92-27 [online]. Accessed 10 July 2009. Technical University of Aachen (RWTH Aachen). URL: `http://sunsite.informatik.rwth-aachen.de/Publications/AIB`

Henkin, L., Monk, J. D. and Tarski, A. 1971. *Cylindric Algebras Part I*, Studies in Logic and the Foundations of Mathematics, no. 115. North Holland, Amsterdam.

Hermenegildo, M. V. and Rossi, F. 1995. Strict and nonstrict independent and-parallelism in logic programs: Correctness, efficiency, and compile-time conditions. *The Journal of Logic Programming 22*, 1, 1–45.

Hill, P. M., Zaffanella, E. and Bagnara, R. 2004. A correct, precise and efficient integration of set-sharing, freeness and linearity for the analysis of finite and rational tree languages. *Theory and Practice of Logic Programming 4*, 3, 289–323.

Howe, J. M. and King, A. 2003. Three optimisations for sharing. *Theory and Practice of Logic Programming 3*, 2 (Jan.), 243–257.

Jacobs, D. and Langen, A. 1992. Static analysis of logic programs for independent AND parallelism. *The Journal of Logic Programming 13*, 2–3, 291–314.

King, A. 1994. A synergistic analysis for sharing and groundness which traces linearity. In *Programming Languages and Systems ESOP '94, Fifth European Symposium on Programming Edinburg, U.K., April 11–13, 1994, Proc.*, D. Sannella, Ed. Lecture Notes in Computer Science, vol. 788. Springer, Berlin Heidelberg, 363–378.

King, A. 2000. Pair-sharing over rational trees. *The Journal of Logic Programming 46*, 1–2, 139–155.

King, A. and Longley, M. 1995. *Abstract Matching Can Improve on Abstract Unification*, Technical Report 4-95*. Computing Laboratory, University of Kent, Canterbury, UK.

Langen, A. 1990. *Static Analysis for Independent And-Parallelism in Logic Programs*, PhD thesis. University of Southern California, Los Angeles, CA.

Le Charlier, B., Musumbu, K. and Van Hentenryck, P. 1991. A generic abstract interpretation algorithm and its complexity analysis. In *Logic Programming: Proc. of the Eighth International Conference*, K, Furukawa, Ed. MIT Press, Cambridge, MA, 64–78.

Le Charlier, B. and Van Hentenryck, P. 1994. Experimental evaluation of a generic abstract interpretation algorithm for prolog. *ACM Transactions on Programming Languages and Systems 16*, 1, 35–101.

Levi, G. and Spoto, F. 2003. Pair-independence and freeness analysis through linear refinement. *Information and Computation 182*, 1, 14–52.

Lloyd, J. W. 1987. *Foundations of Logic Programming*, 2nd ed. Springer, New York.

Marriott, K., Søndergaard, H. and Jones, N. D. 1994. Denotational abstract interpretation of logic programs. *ACM Transactions on Programming Languages and Systems 16*, 3, 607–648.

Muthukumar, K. and Hermenegildo, M. V. 1991. Combined determination of sharing and freeness of program variables through abstract interpretation. In *Logic Programming: Proc. of the Eighth International Conference*, K, Furukawa, Ed. MIT Press, Cambridge, MA, 49–63.

Muthukumar, K. and Hermenegildo, M. V. 1992. Compile-time derivation of variable dependency using abstract interpretation. *The Journal of Logic Programming 13*, 2–3, 315–347.

Palamidessi, C. 1990. Algebraic properties of idempotent substitutions. In *Automata, Languages and Programming, 17th International Colloquium Warwick University, England, July 16–20, 1990, Proc.*, M. Paterson, Ed. Lecture Notes in Computer Science, vol. 443. Springer, Berlin Heidelberg, 386–399.

Shepherdson, J. C. 1994. The role of standardising apart in logic programming. *Theoretical Computer Science 129*, 143–166.

Søndergaard, H. 1986. An application of abstract interpretation of logic programs: Occur check reduction. In *Proc. ESOP 86*, B. Robinet and R. Wilhelm, Eds. Lecture Notes in Computer Science, vol. 213. Springer, Berlin Heidelberg, 327–338.

Van Emden, M. H. and Kowalski, R. A. 1976. The semantics of predicate logic as a programming language. *Journal of the ACM 23*, 4, 733–742.