A general framework for variable aliasing: Towards optimal operators for sharing properties

Gianluca Amato[1] and Francesca Scozzari[2]

[1] Dipartimento di Matematica e Informatica, Università di Udine.
[2] Dipartimento di Informatica, Università di Pisa.
amato@dimi.uniud.it, scozzari@di.unipi.it

Abstract. We face the problem of devising optimal unification operators for sharing and linearity analysis of logic programs by abstract interpretation. We propose a new (infinite) domain $\mathtt{ShLin}^\omega$ which can be thought of as a general framework from which other domains can be easily derived by abstraction. The advantage is that $\mathtt{ShLin}^\omega$ is endowed with very elegant and optimal abstract operators for unification and matching, based on a new concept of *sharing graph* which plays the same role of alternating paths for pair sharing analysis. We also provide an alternative, purely algebraic description of sharing graphs. Starting from the results for $\mathtt{ShLin}^\omega$, we derive optimal abstract operators for two well-known domains which combine sharing and linearity: $\mathtt{ShLin}^2$ by Andy King and the classic $\mathtt{Sharing} \times \mathtt{Lin}$.

1 Introduction

In the field of static analysis of logic programs by abstract interpretation [7, 8], the property of sharing has been the object of many works, both on the theoretical and practical point of view. The goal of (set) sharing analysis is to detect sets of variables which share a common variable in the answer substitutions. Typical applications of sharing analysis are in the fields of optimization of unification [20] and parallelization of logic programs [10].

It is now widely recognized that the original domain proposed for sharing analysis, namely Sharing [17, 13] by Jacobs and Langen, is not very precise, so that it is often combined with other domains for treating freeness, linearity, groundness or structural information (see [3] for a comparative evaluation). In particular, adding some kind of linearity information seems to be very profitable, both for the gain in precision and speed which can be obtained, and for the fact that it can be easily and elegantly embedded inside the sharing groups (see [14]). However, optimal operators for combined analysis of sharing and linearity have never been devised, neither for the domain $\mathtt{ShLin}^2$ [14], nor for the more broadly adopted $\mathtt{Sharing} \times \mathtt{Lin}$ [9, 19] or ASub [4]. The lack of optimal operators brings two kinds of disadvantages: first, the analysis obviously looses in precision when using sub-optimal abstract operators; second, computing approximated abstract objects can lead to a speed-down of the analysis. The latter is typical of sharing analysis, where abstract domains are usually defined in such a way that, the less information we have, the more abstract objects are complex. This is not the case for other kind of analyses, such as groundness analysis, where the complexity of abstract objects may grow accordingly to the amount of groundness information they encode. The lack of optimal operators is due to the fact that the role played by linearity in the unification process has never been fully clarified. The traditional domains which combine sharing and linearity information are too abstract to capture in a clean way the effect of repeated occurrences of a variable in a term and most of the effects of (non-)linearity are obscured by the abstraction process. In this paper, we investigate the interaction between sharing and linearity, and provide optimal abstract operators for two well-known domains which combine these properties. We start by introducing our concrete goal-dependent framework which is based on [6], with the improvements introduced in [1] concerning backward unification. We propose a slightly modified domain of substitutions, which are quotiented modulo an appropriate renaming w.r.t. the variables which are not of interest (see Jacobs and Langen' domain *ESubst* [13]). We define two operations of unification and matching which are used as an intermediate step toward the semantics function for forward and backward unifications.

Inspired by $\mathtt{ShLin}^2$, we propose an abstract domain which is able to encode the *amount* of non-linearity, i.e., which keeps track of the exact number of occurrences of the same variable in a term. The domain we obtain is very simple and elegant, but cannot be directly used for static analysis, at least without resorting to widening operators, since it contains infinite ascending chains. However, in this domain the role played by (non-)linearity is manifest, and the optimal abstract operators for unification and matching [9, 16, 1] assume a very clean form. The cornerstone of the abstract unification is the concept of *sharing graph* which plays the same role of alternating paths [20, 15] for pair sharing. A sharing graph is a graph theoretic notion to figure out sharing groups which are combined during the unification process to obtain a new sharing group. The use of sharing graphs offers a new perspective to look at single variables in the process of unification, and simplify the proofs of correctness and optimality of the abstract operators. We also provide a purely algebraic characterization of the results, which can help in implementing the domain by making use of widening operators and in devising abstract operators for further abstractions of $\mathtt{ShLin}^\omega$. We show that the domains $\mathtt{ShLin}^2$ and $\mathtt{Sharing} \times \mathtt{Lin}$ can be immediately obtained as abstractions of $\mathtt{ShLin}^\omega$ and we provide the optimal operators for unification and matching. We also provide a simplified version of the operators for the domain $\mathtt{Sharing} \times \mathtt{Lin}$ which is still correct, but which is optimal for one-binding substitutions only. We show that unification between an abstract object and a substitution cannot be computed one binding at a time while remaining optimal. Finally, we conclude with some open questions for future work.

2 Notation

Let $\mathbb{N}^+$ be the set of natural numbers without zero. A *(finite) multiset* is a map $X : \mathcal{X} \to \mathbb{N}^+$ where $\mathcal{X}$ is a finite set called the *support* of X and denoted by $\llbracket X \rrbracket$.

We often denote a multiset as $\{\!\{v_1, \ldots, v_n\}\!\}$ where $v_1, \ldots, v_n$ is a sequence of elements with repetitions. We also use the polynomial notation $X = v_1^{i_1}, \ldots, v_n^{i_n}$ to denote a multiset with support $\{v_1, \ldots, v_n\}$ such that $X(v_k) = i_k$. We extend the functional notation for multiset by writing $X(v) = 0$ if $v \notin [\![X]\!]$. If S is a set, a *multiset over* S is a multiset whose support is a subset of S. We denote by $\wp_m(S)$ the set of multisets over S and we write $X \subseteq_m S$ as an alternative for $X \in \wp_m(S)$. Any set S used as an argument to a multiset operator stands for the multiset with support S and such that $S(x) = 1$ for all $x \in S$. We denote by $X|_S$ the multiset defined as $X(v)$ if $v \in S$, 0 otherwise. $|X|$ will denote the number of elements in X including repeated elements, i.e., $\sum_{x \in [\![X]\!]} X(x)$ when $X \neq \emptyset$, 0 otherwise. If E is any expression involving a variable x, we write $\sum_{x \in X} E(x)$ as a short form for $\sum_{x \in [\![X]\!]} X(x) \cdot E(x)$. We use either $\{\!\{\}\!\}$ or $\emptyset$ for the multiset whose support is empty, while union and intersection of multisets are denoted by $\uplus$ and $\Cap$. If $X = \{\!\{X_1, \ldots, X_m\}\!\}$ is a multiset of multisets, then $\biguplus X = X_1 \uplus \cdots \uplus X_n$. Let Atoms, $\mathsf{Clauses}$, Body and Progs be the syntactic categories for atoms, clauses, bodies and programs respectively, where $\lambda \in \mathsf{Body}$ stands for the empty body. We denote by *Subst* and *ISubst* the sets of substitutions and idempotent substitutions respectively, by ϵ the empty substitution and by *Ren* the set of renamings (i.e., invertible substitutions). Given a substitution θ, $\mathrm{dom}(\theta)$ and $\mathrm{rng}(\theta)$ denote the domain and range of θ. Let $\mathcal{V}$ be a denumerable set of variables. Given $v \in \mathcal{V}$ and a term t, we denote by $\mathit{vars}(t)$ the set of variables occurring in t and by $\mathit{occ}(v, t)$ the number of occurrences of v in t.

3 The Concrete Semantics

Our analysis is based on a (collecting) goal-dependent semantics for logic programs. We look for a concrete domain where substitutions explicitly show their variables of interest, which are "independent" from the other variables. This choice is motivated from the fact that, in practice, one needs to keep track of the current variables of interest during the analysis. Several semantics in the literature present these characteristics, e.g. the semantics in [18] based on ex-equations, [6] using idempotent substitutions and that in [13] based on the domain *ESubst* of existential substitutions. The latter semantics is probably the most appropriate to our goal, but it is based on a non-standard definition of substitution and the relation between the unification operator on *ESubst* and the standard one, although clear, is not well stated. Moreover, most of the research in combining sharing and linearity information has been done on the domain defined in [6], so that founding our work on a different framework would make difficult the comparison of our results to the literature. Therefore, we work with [6] and show that with an appropriate equivalence relation on substitutions, one can obtain an equivalent semantics over existential substitutions.

3.1 Concrete Domain and Operators

The concrete domain is $\mathtt{Rsub} = \wp(\mathit{ISubst}) \times \wp_f(\mathcal{V}) \cup \{\bot_{\mathrm{Rs}}, \top_{\mathrm{Rs}}\}$ (see [6] for a detailed introduction). $\mathtt{Rsub}$ is partially ordered as follows: $\bot_{\mathrm{Rs}}$ is the bottom element, $\top_{\mathrm{Rs}}$ the top and $[\Theta_1, U_1] \leq_{\mathrm{Rs}} [\Theta_2, U_2]$ if and only if $U_1 = U_2$ and $\Theta_1 \subseteq \Theta_2$. $\mathtt{Rsub}$ is a complete lattice w.r.t. $\leq_{\mathrm{Rs}}$. The l.u.b. of $\mathtt{Rsub}$ is denoted by $\sqcup_{\mathrm{Rs}}$. We briefly recall the concrete operations from [6] and refined in [1] with the introduction of two different operators for forward and backward unification. The concrete projection $\pi_{\mathrm{Rs}} : \mathtt{Rsub} \times \mathtt{Rsub} \to \mathtt{Rsub}$ is defined as:

$$\begin{aligned}
&\pi_{\mathrm{Rs}}(\bot_{\mathrm{Rs}}, A) = \pi_{\mathrm{Rs}}(A, \bot_{\mathrm{Rs}}) = \bot_{\mathrm{Rs}} \\
&\pi_{\mathrm{Rs}}(\top_{\mathrm{Rs}}, A) = \pi_{\mathrm{Rs}}(A, \top_{\mathrm{Rs}}) = \top_{\mathrm{Rs}} && \text{when } A \neq \bot_{\mathrm{Rs}} \\
&\pi_{\mathrm{Rs}}([\Theta_1, U_1], [\Theta_2, U_2]) = [\Theta_1, U_1 \cap U_2]
\end{aligned}$$

In the following, for the sake of conciseness, we define the behavior of the concrete and abstract operators only in case all the arguments are different from $\bot_{\mathrm{Rs}}$ and $\top_{\mathrm{Rs}}$. We implicitly assume that the result is $\bot_{\mathrm{Rs}}$ if any of the argument is $\bot_{\mathrm{Rs}}$, $\top_{\mathrm{Rs}}$ when any of the argument is $\top_{\mathrm{Rs}}$ and no argument is $\bot_{\mathrm{Rs}}$. The concrete forward unification is $\mathbf{U}^f_{\mathrm{Rs}} : \mathtt{Rsub} \times \wp_f(\mathcal{V}) \times \mathsf{Atoms} \times \mathsf{Atoms} \to \mathtt{Rsub}$ such that:

$$\begin{aligned}
\mathbf{U}^f_{\mathrm{Rs}}([\Theta, U_1], U_2, A_1, A_2) = [\{\mathrm{mgu}(\rho_1(\theta), \delta) \mid \theta \in \Theta,& \\
\delta = \mathrm{mgu}(\rho_1(A_1) = A_2)\}, \rho_1(U_1) \cup U_2]&
\end{aligned}$$

where $(\rho_1, \rho_2) = \mathit{Apart}(U_2)$, provided $\mathit{vars}(A_1) \subseteq U_1$ and $\mathit{vars}(A_2) \subseteq U_2$, $\bot_{\mathrm{Rs}}$ in all the other cases. We still need to define Apart. Given $U_2 \in \wp_f(\mathcal{V})$, take a partition $\{V_1, V_2\}$ of $\mathcal{V}$ such that V_1 and V_2 are infinite and $U_2 \subseteq V_2$. Then $\mathit{Apart}(U_2) = (\rho_1, \rho_2)$ where $\rho_1 : \mathcal{V} \to V_1$ and $\rho_2 : \mathcal{V} \to V_2$ are bijections such that, for each $x \in U_2$, $\rho_2(x) = x$. We apply such bijections to syntactic objects as if they were substitutions. The backward unification exploits the relation among substitutions $\preceq_U \subseteq \mathit{Subst} \times \mathit{Subst}$ defined as follows, for $U \in \wp_f(\mathcal{V})$:

$$\sigma \preceq_U \sigma' \iff \exists \delta \in \mathit{Subst}. \forall x \in U. \sigma(x) = \delta(\sigma'(x)) \ . \qquad (1)$$

The concrete backward unification $\mathbf{U}^b_{\mathrm{Rs}} : \mathtt{Rsub} \times \mathtt{Rsub} \times \mathsf{Atoms} \times \mathsf{Atoms} \to \mathtt{Rsub}$ is given by:

$$\begin{aligned}
\mathbf{U}^b_{\mathrm{Rs}}([\Theta_1, U_1], [\Theta_2, U_2], A_1, A_2) = \{\mathrm{mgu}(\rho_1(\sigma_1), \rho_2(\sigma_2), \delta) \mid \sigma_1 \in \Theta_1, \sigma_2 \in \Theta_2,& \\
\delta = \mathrm{mgu}(\rho_1(A_1), A_2), \rho_1(\sigma_1) \preceq_{\rho_1(U_1)} \mathrm{mgu}(\rho_2(\sigma_2), \delta), \rho_1(U_1) \cup U_2] \ .&
\end{aligned}$$

where $\mathit{vars}(A_1) \subseteq U_1$, $\mathit{vars}(A_2) \subseteq U_2$, $\bot_{\mathrm{Rs}}$ in all the other cases. Here we improve over the standard unification by requiring that $\rho_1(\sigma_1)$ (the exit substitution) is an instance of $\mathrm{mgu}(\rho_2(\sigma_2), \delta)$ (the entry substitution) w.r.t. the variables of the calling atom [9]. By using the previously defined operators, we provide a goal-dependent, bottom-up semantics for logic programs. A denotation is an element in the set of monotonic maps $\mathcal{D}en = \mathsf{Atoms} \to \mathtt{Rsub} \to \mathtt{Rsub}$ and the semantic functions $\mathcal{P} : \mathsf{Progs} \to \mathcal{D}en$, $\mathcal{C} : \mathsf{Clauses} \to \mathcal{D}en \to \mathcal{D}en$ and $\mathcal{B} : \mathsf{Body} \to \mathcal{D}en \to \mathtt{Rsub} \to \mathtt{Rsub}$ are defined according to [1] as follows.

$$\mathcal{P}[\![P]\!] = \mathit{lfp}\lambda d. \left(\bigsqcup_{cl \in P} \mathcal{C}[\![cl]\!] d \right)$$

$$\mathcal{C}[\![H \leftarrow B]\!]dAx = \pi_{\mathrm{Rs}}(\mathbf{U}^b_{\mathrm{Rs}}(x', x, H, A), x)$$
$$\text{where } x' = \mathcal{B}[\![B]\!]d(\pi_{\mathrm{Rs}}(\mathbf{U}^f_{\mathrm{Rs}}(x, \mathit{vars}(H \leftarrow B), A, H), [\emptyset, \mathit{vars}(H \leftarrow B))]$$
$$\mathcal{B}[\![\lambda]\!]dx = x$$
$$\mathcal{B}[\![A, B]\!]dx = \mathcal{B}[\![B]\!]d(dAx)$$

Given a program P and an atom A, the set of computed answers for A in P is given by $\mathcal{P}[\![P]\!]A([\{\epsilon\}, \mathit{vars}(A)])$.

3.2 A Different Concrete Domain

It turns out that the domain of idempotent substitutions is too concrete for the above semantics of logic programs. Given a goal $\mathtt{p(x, y)}$ in a program P, we do not really want to distinguish between the answers $\{x/y\}$, $\{y/x\}$ and $\{x/u, y/u\}$. Note that while $\{x/y\}$ and $\{y/x\}$ can be obtained from each other by renaming, the same does not hold for $\{x/y\}$ and $\{x/u, y/u\}$. Actually, in the literature we find several alternatives to solve this problem, like ex-equations [18], Herbrand constraints and the domain of existential substitutions *ESubst* [13]. The common viewpoint is that substitutions are viewed as constraints and that variables which are not of interest (like u in the previous example) are existentially quantified. We show that such domains naturally arise as appropriate equivalence classes of substitutions. Given two substitutions θ and θ' and a set of variables U, we define the equivalence relation:

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

which is the equivalence relation induced by the preorder $\preceq_U$. By exploiting this relation, we can define a new domain $\mathit{ISubst}_\sim$ of *existential substitutions* as the disjoint union of all the $\mathit{ISubst}_{\sim_U}$, for $U \in \wp_f(\mathcal{V})$.

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

In the following we write $[\theta]_U$ for the equivalence class of θ w.r.t. $\sim_U$. Given $U, V \in \wp_f(\mathcal{V})$, $[\theta_1]_U, [\theta_2]_V \in \mathit{ISubst}_\sim$, we define:

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

where $\theta_1' \sim_U \theta_1$, $\theta_2' \sim_U \theta_2$, $\mathrm{dom}(\theta_1') = U$, $\mathrm{dom}(\theta_2') = V$ and $\mathrm{rng}(\theta_1') \cap \mathrm{rng}(\theta_2') = \emptyset$. It can be proved that the definition does not depend from the choice of representatives, and that $\mathrm{mgu}([\theta_1]_U, [\theta_2]_V)$ is the greatest lower bound of $[\theta_1]_U$ and $[\theta_2]_V$. It is worth noting that the resultant domain $\mathit{ISubst}_\sim$ is isomorphic to the domain *ESubst* by Jacobs and Langen [13], which is based on a non standard definition of substitution. By exploiting the domain $\mathit{ISubst}_\sim$ we can define a domain which is a complete abstraction of $\mathtt{Rsub}$. We lift the equivalence $\sim_U$ to $\mathtt{Rsub}$.

$$[\Theta_1, U] \sim [\Theta_2, U] \iff \forall \theta \in \Theta_1 \exists \theta' \in \Theta_2 . \theta \sim_U \theta' \text{ and vice versa.} \tag{3}$$

As shown in [1], $\sim$ is a congruence w.r.t. to the operations in $\mathtt{Rsub}$. Since $[\{\sigma\}, U] \sim [\{\sigma'\}, U]$ iff $\sigma \sim_U \sigma'$ and moreover $[\Theta, U] = \bigsqcup_{\mathrm{Rs}} \{[\{\sigma\}, U] \mid \sigma \in \Theta\}$, it turns out that $\mathtt{Rsub}_\sim$ is isomorphic to the following domain:

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

The operators and semantic functions over $\mathtt{Rsub}$ induce corresponding operators on $\mathtt{Psub}$ by means of the isomorphisms between $\mathtt{Psub}$ and $\mathtt{Rsub}_\sim$. First of all, let us define the auxiliary operations of unification and matching. Forward and backward unification will be built starting from them. The concrete unification $\mathsf{unif}_{\mathrm{Ps}} : \mathtt{Psub} \times \mathit{ISubst} \times \wp_f(\mathcal{V}) \to \mathtt{Psub}$ is given by:

$$\mathsf{unif}_{\mathrm{Ps}}([\Sigma, U], \delta, V) = [\{\mathrm{mgu}([\sigma]_U, [\delta]_V) \mid [\sigma]_U \in \Sigma\}, U \cup V]$$

provided $\mathit{vars}(\delta) \subseteq V$. It is worth noting that when $V = U$, this is the standard unification which is usually considered in the literature on sharing. It is possible to define $\mathsf{unif}_{\mathrm{Ps}}$ to take an argument in $\mathit{ISubst}_\sim$ instead of a substitution and a set of variables. However, this would make the operation more general, since not all the elements of $\mathit{ISubst}_\sim$ admits a representative $[\theta]_U$ with $\mathit{vars}(\theta) \subseteq U$. Using this definition of $\mathsf{unif}_{\mathrm{Ps}}$, we are actually restricting our attention to this case, which simplifies the presentation of the abstract operators.

We then define the matching operation $\mathsf{match}_{\mathrm{Ps}} : \mathtt{Psub} \times \mathtt{Psub} \to \mathtt{Psub}$ as:

$$\begin{aligned}\mathsf{match}_{\mathrm{Ps}}([\Theta_1, U_1], [\Theta_2, U_2]) = [\{&\mathrm{mgu}([\theta_1]_{U_1}, [\theta_2]_{U_2}) \mid \\ &\theta_1 \preceq_{U_1} \theta_2, [\theta_1]_{U_1} \in \Theta_1, [\theta_2]_{U_2} \in \Theta_2\}, U_2]\end{aligned}$$

provided $U_1 \subseteq U_2$. These can be used to define the forward and backward unification over $\mathtt{Psub}$ as follows:

$$\begin{aligned}\mathbf{U}^f_{\mathrm{Ps}}([\Sigma, U_1], U_2, A_1, A_2) &= \mathsf{unif}_{\mathrm{Ps}}(\rho_1([\Sigma, U_1]), \mathrm{mgu}(\rho_1(A_1) = A_2), U_2] \\ \mathbf{U}^b_{\mathrm{Ps}}([\Sigma_1, U_1], [\Sigma_2, U_2], A_1, A_2) &= \\ \mathsf{match}_{\mathrm{Ps}}(\rho_1([\Sigma_1, U_1]), &\mathsf{unif}_{\mathrm{Ps}}([\Sigma_2, U_2], \mathrm{mgu}(\rho_1(A_1) = A_2), \rho_1(U_1)))\end{aligned}$$

where $(\rho_1, \rho_2) = \mathit{Apart}(U_2)$. These can be easily proved to correspond to the ones for $\mathtt{Rsub}_\sim$ with simple algebraic manipulations.

4 The abstract domain $\mathtt{ShLin}^\omega$

In this section we define a new abstract domain $\mathtt{ShLin}^\omega$. Since it is infinite and contains infinite ascending chains, it cannot be directly used for the analysis. It should be thought of as a general framework from which other domains can be easily derived by abstraction. The idea underlying the construction of $\mathtt{ShLin}^\omega$ is to count the exact number of occurrences of the same variable in a term. In this way, it extends the standard domain $\mathtt{Sharing}$ by recording, for each $v \in \mathcal{V}$ and $\theta \in \mathit{ISubst}$, not only the set $\{w \mid v \in \theta(w)\}$ but the pairs $\{\langle w, \mathit{occ}(v, \theta(w))\rangle \mid v \in \theta(w)\}$ with the use of multisets. We call *ω-sharing group* a multiset of variables and we build a domain which works on ω-sharing groups.

$$\mathtt{ShLin}^\omega = \{[S, U] \mid U \in \wp_f(\mathcal{V}), S \subseteq \wp_m(U), S \neq \emptyset \Rightarrow \emptyset \in S\} \cup \{\bot_\omega, \top_\omega\} \tag{4}$$

where $\bot_\omega$ is the least element, $\top_\omega$ is the greatest and $[S_1, U_1] \leq_\omega [S_2, U_2]$ iff $U_1 = U_2$ and $S_1 \subseteq S_2$. $\mathtt{ShLin}^\omega$ is a complete lattice and the l.u.b. is denoted by $\sqcup_\omega$.

Given a substitution θ and a variable $v \in \mathcal{V}$, we denote by $\theta^{-1}(v)$ the ω-sharing group B with support $\{w \mid v \in \theta(w)\}$ and $B(w) = \mathit{occ}(v, \theta(w))$. Therefore $\theta^{-1}(v)$ maps each variable w to the number of occurrences of v in $\theta(w)$. We define the abstraction for a substitution θ w.r.t. the variables of interest in U:

$$\alpha_U(\theta) = \{\theta^{-1}(v)|_U \mid v \in \mathcal{V}\} \ . \tag{5}$$

Intuitively, each $B \in \alpha_U(\theta)$ corresponds to one or more variables which are shared by all the variables in B, each with the exact number of occurrences. For example, given $\theta = \{x/t(y,u,u), z/y, v/u\}$ and $U = \{w,x,y,z\}$, we have $\theta^{-1}(u) = x^2vu$, $\theta^{-1}(y) = xyz$, $\theta^{-1}(z) = \theta^{-1}(v) = \theta^{-1}(x) = \emptyset$ and $\theta^{-1}(s) = s$ for all the other variables (included w). Projecting over U we obtain $\alpha_U(\theta) = \{x^2, xyz, w, \emptyset\}$. Note that if $\theta_1 \sim_U \theta_2$ then $\alpha_U(\theta_1) = \alpha_U(\theta_2)$. Therefore, we can lift α_U to obtain the Galois insertion between `Psub` and `ShLin`$^\omega$ as follows: $\alpha_\omega(\bot_{\mathrm{Rs}}) = \bot_\omega$, $\alpha_\omega(\top_{\mathrm{Rs}}) = \top_\omega$ and

$$\alpha_\omega([\Sigma, U]) = \left[\bigcup\{\alpha_U(\theta) \mid \theta \in \Sigma\}, U\right] \tag{6}$$

In the following, we will omit to explicitly define abstraction and concretization on the top and bottom elements of the domains. The projection operation is defined pointwise in the obvious way:

$$\pi_\omega([S_1, U_1], [S_2, U_2]) = [\{B|_{U_2} \mid B \in S_1\}, U_1 \cap U_2] \tag{7}$$

4.1 Unification and Matching

The unification is much more complex and we prefer to characterize the operation of unification by means of graph theoretic notions. We first need to define the multiplicity of an ω-sharing group B in a term t as follows:

$$\chi(B, t) = \sum_{v \in \llbracket B \rrbracket} B(v) \cdot \mathit{occ}(v, t) \ . \tag{8}$$

For instance, $\chi(x^3yz^4, t(x, y, f(x, y, z))) = 3 \cdot 2 + 1 \cdot 2 + 4 \cdot 1 = 12$. If $B \in \alpha_U(\theta)$ represents the variable v (i.e., $B = \theta^{-1}(v) \cap U$) then $\chi(B, t)$ is the number of occurrences of v in $\theta(t)$.

A *sharing graph* is a directed multigraph whose nodes are labeled with sharing groups. In formulas, it is a tuple $\langle N, l, E\rangle$ where N is the finite set of nodes, $l : N \to \wp_m(\mathcal{V})$ is the labeling functions and $E \in \wp_m(N \times N)$ is the multiset of edges. A *balanced* sharing graph for the equation $t_1 = t_2$ and a set of ω-sharing groups S is a sharing graph $G = \langle N, l, E\rangle$ such that:

1. G is connected;
2. for each node $s \in N$, $l(s) \in S$;
3. for each node $s \in N$, the out-degree of s is equal to $\chi(l(s), t_1)$ and the in-degree of s is equal to $\chi(l(s), t_2)$.

Given a balanced sharing graph $G = \langle N, l, E\rangle$, we define the *resultant ω-sharing group* of G as $\mathit{res}(G) = \biguplus_{s \in N} l(s)$. The set of resultants ω-sharing groups for $t_1 = t_2$ given a set S of sharing groups is denoted by:

$$\mathrm{mgu}(S, t_1 = t_2) = \{res(G) \mid G \text{ is a balanced sharing graph for } S \text{ and } t_1 = t_2\ \ .\}$$

A sharing graph represents a possible way to merge together several sharing groups by unifying them with a given binding. Assume, for $j \in \{1, 2\}$, $B_j \in \alpha_U(\theta)$, i.e., there exist $v_j \in \mathcal{V}$ such that $B_j = \theta^{-1}(v_j) \cap U$. When unifying θ with the binding $t_1 = t_2$, we know that $\mathrm{mgu}(\theta, t_1 = t_2) = \mathrm{mgu}(\theta(t_1) = \theta(t_2)) \circ \theta$ and that $\theta(t_i)$ contains $\chi(B_j, t_i)$ instances of v_j. An arrow from the sharing group B_1 to B_2 represents the fact that, in $\mathrm{mgu}(\theta(t_1) = \theta(t_2))$, one of the copies of v_1 is aliased with one of the copies of v_2. The third condition for balanced sharing graphs implies that all the copies of each v_j are aliased with some other variable. Therefore, we are considering the case when $\theta(t_1)$ and $\theta(t_2)$ only differs for the variables occurring in them. Although this is restrictive in general, it is enough to reach optimality when equations are reduced to solved normal form.

Example 1. Let $S = \{ux^2, xy, vz, wz, xyz\}$. The following is a balanced sharing graph for $t(x) = r(y, z)$ and S:

$$\begin{tikzcd}
\boxed{ux^2}^{0}_{2} \arrow[r] \arrow[dr] & \boxed{xy}^{1}_{1} \arrow[r] & \boxed{vz}^{1}_{0} \\
 & \boxed{xy}^{1}_{1} \arrow[r] & \boxed{wz}^{1}_{0}
\end{tikzcd}$$

where pedices and apices on a sharing group B are respectively the value of $\chi(B, t(x))$ and $\chi(B, r(y, z))$. Therefore $uvwx^4y^2z^2 \in \mathrm{mgu}(S, t(x) = r(y, z))$.

Example 2. Let $S = \{ux^2, xy, vz, wz, xyz\}$ and $U = \{u, v, w, x, y, z\}$. The following is a balanced sharing graph for $x = r(y, y, z)$ and S:

$$\begin{tikzcd}
\boxed{ux^2}^{0}_{2} \arrow[r, bend left] \arrow[r, bend right] & \boxed{xyz}^{3}_{1} \arrow[loop right]
\end{tikzcd}$$

where pedices and apices on a sharing group B are respectively the value of $\chi(B, x)$ and $\chi(B, r(y, y, z))$. Therefore $ux^3yz \in \mathrm{mgu}(S, x = r(y, y, z))$. Note that this sharing group can actually be generated by the substitution $\theta = \{x/r(v_1, v_1, v_2), y/v_2, z/v_2, u/v_1, v/a, w/a\}$ where a is a ground term. It is the case that $\alpha_U(\theta) \subseteq S$ and $\mathrm{mgu}(\theta, x = r(y, y, z))$ performs exactly the unification depicted by the sharing graph.

We define $\mathrm{mgu}(S, \theta)$ with $\theta \in \mathit{ISubst}$ by induction on the number of bindings.

$$\mathrm{mgu}(S, \epsilon) = \epsilon \qquad \mathrm{mgu}(S, \{x/t\} \uplus \theta) = \mathrm{mgu}(\mathrm{mgu}(S, x = t), \theta)$$

Now, we are ready to define the abstract unification in $\mathtt{ShLin}^{\omega}$ as:

$$\mathsf{unif}_{\omega}([S, U_1], \delta, U_2) = [\mathrm{mgu}(S \cup \{\{\!\{v\}\!\} \mid v \in U_2 \setminus U_1\}, \delta), U_1 \cup U_2]$$

provided that $vars(\delta) \subseteq U_2$.

Theorem 1. *The operation* unif_{ω} *is optimal and correct w.r.t.* $\mathsf{unif}_{\mathrm{Ps}}$

Note that the operation unif_{ω} is designed by first extending the domain in order to include all the variables in U_2 and then performing the operation, and that this construction yields to an optimal abstraction of the concrete unification. This is not the case for other abstract domains, e.g. `Sharing`, as shown in [1]. The proof of correctness is by induction on the number of bindings in the substitution. The proof of optimality is more complex and it is based on a notion of parallel abstract unification of multigraphs. We show that parallel unification gives the same results of the iterated use of the single binding unification.

As far as the matching operation is concerning, assume $\alpha_\omega([\Sigma_i, U_i]) = [S_i, U_i]$ for $i \in \{1, 2\}$. If we unify a substitution $\sigma_1 \in \Sigma_1$ with $\sigma_2 \in \Sigma_2$ such that $\sigma_1 \preceq_{U_1} \sigma_2$, then σ_1 will not be further instantiated and thus $\alpha_{U_1}(\mathrm{mgu}(\sigma_1, \sigma_2)) \in S_1$. Moreover, the sharing groups in S_2 which do not contain any variable in U_1 are not affected by the unification, since the corresponding existential variable does not appear in $\sigma_2(v)$ for any $v \in U_1$. We can now design an abstract matching operation which satisfies the above conditions.

$$\mathsf{match}_\omega([S_1, U_1], [S_2, U_2]) = \left[S'_2 \cup \left\{X \in (S''_2)^* \mid X_{|U_1} \in S_1\right\}, U_1\right]$$

where $S'_2 = \{B \in S_2 \mid B_{|U_1} = \emptyset\}$, $S''_2 = S_2 \setminus S'_2$ and $U_1 \subseteq U_2$. Here we also use the auxiliary operation $(\_)^*$ defined as:

$$S^* = \{\uplus \mathcal{S} \mid \mathcal{S} \in \wp_m(S)\} \quad . \tag{9}$$

Note that match_ω is very similar to the analogous operation for `Sharing` defined in [1].

Theorem 2. *The operation* match_ω *is optimal and correct w.r.t.* $\mathsf{match}_{\mathrm{Ps}}$. *Furthermore, it is complete when the second argument of* $\mathsf{match}_{\mathrm{Ps}}$ *contains a single substitution.*

The forward and backward unification operators $\mathbf{U}^f_\omega$ and $\mathbf{U}^b_\omega$for `ShLin`$^\omega$ are obtained by the corresponding definitions $\mathbf{U}^f_{\mathrm{Ps}}$ and $\mathbf{U}^b_{\mathrm{Ps}}$ for `Psub`, by replacing the matching and unification operations with their abstract counterparts. By exploiting the above results, it is now an easy task to show the following corollary.

Corollary 1. *The operators* $\mathbf{U}^f_\omega$ *and* $\mathbf{U}^b_\omega$ *are correct and optimal w.r.t.* $\mathbf{U}^f_{\mathrm{Ps}}$ *and* $\mathbf{U}^b_{\mathrm{Ps}}$.

4.2 A Characterization for Resultant Sharing Groups

The concept of resultant ω-sharing group, while suggestive and very intuitive, does not help in practice in the implementation of the operations. Although `ShLin`$^\omega$ has not been designed to be directly implemented, some of its abstractions could. Providing a simpler definition for the set of resultant ω-sharing groups could help in developing the abstract operators for its abstractions. We show that given a set S of ω-sharing groups and an equation $t_1 = t_2$, the set of resultant ω-sharing groups has an elegant algebraic characterization.

Theorem 3. *Let S be a set of ω-sharing groups and t_1, t_2 be terms. Then* $B \in \mathrm{mgu}(S, t_1 = t_2)$ *iff* $B = \uplus_{i\in I} B_i$ *where I is a finite set and* $\{\!\{B_i\}\!\}_{i\in I} \in \wp_m(S)$ *such that:*

$$\sum_{i\in I} \chi(B_i, t_1) = \sum_{i\in I} \chi(B_i, t_2) \geq |I| - 1 \quad .$$

From the above theorem, we can now give an algebraic characterization of the abstract unification operator as follows.

$$\mathrm{mgu}(S, t_1 = t_2) = \left\{ \Uplus \mathcal{S} \mid \mathcal{S} \in \wp_m(S), \sum_{B \in \mathcal{S}} \chi(B, t_1) = \sum_{B \in \mathcal{S}} \chi(B, t_2) \geq |\mathcal{S}| - 1 \right\} .$$

Example 3. Consider $S = \{xa, xb, z^2, zc\}$ and the equation $x = z$. Then if we choose $X = \{\!\{xa, xb, z^2\}\!\}$, we have $\chi(X, x) = 2 = \chi(X, z) \geq |X| - 1$. Therefore $x^2z^2ab \in \mathrm{mgu}(S, x = z)$. If we take $X = \{\!\{xa, xb, zc, zc\}\!\}$, although $\chi(X, x) = 2 = \chi(X, z)$, we have $|X| - 1 = 3$. Actually, $z^2c^2x^2ab \notin \mathrm{mgu}(S, x = z)$.

5 Domains for Linearity and Aliasing

In this section we show that two domains for sharing analysis with linearity information, namely the domain proposed by King in [14] and the classic reduced product `Sharing`$\times$`Lin`, can be obtained as abstraction of `ShLin`$^\omega$. This allows us to design optimal abstract operators for both domains, by exploiting the results for `ShLin`$^\omega$.

5.1 King's Domain

We first consider the domain for combined analysis of sharing and linearity introduced by King in [14]. We call *2-sharing group* a map $o : \mathcal{V} \to \{0, 1, \infty\}$ such that its support $\llbracket o \rrbracket = \{v \in \mathcal{V} \mid o(v) \neq 0\}$ is finite. We write $o_m(x)$ to denote $o(x)$ if $o(x) \leq 1$, 2 otherwise (where $n \leq \infty$ for each $n \in \mathbb{N}$). Intuitively, a 2-sharing group o represents the sets $\gamma_2(o)$ of ω-sharing group given by:

$$\gamma_2(o) = \{B \in \wp_m(\mathcal{V}) \mid \llbracket o \rrbracket = \llbracket B \rrbracket \wedge \forall x \in \llbracket o \rrbracket . o_m(x) \leq B(x) \leq o(x) \ .$$

We denote by $Sg^2(V)$ the set of 2-sharing groups whose support is a subset of V. We use a polynomial notation for 2-sharing groups as for ω-sharing groups: a group o such that $\llbracket o \rrbracket = \{x, y, z\}$, $o(x) = o(y) = 1$ and $o(z) = \infty$ will be denoted by xyz^∞. We also use $\emptyset$ for the 2-sharing group with empty support.

The idea is to use 2-sharing groups to keep track of linearity: If $o(x) = \infty$, it means that the variable x is not linear in the sharing group o. In [14] the number 2 is used as an exponent instead of ∞, but we prefer this notation to be coherent with ω-sharing groups. In the rest of this subsection, we use the term "sharing group" as a short form of 2-sharing group.

We first need to define an order relation over sharing groups as follows.

$$o \leq o' \iff \llbracket o \rrbracket = \llbracket o' \rrbracket \wedge \forall x \in \llbracket o \rrbracket . \ o(x) \leq o'(x) \ . \tag{10}$$

Given a sharing group o, we also define the *delinearization* operator o^2 as the sharing group $o' \geq o$ such that $\forall x \in \llbracket o \rrbracket . o'(x) = \infty$. The operator is extended pointwise to sets and multisets. The domain we are interested in is the following.

$$\mathtt{ShLin}^2 = \left\{[S, U] \mid S \in \wp_\downarrow(Sg^2(U)), U \in \wp_f(\mathcal{V}), S \neq \emptyset \Rightarrow \emptyset \in S\right\} \cup \{\top_2, \bot_2\} \tag{11}$$

where $\wp_\downarrow(Sg^2(U))$ is the powerset of downward closed 2-sharing groups according to $\leq$ and $[S_1, U_1] \leq_2 [S_2, U_2]$ iff $U_1 = U_2$ and $S_1 \subseteq S_2$. Since we only consider downward closed sets, we are not able to state that some variable is definitively non-linear. We define an adjunction with $\texttt{ShLin}^\omega$ via the following concretization map $\gamma : \texttt{ShLin}^2 \to \texttt{ShLin}^\omega$.

$$\gamma([S,U]) = \left[\bigcup\{\gamma(o) \mid o \in S\}, U\right] \ . \tag{12}$$

The l.u.b. is given by the downward closure of the unions of the first components. Projection is given by:

$$\pi_2([S_1,U_1],[S_2,U_2]) = [\{o_{|U_2} \mid o \in S_1\}, U_1 \cap U_2] \ . \tag{13}$$

where $o_{|X}(v)$ is $o(v)$ if $v \in X$, 0 otherwise. Given two sharing groups o and o' we define:

$$o \Box o' = \lambda v \in \mathcal{V}.o(v) \oplus o'(v) \tag{14}$$

where $0 \oplus x = x \oplus 0 = x$ and $\infty \oplus x = x \oplus \infty = 1 \oplus 1 = \infty$. We will use $\Box\{\!\{o_1,\ldots,o_n\}\!\}$ for $o_1 \Box \cdots \Box o_n$. Note that $o^2 = o \Box o$. According to the corresponding definition for ω-sharing groups, we also need to define the following auxiliary function.

$$S^* = \left\{\Box\mathcal{S} \mid \mathcal{S} \in \wp_m(S)\right\} \ . \tag{15}$$

The minimum and maximum multiplicity of o in t are defined as follows:

$$\chi_m(o,t) = \sum_{v \in \llbracket o \rrbracket} o_m(v) \cdot occ(v,t) \qquad \chi_M(o,t) = \sum_{v \in \llbracket o \rrbracket} o(v) \cdot occ(v,t) \tag{16}$$

If B is an ω-sharing group represented by o, i.e., $B \in \gamma_2(o)$, then $\chi_m(o,t) \leq \chi(B,t) \leq \chi_M(o,t)$. Actually, not all the values between $\chi_m(o,t)$ and $\chi_M(o,t)$ may be assumed by $\chi(B,t)$, but this will not affect the precision of the abstract operators. Note that the maximum multiplicity $\chi_M(o,t)$ either is equal to the minimum multiplicity $\chi_m(o,t)$ or it is infinite. According to the above definitions, we can now define the multiplicity of a multiset of sharing groups.

$$\chi(Y,t) = \Big\{ n \mid \sum_{o \in \llbracket Y \rrbracket} Y(o) \cdot \chi_m(o,t) \leq n \leq \sum_{o \in \llbracket Y \rrbracket} Y(o) \cdot \chi_M(o,t) \Big\} \ . \tag{17}$$

Again, this is a superset of all the possible values which can be obtained by combining the multiplicities of all the sharing groups in Y. But, as we will show later, this definition is sufficiently accurate to allows us to design the optimal abstract unification operator. This can actually be defined as follows.

$$\mathrm{mgu}(S, x = t) = \downarrow\left\{\Box Y \mid Y \subseteq_m S, \exists n \in \chi(Y,x) \cap \chi(Y,t).\ n \geq |Y| - 1\right\} \ , \tag{18}$$

where $\downarrow X$ is the downward closure of X w.r.t. $\leq$. The basic idea is to check, for each $Y \subseteq_m S$, if there exists an instance of the ω-sharing groups of Y which satisfies the condition in Theorem 3.

Example 4. Let $S = \downarrow\{x^\infty a, x^\infty b, x^\infty c, z^\infty\}$ and $Y = \{\!\{x^\infty a, x^\infty b, xc, z^\infty\}\!\}$. We have $\chi(Y,x) = \{n \mid n \geq 5\}$ and $\chi(Y, t(z,z)) = \{n \mid n \geq 4\}$. Since $t(z,z)$

contains two occurrences of z, the "actual" multiplicity of the sharing group z^∞ in $t(z, z)$ should be a multiple of 2. But we do not need to check this condition and can safely approximate this set with $\{n \mid n \geq 4\}$. Intuitively, this works because we can always choose a multiple which is contained in both $\chi(Y, x)$ and $\chi(Y, t)$ and which is an "actual" multiplicity. For instance, we can take $n = 6 \in \chi(Y, x) \cap \chi(Y, t(z, z))$ and since we have $6 \geq 3 = |Y| - 1$, we get that the sharing group $\Box Y = x^\infty abcz^\infty$ belongs to $\text{mgu}(S, x = t(z, z))$. This sharing group can be generated by the substitution $\{x/t(t(a, a, c), t(b, b, c)), z/t(v, v, v)\}$ when the variables of interest are $\{x, z, a, b, c\}$.

By exploiting the particular structure of 2-sharing groups, we can rewrite the mgu operator in a much simpler way, to be used in practice to implement the abstract operator. Given a set of sharing groups S and an equation $t_1 = t_2$, we define, for $i \in \{1, 2\}$ and $j \in \mathbb{N}$, $S_i^j = \{o \in S \mid \chi_M(o, t_i) = j\}$, $S_i = \{o \in S \mid \chi_M(o, t_i) \neq 0\}$, $P_i^j = S_i^j \setminus S_{3-i}$, $P_i = S_i \setminus S_{3-i}$ and $C^i = S_1^i \cap S_2^i$. We also write $S_i^{nl} = \{o \in S \mid \chi_M(o, t_i) > 1\}$. Note that, if t_1 is a variable, $S_1^{nl} = S_1^\infty$. Then, by considering x as t_1 and t as t_2, we can rewrite the mgu operator as follows.

$$
\begin{aligned}
&\text{mgu}(S, x = t) = C^0 \cup \\
&\quad\downarrow\Big(\{\Box X^2 \mid X \subseteq S_1 \cup S_2, X \cap S_1^{nl} \neq \emptyset, X \cap S_2^{nl} \neq \emptyset\} \cup \\
&\qquad \{\Box X^2 \mid X \subseteq S_2^1, X \cap S_1^{nl} \neq \emptyset\} \cup \\
&\qquad \{o \Box (\Box X^2) \mid o \in P_1, X \subseteq S_2^1, X \cap S_1^{nl} \neq \emptyset \vee o \in P_1^\infty, X \cap P_2 \neq \emptyset\} \cup \\
&\qquad \{\Box X^2 \mid X \subseteq S_1^1, X \cap S_2^{nl} \neq \emptyset\} \cup \\
&\qquad \{o \Box (\Box X^2) \mid o \in P_2, X \subseteq S_1^1, X \cap S_2^{nl} \neq \emptyset \vee o \in P_2^\infty, X \cap P_1 \neq \emptyset\} \cup \\
&\qquad \{\Box X^2 \mid X \subseteq C^1\} \cup \\
&\qquad \{o \Box Y \Box (\Box X^2) \mid o \in P_2, X \subseteq C^1, Y \subseteq_m P_1^1, |Y| = \chi_M(o, t) \in \mathbb{N}^+\}\Big)
\end{aligned}
\tag{19}
$$

The seven cases above correspond to the different choices of a multiset $\mathcal{S} \subseteq_m S_1 \cup S_2$ of sharing groups which we want to merge. In the first case, we require that there is at least a non-linear sharing group for x and t, while in the second and third case we only require the existence of a non-linear sharing group for x. The second line corresponds to the case when we do not have any element in P_1, while the third case is applied when there is exactly one element of P_1 in $\mathcal{S}$. The fourth and fifth case are symmetric to the second and third, when all the sharing groups are linear for x and non-linear for t. Note a subtlety of the fifth case, where a non-linear sharing group in S_2^{nl} only needs to have a maximum multiplicity bigger than one, while a non-linear in P_2 needs to have an infinite maximum multiplicity. The sixth and seventh case are applied when all the sharing groups are linear for x and t, but at most elements in P_2 which may have a finite maximum multiplicity. An interesting property of (19) is that it also works when S is not downward closed: If $\downarrow S = \downarrow R$ then $\text{mgu}(S, x = t) = \text{mgu}(R, x = t)$. This means that we do not have to compute and carry on the downward closure of a set but only its maximal elements. This simplifies the implementation of the abstract mgu. Moreover, the seven cases are obtained by disjoint choices of the multiset $\mathcal{S} \subseteq_m S_1 \cup S_2$ of sharing groups, to avoid as much as possible any duplication.

Example 5. Let $S = \{x^\infty y, y^\infty b, y, xa, z\}$ and consider the equation $x = y$. By the first case of (19) we obtain $x^\infty y^\infty b^\infty$ and $x^\infty y^\infty b^\infty a^\infty$. From the second and third case we obtain respectively $x^\infty y^\infty$ and $x^\infty y^\infty a$. The fourth and sixth case do not generate any sharing group, while from the fifth and seventh we have respectively $y^\infty x^\infty a^\infty b$ and xya, which are redundant. We also add the original sharing group z which is not related to either x nor y. The final result is $\mathrm{mgu}(S, x = y) = \downarrow\{x^\infty y^\infty b^\infty, x^\infty y^\infty b^\infty a^\infty, x^\infty y^\infty, x^\infty y^\infty a, z\}$. It is worth noting that $S \neq \downarrow S$ and that $\mathrm{mgu}(S, x = y) = \mathrm{mgu}(\downarrow S, x = y)$.

We can now define the abstract unification on $\mathtt{ShLin}^2$ by enlarging the domain before computing the mgu:

$$\mathsf{unif}_2([S, U_1], \theta, U_2) = [\{\mathrm{mgu}(S \cup \{\{\!\{v\}\!\} \mid v \in U_2 \setminus U_1\}, \theta\}, U_2] \qquad (20)$$

The abstract matching follows the same pattern as match_ω, and it is defined as:

$$\mathsf{match}_2([S_1, U_1], [S_2, U_2]) = \left[S'_2 \cup \downarrow\left\{o \in (S''_2)^* \mid o_{|U_1} \in S_1\right\}, U_2\right] \qquad (21)$$

where $S'_2 = \{B \in S_2 \mid B_{|U_1} = \emptyset\}$, $S''_2 = S_2 \setminus S'_2$ and $U_1 \subseteq U_2$.

We can prove that both the operators are correct and optimal, and match_2 is complete w.r.t. single substitutions in its second argument. These conditions suffice to prove that the $\mathbf{U}_2^f$ and $\mathbf{U}_2^b$ are correct and optimal.

5.2 The Domain `Sharing` × `Lin`

In this section we deal with the reduced product $\mathtt{ShLin} = \mathtt{Sharing} \times \mathtt{Lin}$. We briefly recall the definition of the abstract domain and show the abstraction function from King's domain $\mathtt{ShLin}^2$ to $\mathtt{ShLin}$.

$$\begin{aligned}\mathtt{ShLin} = \{[S, L, U] \mid S \subseteq \wp(U), (S \neq \emptyset \Rightarrow \emptyset \in S),\\ L \supseteq U \setminus \mathit{vars}(S), U \in \wp_f(\mathcal{V})\} \cup \{\bot_{sl}, \top_{sl}\}\ .\end{aligned}$$

Moreover, $[S, L, U] \leq_{sl} [S', L', U']$ iff $U = U'$, $S \subseteq S'$, $L \supseteq L'$. The abstraction map from $\mathtt{ShLin}^2$ to $\mathtt{ShLin}$ is defined in the obvious way.

$$\alpha([S, U]) = [\{\llbracket o \rrbracket \mid o \in S\}, \{x \mid \forall o \in S.\ o(x) \leq 1\}, U]\ . \qquad (22)$$

We call *sharing group* an element of $\wp_f(\mathcal{V})$. As for the previous domains, we use the polynomial notation to represent sharing groups, but now all the exponents are fixed to one. Note that the last component U is redundant since it can be computed as $L \cup \mathit{vars}(S)$. The abstract operator for projection is straightforward.

$$\pi_{sl}([S_1, L_1, U_1], [S_2, L_2, U_2]) = [\{B \cap U_2 \mid B \in S_1\}, L_1 \cap U_2, U_1 \cap U_2]\ . \qquad (23)$$

As far as the abstract unification is concerning, we want to design an abstract operator over $\mathtt{ShLin}$ which is optimal for the unification of a single binding. Fixed a set L of linear variables, we define the maximum multiplicity of a sharing group B in a term t as follows:

$$\chi_M^L(B,t) = \begin{cases} \sum_{v \in B} occ(v,t) & \text{if } B \cap vars(t) \subseteq L \\ \infty & \text{otherwise} \end{cases} \tag{24}$$

We also define the maximum multiplicity of a term t in (S, L) as:

$$\chi(S,L,t) = \max_{B \in S} \chi_M^L(B,t) \ . \tag{25}$$

Then we define the abstract unification $\mathrm{mgu}(S, L, x = t)$ as the pair (S', L') where S' is computed as in (19) with χ_M and $\Box$ replaced by χ_M^L and $\cup$ respectively (we can obviously ignore the delinearization operator $(\_)^2$ since $B \cup B = B$). The set L' is computed according to the following definition:

$$L' = (U \setminus vars(S')) \cup \begin{cases} L \setminus (vars(S_1) \cap vars(S_2)) & \text{if } x \in L \text{ and } \chi(S,L,t) \leq 1 \\ L \setminus vars(S_1) & \text{otherwise, if } x \in L \\ L \setminus vars(S_2) & \text{otherwise, if } \chi(S,L,t) \leq 1 \\ L \setminus (vars(S_1) \cup vars(S_2)) & \text{otherwise} \end{cases} \tag{26}$$

where $S_1 = \{B \in S \mid \chi_M^L(B,x) \neq 0\}$ and $S_2 = \{B \in S \mid \chi_M^L(B,t) \neq 0\}$.

Example 6. Let $S = \{xv, xy, zw\}, L = \{x, y, v, w\}$ and consider the binding $x = t(y, z)$. Then $\chi_M^L(xv, t) = 0$, since $xv \cap vars(t) = \emptyset$, $\chi_M^L(xy, t) = 1$ and $\chi_M^L(zw, t) = \infty$. As a result $\chi(S, L, t) = \infty$. In words, it means that the sharing group zw is not linear in t and that t itself is not linear. Note that all the other sharing groups are linear w.r.t. x since $x \in L$. Applying equation (19) as stated above, we obtain $S' = \{xy, xyzvw, xzvw\}$ and $L' = \{w\}$. This is more precise that the standard operators for `Sharing` $\times$ `Lin` [9]. Actually even with the optimizations proposed in [12, 11] or [3], the standard operator is not able to infer that the sharing group xyv is not a possible result of the concrete unification. Note that it would be possible in a domain for rational trees, where the unification of $\{x/t(t(v, y), c), z/w\}$ with $x/t(y, z)$ succeeds with $\{x/t(t(v, y), c), z/c, w/c, y/t(v, y)\}$. This means that we are able to exploit the occur-check of the unification in finite trees. As a consequence, our abstract unification operator is not correct w.r.t. a concrete domain of rational substitutions [15]. However, our results improve over the abstract unification operators of the domains in the literature even in some cases which do not involve the occur-check. For example, if $S = \{xa, xb, xy, z\}$, $L = \{x, a, b, z\}$ and given the binding $x/t(y, z)$, we are able to state that $xzab$ is not a member of $\mathrm{mgu}(S, x = t(y, z))$, but the domains in [12, 3, 11] cannot.

Although the abstract operator for $\mathrm{mgu}(S, L, x = t)$ over `ShLin` is optimal for the unification with a single binding, the optimal operator $\mathrm{mgu}(S, L, \theta)$ for a generic substitution θ cannot be obtained by considering one binding at a time. Actually, let us take $\theta = \{x/t(y, z), s/t(a, b)\}$, $S = \{xs, y, z, a, b\}$, $L = \{x, z, s, a, b\}$. After the first binding we obtain $S' = \{xsy, xsz, a, b\}$ and $L' = \{z, a, b\}$. After the second binding $S'' = \{xsya, xsyb, xsyab, xsza, xszb, xszab\}$

and $L'' = \{z\}$. However, the sharing group $xszab$ cannot be obtained in practice since x is linear in the sharing group xsz, although $x \notin L'$. If we work in the domain $\mathtt{ShLin}^2$ from $Z = \{xs, y^\infty, y, z, a, b\}$, we obtain $Z' = \downarrow\{x^\infty s^\infty y^\infty, xsz, a, b\}$ and $Z'' = \downarrow\{x^\infty s^\infty y^\infty a^\infty b^\infty, x^\infty s^\infty y^\infty a^\infty, x^\infty s^\infty y^\infty b^\infty, xsza, xszb\}$.

In order to obtain an optimal operator for a substitution θ, the obvious solution is to perform the computation over $\mathtt{ShLin}^2$ and to abstract the solution into $\mathtt{ShLin}$. We believe that the implementation of the abstract unification on $\mathtt{ShLin}^2$ could be optimized for this particular case. The same happens to the matching operation. Also in this case, we believe that the easiest approach is to compute over $\mathtt{ShLin}^2$, which is not particularly onerous since the abstract constraint does not increase in size when moving from $\mathtt{ShLin}$ to $\mathtt{ShLin}^2$. Corresponding definitions for unif_{sl}, match_{sl}, $\mathbf{U}^f_{sl}$ and $\mathbf{U}^b_{sl}$ are immediate.

6 Conclusion and Future Work

We summarize the main results of this paper.

- We clarify the relationship between domains of substitutions with existentially quantified variables, such as *ESubst* [13], and the standard domain of idempotent substitutions. To the best of our knowledge, this is the first time that a direct correspondence between *ESubst* and a quotient of idempotent substitutions has been showed.
- We propose a new domain $\mathtt{ShLin}^\omega$ as a general framework for investigating sharing and linearity properties. We introduce the notion of *(balanced) sharing graph* as a generalization of the concept of alternating path [20, 15] used for pair sharing analysis and provide optimal abstract operators for $\mathtt{ShLin}^\omega$.
- We show that $\mathtt{ShLin}^\omega$ is a useful starting point for studying further abstractions. We obtain the optimal operators for forward and backward unification in $\mathtt{Sharing} \times \mathtt{Lin}$ and King's domain $\mathtt{ShLin}^2$. This is the first paper which shows optimality results for a domain obtained by combining sharing and linearity information. Actually in [5] a variant of $\mathtt{Sharing} \times \mathtt{Lin}$ is proposed, based on *set logic programs*. However, despite the claim in the paper, the proposed operators are not optimal, as shown in [11]. Also the operators in [15] for `ASub` are not optimal when working over finite trees.

Several things remain to be explored: first of all, we plan to analyze the domain $\mathtt{PSD} \times \mathtt{Lin}$ [2] in our framework and, possibly, to devise a variant of $\mathtt{ShLin}^2$ which enjoys a similar closure property for redundant sharing groups. This could be of great impact on the efficiency of the analysis. Moreover, we need to study the impact on the precision and performance by adopting the new optimal operators, possibly by implementing our operators in some well-known static analyzer.

In the recent years, many efforts has been made to study the behavior of logic programs in the domain of *rational trees* [15, 21], since they formalize the standard implementations of logic languages. We have shown that our operators, which are optimal for finite trees, are not correct for rational trees , since they exploit the occur-check to reduce the sharing groups generated by the abstract unification (see Ex. 6). It would be interesting to adapt our framework to work with rational trees, in order to obtain optimal operators also for this case.

References

1. G. Amato and F. Scozzari. Optimality in goal-dependent analysis of sharing. Technical Report TR-02-06, Dipartimento di Informatica, Univ. di Pisa, May 2002.
2. R. Bagnara, P. Hill, and E. Zaffanella. Set-sharing is redundant for pair-sharing. *Theoretical Computer Science*, 2002. To appear.
3. R. Bagnara, E. Zaffanella, and P. M. Hill. Enhanced sharing analysis techniques: A comprehensive evaluation. In *Proc. of ACM Conf. PPDP*, pp. 103–114, 2000.
4. M. Codish, D. Dams, and E. Yardeni. Derivation and safety of an abstract unification algorithm for groundness and aliasing analysis. In *ICLP*, pp. 79–93, 1991.
5. M. Codish, V. Lagoon, and F. Bueno. An algebraic approach to sharing analysis of logic programs. In *Static Analysis Symposium*, pp. 68–82, 1997.
6. A. Cortesi, G. Filé, and W. W. Winsborough. Optimal groundness analysis using propositional logic. *Journal of Logic Programming*, 27(2):137–167, 1996.
7. P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Proc. ACM POPL*, pp. 269–282, 1979.
8. P. Cousot and R. Cousot. Abstract Interpretation and Applications to Logic Programs. *Journal of Logic Programming*, 13(2 & 3):103–179, 1992.
9. W. Hans and S. Winkler. Aliasing and groundness analysis of logic programs through abstract interpretation and its safety. Technical Report 92–27, Technical University of Aachen (RWTH Aachen), 1992.
10. M. V. Hermenegildo and F. Rossi. Strict and nonstrict independent and-parallelism in logic programs: Correctness, efficiency, and compile-time conditions. *Journal of Logic Programming*, 22(1):1–45, 1995.
11. P. M. Hill, E. Zaffanella, and R. Bagnara. A correct, precise and efficient integration of set-sharing, freeness and linearity for the analysis of finite and rational tree languages. Available at http://www.cs.unipr.it/~bagnara/.
12. J. Howe and A. King. Three Optimisations for Sharing. Technical Report 11-01, Computing Laboratory, Univ. of Kent at Canterbury, 2001. To appear in TPLP.
13. D. Jacobs and A. Langen. Static Analysis of Logic Programs for Independent AND Parallelism. *Journal of Logic Programming*, 13(2 & 3):291–314, 1992.
14. A. King. A synergistic analysis for sharing and groundness which traces linearity. In *ESOP*, vol. 788 of *LNCS*, pp. 363–378, 1994.
15. A. King. Pair-sharing over rational trees. *JLP*, 46(1-2):139–155, Nov. 2000.
16. A. King and M. Longley. Abstract matching can improve on abstract unification. Technical Report 4-95*, Computing Laboratory, Univ. of Kent, Canterbury, 1995.
17. A. Langen. *Static Analysis for Independent And-parallelism in Logic Programs.* PhD thesis, University of Southern California, Los Angeles, California, 1990.
18. K. Marriott, H. Søndergaard, and N. D. Jones. Denotational abstract interpretation of logic programs. *ACM TOPLAS*, 16(3):607–648, 1994.
19. K. Muthukumar and M. V. Hermenegildo. Compile-time derivation of variable dependency using abstract interpretation. *JLP*, 13(2&3):315–347, 1992.
20. H. Søndergaard. An application of abstract interpretation of logic programs: Occur check reduction. In *Proc. ESOP 86*, vol. 213 of LNCS, pp. 327–338, 1986.
21. E. Zaffanella. *Correctness, Precision and Efficiency in the Sharing Analysis of Real Logic Languages.* PhD thesis, School of Computing, University of Leeds, Leeds, U.K., 2001. Available at http://www.cs.unipr.it/~zaffanella/.