$Share \times Free$ Revisited

Andy King

Computing Laboratory,
University of Kent at Canterbury,
Canterbury, CT2 7NF, UK.
e-mail: amk1@ukc.ac.uk

Abstract

Analyses for possible variable sharing and definite freeness are important both in the automatic parallelisation and in the optimisation of sequential logic programs. In this paper, a new efficient approach to analysis is described which can infer sharing and freeness information to an unusually high degree of accuracy. The analysis exploits a confluence property of the unification algorithm to split the analysis into two distinct phases. The two phase analysis improves efficiency by enabling each phase of the analysis to manipulate relatively simple data-structures. The precision follows from the combination of domains. The analysis propagates groundness with the accuracy of sharing groups and yet infers sharing and freeness to a precision which exceeds that of a normal freeness analysis. High precision compoundness information can be derived too. The usefulness of the analysis is demonstrated with worked examples. Correctness is formally proven.

1 Introduction

Abstract interpretation for sharing and freeness are important topics of logic programming. Sharing (or aliasing) analysis conventionally infers which program variables are definitely grounded and which variables can possibly be bound to terms containing a common variable. Freeness analysis usually infers which program variables are free, that is, which variables can never be bound to non-variable terms. Compoundness analysis [21, 9] is the dual of freeness analysis in that it detects which variables are guaranteed to be bound to non-variable (compound) terms. Compoundness analysis, as applied in [9], additionally traces the principal functor of variable bindings. Compoundness information can aid indexing. Applications of sharing and freeness information are numerous and include: the sound removal of the occur-check [25]; optimisation of backtracking [5]; the specialisation of unification [27]; and the identification [28, 13] and efficient exploitation [23, 14, 24] of independent and-parallelism. Early proposals for sharing, freeness and compoundness analyses include [29, 12, 20], [23] and [21].

This paper is concerned with a semantic basis for sharing, freeness and compoundness analysis, and in particular, the justification of a high precision abstract unification algorithm. Following the approach of abstract interpretation [10], an abstract unification algorithm (the abstract operation) is constructed by mimicking the substitutions (the concrete data) which arise in a standard unification algorithm [17] (the concrete operation) with finite sharing, freeness and compoundness abstractions (the abstract data).

The accuracy of the analysis depends, in part, on the substitution properties that the sharing abstractions capture. The popular sharing and freeness domain $Share \times Free$ [23], for instance, captures possible sharing and definite groundness in its $Share$ component; and definite freeness in its $Free$ component. The structure of $Share$ is particularly rich, implicitly encoding covering information [9]. Covering, in short, permits groundness to interact nicely with sharing to remove redundant aliasing. For finiteness, $Share \times Free$ is parametrised by a finite set of program variables, $Pvar$, which typically equate to the variables of a clause. To be precise, $Share_{Pvar} = \wp(\wp(Pvar))$ and $Free_{Pvar} = \wp(Pvar)$. If $Pvar = \{x, y, z\}$, for instance, the substitution $\phi = \{x \mapsto f(a, y, b),\ z \mapsto g(y)\}$ would be encoded by the pair $\langle \phi^{\mathcal{S}}, \phi^{\mathcal{F}} \rangle$ where $\phi^{\mathcal{S}} = \{\{x,\ y,\ z\},\ \emptyset\}$ and $\phi^{\mathcal{F}} = \{y\}$. The pair indicates that a variable occurs through x, y and z, that is, they share; and that y is free.

Tracking freeness often brings a two-fold win: first, it enlarges the class of ensuing optimisations [28, 13]; second, it improves the groundness and sharing [23]. Groundness and sharing is refined since freeness relates to the structural or type properties of a substitution. Precision is improved as a result of the synergy between sharing and type analysis. By keeping track of type information, it is possible to infer more accurate sharing information. Conversely, more accurate type information can be deduced if sharing is traced. Specifically, by tracking freeness [23, 9, 26] (or alternatively a type property called linearity [25, 7, 15]), a sharing analysis does not always have to assume that aliasing is transitive [7]. If variables can be inferred to be free, worst case aliasing need not be assumed in an analysis.

$Share_{Pvar} \times Free_{Pvar}$, however, only captures shallow type information: it traces the freeness of terms to which a variable can be bound but not the freeness of sub-terms. The usefulness of tracing sharing and freeness to the level of sub-terms has been reported before [16] but the analysis proposed in [16] is difficult to implement efficiently. This paper remedies this deficiency by augmenting the domain $Share_{Pvar} \times Free_{Pvar}$ with a simple but powerful type component $Sub^{\mathcal{C}}_{Pvar}$, and by adopting a new, modular approach to analysis. The domain $Sub^{\mathcal{C}}_{Pvar}$ consists of a set of canonical substitutions on $Pvar$ which encode structure. The composite domain is a subset of $Sub^{\mathcal{C}}_{Pvar} \times Share_{Pvar} \times Free_{Pvar}$, denoted $Type_{Pvar}$, and might typically represent ϕ by

$$\langle \phi^{\mathcal{C}}, \phi^{\mathcal{S}}, \phi^{\mathcal{F}} \rangle \text{ where } \begin{array}{l} \phi^{\mathcal{C}} = \{x \mapsto f(a, x^2, b), y \mapsto y^{\lambda}, z \mapsto g(z^1)\}, \\ \phi^{\mathcal{S}} = \{\{x^2, y^{\lambda}, z^1\}, \emptyset\}, \\ \phi^{\mathcal{F}} = \{x^2, y^{\lambda}\} \end{array}$$

The triple indicates sharing between the second argument of term bound to x (x^2), y (y^{λ}), and the first argument of the term bound to z (z^1). The triple also records the freeness of both x^2 and y^{λ}. It thus represents sharing and freeness to the precision of sub-terms. In addition, it captures useful compoundness information too, for instance, that x is bound to a term with a principal functor f and an arity of 3.

In contrast to other approaches [16], high precision does not come at the expense of gross inefficiency. The analysis exploits a confluence property of the unification algorithm (that all unifiers are equal up to renaming [17]) to split the analysis into two distinct phases. In the first phase compoundness information is tracked. In the second phase sharing and freeness is traced. The compoundness phase only operates on the compoundness component of the domain. Similarly, the sharing and freeness phase only operates on the sharing and freeness component of the domain. Since each phase of the analysis need only manipulate its own (relatively simple) data-structure, efficiency is maintained without sacrificing precision. The modularity also leads to a well-structured proof of correctness.

The exposition is structured as follows. Section 2 describes the notation and preliminary definitions which will be used throughout. In section 3, the focus is on abstracting data. An abstraction for substitutions is constructed which expressively captures sharing, freeness and compoundness properties of substitutions. In section 4, the emphasis changes to abstracting operations. Abstract analogs for renaming, unification, composition and restriction are defined in terms of an abstract *unify* operator [14]. An abstract unification algorithm is defined which, in turn, describes an abstract analog of *unify*. (Once an abstract *unify* operator is specified and proved safe, a complete and correct abstract interpreter is practically defined by virtue of existing abstract interpretation frameworks [2, 18, 24].) Correctness is also proved. For reasons of brevity and continuity, however, the proofs are relegated to an appendix, section 7. Finally, sections 5 and 6 present the related work and the concluding discussion.

2 Notation and preliminaries

To introduce the analysis some notation and preliminary definitions are required. The reader is assumed to be familiar with the standard constructs used in logic programming [19] such as a universe of (possibly super- and sub-scripted) variables $(u, v \in) Uvar$; the set of terms $(t \in) Term$ formed from $Uvar$ and the set of functors $(f, g, h \in) Func$ (of the first-order language underlying the program), and the set of program atoms $Atom$. $Func$ is considered to include the set of constants $Const$. It is sometimes convenient to abbreviate $f(t_1, \ldots, t_n)$ to $f(\underline{t_i})$.

Let $Pvar$ denote a finite set of program variables – the variables that are in the text of the program; and let $var(o)$ denote the set of variables in a syntactic object o. Also suppose that the set of finite sequences of positive integers is denoted by $\mathcal{S} = \{1, 2, \ldots\}^*$. $\mathcal{S}$ is considered to include the empty sequence λ. If $\cdot$ denotes concatenation of sequences, the application of a term t to the sequence s, $t[s]$, can be defined by the partial mapping $t[\lambda] = t$ and $f(t_1, \ldots, t_n)[i{\cdot}s] = t_i[s]$. Hence, $f(u, g(v))[2{\cdot}1{\cdot}\lambda] = v$, $f(u, g(v))[1{\cdot}\lambda] = u$, and $f(u, g(v))[\lambda] = f(u, g(v))$.

2.1 Substitutions

A substitution ϕ is a total mapping $\phi : Uvar \to Term$ such that its domain $dom(\phi) = \{u \in Uvar \mid \phi(u) \neq u\}$ is finite. The application of a substitution ϕ to a variable u is denoted by $\phi(u)$. Thus the codomain is give by $cod(\phi) = \cup_{u \in dom(\phi)} var(\phi(u))$. A substitution ϕ is sometimes represented as a finite set of variable and term pairs $\{u \mapsto \phi(u) \mid u \in dom(\phi)\}$. The identity mapping on $Uvar$ is called the empty substitution and is denoted by ϵ. Substitutions, sets of substitutions, and the set of substitutions on $Uvar$ are denoted by lower-case Greek letters, upper-case Greek letters, and Sub.

Substitutions are extended in the usual way from variables to functions, from functions to terms, and from terms to atoms. The restriction of a substitution ϕ to a set of variables $U \subseteq Uvar$ and the composition of two substitutions ϕ and φ, are denoted by $\phi \restriction U$ and $\phi \circ \varphi$ respectively, and defined so that $(\phi \circ \varphi)(u) = \phi(\varphi(u))$. Restriction lifts to sets of substitutions by: $\Phi \restriction U = \{\phi \restriction U \mid \phi \in \Phi\}$. The preorder Sub $(\sqsubseteq)$, ϕ is more general than φ, is defined by: $\phi \sqsubseteq \varphi$ if and only if there exists a substitution $\psi \in Sub$ such that $\varphi = \psi \circ \phi$. The preorder induces an equivalence relation $\approx$ on Sub, that is: $\phi \approx \varphi$ if and only if $\phi \sqsubseteq \varphi$ and $\varphi \sqsubseteq \phi$.

A useful related preorder Sub $(\sqsubseteq_{Pvar})$ is defined by: $\phi \sqsubseteq_{Pvar} \varphi$ if and only if there exists a substitution $\psi \in Sub$ such that $\varphi \restriction Pvar = (\psi \circ \phi) \restriction Pvar$.

2.2 Equations and most general unifiers

An equation is an equality constraint of the form $a = b$ where a and b are terms or atoms. Let $(e \in) Eqn$ denote the set of finite sets of equations. The equation set $\{e\} \cup E$, following [7], is abbreviated by $e\!:\!E$. There is a natural mapping from substitutions to equations, that is, $eqn(\phi) = \{u = t_u \mid u \mapsto t_u \in \phi\}$. Thus, when unambiguous, substitutions will be expressed as equations. The set of most general unifiers of E, $mgu(E)$, is defined operationally [14] in terms of a predicate mgu. The predicate $mgu(E, \phi)$ which is true if ϕ is a most general unifier of E.

Definition 2.1 (mgu) *The set of most general unifiers $mgu(E) \in \wp(Sub)$ is defined by: $mgu(E) = \{\phi \mid mgu(E, \phi)\}$ where*

$$\begin{array}{r}
mgu(\emptyset, \epsilon) \\
mgu(v\!=\!v\!:\!E, \zeta)\; if\, mgu(E, \zeta) \\
mgu(t\!=\!v\!:\!E, \zeta)\; if\, mgu(v\!=\!t\!:\!E, \zeta) \\
mgu(v\!=\!t\!:\!E, \zeta \circ \eta)\; if\, mgu(\eta(E), \zeta) \wedge v \notin var(t) \wedge \eta = \{v \mapsto t\} \\
mgu(f(\underline{t_i}) = f(\underline{t_i'})\!:\!E, \zeta)\; if\, mgu(\{t_i = t_i'\}_{i=1}^{n} \cup E, \zeta)
\end{array}$$

By induction it follows that $dom(\phi) \cap cod(\phi) = \emptyset$ if $\phi \in mgu(E)$, or put another way, that the most general unifiers are idempotent [17].

Following [14], the semantics of a logic program is formulated in terms of a single $unify$ operator. To construct $unify$, and specifically to rename apart program variables, an invertible substitution [17], Υ, is introduced. It is convenient to let $Rvar$ denote a universe of renaming variables distinct from $Uvar$, $Uvar \cap Rvar = \emptyset$, and suppose that $\Upsilon : Uvar \to Rvar$. Υ is assumed to consistently rename super-scripted variables, that is, $\Upsilon(u^s) = \Upsilon(u)^s$.

Definition 2.2 ($unify$) *The partial mapping $unify : Atom \times Sub \times Atom \times Sub \to Sub$ is defined by:*

$$unify(a, \phi, b, \psi) = (\varphi \circ \phi) \restriction Pvar \text{ where } \varphi \in mgu(\{\phi(a) = \Upsilon(\psi(b))\})$$

To approximate the $unify$ operation it is convenient to introduce a collecting semantics, concerned with sets of substitutions, to record the substitutions that occur at various program points. In the collecting semantics interpretation, $unify$ is extended to $unify^c$, which manipulates (possibly infinite) sets of substitutions.

Definition 2.3 ($unify^c$) *The mapping $unify^c : Atom \times \wp(Sub) \times Atom \times \wp(Sub) \to \wp(Sub)$ is defined by:*

$$unify^c(a, \Phi, b, \Psi) = \{\theta \mid \phi \in \Phi \wedge \psi \in \Psi \wedge \theta = unify(a, \phi, b, \psi)\}$$

2.3 Linearity

Linearity relates to the number of times a variable occurs in a term [25, 7, 15]. A term is linear if it definitely does not contain multiple occurrences of a variable; otherwise it is non-linear. The significance of linearity is that the unification of linear terms only yields restricted forms of aliasing. This is exploited (in proof 7.10) to simplify the proof of correctness. To be more precise about linearity, it is necessary to introduce the variable multiplicity of a term t, denoted $\chi(t)$.

Definition 2.4 (variable multiplicity, χ [7]) *The variable multiplicity operator $\chi : Term \rightarrow \{0, 1, 2\}$ is defined by:*

$$\chi(t) = \max(\{\chi_u(t) \mid u \in Uvar\}) \text{ where } \chi_u(t) = \begin{cases} 0 \text{ if } u \text{ does not occur in } t \\ 1 \text{ if } u \text{ occurs only once in } t \\ 2 \text{ if } u \text{ occurs many times in } t \end{cases}$$

Lemma 2.1 states one restriction on a most general unifier which follow from unification with a linear term.

Lemma 2.1 $\chi(b) \neq 2 \wedge var(a) \cap var(b) = \emptyset \wedge \phi \in mgu(\{a = b\}) \Rightarrow$

1. $\forall u, u' \in Uvar\,.\, u \neq u' \wedge var(\phi(u)) \cap var(\phi(u')) \neq \emptyset \Rightarrow u \notin var(a) \vee u' \notin var(a)$.

Lemma 2.1 represents one case of a three part result which is formally established in [15]. The lemma differs from the corresponding lemma in [7] (lemma 2.2) because lemma 2.1 requires that a and b do not share variables. This is essentially a work-around for a subtle mistake in lemma 2.2 [11].

3 Abstracting substitutions

Abstract interpretation clarifies how data is represented in the abstract by requiring the relationship between the data and the abstract data to be made explicit. Because of the compositional nature of $Type_{Pvar}$ this relationship is detailed in three steps. In the first step, section 3.1, the focus is on the $Sub^{\mathcal{C}}_{Pvar}$ component and its abstraction and concretisation mappings. Section 3.2, the second step, is concerned with the $Share_{Pvar} \times Free_{Pvar}$ component. Finally the third step, section 3.3, explains how the $Sub^{\mathcal{C}}_{Pvar}$, $Share_{Pvar}$ and $Free_{Pvar}$ components fit together to produce $Type_{Pvar}$.

3.1 Abstracting substitutions with $Sub^{\mathcal{C}}_{Pvar}$

The ϕ example of section 1 captures the compoundness of ϕ in $\phi^{\mathcal{C}} = \{x \mapsto f(a, x^2, b), y \mapsto y^{\lambda}, z \mapsto g(z^1)\}$ and the sharing and freeness of the variables $cod(\phi^{\mathcal{C}}) = \{x^2, y^{\lambda}, z^1\}$ by $\phi^{\mathcal{S}}$ and $\phi^{\mathcal{F}}$. The intuition behind the compoundness abstraction $\phi^{\mathcal{C}}$ is that it represents substitutions ϕ for which: $\phi^{\mathcal{C}}$ is more general than any ϕ. One advance of this approach is that it permits compoundness to be expressed in simple terms. One disadvantage, however, is that since Sub is a pre-order (rather than a poset), the compoundness of ϕ could equally be represented by $\{x \mapsto f(a, u, b), y \mapsto v, z \mapsto g(w)\}$. Thus, in general, there is not a unique best substitution for representing the compoundness of ϕ.

3.1.1 On the domain $Sub^{\mathcal{C}}_{Pvar}$

Representation problems can be straightforwardly avoided, however, by defining the compoundness domain to coincide with a poset of substitutions. One suitable and particularly simple poset can be constructed from substitutions with codomain variables super-scripted by integer sequences drawn from $\mathcal{S}$.

To be more precise, if $Uvar^{\mathcal{S}}$ denotes the variables of $Uvar$ which are super-scripted by $\mathcal{S}$, then the substitutions of the poset are defined to map variables of $Uvar$ into a set of terms with super-scripted variables dubbed $Term^{\mathcal{S}}$. $Term^{\mathcal{S}}$ is formally defined by: $Term^{\mathcal{S}} = \{t \in Term \mid var(t) \subseteq Uvar^{\mathcal{S}}\}$. In fact, to conform to a poset, the variables must be consistently super-scripted. The particular notion of consistency is formalised by the term set $Term^{\lambda}_{u}$.

Definition 3.1 ($Term^{s}_{u}$) *$Term^{s}_{u}$ is the least set defined by:*

$$Term^{s}_{u} = \{u^{s}\} \cup \left\{ f(t_1, \ldots, t_n) \in Term^{s}_{u} \;\middle|\; \begin{array}{c} f \in Func \wedge \\ t_1 \in Term^{s \cdot 1}_{u} \wedge \ldots \wedge t_n \in Term^{s \cdot n}_{u} \end{array} \right\}$$

In $Term^{s}_{u}$ variables are consistently super-scripted in the sense that the position p of a variable in a term t is give by its super-script, that is, $t[p] = u^{s \cdot p}$.

Example 3.1 *Returning to the ϕ example of section 1, observe that the terms $\phi^{\mathcal{C}}(x)$, $\phi^{\mathcal{C}}(y)$ and $\phi^{\mathcal{C}}(z)$ are consistently super-scripted in that*

$$f(a, x^2, b) \in Term^{\lambda}_{x}, \quad y^{\lambda} \in Term^{\lambda}_{y}, \quad g(z^1) \in Term^{\lambda}_{z}$$

To flag the domain and codomain of a substitution, it is helpful to identify three classes of substitution. Specifically, let $\phi^{\mathcal{D}} \in Sub^{\mathcal{D}}$ denote substitutions for which $\phi^{\mathcal{D}} : Uvar^{\mathcal{S}} \to Term$; let $\phi^{\mathcal{C}} \in Sub^{\mathcal{C}}$ denote substitutions such that $\phi^{\mathcal{C}} : Uvar \to Term^{\mathcal{S}}$; and finally let $\phi^{\mathcal{B}} \in Sub^{\mathcal{B}}$ denote substitutions with $\phi^{\mathcal{B}} : Uvar^{\mathcal{S}} \to Term^{\mathcal{S}}$. Informally, the $\mathcal{D}$, $\mathcal{C}$ and $\mathcal{B}$ indicate whether or not the domain, codomain or both annotated with sequence information.

The domain $Sub^{\mathcal{C}}_{Pvar}$ is finally fleshed out by restricting the variable term bindings $u \mapsto t_u$ of $Sub^{\mathcal{C}}$ to a particular form where $u \in Pvar$ and $t_u \in Term^{\lambda}_{u}$.

Definition 3.2 ($Sub^{\mathcal{C}}_{Pvar}$) *The domain $Sub^{\mathcal{C}}_{Pvar}$ is defined by:*

$$Sub^{\mathcal{C}}_{Pvar} = \left\{ \phi^{\mathcal{C}} \in Sub^{\mathcal{C}} \;\middle|\; \begin{array}{c} dom(\phi^{\mathcal{C}}) = Pvar \wedge \\ \forall u \mapsto t_u \in \phi^{\mathcal{C}} . \, t_u \in Term^{\lambda}_{u} \end{array} \right\}$$

The significance of the construction is that $Sub^{\mathcal{C}}_{Pvar}$ ($\sqsubseteq_{Pvar}$) is a poset rather than a pre-order. Moreover, when equipped with a top element $fail^{\mathcal{C}}$, a lub $\sqcup^{\mathcal{C}}$ (corresponding to unification) and a glb $\sqcap^{\mathcal{C}}$ (corresponding to anti-unification [17]), $Sub^{\mathcal{C}}_{Pvar} \cup \{fail^{\mathcal{C}}\}$ ($\sqsubseteq_{Pvar}$) is a complete lattice.

3.1.2 On the mappings $\alpha^{\mathcal{C}}_{Pvar}$ and $\gamma^{\mathcal{C}}_{Pvar}$

The mappings $\alpha^{\mathcal{C}}_{Pvar}$ and $\gamma^{\mathcal{C}}_{Pvar}$ are introduced to explain how compoundness abstractions connect with substitutions.

Definition 3.3 ($\alpha^{\mathcal{C}}_{Pvar}$ **and** $\gamma^{\mathcal{C}}_{Pvar}$) *The abstraction and concretisation mappings* $\alpha^{\mathcal{C}}_{Pvar} : \wp(Sub) \to Sub^{\mathcal{C}}_{Pvar}$ *and* $\gamma^{\mathcal{C}}_{Pvar} : Sub^{\mathcal{C}}_{Pvar} \to \wp(Sub)$ *are defined by:*

$$\alpha^{\mathcal{C}}_{Pvar}(\Phi) = \sqcup^{\mathcal{C}}\{\phi^{\mathcal{C}} \in Sub^{\mathcal{C}}_{Pvar} \mid \phi^{\mathcal{C}} \sqsubseteq_{Pvar} \phi \wedge \phi \in \Phi\}$$

$$\gamma^{\mathcal{C}}_{Pvar}(\phi^{\mathcal{C}}) = \{\phi \in Sub \mid \phi^{\mathcal{C}} \sqsubseteq_{Pvar} \phi\}$$

Notice that $\alpha^{\mathcal{C}}_{Pvar}(\{\epsilon\}) = \epsilon^{\mathcal{C}}$ where $\epsilon^{\mathcal{C}} = \{u \mapsto u^{\lambda} \mid u \in Pvar\}$ and more generally $u \mapsto t_u \in \alpha^{\mathcal{C}}_{Pvar}(\Phi)$ for all $u \in Pvar$.

Example 3.2 *If* $\mu = \{x \mapsto f(a, y, b),\ z \mapsto g(y)\}$ *and* $\nu = \{x \mapsto f(y, z, c)\}$ *with* $Pvar = \{x, y, z\}$ *then*

$$\alpha^{\mathcal{C}}_{Pvar}(\{\mu\}) = \{x \mapsto f(a, x^2, b), y \mapsto y^{\lambda}, z \mapsto g(z^1)\}$$

$$\alpha^{\mathcal{C}}_{Pvar}(\{\nu\}) = \{x \mapsto f(x^1, x^2, c), y \mapsto y^{\lambda}, z \mapsto z^{\lambda}\}$$

$$\alpha^{\mathcal{C}}_{Pvar}(\{\mu, \nu\}) = \{x \mapsto f(x^1, x^2, x^3), y \mapsto y^{\lambda}, z \mapsto z^{\lambda}\}$$

3.1.3 On finiteness

For a given program $Func$ is finite and therefore termination can be enforced by representing compoundness information to a predetermined depth bound k. The notion of depth is formalised by a mapping $depth : Term \to \{1, 2, \ldots\}$ defined by:

$$depth(t) = \begin{cases} 1 & \text{if } t \in Uvar \vee t \in Const \\ 1 + \max(\{depth(t_i)\}_{i=1}^{n}) & \text{if } t \in Func \wedge t = f(t_1, \ldots, t_n) \end{cases}$$

A natural depth-k widening can be defined by extending the $depth$ mapping to substitutions:

$$depth(t) = \max(\{depth(t_u) \mid u \mapsto t_u \in \phi\})$$

Definition 3.4 ($\bigtriangledown^{\mathcal{C}}_{k}$) *The widening* $\bigtriangledown^{\mathcal{C}}_{k} : Sub^{\mathcal{C}}_{Pvar} \to Sub^{\mathcal{C}}_{Pvar}$ *is defined by:*

$$\bigtriangledown^{\mathcal{C}}_{k}(\phi^{\mathcal{C}}) = \sqcup^{\mathcal{C}}\{\varphi^{\mathcal{C}} \in Sub^{\mathcal{C}}_{Pvar} \mid \varphi^{\mathcal{C}} \sqsubseteq_{Pvar} \phi^{\mathcal{C}} \wedge depth(\varphi^{\mathcal{C}}) \leq k\}$$

Observe that $\gamma^{\mathcal{C}}_{Pvar}(\phi^{\mathcal{C}}) \subseteq \gamma^{\mathcal{C}}_{Pvar}(\bigtriangledown^{\mathcal{C}}_{k}(\phi^{\mathcal{C}}))$ as required for correctness.

Example 3.3 *Adopting the* μ *of example 3.2*

$$\bigtriangledown^{\mathcal{C}}_{1}(\alpha^{\mathcal{C}}_{Pvar}(\{\mu\})) = \{x \mapsto x^{\lambda}, y \mapsto y^{\lambda}, z \mapsto z^{\lambda}\}$$

$$\bigtriangledown^{\mathcal{C}}_{2}(\alpha^{\mathcal{C}}_{Pvar}(\{\mu\})) = \{x \mapsto f(a, x^2, b), y \mapsto y^{\lambda}, z \mapsto g(z^1)\}$$

3.2 Abstracting substitutions with $Share_{Pvar} \times Free_{Pvar}$

To keep the paper self-contained the $Share_{Pvar} \times Free_{Pvar}$ domain and its mappings will be briefly reviewed.

3.2.1 On the domain $Share_{Pvar} \times Free_{Pvar}$

$Share_{Pvar}$ is formulated in terms of sharing groups [14, 24] which record which program variables potentially share variables. A sharing group is a (possibly empty) set of program variables. $Free_{Pvar}$, on the other hand, represents the free program variables as a set.

Definition 3.5 **($Share_{Pvar}$ and $Free_{Pvar}$)** *The domains $Share_{Pvar}$ and $Free_{Pvar}$ are defined by:*

$$Share_{Pvar} = \wp(\wp(Pvar)), \quad Free_{Pvar} = \wp(Pvar)$$

The intuition is that a sharing group records which program variables are bound to terms that share a variable. $Share_{Pvar} \times Free_{Pvar}$ is finite since $Pvar$ is finite.

3.2.2 On the mappings $\alpha^{\mathcal{SF}}_{Pvar}$ and $\gamma^{\mathcal{SF}}_{Pvar}$

In the spirit of [24], the abstraction and concretisation mappings are constructed by lifting two mappings, sh_{Pvar} and fr_{Pvar}, to sets of substitutions. The mappings sh_{Pvar} and fr_{Pvar} detail how a single substitution is abstracted.

Definition 3.6 **(sh_{Pvar} and fr_{Pvar})** *The abstraction mappings $sh_{Pvar} : Sub \rightarrow Share_{Pvar}$ and $fr_{Pvar} : Sub \rightarrow Free_{Pvar}$ are defined by:*

$$sh_{Pvar}(\phi) = \{occ_{Pvar}(u, \phi) \mid u \in Uvar\}, \quad occ_{Pvar}(u, \phi) = \{v \in Pvar \mid u \in var(\phi(v))\}$$

$$fr_{Pvar}(\phi) = \{v \in Pvar \mid var(\phi(v)) \in Uvar\}$$

Example 3.4 *Using the μ, ν and $Pvar = \{x, y, z\}$ of example 3.2*

$$sh_{Pvar}(\mu) = \{occ_{Pvar}(y, \mu), \emptyset\} = \{\{x, y, z\}, \emptyset\}$$

$$sh_{Pvar}(\nu) = \{occ_{Pvar}(y, \nu), occ_{Pvar}(z, \nu), \emptyset\} = \{\{x, y\}, \{x, z\}, \emptyset\}$$

$$fr_{Pvar}(\mu) = \{y\}, \quad fr_{Pvar}(\nu) = \{y, z\}$$

The abstraction sh_{Pvar} is analogous to the abstraction $\mathcal{A}$ used in [24]. Observe that for $\phi \approx \varphi$, $sh_{Pvar}(\phi) = sh_{Pvar}(\varphi)$ and $fr_{Pvar}(\phi) = fr_{Pvar}(\varphi)$. The mapping $\alpha^{\mathcal{SF}}_{Pvar}$ and $\gamma^{\mathcal{SF}}_{Pvar}$ follow directly from sh_{Pvar} and fr_{Pvar}.

Definition 3.7 **($\alpha^{\mathcal{SF}}_{Pvar}$ and $\gamma^{\mathcal{SF}}_{Pvar}$)** *The abstraction and concretisation mappings $\alpha^{\mathcal{SF}}_{Pvar} : \wp(Sub) \rightarrow Share_{Pvar} \times Free_{Pvar}$ and $\gamma^{\mathcal{SF}}_{Pvar} : Share_{Pvar} \times Free_{Pvar} \rightarrow \wp(Sub)$ are defined by:*

$$\alpha^{\mathcal{SF}}_{Pvar}(\Phi) = \langle \alpha^{\mathcal{S}}_{Pvar}(\Phi), \alpha^{\mathcal{F}}_{Pvar}(\Phi) \rangle, \quad \gamma^{\mathcal{SF}}_{Pvar}(\phi^{SF}) = \gamma^{\mathcal{S}}_{Pvar}(\phi^{\mathcal{S}}) \cap \gamma^{\mathcal{F}}_{Pvar}(\phi^{\mathcal{F}})$$

where

$$\alpha^{\mathcal{S}}_{Pvar}(\Phi) = \cup_{\phi \in \Phi} sh_{Pvar}(\phi), \quad \gamma^{\mathcal{S}}_{Pvar}(\phi^{\mathcal{S}}) = \{\phi \mid sh_{Pvar}(\phi) \subseteq \phi^{\mathcal{S}}\}$$
$$\alpha^{\mathcal{F}}_{Pvar}(\Phi) = \cap_{\phi \in \Phi} fr_{Pvar}(\phi), \quad \gamma^{\mathcal{F}}_{Pvar}(\phi^{\mathcal{F}}) = \{\phi \mid \phi^{\mathcal{F}} \subseteq fr_{Pvar}(\phi)\}$$

Note that $\alpha^{\mathcal{S}}(\emptyset) = \emptyset$ whereas $\alpha^{\mathcal{S}}(\Phi) = \{\emptyset\}$ if Φ is a set of substitutions which all ground $Pvar$. This distinction is preserved in both $Share_{Pvar} \times Free_{Pvar}$ and $Type_{Pvar}$.

Example 3.5 *Continuing with example 3.4*

$$\alpha^{\mathcal{SF}}_{Pvar}(\{\mu, \nu\}) = \langle \{\{x, y, z\}, \{x, y\}, \{x, z\}, \emptyset\}, \{y\} \rangle$$

3.3 Abstracting substitutions with $Type_{Pvar}$

The domain $Type_{Pvar}$ is a strict subset of $Sub^{\mathcal{C}}_{Pvar} \times Share_{Pvar} \times Free_{Pvar}$, crafted so that $Type_{Pvar}$ is a complete lattice. The construction proceeds by first specifying the structure of $Type_{Pvar}$ in section 3.3.1. Second, in section 3.3.2, the concretisation mapping $\gamma^{\mathcal{CSF}}_{Pvar}$ is introduced to formalise the relationship between $Type_{Pvar}$ and $\wp(Sub)$. Third, in section 3.3.3, the lattice $\wp(Sub)$ $(\subseteq)$ is used to induce a poset $Type_{Pvar}$ $(\sqsubseteq^{\mathcal{CSF}})$ which, when equipped with a lub $\sqcup^{\mathcal{CSF}}$ and a glb $\sqcap^{\mathcal{CSF}}$, is elevated to a complete lattice. Fourth, in section 3.3.4, the glb $\sqcup^{\mathcal{CSF}}$ and the $\gamma^{\mathcal{CSF}}_{Pvar}$ are used to finally formulate the abstraction mapping $\alpha^{\mathcal{CSF}}_{Pvar}$, and thus complete the construction of a Galois connection [10].

3.3.1 On the domain $Type_{Pvar}$

The domain $Type_{Pvar}$ is formulated so that $\gamma^{\mathcal{CSF}}_{Pvar}$ is injective. An injective concretisation mapping is useful since it can be used to straightforwardly induce a poset (rather an a pre-order) on $Type_{Pvar}$ from Sub $(\subseteq)$. In short, the domain is constructed so that different domain elements (in the abstract) represent different sets of substitutions (in the concrete).

Definition 3.8 ($Type_{Pvar}$) *The abstract domain, $Type_{Pvar}$, is defined by:*

$$Type_{Pvar} = \{\langle fail, \emptyset, \emptyset\rangle\} \cup \left\{ \langle \phi^{\mathcal{C}}, \mathcal{U}, U\rangle \;\middle|\; \begin{array}{c} \phi^{\mathcal{C}} \in Sub^{\mathcal{C}}_{Pvar} \wedge \\ \mathcal{U} \in Share_{cod(\phi^{\mathcal{C}})} \wedge \mathcal{U} \neq \emptyset \wedge \\ U \in Free_{cod(\phi^{\mathcal{C}})} \end{array} \right\}$$

3.3.2 On the mapping $\gamma^{\mathcal{CSF}}_{Pvar}$

The relationship between $Type_{Pvar}$ and $\wp(Sub)$ is made explicit by the mapping $\gamma^{\mathcal{CSF}}_{Pvar}$.

Definition 3.9 ($\gamma^{\mathcal{CSF}}_{Pvar}$) *The concretisation mapping $\gamma^{\mathcal{CSF}}_{Pvar} : Type_{Pvar} \rightarrow \wp(Sub)$ is defined by:*

$$\gamma^{\mathcal{CSF}}_{Pvar}(\langle \phi^{\mathcal{C}}, \phi^{\mathcal{S}}, \phi^{\mathcal{F}}\rangle) = \left\{ \phi \;\middle|\; \begin{array}{c} \phi \restriction Pvar = \phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar \wedge \\ sh_{cod(\phi^{\mathcal{C}})}(\phi^{\mathcal{D}}) \subseteq \phi^{\mathcal{S}} \wedge \phi^{\mathcal{F}} \subseteq fr_{cod(\phi^{\mathcal{C}})}(\phi^{\mathcal{D}}) \end{array} \right\}$$

It is assumed that $\gamma^{\mathcal{CSF}}_{Pvar}(\langle fail, \emptyset, \emptyset\rangle) = \emptyset$. Notice how the definitions of $\alpha^{\mathcal{CSF}}_{Pvar}$ requires the sh_{Pvar} and fr_{Pvar} mappings to be parametrised by $Pvar$. The parameter is required because the $Share$ and $Free$ components of $Type_{Pvar}$ record the sharing and freeness of the variables of $cod(\phi^{\mathcal{C}})$ which is not fixed. Note also that for $\phi \approx \varphi$, $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ if and only if $\varphi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$. This follows because $sh_{Pvar}(\theta) = sh_{Pvar}(\vartheta)$ and $fr_{Pvar}(\theta) = fr_{Pvar}(\vartheta)$ if $\theta \approx \vartheta$.

Example 3.6 *To illustrate the expressiveness of $Type_{Pvar}$, consider the properties of an arbitrary substitution ϕ taken from $\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ where*

$$\begin{aligned} \phi^{\mathcal{CSF}} = \langle & \{x \mapsto f(x^1, x^2, x^3), y \mapsto y^\lambda, z \mapsto z^\lambda\}, \\ & \{\{x^1, y^\lambda\}, \{x^2, y^\lambda, z^\lambda\}, \{x^2, z^\lambda\}, \emptyset\}, \{x^2, y^\lambda\}\rangle \end{aligned}$$

sharing: *The first argument of* $\phi(x)$ *and* $\phi(z)$ *do not share. This follows because* x^1 *and* z^λ *do not occur in a sharing group of* $\phi^{\mathcal{S}}$. *Likewise,* x *never has any internal aliasing under* ϕ. *Put another way,* x *is linear [25, 7, 15], which in this case means that the first, second and third argument of* x *are independent.*

groundness: *The third argument of* $\phi(x)$ *is ground. This follows since the variables of* $cod(\phi^{\mathcal{C}})$ *which* ϕ *ground, do not appear in* $\phi^{\mathcal{S}}$.

freeness: *The variable* y *is free under* ϕ *and the second argument of* $\phi(x)$ *is also free. This follows immediately from* $\phi^{\mathcal{F}}$.

compoundness: *Compoundness is captured in that* $\phi^{\mathcal{C}}$ *shows that* $\phi(x)$ *is compound with a principal functor of* f *and an arity of 3.*

covering: *Covering is also implicitly captured by virtue of* $\phi^{\mathcal{S}}$. *For instance,* x *covers both* y, *or more exactly, the first and second arguments of* x *cover* y. *Thus grounding* x *grounds* y*; or more precisely, grounding the first and second arguments of* x *grounds* y.

3.3.3 On the domain $Type_{Pvar}$ (continued)

A natural ordering $Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) can be induced from Sub ($\subseteq$) by $\gamma^{\mathcal{CSF}}_{Pvar}$.

Definition 3.10 *The preorder* $Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) *is defined by:* $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \varphi^{\mathcal{CSF}}$ *if and only if* $\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}}) \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\varphi^{\mathcal{CSF}})$.

Note that $\gamma^{\mathcal{CSF}}_{Pvar}$ is monotonically increasing by definition. $Type_{Pvar}$ is carefully engineered so that the preorder $Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) is also a poset. This is formally stated as lemma 3.1 and established by proof 3.1.

Lemma 3.1 $Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) *is a poset.*

The poset has a top element $\top^{\mathcal{CSF}} = \langle \epsilon^{\mathcal{C}}, \wp(\wp(cod(\epsilon^{\mathcal{C}}))), \emptyset\rangle$ and a bottom element $\bot^{\mathcal{CSF}} = \langle fail, \emptyset, \emptyset\rangle$ since $\gamma^{\mathcal{CSF}}_{Pvar}(\top^{\mathcal{CSF}}) = Sub$ and $\gamma^{\mathcal{CSF}}_{Pvar}(\bot^{\mathcal{CSF}}) = \emptyset$. The bottom element is meaningful and, in fact, represents goal failure. Specifically, in a top-down abstract interpretation framework [2, 18, 24], it is possible for a goal to fail if it is called with some abstract substitutions. Returning $\bot^{\mathcal{CSF}}$ for the success abstract substitution indicates that the goal can never be satisfied under any calling substitution which the calling abstract substitution abstracts.

For the purposes of abstract interpretation [10], $Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) is required to be a complete lattice, preferably equipped with a lub $\sqcup^{\mathcal{CSF}}$ which is straightforward to compute. To succinctly define $\sqcup^{\mathcal{CSF}}$ (and later a widening $\bigtriangledown^{\mathcal{CSF}}_k$) it is convenient to introduce an auxiliary mapping. The mapping is formulated as a substitution. More exactly, if $\phi^{\mathcal{B}} = \{x^1 \mapsto a, x^3 \mapsto b, z^\lambda \mapsto g(z^1)\}$, it is helpful to define $\widehat{\phi^{\mathcal{B}}} = \{z^1 \mapsto z^\lambda\}$. Informally, $\widehat{\phi^{\mathcal{B}}}$ is constructed to map the codomain variables of $\phi^{\mathcal{B}}$ to the domain variables of $\phi^{\mathcal{B}}$. More formally, $\widehat{\psi^{\mathcal{B}}} = \{u_t \mapsto u^s \mid u^s \mapsto t_{u_s} \in \psi^{\mathcal{B}} \wedge u_t \in var(t_{u^s})\}$. A sufficient condition for $\widehat{\psi^{\mathcal{B}}}$ to be well-defined is that $t_{u_s} \in Term^s_u$ for all $u_t \mapsto t_{u^s} \in \psi^{\mathcal{B}}$. This, in fact, will always be guaranteed by the context.

Definition 3.11 ($\sqcup^{\mathcal{CSF}}$) *The mapping* $\sqcup^{\mathcal{CSF}} : \wp(Type_{Pvar}) \to Type_{Pvar}$ *is defined by:*

$$
\begin{array}{l}
\sqcup^{\mathcal{CSF}}(\Phi^{\mathcal{CSF}}) = \langle \varphi^{\mathcal{C}}, \varphi^{\mathcal{S}}, \varphi^{\mathcal{F}} \rangle \textit{ where} \\
\quad \varphi^{\mathcal{C}} = \sqcap^{\mathcal{C}}(\{ \quad \phi^{\mathcal{C}} \quad \mid \phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}\}) \\
\quad \varphi^{\mathcal{S}} = \cup \;(\{ \quad \widehat{\phi^{\mathcal{B}}}(\phi^{\mathcal{S}}) \quad \mid \phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}} \wedge \phi^{\mathcal{B}} \circ \varphi^{\mathcal{C}} \restriction Pvar = \phi^{\mathcal{C}}\}) \\
\quad \varphi^{\mathcal{F}} = \cap \;(\{\phi^{\mathcal{F}} \setminus cod(\phi^{\mathcal{B}}) \mid \phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}} \wedge \phi^{\mathcal{B}} \circ \varphi^{\mathcal{C}} \restriction Pvar = \phi^{\mathcal{C}}\})
\end{array}
$$

It is assumed that $\sqcap^{\mathcal{C}}(\emptyset) = fail$ so that $\sqcup^{\mathcal{CSF}}(\emptyset) = \perp^{\mathcal{CSF}}$. Computationally, the lub $\sqcup^{\mathcal{CSF}}$ inherits much of its simplicity from the pair-wise union and intersection lub of $Share_{Pvar} \times Free_{Pvar}$. The main novelty is that anti-unification [17] is required to compute the glb for the $Sub^{\mathcal{C}}_{Pvar}$ component.

Lemma 3.2 *$Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) has a lub $\sqcup^{\mathcal{CSF}}$.*

Example 3.7 *To illustrate the calculation of* $\sqcup^{\mathcal{CSF}}$ *consider* $\sqcup^{\mathcal{CSF}}(\{\mu^{\mathcal{CSF}}, \nu^{\mathcal{CSF}}\})$ *where* $Pvar = \{x, y, z\}$ *and*

$$\mu^{\mathcal{CSF}} = \langle \{x \mapsto f(a, x^2, b),\ y \mapsto y^{\lambda},\ z \mapsto g(z^1)\}, \{\{x^2, y^{\lambda}, z^1\}, \emptyset\}, \{x^2, y^{\lambda}, z^1\}\rangle$$

$$
\begin{array}{rl}
\nu^{\mathcal{CSF}} = & \langle \{x \mapsto f(x^1, x^2, c),\ y \mapsto y^{\lambda},\ z \mapsto z^{\lambda}\}, \\
& \{\{x^1, y^{\lambda}\}, \{x^2, z^{\lambda}\}, \emptyset\}, \{x^1, x^2, y^{\lambda}, z^{\lambda}\}\rangle
\end{array}
$$

For conciseness put $\varphi^{\mathcal{CSF}} = \sqcup^{\mathcal{CSF}}(\{\mu^{\mathcal{CSF}}, \nu^{\mathcal{CSF}}\})$ *and thus*

$$\varphi^{\mathcal{C}} = \mu^{\mathcal{C}} \sqcap^{\mathcal{C}} \nu^{\mathcal{C}} = \{x \mapsto f(x^1, x^2, x^3), y \mapsto y^{\lambda}, z \mapsto z^{\lambda}\}$$

$$\varphi^{\mathcal{S}} = \widehat{\mu^{\mathcal{B}}}(\mu^{\mathcal{S}}) \cup \widehat{\nu^{\mathcal{B}}}(\nu^{\mathcal{S}}) = \{\{x^1, y^{\lambda}\}, \{x^2, y^{\lambda}, z^{\lambda}\}, \{x^2, z^{\lambda}\}, \emptyset\}$$

$$\varphi^{\mathcal{F}} = (\mu^{\mathcal{F}} \setminus cod(\mu^{\mathcal{B}})) \cap (\nu^{\mathcal{F}} \setminus cod(\nu^{\mathcal{B}})) = \{x^2, y^{\lambda}\}$$

since

$$\mu^{\mathcal{B}} = \{x^1 \mapsto a, x^3 \mapsto b, z^{\lambda} \mapsto g(z^1)\}, \quad \nu^{\mathcal{B}} = \{x^3 \mapsto c\}$$

$$\widehat{\mu^{\mathcal{B}}} = \{z^1 \mapsto z^{\lambda}\}, \quad \widehat{\nu^{\mathcal{B}}} = \epsilon$$

with $\mu^{\mathcal{B}} \circ \varphi^{\mathcal{C}} \restriction Pvar = \mu^{\mathcal{C}}$ *and* $\nu^{\mathcal{B}} \circ \varphi^{\mathcal{C}} \restriction Pvar = \nu^{\mathcal{C}}$.

The glb $\sqcap^{\mathcal{CSF}}$ can be defined (in the standard way [1]) in terms of the lub $\sqcup^{\mathcal{CSF}}$. To be more precise, $\sqcap^{\mathcal{CSF}}(\Phi^{\mathcal{CSF}}) = \sqcup^{\mathcal{CSF}}(\{\varphi^{\mathcal{CSF}} \mid \forall \phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}. \varphi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \phi^{\mathcal{CSF}}\})$. Corollary 3.1 and proof 7.3 document that $Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) is a complete lattice as a consequence.

Corollary 3.1 *$Type_{Pvar}$ ($\sqsubseteq^{\mathcal{CSF}}$) is a complete lattice.*

3.3.4 On the mapping $\alpha^{\mathcal{CSF}}_{Pvar}$

Once the glb $\sqcap^{\mathcal{CSF}}$ is defined, the concretisation mapping $\gamma^{\mathcal{CSF}}_{Pvar}$ determines the abstraction mapping $\alpha^{\mathcal{CSF}}_{Pvar}$. This is spelt out in definition 3.12. The reason for explicitly introducing $\alpha^{\mathcal{CSF}}_{Pvar}$ (rather than leaving its definition implicit) is to aid the synthesis of a constructive version of $\alpha^{\mathcal{CSF}}_{Pvar}$.

Definition 3.12 ($\alpha^{\mathcal{CSF}}_{Pvar}$) *The abstraction mapping* $\alpha^{\mathcal{CSF}}_{Pvar} : \wp(Sub) \to Type_{Pvar}$ *is defined by:*

$$\alpha^{\mathcal{CSF}}_{Pvar}(\Phi) = \sqcap^{\mathcal{CSF}}(\{\phi^{\mathcal{CSF}} \mid \Phi \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})\})$$

Note that $\alpha^{\mathcal{CSF}}_{Pvar}$ is monotonically increasing since $\sqcap^{\mathcal{CSF}}$ is monotonically decreasing. Furthermore $\Phi \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\alpha^{\mathcal{CSF}}_{Pvar}(\Phi))$ for all $\Phi \in \wp(Sub)$ and $\alpha^{\mathcal{CSF}}_{Pvar}(\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})) \sqsubseteq^{\mathcal{CSF}} \phi^{\mathcal{CSF}}$ for all $\phi^{\mathcal{CSF}} \in Type_{Pvar}$.

Following [10], the relationship between $Type_{Pvar}$ and $\wp(Sub)$ can thus be stated as a Galois connection. This gives lemma 3.3. The corresponding proof, numbered 7.5, is almost immediate.

Lemma 3.3 (Galois connection) $\alpha^{\mathcal{CSF}}_{Pvar}$ *and* $\gamma^{\mathcal{CSF}}_{Pvar}$ *form a Galois connection between* $Type_{Pvar}$ $(\sqsubseteq^{\mathcal{CSF}})$ *and* $\wp(Sub)$ $(\subseteq)$.

In fact, to be precise, $\alpha^{\mathcal{CSF}}_{Pvar}$ and $\gamma^{\mathcal{CSF}}_{Pvar}$ form a Galois insertion [22] since $\gamma^{\mathcal{CSF}}_{Pvar}$ is injective. In more pragmatic terms, an insertion means that the abstract domain does not contain any redundant elements.

Returning to the abstraction mapping, lemma 3.4 presents one constructive version of $\alpha^{\mathcal{CSF}}_{Pvar}$. Interestingly, the reformulation not only details how $\alpha^{\mathcal{CSF}}_{Pvar}$ can be computed, but it also emphases the connection between $\alpha^{\mathcal{CSF}}_{Pvar}$ and $\alpha^{\mathcal{SF}}_{Pvar}$. Correctness is established in proof 7.4 and example 3.8 demonstrates one application of the new, constructive abstraction mapping.

Lemma 3.4

$$\alpha^{\mathcal{CSF}}_{Pvar}(\Phi) = \left\langle \phi^{\mathcal{C}}, \bigcup_{\phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar \in \Phi \restriction Pvar} sh_{cod(\phi^{\mathcal{C}})}(\phi^{\mathcal{D}}), \bigcap_{\phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar \in \Phi \restriction Pvar} fr_{cod(\phi^{\mathcal{C}})}(\phi^{\mathcal{D}}) \right\rangle$$

where $\phi^{\mathcal{C}} = \alpha^{\mathcal{C}}_{Pvar}(\Phi)$.

Example 3.8 *Building on example 3.2, let* $\phi^{\mathcal{C}} = \alpha^{\mathcal{C}}_{Pvar}(\{\mu, \nu\})$ *and consider the calculation of* $\alpha^{\mathcal{CSF}}_{Pvar}(\Phi)$ *where* $\Phi = \{\mu, \nu\}$ *and* $Pvar = \{x, y, z\}$. *Thus*

$$\alpha^{\mathcal{CSF}}_{Pvar}(\Phi) = \langle \phi^{\mathcal{C}}, sh_{cod(\phi^{\mathcal{C}})}(\mu^{\mathcal{D}}) \cup sh_{cod(\phi^{\mathcal{C}})}(\nu^{\mathcal{D}}), fr_{cod(\phi^{\mathcal{C}})}(\mu^{\mathcal{D}}) \cap fr_{cod(\phi^{\mathcal{C}})}(\nu^{\mathcal{D}}) \rangle$$

where

$$\mu^{\mathcal{D}} = \left\{ \begin{array}{l} x^1 \mapsto a, \\ x^2 \mapsto y, \\ x^3 \mapsto b, \\ y^\lambda \mapsto y, \\ z^\lambda \mapsto g(y) \end{array} \right\}, \quad \nu^{\mathcal{D}} = \left\{ \begin{array}{l} x^1 \mapsto y, \\ x^2 \mapsto z, \\ x^3 \mapsto c, \\ y^\lambda \mapsto y, \\ z^\lambda \mapsto z \end{array} \right\}$$

Hence

$$\begin{array}{rl} \alpha^{\mathcal{CSF}}_{Pvar}(\Phi) = \langle & \{x \mapsto f(x^1, x^2, x^3), y \mapsto y^\lambda, z \mapsto z^\lambda\}, \\ & \{\{x^1, y^\lambda\}, \{x^2, y^\lambda, z^\lambda\}, \{x^2, z^\lambda\}, \emptyset\}, \{x^2, y^\lambda\}\rangle \end{array}$$

Notice that $\alpha^{\mathcal{CSF}}_{Pvar}(\Phi)$ *tallies with the* $\phi^{\mathcal{CSF}}$ *of example 3.6. A comparison of the* Φ *(of example 3.8) to the* $\gamma^{\mathcal{CSF}}_{Pvar}(\alpha^{\mathcal{CSF}}_{Pvar}(\Phi))$ *(of example 3.6) demonstrates the precision with which* $Type_{Pvar}$ *captures sharing, groundness, freeness, compoundness and covering information.*

3.3.5 On finiteness

Since widening is required to induce termination on $Sub^{\mathcal{C}}_{Pvar}$, widening is also necessary to obtain finiteness on $Type_{Pvar}$. A suitable widening can be constructed from $\bigtriangledown^{\mathcal{C}}_{k}$, however, by borrowing some machinery from $\sqcup^{\mathcal{CSF}}$. Specifically, if $\bigtriangledown^{\mathcal{C}}_{k}$ details how the $Sub^{\mathcal{C}}_{Pvar}$ domain can be widened, then $\sqcup^{\mathcal{CSF}}$ explains how the $Share$ and $Free$ components can be amended to fit with the widened $Sub^{\mathcal{C}}_{Pvar}$ component.

Definition 3.13 ($\bigtriangledown^{\mathcal{CSF}}_{k}$) *The widening* $\bigtriangledown^{\mathcal{CSF}}_{k} : Type_{Pvar} \rightarrow Type_{Pvar}$ *is defined by:*

$$\bigtriangledown^{\mathcal{CSF}}_{k}(\phi^{\mathcal{CSF}}) = \langle \bigtriangledown^{\mathcal{CSF}}_{k}(\phi^{\mathcal{C}}), \widehat{\phi^{\mathcal{B}}}(\phi^{\mathcal{S}}), \phi^{\mathcal{F}} \setminus cod(\phi^{\mathcal{B}}) \rangle$$

where $\phi^{\mathcal{B}} \circ \bigtriangledown^{\mathcal{CSF}}_{k}(\phi^{\mathcal{C}}) \restriction Pvar = \phi^{\mathcal{C}}$.

Note that the k does not necessarily need to be fixed but can be varied during analysis – just so long as termination is achieved. The correctness of the widening is stated as lemma 3.5 and proven in proof 7.6.

Lemma 3.5 $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \bigtriangledown^{\mathcal{CSF}}_{k}(\phi^{\mathcal{CSF}})$

4 Abstracting unification

Abstract interpretation can help to focus the development of an analysis by illuminating the connection between an operation (like unification) and its abstract counterpart. In this case, the abstract counterpart for unification is divided into two distinct algorithmic phases. This is a consequence of exploiting confluence. The first phase, detailed in section 4.2, traces compoundness information; whereas the second phase, documented in section 4.1, infers sharing and freeness information. The order of presentation reflects the construction of the analysis: first, an efficient sharing and freeness analysis is synthesised in section 4.1; second, the analysis is extended to trace compoundness to bounded depth in section 4.2.

4.1 Abstracting unification with $Share_{Pvar} \times Free_{Pvar}$

The sharing and freeness component of the analysis is, in fact, interesting within its own right. It is interesting for a number of reasons. First, the sharing and freeness analysis can used by itself, stand-alone. Second, although the analysis applies many of the ideas of [23], it is both simple and efficient. Third, the analysis has been proved correct. In short, the analysis can be regarded as a systematic and efficient reformulation of the abstract unification algorithm of [23].

Unification is abstracted by tracing the steps of a standard unification algorithm [17]. To trace unification, the abstract algorithm mimicks the recursive simplification steps of mgu, relegating the solution of simplified equations of the form $v = t$ to a mapping $mgu^{\mathcal{SF}}$. Similar simplification steps, dubbed pre-unification in [7], are applied in other abstract unification algorithms [23, 7]. The mapping $mgu^{\mathcal{SF}}$ is defined to abstract a slight variant of mgu. Specifically, if $\varphi \in mgu(\{\phi(v) = \phi(t)\})$ and $\phi \in \gamma^{\mathcal{SF}}_{Pvar}(\phi^{\mathcal{SF}})$ then $mgu^{\mathcal{SF}}(u, t, \phi^{\mathcal{SF}})$ abstracts the composition $\varphi \circ \phi$ (rather than φ), that is, $\varphi \circ \phi \in \gamma^{\mathcal{SF}}_{Pvar}(mgu^{\mathcal{SF}}(u, t, \phi^{\mathcal{SF}}))$. This spares the need to define an extra (composition) operator.

Definition 4.1 ($mgu^{\mathcal{SF}}$) *The relation* $mgu^{\mathcal{SF}} : Eqn \times (Share_{Pvar} \times Free_{Pvar}) \times (Share_{Pvar} \times Free_{Pvar})$ *is defined by:*

$$\begin{array}{r}
mgu^{\mathcal{SF}}(\emptyset, \phi^{\mathcal{SF}}, \phi^{\mathcal{SF}}) \\
mgu^{\mathcal{SF}}(u = u : E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \text{ if } mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \\
mgu^{\mathcal{SF}}(t = u : E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \text{ if } mgu^{\mathcal{SF}}(u = t : E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \\
mgu^{\mathcal{SF}}(u = t : E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \text{ if } mgu^{\mathcal{SF}}(E, \tau^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \wedge u \notin var(t) \\
mgu^{\mathcal{SF}}(f(\underline{t_i}) = f(\underline{t'_i}) : E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \text{ if } mgu^{\mathcal{SF}}(\{t_i = t'_i\}_{i=1}^{n} \cup E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}})
\end{array}$$

where $\tau^{\mathcal{S}} = mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$ *and* $\tau^{\mathcal{F}} = mgu^{\mathcal{F}}(u, t, \phi^{\mathcal{SF}})$.

To define the mappings $mgu^{\mathcal{S}}$ and $mgu^{\mathcal{F}}$ (and thus the relation $mgu^{\mathcal{SF}}$) a number of standard auxiliary operators are required [14, 24]. First, $rel(t, \phi^{\mathcal{S}})$ represents the sharing groups of $\phi^{\mathcal{S}}$ which are relevant to the term t, that is, those sharing groups of $\phi^{\mathcal{SF}}$ which share variables with t. Second, in the absence of useful freeness information worst-case aliasing is assumed. Thus, as in [14, 24], a closure under union operator, *, is employed to enumerate all the possible sharing groups that can possibly arise in unification. Third, to succinctly define $mgu^{\mathcal{SF}}$, it is convenient to lift $\cup$ to sets of sharing groups with a pair-wise union operator, denoted $\Box$.

Definition 4.2 (rel, * **[14, 24] and** $\Box$)

$$rel(t, \phi^{\mathcal{S}}) = \{U \in \phi^{\mathcal{S}} \mid U \cap var(t) \neq \emptyset\}$$

$$\phi^{\mathcal{S}\star} = \phi^{\mathcal{S}} \cup \{U \cup U' \mid U, U' \in \phi^{\mathcal{S}\star}\}, \quad \phi^{\mathcal{S}} \Box \phi'^{\mathcal{S}} = \{U \cup U' \mid U \in \phi^{\mathcal{S}} \wedge U' \in \phi'^{\mathcal{S}}\}$$

The mappings $mgu^{\mathcal{S}}$ and $mgu^{\mathcal{F}}$ apply different analysis strategies according to the freeness of $\phi(v)$ and $\phi(t)$ for $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{SF}})$. The default strategy of $mgu^{\mathcal{S}}$ corresponds to the standard treatment in the abstract solver *amgu* of [14].

Definition 4.3 ($mgu^{\mathcal{S}}$ **and** $mgu^{\mathcal{F}}$)

$$mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}}) = \begin{cases} \psi^{\mathcal{S}} \cup (rel(u, \phi^{\mathcal{S}}) \Box rel(t, \phi^{\mathcal{S}})) & \text{if } u \in \phi^{\mathcal{F}} \vee t \in \phi^{\mathcal{F}} \\ \psi^{\mathcal{S}} \cup (rel(u, \phi^{\mathcal{S}})^{\star} \Box rel(t, \phi^{\mathcal{S}})^{\star}) & \text{otherwise} \end{cases}$$

where $\psi^{\mathcal{S}} = \phi^{\mathcal{S}} \setminus (rel(u, \phi^{\mathcal{S}}) \cup rel(t, \phi^{\mathcal{S}}))$.

$$mgu^{\mathcal{F}}(u, t, \phi^{\mathcal{SF}}) = \begin{cases} \phi^{\mathcal{F}} & \text{if } u \in \phi^{\mathcal{F}} \wedge t \in \phi^{\mathcal{F}} \\ \phi^{\mathcal{F}} \setminus var(rel(u, \phi^{\mathcal{S}})) & \text{else if } u \in \phi^{\mathcal{F}} \\ \phi^{\mathcal{F}} \setminus var(rel(t, \phi^{\mathcal{S}})) & \text{else if } t \in \phi^{\mathcal{F}} \\ \phi^{\mathcal{F}} \setminus var(rel(u, \phi^{\mathcal{S}}) \cup rel(t, \phi^{\mathcal{S}})) & \text{otherwise} \end{cases}$$

Note that $rel(u, \phi^{\mathcal{S}}) \Box rel(t, \phi^{\mathcal{S}}) = \emptyset$ and $rel(u, \phi^{\mathcal{S}})^{\star} \Box rel(t, \phi^{\mathcal{S}})^{\star} = \emptyset$ if $rel(u, \phi^{\mathcal{S}}) = \emptyset$. Thus, in case one of $mgu^{\mathcal{S}}$, $rel(t, \phi^{\mathcal{S}})$ need not be calculated if $rel(u, \phi^{\mathcal{S}}) = \emptyset$ and similarly in case two, $rel(t, \phi^{\mathcal{S}})$ need not be computed or closed under union if $rel(u, \phi^{\mathcal{S}}) = \emptyset$. Analogous refinements follow if $rel(t, \phi^{\mathcal{S}}) = \emptyset$. In addition, observe that $mgu^{\mathcal{S}}$ applies the refinement suggested in [23], that is, if either u or t are free, then the calculation of a closure can be avoided. This contrasts with other freeness algorithms, for example [26], which always calculate a closure unless both u and t are free. The correctness of the mappings $mgu^{\mathcal{S}}$ and $mgu^{\mathcal{F}}$ is asserted in lemma 4.1 and 4.2. The corresponding proofs are numbered 7.7 and 7.8.

Lemma 4.1

$$\begin{array}{l} \phi \in \gamma^{\mathcal{S}}_{Pvar}(\phi^{\mathcal{S}}) \wedge \varphi \in mgu(\{\phi(u) = \phi(t)\}) \wedge \\ \quad \{u\} \cup var(t) \subseteq Pvar \wedge u \notin var(t) \Rightarrow \varphi \circ \phi \in \gamma^{\mathcal{S}}_{Pvar}(mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}}) \end{array}$$

Lemma 4.2

$$\begin{array}{l} \phi \in \gamma^{\mathcal{F}}_{Pvar}(\phi^{\mathcal{F}}) \wedge \varphi \in mgu(\{\phi(u) = \phi(t)\}) \wedge \\ \quad \{u\} \cup var(t) \subseteq Pvar \wedge u \notin var(t) \Rightarrow \varphi \circ \phi \in \gamma^{\mathcal{F}}_{Pvar}(mgu^{\mathcal{F}}(u, t, \phi^{\mathcal{SF}}) \end{array}$$

The correctness of the relation $mgu^{\mathcal{SF}}$ follows from lemma 4.1 and 4.2 and is stated as theorem 4.1. The corresponding proof is numbered 7.9.

Theorem 4.1

$$\begin{array}{l} \phi \in \gamma^{\mathcal{SF}}_{Pvar}(\phi^{\mathcal{SF}}) \wedge \varphi \in mgu(\phi(E)) \wedge \\ \quad mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}, \mu^{\mathcal{SF}}) \wedge var(E) \subseteq Pvar \Rightarrow \varphi \circ \phi \in \gamma^{\mathcal{SF}}_{Pvar}(\mu^{\mathcal{SF}}) \end{array}$$

It is convenient shorthand to regard $mgu^{\mathcal{SF}}$ as a mapping, that is, $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}) = \psi^{\mathcal{SF}}$ if $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}, \psi^{\mathcal{SF}})$. Strictly, it is necessary to show that $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}, \psi^{\mathcal{SF}})$ is deterministic for $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}})$ to be well-defined. Like in [7], the conjecture is that $mgu^{\mathcal{SF}}$ yields a unique abstract substitution $\psi^{\mathcal{SF}}$ for $\phi^{\mathcal{SF}}$ regardless of the order in which E is solved (though, in practice, any $\psi^{\mathcal{SF}}$ is safe).

Example 4.1 illustrates that the simplicity of the analysis is not gained at the expense of precision. Indeed the analysis seems to possess much of the power of the original sharing and freeness analysis of [23].

Example 4.1 *Adapting an example from [23], consider the computation of* $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}})$ *where*

$$E = \left\{ \begin{array}{l} x_1 = f(y_1, y_2), \\ x_1 = f(y_3, y_4), \\ y_3 = a, \\ y_5 = x_2, \\ y_6 = x_2, \\ y_6 = f(x_1, x_3) \end{array} \right\}, \qquad \begin{array}{rl} \phi^{\mathcal{SF}} = & \langle \{\emptyset, \{x_1\}, \{x_2\}, \{x_3\}, \\ & \{y_1, y_2\}, \{y_2\}, \{y_3\}, \{y_5\}, \{y_6\}\}, \\ & \{x_1, x_2, x_3, y_1, y_3, y_5, y_6\}\rangle \end{array}$$

Thus, putting $\phi^{\mathcal{SF}}_1 = \phi^{\mathcal{SF}}$, *and considering each equation of* $E = \{u_i = t_i\}_{i=1}^{6}$ *in turn, then* $\phi^{\mathcal{SF}}_{i+1} = \langle mgu^{\mathcal{S}}(u_i, t_i, \phi^{\mathcal{SF}}_i), mgu^{\mathcal{F}}(u_i, t_i, \phi^{\mathcal{SF}}_i)\rangle$ *where*

$$\begin{array}{rl} \phi^{\mathcal{SF}}_1 = & \langle \{\emptyset, \{x_1\}, \{x_2\}, \{x_3\}, \{y_1, y_2\}, \{y_2\}, \{y_3\}, \{y_5\}, \{y_6\}\}, \\ & \{x_1, x_2, x_3, y_1, y_3, y_5, y_6\}\rangle \\ \phi^{\mathcal{SF}}_2 = & \langle mgu^{\mathcal{S}}(x_1, f(y_1, y_2), \phi^{\mathcal{SF}}_1), mgu^{\mathcal{F}}(x_1, f(y_1, y_2), \phi^{\mathcal{SF}}_1)\rangle \\ = & \langle \{\emptyset, \{x_1, y_1, y_2\}, \{x_1, y_2\}, \{x_2\}, \{x_3\}, \{y_3\}, \{y_5\}, \{y_6\}\}, \\ & \{x_2, x_3, y_1, y_3, y_5, y_6\}\rangle \\ \phi^{\mathcal{SF}}_3 = & \langle \{\emptyset, \{x_1, y_1, y_2, y_3\}, \{x_1, y_2, y_3\}, \{x_2\}, \{x_3\}, \{y_5\}, \{y_6\}\}, \\ & \{x_2, x_3, y_5, y_6\}\rangle \\ \phi^{\mathcal{SF}}_4 = & \langle \{\emptyset, \{x_2\}, \{x_3\}, \{y_5\}, \{y_6\}\}, \{x_2, x_3, y_5, y_6\}\rangle \\ \phi^{\mathcal{SF}}_5 = & \langle \{\emptyset, \{x_2, y_5\}, \{x_3\}, \{y_6\}\}, \{x_2, x_3, y_5, y_6\}\rangle \\ \phi^{\mathcal{SF}}_6 = & \langle \{\emptyset, \{x_2, y_5, y_6\}, \{x_3\}\}, \{x_2, x_3, y_5, y_6\}\rangle \\ \phi^{\mathcal{SF}}_7 = & \langle \{\emptyset, \{x_2, x_3, y_5, y_6\}\}, \{x_3\}\rangle \end{array}$$

Therefore $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}) = \langle\{\emptyset, \{x_2, x_3, y_5, y_6\}\}, \{x_3\}\rangle$*. The freeness analysis of [23] similarly infers (modulo a projection operation [23]) that only* x_2*,* x_3*,* y_5 *and* y_6 *are possibly aliased and that* x_3 *is free.*

4.2 Abstracting unification with $Type_{Pvar}$

The task of extending abstract unification from $Share_{Pvar} \times Free_{Pvar}$ to $Type_{Pvar}$ boils down to defining a mapping $mgu^{\mathcal{CSF}}$. The specific requirement is for a mapping $mgu^{\mathcal{CSF}}$ such that if $\varphi \in mgu(\phi(E))$, $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ and $\mu^{\mathcal{CSF}} = mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}})$ then $\varphi \circ \phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\mu^{\mathcal{CSF}})$. As with $mgu^{\mathcal{SF}}$, the problem is cast in a way so as to avoid a composition operation.

The mapping $mgu^{\mathcal{CSF}}$ builds on $mgu^{\mathcal{SF}}$ by detailing how $\mu^{\mathcal{C}}$ can be computed. In addition, $mgu^{\mathcal{CSF}}$ explains how $\mu^{\mathcal{SF}}$ can be calculated with $mgu^{\mathcal{SF}}$ from $\mu^{\mathcal{C}}$, $\phi^{\mathcal{SF}}$ and E. Interestingly, $\mu^{\mathcal{C}}$ is calculated by applying (concrete) unification to solve the equation set $\phi^{\mathcal{C}}(E)$. The intuition is that if $\varphi^{\mathcal{B}}$ is a unifier of $mgu(\phi^{\mathcal{C}}(E))$ then $\varphi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar$ is a likely to be a good candidate for $\mu^{\mathcal{C}}$. One technical point, however, is that the substitution $\varphi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar$ is not always an element of $Sub^{\mathcal{C}}_{Pvar}$.

Example 4.2 *Consider, for example, the abstraction of* $mgu(E, \phi)$ *where* ϕ *is represented by* $\phi^{\mathcal{CSF}}$*,* E *is defined in example 4.1 and* $\phi^{\mathcal{C}} = \{x_1 \mapsto x_1^\lambda, x_2 \mapsto x_2^\lambda, x_3 \mapsto x_3^\lambda, y_1 \mapsto y_1^\lambda, \ldots, y_6 \mapsto y_6^\lambda\}$*. Thus, calculating* $\varphi^{\mathcal{B}} \in mgu(\phi^{\mathcal{C}}(E))$ *and* $\varphi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar$ *yields*

$$\varphi^{\mathcal{B}} = \left\{ \begin{array}{l} x_1^\lambda \mapsto f(a, y_2^\lambda), \\ x_2^\lambda \mapsto f(f(a, y_2^\lambda), x_3^\lambda), \\ y_1^\lambda \mapsto a, \\ y_3^\lambda \mapsto a, \\ y_4^\lambda \mapsto y_2^\lambda, \\ y_5^\lambda \mapsto f(f(a, y_2^\lambda), x_3^\lambda), \\ y_6^\lambda \mapsto f(f(a, y_2^\lambda), x_3^\lambda) \end{array} \right\}, \quad \varphi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar = \left\{ \begin{array}{l} x_1 \mapsto f(a, y_2^\lambda), \\ x_2 \mapsto f(f(a, y_2^\lambda), x_3^\lambda), \\ y_1 \mapsto a, \\ y_3 \mapsto a, \\ y_4 \mapsto y_2^\lambda, \\ y_5 \mapsto f(f(a, y_2^\lambda), x_3^\lambda), \\ y_6 \mapsto f(f(a, y_2^\lambda), x_3^\lambda) \end{array} \right\}$$

Therefore, in general, $\psi \circ \phi^{\mathcal{C}} \upharpoonright Pvar \notin Sub^{\mathcal{C}}_{Pvar}$*.*

One way to capture the compoundness of $\varphi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar$ in terms of $Sub^{\mathcal{C}}_{Pvar}$ is to find a substitution $\psi^{\mathcal{B}}$ such that $\psi^{\mathcal{B}} \sqsubseteq \varphi^{\mathcal{B}}$ with the required $\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar \in Sub^{\mathcal{C}}_{Pvar}$. Safety follows since $\psi^{\mathcal{B}} \sqsubseteq \varphi^{\mathcal{B}}$. To maximise the precision, however, it is important to choose the least general $\psi^{\mathcal{B}}$ with the $\psi^{\mathcal{B}} \sqsubseteq \varphi^{\mathcal{B}}$ property. This observation on precision and ordering leads to the definition of the poset $Sub^{\mathcal{B}}_{Pvar}$ $(\sqsubseteq_{Pvar})$.

Definition 4.4 **(**$Sub^{\mathcal{B}}_{Pvar}$**)** *The poset* $Sub^{\mathcal{B}}_{Pvar}$ $(\sqsubseteq_{Pvar})$ *is defined by:*

$$Sub^{\mathcal{B}}_{Pvar} = \left\{ \phi^{\mathcal{B}} \in Sub^{\mathcal{B}} \;\middle|\; \begin{array}{c} dom(\phi^{\mathcal{B}}) \subseteq Pvar \wedge \\ \forall u^s \mapsto t_u \in \phi^{\mathcal{B}}. t_u \in Term^s_u \end{array} \right\}$$

$Sub^{\mathcal{B}}_{Pvar}$ $(\sqsubseteq_{Pvar})$ is a poset (rather than a preorder) by virtue of its restricted variable term bindings. Also, when equipped with a top element $fail^{\mathcal{B}}$, a lub $\sqcup^{\mathcal{B}}$ (corresponding to unification) and a glb $\sqcap^{\mathcal{B}}$ (corresponding to anti-unification [17]), $Sub^{\mathcal{B}}_{Pvar} \cup \{fail^{\mathcal{B}}\}$ $(\sqsubseteq_{Pvar})$ is a complete lattice.

With the aid of the lub, a mapping sub_{Pvar} can be formulated which calculates the most accurate, safe substitution $\varphi^{\mathcal{B}}$ such that $\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \upharpoonright Pvar \in Sub^{\mathcal{C}}_{Pvar}$.

Definition 4.5 (sub_{Pvar}) *The mapping* $sub_{Pvar} : Sub^{\mathcal{B}}_{Pvar} \rightarrow Sub^{\mathcal{B}}_{Pvar}$ *is defined by:*

$$sub_{Pvar}(\phi^{\mathcal{B}}) = \sqcup^{\mathcal{B}}\{\varphi^{\mathcal{B}} \in Sub^{\mathcal{B}}_{Pvar} \mid \varphi^{\mathcal{B}} \sqsubseteq_{Pvar} \phi^{\mathcal{B}}\}$$

Although the mapping sub_{Pvar} is defined in terms of the lub of a possibly large set, sub_{Pvar} can be implemented efficiently.

Example 4.3 *Continuing with example 4.2, put* $\psi^{\mathcal{B}} = sub_{Pvar}(\varphi^{\mathcal{B}})$ *and thus*

$$\psi^{\mathcal{B}} = \left\{ \begin{array}{l} x_1^{\lambda} \mapsto f(a, x_1^1), \\ x_2^{\lambda} \mapsto f(f(a, x_2^{1\cdot 2}), x_2^2), \\ y_1^{\lambda} \mapsto a, \\ y_3^{\lambda} \mapsto a, \\ y_5^{\lambda} \mapsto f(f(a, y_5^{1\cdot 2}), y_5^2), \\ y_6^{\lambda} \mapsto f(f(a, y_6^{1\cdot 2}), y_6^2) \end{array} \right\}, \quad \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar = \left\{ \begin{array}{l} x_1 \mapsto f(a, x_1^1), \\ x_2 \mapsto f(f(a, x_2^{1\cdot 2}), x_2^2), \\ y_1 \mapsto a, \\ y_3 \mapsto a, \\ y_5 \mapsto f(f(a, y_5^{1\cdot 2}), y_5^2), \\ y_6 \mapsto f(f(a, y_6^{1\cdot 2}), y_6^2) \end{array} \right\}$$

Observe that $\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar \in Sub^{\mathcal{C}}_{Pvar}$ *as required.*

With the compoundness component $\mu^{\mathcal{C}} = \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar$ computed, the problem reduces to calculating the sharing and freeness component $\mu^{\mathcal{SF}}$. The sharing and freeness component is computed by transforming the $Type_{Pvar}$ problem into a $Share_{Pvar} \times Free_{Pvar}$ problem. The idea is that if $mgu^{\mathcal{SF}}$ is efficient and fast, then $mgu^{\mathcal{SF}}$ should be used wherever possible. The transformation is under-pinned by lemma 4.3 and theorem 4.2. In theorem 4.2 confluence is used to show that $[\psi^{\mathcal{D}} \circ (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)] \restriction Pvar = \varphi \circ \phi$ where $\psi^{\mathcal{D}} = mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta)$ and $\vartheta = mgu(\theta \cup \psi^{\mathcal{B}})$. This closes the gap between $Type_{Pvar}$ and $Share_{Pvar} \times Free_{Pvar}$ since if it is possible to find a $\vartheta^{\mathcal{SF}}$ such that $\vartheta \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta^{\mathcal{SF}})$ then it is also possible to compute $\mu^{\mathcal{SF}}$ with $mgu^{\mathcal{SF}}$. More exactly, $mgu^{\mathcal{SF}}(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E), \vartheta^{\mathcal{SF}}) = \mu^{\mathcal{SF}}$. This is the first result. The second result, lemma 4.3, asserts that an auxiliary operation $mgu^{\mathcal{BSF}}$ can provide the required $\vartheta^{\mathcal{SF}}$. To be precise, if $mgu^{\mathcal{BSF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}}, \vartheta^{\mathcal{SF}})$ then $\vartheta \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta^{\mathcal{SF}})$. Thus both results, when applied together, define a procedure for computing $\mu^{\mathcal{SF}}$. In short, $\mu^{\mathcal{SF}} = mgu^{\mathcal{SF}}(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E), \vartheta^{\mathcal{SF}})$ where $mgu^{\mathcal{BSF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}}, \vartheta^{\mathcal{SF}})$.

The operation $mgu^{\mathcal{BSF}}$ is designed to calculate an abstraction for $\vartheta = mgu(\theta \cup \psi^{\mathcal{B}})$. This, in fact, can be calculated relatively straightforwardly because of the simple structure of $\psi^{\mathcal{B}}$. First, the domain variables of $\psi^{\mathcal{B}}$ do not share in $\psi^{\mathcal{B}}$: $var(\psi^{\mathcal{B}}(u)) \cap var(\psi^{\mathcal{B}}(v)) = \emptyset$ for any pair of distinct variables $u, v \in dom(\psi^{\mathcal{B}})$. Second, the domain variables of $\psi^{\mathcal{B}}$ are guaranteed to be linear: $\chi(\psi^{\mathcal{B}}(u)) = 1$ for all $u \in dom(\psi^{\mathcal{B}})$. This means that $mgu^{\mathcal{BSF}}$ can calculate an accurate abstraction for ϑ.

Definition 4.6 ($mgu^{\mathcal{BSF}}$) *The relation* $mgu^{\mathcal{BSF}} : Sub^{\mathcal{B}}_{Pvar} \times (Share_{Pvar} \times Free_{Pvar}) \times (Share_{Pvar} \times Free_{Pvar})$ *is defined by:*

$$\begin{array}{c} mgu^{\mathcal{BSF}}(\emptyset, \phi^{\mathcal{SF}}, \phi^{\mathcal{SF}}) \\ mgu^{\mathcal{BSF}}(u = t : E, \phi^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \text{ if } mgu^{\mathcal{BSF}}(E, \tau^{\mathcal{SF}}, \varphi^{\mathcal{SF}}) \end{array}$$

where $\tau^{\mathcal{S}} = mgu^{\mathcal{BS}}(u, t, \phi^{\mathcal{SF}})$ *and* $\tau^{\mathcal{F}} = mgu^{\mathcal{BF}}(u, t, \phi^{\mathcal{SF}})$.

Definition 4.7 ($mgu^{\mathcal{BS}}$ **and** $mgu^{\mathcal{BF}}$)

$$mgu^{\mathcal{BS}}(u, t, \phi^{\mathcal{SF}}) = \left\{ \begin{array}{l} \tau^{\mathcal{S}} \cup (\mathcal{U}_u \,\Box\, U_t) \text{ if } u \in \phi^{\mathcal{F}} \\ \tau^{\mathcal{S}} \cup (\mathcal{U}_u \,\Box\, \mathcal{U}_t) \text{ otherwise} \end{array} \right.$$

$$
\begin{aligned}
\textit{where}\ \tau^{\mathcal{S}} &= \phi^{\mathcal{S}} \setminus rel(u, \phi^{\mathcal{S}}) \\
\mathcal{U}_u &= \{U_u \setminus \{u\} \mid U_u \in rel(u, \phi^{\mathcal{S}})\} \\
U_t &= \{\{u_t\} \mid u_t \in var(t)\} \\
\mathcal{U}_t &= \{U_t \subseteq var(t) \mid U_t \neq \emptyset\}
\end{aligned}
$$

$$
mgu^{\mathcal{BF}}(u, t, \phi^{\mathcal{SF}}) = \begin{cases} (\phi^{\mathcal{F}} \setminus var(rel(u, \phi^{\mathcal{S}}))) \cup var(t) & \textit{if}\ u \in \phi^{\mathcal{F}} \\ (\phi^{\mathcal{F}} \setminus var(rel(u, \phi^{\mathcal{S}}))) & \textit{otherwise} \end{cases}
$$

The irregular structure of $mgu^{\mathcal{BS}}$ and $mgu^{\mathcal{BF}}$ stems from the requirement that $\vartheta \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta^{\mathcal{SF}})$ whereas $\phi^{\mathcal{D}} \in \gamma^{\mathcal{SF}}_{cod(\phi^{\mathcal{C}})}(\phi^{\mathcal{SF}})$ (referring to the $\phi^{\mathcal{D}}$ of lemma 4.3). Put another way, the $\vartheta^{\mathcal{SF}}$ is expressed in terms of the variables $cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)$ whereas $\phi^{\mathcal{D}}$ is represented in terms of the variables $cod(\phi^{\mathcal{C}})$. This means that $mgu^{\mathcal{BSF}}$ has to incrementally extend from $Share_{cod(\phi^{\mathcal{C}})} \times Free_{cod(\phi^{\mathcal{C}})}$ to $Share_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)} \times Free_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}$. This is the main technicality involved in establishing lemma 4.3 in proof 4.3.

As with $mgu^{\mathcal{SF}}$, it is a convenient shorthand to regard $mgu^{\mathcal{BSF}}$ as a mapping. Like before, $mgu^{\mathcal{BSF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}}, \vartheta^{\mathcal{SF}})$ yields a safe $\vartheta^{\mathcal{SF}}$ regardless of the order in which $\psi^{\mathcal{B}}$ is solved.

Lemma 4.3

$$
\begin{array}{ll}
\phi \restriction Pvar = \phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar & \wedge \\
\varphi^{\mathcal{B}} \in mgu(\phi^{\mathcal{C}}(E)) & \wedge \\
\psi^{\mathcal{B}} = sub_{Pvar}(\varphi^{\mathcal{B}}) & \wedge \\
\vartheta \in mgu(\psi^{\mathcal{B}} \cup \phi^{\mathcal{D}}) & \wedge \\
mgu^{\mathcal{BSF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}}, \vartheta^{\mathcal{SF}}) & \Rightarrow \\
\quad \vartheta \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta^{\mathcal{SF}}) &
\end{array}
$$

Definition 4.8 and theorem 4.2 summarise how $mgu^{\mathcal{BSF}}$ and $mgu^{\mathcal{SF}}$ fit together to form $mgu^{\mathcal{CSF}}$. The corresponding proof is labelled 7.11.

Definition 4.8 *The mapping* $mgu^{\mathcal{CSF}} : Eqn \times Type_{Pvar} \to Type_{Pvar}$ *is defined by:*

$$
\begin{aligned}
& mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}}) = \langle \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar, \mu^{\mathcal{S}}, \mu^{\mathcal{F}} \rangle \ \textit{where} \\
& \quad \varphi^{\mathcal{B}} \ \in mgu(\phi^{\mathcal{C}}(E)) \wedge \\
& \quad \psi^{\mathcal{B}} \ = sub_{Pvar}(\varphi^{\mathcal{B}}) \wedge \\
& \quad \mu^{\mathcal{SF}} = mgu^{\mathcal{SF}}(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E), mgu^{\mathcal{BSF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}}))
\end{aligned}
$$

Theorem 4.2

$$
\begin{aligned}
& \phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}}) \wedge \varphi \in mgu(\phi(E)) \wedge \\
& \quad var(E) \subseteq Pvar \Rightarrow \varphi \circ \phi \in \gamma^{\mathcal{CSF}}_{Pvar}(mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}}))
\end{aligned}
$$

Example 4.4 *Consider the calculation of* $mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}}) = \mu^{\mathcal{CSF}}$ *adopting the* E *of example 4.1 and where*

$$
\begin{aligned}
\phi^{\mathcal{C}} &= \{x_1 \mapsto x_1^{\lambda}, x_2 \mapsto x_2^{\lambda}, x_3 \mapsto x_3^{\lambda}, y_1 \mapsto y_1^{\lambda}, \ldots, y_6 \mapsto y_6^{\lambda}\} \\
\phi^{\mathcal{S}} &= \{\emptyset, \{x_1^{\lambda}\}, \{x_2^{\lambda}\}, \{x_3^{\lambda}\}, \{y_1^{\lambda}, y_2^{\lambda}\}, \{y_2^{\lambda}\}, \{y_3^{\lambda}\}, \{y_5^{\lambda}\}, \{y_6^{\lambda}\}\} \\
\phi^{\mathcal{F}} &= \{x_1^{\lambda}, x_2^{\lambda}, x_3^{\lambda}, y_1^{\lambda}, y_3^{\lambda}, y_5^{\lambda}, y_6^{\lambda}\}
\end{aligned}
$$

Thus, using the $\varphi^{\mathcal{B}} \in mgu(\phi^{\mathcal{C}}(E))$ *and* $\psi^{\mathcal{B}} = sub_{Pvar}(\varphi^{\mathcal{B}})$ *of example 4.3, then if* $\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E)$ *is simplified to* E' *through pre-unification*

$$\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) = \left\{ \begin{array}{c} f(a, x_1^1) = f(a, y_2^\lambda), \\ f(a, x_1^1) = f(a, y_4^\lambda), \\ a = a, \\ f(f(a, y_5^{1\cdot 2}), y_5^2) = f(f(a, x_2^{1\cdot 2}), x_2^2), \\ f(f(a, y_6^{1\cdot 2}), y_6^2) = f(f(a, x_2^{1\cdot 2}), x_2^2), \\ f(f(a, y_6^{1\cdot 2}), y_6^2) = f(f(a, x_1^1), x_3^\lambda) \end{array} \right\}, \quad E' = \left\{ \begin{array}{c} x_1^1 = y_2^\lambda, \\ x_1^1 = y_4^\lambda, \\ y_5^{1\cdot 2} = x_2^{1\cdot 2}, \\ y_5^2 = x_2^2, \\ y_6^{1\cdot 2} = x_2^{1\cdot 2}, \\ y_6^2 = x_2^2, \\ y_6^{1\cdot 2} = x_1^1, \\ y_6^2 = x_3^\lambda \end{array} \right\}$$

Moving onto the calculation of $mgu^{\mathcal{BSF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}})$, *putting* $\vartheta_1^{\mathcal{CS}} = \phi^{\mathcal{CSF}}$, *and considering each equation of* $\phi^{\mathcal{B}} = \{u_i = t_i\}_{i=1}^6$ *in turn*

$$\begin{aligned}
\vartheta_1^{\mathcal{CS}} &= \langle \{\emptyset, \{x_1^\lambda\}, \{x_2^\lambda\}, \{x_3^\lambda\}, \{y_1^\lambda, y_2^\lambda\}, \{y_2^\lambda\}, \{y_3^\lambda\}, \{y_5^\lambda\}, \{y_6^\lambda\}\}, \\
&\quad \{x_1^\lambda, x_2^\lambda, x_3^\lambda, y_1^\lambda, y_3^\lambda, y_5^\lambda, y_6^\lambda\}\rangle \\
\vartheta_2^{\mathcal{CS}} &= \langle mgu^{\mathcal{BS}}(x_1^\lambda, f(a, x_1^1), \vartheta_1^{\mathcal{CS}}), mgu^{\mathcal{BF}}(x_1^\lambda, f(a, x_1^1), \vartheta_1^{\mathcal{CS}})\rangle \\
&= \langle \{\emptyset, \{x_1^1\}, \{x_2^\lambda\}, \{x_3^\lambda\}, \{y_1^\lambda, y_2^\lambda\}, \{y_2^\lambda\}, \{y_3^\lambda\}, \{y_5^\lambda\}, \{y_6^\lambda\}\}, \\
&\quad \{x_1^1, x_2^\lambda, x_3^\lambda, y_1^\lambda, y_3^\lambda, y_5^\lambda, y_6^\lambda\}\rangle \\
\vartheta_3^{\mathcal{CS}} &= \langle \{\emptyset, \{x_1^1\}, \{x_2^{1\cdot 2}\}, \{x_2^2\}, \{x_3^\lambda\}, \{y_1^\lambda, y_2^\lambda\}, \{y_2^\lambda\}, \{y_3^\lambda\}, \{y_5^\lambda\}, \{y_6^\lambda\}\}, \\
&\quad \{x_1^1, x_2^{1\cdot 2}, x_2^2, x_3^\lambda, y_1^\lambda, y_3^\lambda, y_5^\lambda, y_6^\lambda\}\rangle \\
\vartheta_4^{\mathcal{CS}} &= \langle \{\emptyset, \{x_1^1\}, \{x_2^{1\cdot 2}\}, \{x_2^2\}, \{x_3^\lambda\}, \{y_2^\lambda\}, \{y_3^\lambda\}, \{y_5^\lambda\}, \{y_6^\lambda\}\}, \\
&\quad \{x_1^1, x_2^{1\cdot 2}, x_2^2, x_3^\lambda, y_3^\lambda, y_5^\lambda, y_6^\lambda\}\rangle \\
\vartheta_5^{\mathcal{CS}} &= \langle \{\emptyset, \{x_1^1\}, \{x_2^{1\cdot 2}\}, \{x_2^2\}, \{x_3^\lambda\}, \{y_2^\lambda\}, \{y_5^\lambda\}, \{y_6^\lambda\}\}, \\
&\quad \{x_1^1, x_2^{1\cdot 2}, x_2^2, x_3^\lambda, y_5^\lambda, y_6^\lambda\}\rangle \\
\vartheta_6^{\mathcal{CS}} &= \langle \{\emptyset, \{x_1^1\}, \{x_2^{1\cdot 2}\}, \{x_2^2\}, \{x_3^\lambda\}, \{y_2^\lambda\}, \{y_5^{1\cdot 2}\}, \{y_5^2\}, \{y_6^\lambda\}\}, \\
&\quad \{x_1^1, x_2^{1\cdot 2}, x_2^2, x_3^\lambda, y_5^{1\cdot 2}, y_5^2, y_6^{1\cdot 2}, y_6^2\}\rangle \\
\vartheta_7^{\mathcal{CS}} &= \langle \{\emptyset, \{x_1^1\}, \{x_2^{1\cdot 2}\}, \{x_2^2\}, \{x_3^\lambda\}, \{y_2^\lambda\}, \{y_5^{1\cdot 2}\}, \{y_5^2\}, \{y_6^{1\cdot 2}\}, \{y_6^2\}\}, \\
&\quad \{x_1^1, x_2^{1\cdot 2}, x_2^2, x_3^\lambda, y_5^{1\cdot 2}, y_5^2, y_6^{1\cdot 2}, y_6^2\}\rangle
\end{aligned}$$

With $\vartheta^{\mathcal{SF}} = \vartheta_7^{\mathcal{SF}}$, $mgu^{\mathcal{SF}}(E', \vartheta^{\mathcal{SF}}) = \mu^{\mathcal{SF}} = \langle \{\emptyset, \{x_2^2, x_3^\lambda, y_5^2, y_6^2\}\},\ \{x_2^2, x_3^\lambda, y_5^2, y_6^2\}\rangle$. *Thus* $\mu^{\mathcal{CSF}}$ *is given by*

$$\mu^{\mathcal{C}} = \left\{ \begin{array}{l} x_1 \mapsto f(a, x_1^1), \\ x_2 \mapsto f(f(a, x_2^{1\cdot 2}), x_2^2), \\ x_3 \mapsto x_3^\lambda, \\ y_1 \mapsto a, \\ y_2 \mapsto y_2^\lambda, \\ y_3 \mapsto a, \\ y_4 \mapsto y_4^\lambda, \\ y_5 \mapsto f(f(a, y_5^{1\cdot 2}), y_5^2), \\ y_6 \mapsto f(f(a, y_6^{1\cdot 2}), y_6^2) \end{array} \right\}, \quad \begin{array}{l} \mu^{\mathcal{S}} = \{\emptyset, \{x_2^2, x_3^\lambda, y_5^2, y_6^2\}\}, \\ \\ \mu^{\mathcal{F}} = \{x_2^2, x_3^\lambda, y_5^2, y_6^2\} \end{array}$$

Clearly $\mu^{\mathcal{CSF}}$ *is a refinement of the* $\langle \{\emptyset,\ \{x_2, x_3, y_5, y_6\}\},\ \{x_3\}\rangle$ *derived in example 4.1. In addition to recording the sharing between* x_2, x_3, y_5 *and* y_6 *and freeness of* x_3, $\mu^{\mathcal{CSF}}$ *records term structure in* $\mu^{\mathcal{C}}$, *details sharing to the precision of sub-terms in* $\mu^{\mathcal{S}}$, *and infers which sub-terms are free in* $\mu^{\mathcal{F}}$.

Example 4.5 *By widening $\psi^{\mathcal{B}}$, finiteness can be enforced and the precision of the analysis can be adjusted to suit the application. Thus, returning to the calculation of $mgu^{\mathcal{CSF}}(\{a = b\}, \phi^{\mathcal{CSF}}) = \mu^{\mathcal{CSF}}$ in example 4.4, consider the effect of throttling $\psi^{\mathcal{B}}$ to depth-0 and depth-1 to obtain*

$$\psi_0^{\mathcal{B}} = \left\{ \begin{array}{l} y_1^\lambda \mapsto a, \\ y_3^\lambda \mapsto a \end{array} \right\}, \quad \psi_1^{\mathcal{B}} = \left\{ \begin{array}{l} x_1^\lambda \mapsto f(a, x_1^1), \\ x_2^\lambda \mapsto f(x_2^1, x_2^2), \\ y_1^\lambda \mapsto a, \\ y_3^\lambda \mapsto a, \\ y_5^\lambda \mapsto f(y_5^1, y_5^2), \\ y_6^\lambda \mapsto f(y_6^1, y_6^2) \end{array} \right\}$$

Like before, $\psi_i^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E)$ can be simplified via pre-unification to obtain E_i'

$$E_0' = \left\{ \begin{array}{l} x_1^\lambda = f(a, y_2^\lambda), \\ x_1^\lambda = f(a, y_4^\lambda), \\ y_5^\lambda = x_2^\lambda, \\ y_6^\lambda = x_2^\lambda, \\ y_6^\lambda = f(x_1^\lambda, x_3^\lambda) \end{array} \right\}, \quad E_1' = \left\{ \begin{array}{l} x_1^1 = y_2^\lambda, \\ x_1^1 = y_4^\lambda, \\ y_5^1 = x_2^1, \\ y_5^2 = x_2^2, \\ y_6^1 = x_2^1, \\ y_6^2 = x_2^2, \\ y_6^1 = f(a, x_1^1), \\ y_6^2 = x_3^\lambda \end{array} \right\}$$

Calculating $mgu^{\mathcal{BSF}}(\psi_i^{\mathcal{B}}, \phi^{\mathcal{SF}}) = \vartheta_i^{\mathcal{SF}}$ gives

$$\vartheta_0^{\mathcal{CS}} = \langle \{\emptyset, \{x_1^\lambda\}, \{x_2^\lambda\}, \{x_3^\lambda\}, \{y_2^\lambda\}, \{y_5^\lambda\}, \{y_6^\lambda\}\}, \{x_1^\lambda, x_2^\lambda, x_3^\lambda, y_5^\lambda, y_6^\lambda\} \rangle$$

$$\begin{array}{rl} \vartheta_1^{\mathcal{CS}} = & \langle \{\emptyset, \{x_1^1\}, \{x_2^1\}, \{x_2^2\}, \{x_3^\lambda\}, \{y_2^\lambda\}, \{y_5^1\}, \{y_5^2\}, \{y_6^1\}, \{y_6^2\}\}, \\ & \{x_1^1, x_2^1, x_2^2, x_3^\lambda, y_5^1, y_5^2, y_6^1, y_6^2\} \rangle \end{array}$$

Finally $mgu^{\mathcal{SF}}(E_i', \vartheta_i^{\mathcal{SF}})$ defines $\mu_i^{\mathcal{SF}}$ and $\psi_i^{\mathcal{B}} \circ \phi^{\mathcal{C}}$ defines $\mu_i^{\mathcal{C}}$

$$\mu_0^{\mathcal{SF}} = \langle \{\emptyset, \{x_2^\lambda, x_3^\lambda, y_5^\lambda, y_6^\lambda\}\}, \{x_3^\lambda\} \rangle$$

$$\mu_1^{\mathcal{SF}} = \langle \{\emptyset, \{x_2^2, x_3^\lambda, y_5^2, y_6^2\}\}, \{x_2^2, x_3^\lambda, y_5^2, y_6^2\} \rangle$$

$$\mu_0^{\mathcal{C}} = \left\{ \begin{array}{l} x_1 \mapsto x_1^\lambda, \\ x_2 \mapsto x_2^\lambda, \\ x_3 \mapsto x_3^\lambda, \\ y_1 \mapsto a, \\ y_2 \mapsto y_2^\lambda, \\ y_3 \mapsto a, \\ y_4 \mapsto y_4^\lambda, \\ y_5 \mapsto y_5^\lambda, \\ y_6 \mapsto y_6^\lambda \end{array} \right\}, \quad \mu_1^{\mathcal{C}} = \left\{ \begin{array}{l} x_1 \mapsto f(a, x_1^1), \\ x_2 \mapsto f(x_2^1, x_2^2), \\ x_3 \mapsto x_3^\lambda, \\ y_1 \mapsto a, \\ y_2 \mapsto y_2^\lambda, \\ y_3 \mapsto a, \\ y_4 \mapsto y_4^\lambda, \\ y_5 \mapsto f(y_5^1, y_5^2), \\ y_6 \mapsto f(y_6^1, y_6^2) \end{array} \right\}$$

Note that even at depth-0, the analysis captures compoundness information that cannot be derived by a conventional Share $\times$ Free analysis. Note too that since variables only occur in μ at level 1, then in terms of sharing and freeness, depth-1 analysis is just as accurate as depth-2 analysis. Finally observe that depth-0 analysis fails to infer the sharing and freeness of sub-terms.

Note that $mgu^{\mathcal{CSF}}$ and each of its constituent parts are independent of k. Thus $mgu^{\mathcal{CSF}}$ is an abstract equation solver for depth-k abstractions of arbitrary k. However, arbitrary k can lead to non-terminating computations and therefore, in general, some method for enforcing convergence and finiteness is required. One simple way of ensuring termination is to widening at the level of $mgu^{\mathcal{CSF}}$. This approach requires just one additional construction, $mgu_k^{\mathcal{CSF}}$, which thresholds the abstract unifier to depth-k. This approach compares very favourably with the finiteness machinery which is detailed in [16].

Definition 4.9 ($mgu_k^{\mathcal{CSF}}$) *The mapping* $mgu_k^{\mathcal{CSF}} : Eqn \times Sub_{Pvar}^{\mathcal{CSF}} \rightarrow Sub_{Pvar}^{\mathcal{CSF}}$ *is defined by:*

$$mgu_k^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}}) = \bigtriangledown_k^{\mathcal{CSF}}(mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}}))$$

Then, with the addition of some renaming machinery, $mgu_k^{\mathcal{CSF}}$ defines a depth-k version of $unify^c$, $unify_k^{\mathcal{CSF}}$. To define $unify_k^{\mathcal{CSF}}$ and prove safety it is necessary to introduce an abstract restriction operator.

Definition 4.10 (abstract restriction) *The abstract restriction operator,* $\cdot \restriction^{\mathcal{CSF}} \cdot$, *is defined by:*

$$\phi^{\mathcal{CSF}} \restriction^{\mathcal{CSF}} U = \langle \phi^{\mathcal{C}} \restriction^{\mathcal{C}} U, \phi^{\mathcal{S}} \restriction^{\mathcal{S}} U, \phi^{\mathcal{F}} \restriction^{\mathcal{F}} U \rangle \text{ where } \begin{array}{l} \phi^{\mathcal{C}} \restriction^{\mathcal{C}} U = \phi^{\mathcal{C}} \restriction U \\ \phi^{\mathcal{S}} \restriction^{\mathcal{S}} U = \{U \cap U' \mid U' \in \phi^{\mathcal{S}}\} \\ \phi^{\mathcal{F}} \restriction^{\mathcal{F}} U = U \cap \phi^{\mathcal{F}} \end{array}$$

The definition of $unify_k^{\mathcal{CSF}}$ is given below with its safety stated as theorem 4.3. Theorem 4.3 assumes $var(a) \cup var(b) \subseteq Pvar$ and is established by proof 7.12.

Definition 4.11 ($unify_k^{\mathcal{CSF}}$) *The mapping* $unify_k^{\mathcal{CSF}} : Atom \times Type_{Pvar} \times Atom \times Type_{Pvar} \rightarrow Type_{Pvar}$ *is defined by:*

$$unify_k^{\mathcal{CSF}}(a, \phi^{\mathcal{CSF}}, b, \psi^{\mathcal{CSF}}) = mgu_k^{\mathcal{CSF}}(\{a = \Upsilon(b)\}, \phi^{\mathcal{CSF}} \cup \Upsilon(\psi^{\mathcal{CSF}})) \restriction^{\mathcal{CSF}} Pvar$$

Theorem 4.3 (local safety of $unify_k^{\mathcal{CSF}}$**)**

$$\begin{array}{l} \Phi \subseteq \gamma_{Pvar}^{\mathcal{CSF}}(\phi^{\mathcal{CSF}}) \wedge \Psi \subseteq \gamma_{Pvar}^{\mathcal{CSF}}(\psi^{\mathcal{CSF}}) \Rightarrow \\ \quad unify^c(a, \Phi, b, \Psi) \subseteq \gamma_{Pvar}^{\mathcal{CSF}}(unify_k^{\mathcal{CSF}}(a, \phi^{\mathcal{CSF}}, b, \psi^{\mathcal{CSF}})) \end{array}$$

Example 4.6 *For the sake of comparison with the freeness analysis of [23], consider the calculation of* $unify_k^{\mathcal{CSF}}(a, \phi_a^{\mathcal{CSF}}, b, \phi_b^{\mathcal{CSF}})$ *for* $k = 1$, 2 *and* 3 *where* $a = p(x_1, x_1, a, x_2, x_2, f(x_1, x_3))$, $b = p(f(x_1, x_2), f(x_3, x_4), x_3, x_5, x_6, x_6)$ *and*

$$\begin{array}{ll} \phi_a^{\mathcal{C}} = \{x_1 \mapsto x_1^{\lambda}, x_2 \mapsto x_2^{\lambda}, x_3 \mapsto x_3^{\lambda}\}, & \phi_b^{\mathcal{C}} = \{x_1 \mapsto x_1^{\lambda}, \ldots, x_6 \mapsto x_6^{\lambda}\}, \\ \phi_a^{\mathcal{S}} = \{\emptyset, \{x_1^{\lambda}\}, \{x_2^{\lambda}\}, \{x_3^{\lambda}\}\}, & \phi_b^{\mathcal{S}} = \{\emptyset, \{x_1^{\lambda}, x_2^{\lambda}\}, \{x_2^{\lambda}\}, \{x_3^{\lambda}\}, \{x_5^{\lambda}\}, \{x_6^{\lambda}\}\}, \\ \phi_a^{\mathcal{F}} = \{x_1^{\lambda}, x_2^{\lambda}, x_3^{\lambda}\}, & \phi_b^{\mathcal{F}} = \{x_1^{\lambda}, x_3^{\lambda}, x_5^{\lambda}, x_6^{\lambda}\} \end{array}$$

Thus, supposing $\Upsilon(x_i) = y_i$ *where* $x_i \in Pvar$ *and* $y_i \in Rvar$, *then the computation reduces to* $mgu_k^{\mathcal{CSF}}(\{a = \Upsilon(b)\}, \phi_a^{\mathcal{CSF}} \cup \Upsilon(\phi_b^{\mathcal{CSF}})) \restriction^{\mathcal{CSF}} Pvar$ *which, in turn, simplifies to* $\bigtriangledown_k^{\mathcal{CSF}}(mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}})) \restriction^{\mathcal{CSF}} Pvar$ *adopting the* E *and* $\phi^{\mathcal{CSF}}$ *of example 4.4. Thus, by example 4.4, it follows that* $unify_k^{\mathcal{CSF}}(a, \phi_a^{\mathcal{CSF}}, b, \phi_b^{\mathcal{CSF}}) = \varphi_k^{\mathcal{CSF}}$ *where*

$$\varphi_1^{\mathcal{C}} = \left\{ \begin{array}{l} x_1 \mapsto x_1^{\lambda}, \\ x_2 \mapsto x_2^{\lambda}, \\ x_3 \mapsto x_3^{\lambda}, \end{array} \right\}, \quad \varphi_1^{\mathcal{S}} = \{\emptyset, \{x_2^{\lambda}, x_3^{\lambda}\}\}, \quad \varphi_1^{\mathcal{F}} = \{x_2^{\lambda}, x_3^{\lambda}, \}$$

$$\varphi_2^{\mathcal{C}} = \left\{ \begin{array}{l} x_1 \mapsto f(a, x_1^1), \\ x_2 \mapsto f(x_2^1, x_2^2), \\ x_3 \mapsto x_3^{\lambda}, \end{array} \right\}, \quad \varphi_2^{\mathcal{S}} = \{\emptyset, \{x_2^1, x_3^{\lambda}\}\}, \quad \varphi_2^{\mathcal{F}} = \{x_2^1, x_3^{\lambda}, \}$$

$$\varphi_3^{\mathcal{C}} = \left\{ \begin{array}{l} x_1 \mapsto f(a, x_1^1), \\ x_2 \mapsto f(f(a, x_2^{1\cdot 2}), x_2^2), \\ x_3 \mapsto x_3^{\lambda}, \end{array} \right\}, \quad \varphi_3^{\mathcal{S}} = \{\emptyset, \{x_2^2, x_3^{\lambda}\}\}, \quad \varphi_3^{\mathcal{F}} = \{x_2^2, x_3^{\lambda}, \}$$

By comparison, the freeness analysis of [23] likewise infers that x_3 is free and that x_2 and x_3 share. However, the analysis of [23] cannot infer the compoundness of x_1 and x_2, nor which sub-terms of x_2 and x_3 share, nor which sub-terms of x_2 are free.

5 Related and future work

Recently, four relevant proposals for computing freeness information have been put forward in the literature. In the first proposal [8], multiple domains and analyses are run in lock step. At each step, the abstract substitutions derived by the different analyses are compared and refined in order to improve the precision. This paper follows the trend for simultaneously tracing different properties (namely sharing, freeness and compoundness), explaining how accuracy and efficiency can be further improved by exploiting confluence. In particular the paper reports a depth-k analysis which cannot be synthesised in terms of the combined domain approach.

In the second proposal [6], the correctness of a sharing and freeness analysis is considered. An abstract unification algorithm is proposed as a basis for constructing accurate freeness analyses with a domain formulated in terms of a system of abstract equations. Safety follows because the abstract algorithm mimics the unification algorithm in an intuitive way. Correctness is established likewise here. One important distinction between the two works is that this paper uses the domain $Sub_{Pvar}^{\mathcal{C}}$ to potentially encode more accurate sharing and freeness information than the abstract equations of [6]. Consequently, a depth-k analysis cannot be derived from the abstract equations of [6]. Also, as pointed out in [3], "it is doubtful whether it (the abstract unification algorithm of [6]) can be the basis for a very efficient analysis".

Third, in [16], the format of sharing groups is revised to capture structural properties of substitutions. An abstract substitution is represented as a set of sharing groups where a sharing group is a (possibly empty) set of program variable and path pairs. The paths permit sharing groups to record the positions of shared variables within a binding, that is, where the shared variable occurs in the terms to which the program variables are bound. Correctness is proved. The usefulness of the approach, however, is compromised by its potential inefficiency. The essential problem is that paths are required to be concatenated, compared and truncated at almost every stage of abstract unification algorithm. This can be expensive. Moreover, because of the way paths are widened, much of the formal analysis machinery has to be duplicated: first, a depth-∞ analysis is formulated; second, a depth-k analysis is constructed. In contrast, the $Sub_{Pvar}^{\mathcal{C}}$ domain of this paper was chosen carefully to simplify widening and ease the construction and presentation of the analysis. Also, in terms of implementation and practicality, the analysis presented in this paper applies confluence in a novel way to split the analysis into small, simple and efficient units.

Very recently, in a fourth proposal [4], a sharing and freeness analysis is formulated in terms of a transition system which reduces a set of abstract equations to an abstract solved form. Sharing is represented in a sharing group fashion with variables enriched with linearity and freeness information by an annotation mapping. Depth-k sharing, groundness, freeness and compoundness information can be represented to a bounded depth by virtue of the abstract equations. To be precise, the domain is formulated as a set of equivalence classes of abstract equations. The domain is similar in spirit to $Type_{Pvar}$: sharing groups and freeness sets record the aliasing between variable place markers that are introduced in the structural component of the domain. One distinction, however, is the $Sub^{\mathcal{C}}_{Pvar}$ domain is engineered to be a poset whereas the abstract equations of [4] lead to a preorder which makes termination more subtle. A second difference is that in this paper the frequently used operations like projection $\restriction^{\mathcal{CSF}}$, lub $\sqcup^{\mathcal{CSF}}$, and widening $\bigtriangledown^{\mathcal{CSF}}_k$ are designed to be efficient. In [4], however, these frequently used operators are not discussed. A third distinction, is the emphasis this paper puts on modularity. Modularity follows by using confluence to split an analysis into its constituent parts. Modularity is advantageous since it simplifies both the presentation and the implementation. On a software engineering perspective it also permits an existing $Share_{Pvar} \times Free_{Pvar}$ unification code, for instance [23], to be plugged into the implementation to reuse valuable code. To be fair, however, the analysis of [4] does trace linearity and capture compoundness and definite sharing between the variables of the abstract equations. This might be useful. Of these three differences, linearity is probably the most significant, and the confluence approach (and in particular lemma 4.3) can extended to accommodate linearity. This is not difficult. The principal reason why linearity has not been directly addressed in this paper is that it is simply not (yet) clear that the extra complication is worthwhile if structure can be traced to depth-k. Future work will focus on implementation and benchmarking (will is a non-trivial study within itself) to suggest suitable k and to determine whether or not complexity of tracing linearity is worthwhile.

6 Conclusions

A powerful and formally justified analysis has been presented for inferring definite groundness, freeness and compoundness, and possible sharing to a bounded depth k. The analysis exploits confluence to split the analysis into its constituent components and introduce modularity. Modularity simplifies the implementation, aids the presentation, and leads to a well-structured proof of correctness. High precision follows from the combination of domains. The analysis propagates groundness with the accuracy of sharing groups and yet infers sharing and freeness to a precision which exceeds that of a normal freeness analysis. The analysis is significant because it can under-pin many optimisations in logic programming. It is likely to be particularly useful in the detection of independent and-parallelism.

Acknowledgements

Thanks are due to Manuel Hermenegildo, Francisco Bueno and Bob Kemp for useful discussions on sharing and freeness. This work was supported, in part, by ESPRIT project (6707) "ParForce".

References

[1] G. Birkhoff. *Lattice theory.* American Mathematical Society, 1948.

[2] M. Bruynooghe. A practical framework for the abstract interpretation of logic programs. *J. Logic Programming*, 10:91–124, 1991.

[3] M. Bruynooghe and M. Codish. Freeness, sharing, linearity and correctness – all at once. In *WSA'93*, pages 153–164, September 1993.

[4] M. Bruynooghe, M. Codish, and A. Mulkers. Abstract unification for a composite domain deriving sharing and freeness properties of program variables. In *ICLP'94 post-conference workshop on the verification and analysis of logic programs*, pages 213–230, Santa Margherita Ligure, Italy, 1994. June.

[5] J.-H. Chang and A. M. Despain. Semi-intelligent backtracking of prolog based static data dependency analysis. In *JICSLP'85*. IEEE Computer Society, 1985.

[6] M. Codish, D. Dams, G. Filé, and M. Bruynooghe. Freeness analysis for logic programs - and correctness? In *ICLP'93*, pages 116–131. MIT Press, June 1993.

[7] M. Codish, D. Dams, and E. Yardeni. Derivation and safety of an abstract unification algorithm for groundness and aliasing analysis. In *ICLP'91*, pages 79–93, Paris, France, 1991. MIT Press.

[8] M. Codish, A. Mulkers, M. Bruynooghe, M. J. García de la Banda, and M. Hermenegildo. Improving abstract interpretation by combining domains. In *PEPM'93*. ACM Press, 1993.

[9] A. Cortesi and G. Filé. Abstract interpretation of logic programs: an abstract domain for groundness, sharing, freeness and compoundness analysis. In *PEPM'91*, pages 52–61. ACM Press, 1991.

[10] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL'77*, pages 238–252. ACM Press, 1977.

[11] D. Dams. Personal communication on linearity lemma 2.2. July, 1993.

[12] S. K. Debray. Static inference of modes and data dependencies in logic programs. *ACM TOPLAS*, 11(3):418–450, July 1989.

[13] M. Hermenegildo and F. Rossi. Non-strict independent and-parallelism. In *ICLP'90*, pages 237–252, Jerusalem, 1990. MIT Press.

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

[15] A. King. A synergistic analysis for sharing and groundness which traces linearity. In *ESOP'94*, pages 363–378, Edinburgh, UK, 1994. Springer-Verlag.

[16] A. King and P. Soper. Depth-k sharing and freeness. In *ICLP'94*, Santa Margherita Ligure, Italy, 1994. MIT Press.

[17] J. Lassez, M. J. Maher, and K. Marriott. *Foundations of Deductive Databases and Logic Programming*, chapter Unification Revisited. Morgan Kaufmann, 1987.

[18] B. Le Charlier, K. Musumbu, and P. Van Hentenryck. A generic abstract interpretation algorithm and its complexity. In *ICLP'91*, pages 64–78. MIT Press, 1991.

[19] J. W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, 1987.

[20] K. Marriott and H. Søndergaard. Analysis of constraint logic programs. In *NACLP'90*, pages 531–547. MIT Press, 1990.

[21] C. Mellish. The automatic generation of mode declarations for prolog programs. In *Workshop on Logic Programming for Intelligent Systems*, Los Angeles, August 1981. Also available as DAI Research Paper 163, Department of Artificial Intelligence, University of Edinburgh.

[22] A. Melton, D. A. Schmidt, and D. E. Strecker. *Category Theory and Computer Programming*, chapter Galois Connections and Computer Science Applications, pages 299–312. Springer-Verlag, Berlin, 1986.

[23] K. Muthukumar and M. Hermenegildo. Combined determination of sharing and freeness of program variables through abstract interpretation. In *ICLP'91*, pages 49–63, Paris, France, 1991. MIT Press.

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

[25] H. Søndergaard. An application of the abstract interpretation of logic programs: occur-check reduction. In *ESOP'86*, pages 327–338, New York, 1986. Springer-Verlag.

[26] R. Sundararajan and J. Conery. An abstract interpretation scheme for groundness, freeness, and sharing analysis of logic programs. In *12^{th} FST and TCS Conference*, New Delhi, India, December 1992. Springer-Verlag.

[27] A. Taylor. *High Performance Prolog Implementation.* PhD thesis, Basser Department of Computer Science, NSW 2006, Australia, July 1991.

[28] W. Winsborough and A. Wærn. Transparent and-parallelism in the presence of shared free variables. In *ICLP'88*, pages 749–764. MIT Press, 1988.

[29] H. Xia. *Analyzing Data Dependencies, Detecting And-Parallelism and Optimizing Backtracking in Prolog Programs.* PhD thesis, University of Berlin, April 1989.

7 Appendix

Proof 7.1 (for lemma 3.1) *Let* $\phi^{\mathcal{CSF}}$, $\varphi^{\mathcal{CSF}} \in Type_{Pvar}$.

1. *Suppose* $\phi^{\mathcal{S}} = \emptyset$ *and* $\varphi^{\mathcal{S}} = \emptyset$. *Thus* $\phi^{\mathcal{CSF}} = \varphi^{\mathcal{CSF}}$ *and* $\gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}}) = \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$.
2. *Suppose* $\phi^{\mathcal{S}} = \emptyset$ *and* $\varphi^{\mathcal{S}} \neq \emptyset$. *Thus* $\gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}}) = \emptyset$ *whereas* $\gamma(\varphi^{\mathcal{CSF}}) \neq \emptyset$.
3. *Suppose* $\phi^{\mathcal{S}} \neq \emptyset$ *and* $\varphi^{\mathcal{S}} = \emptyset$. *Thus* $\gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}}) \neq \emptyset$ *whereas* $\gamma(\varphi^{\mathcal{CSF}}) = \emptyset$.
4. *Suppose* $\phi^{\mathcal{S}} \neq \emptyset$ *and* $\varphi^{\mathcal{S}} \neq \emptyset$.
   (a) *Suppose* $\phi^{\mathcal{C}} \neq \varphi^{\mathcal{C}}$.
      i. *Suppose* $U_\phi \in \phi^{\mathcal{S}}$ *and* $U_\phi \notin \varphi^{\mathcal{S}}$. *Thus* $\phi \in \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$ *such that* $\phi \notin \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$.
      ii. *Suppose* $U_\varphi \in \varphi^{\mathcal{S}}$ *and* $U_\varphi \notin \phi^{\mathcal{S}}$. *Thus* $\varphi \in \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$ *such that* $\varphi \notin \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$.
      iii. *Suppose* $\phi^{\mathcal{S}} = \varphi^{\mathcal{S}}$.
         A. *Suppose* $u^s \in \phi^{\mathcal{F}}$ *and* $u^s \notin \varphi^{\mathcal{F}}$. *Thus* $\varphi \in \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$ *such that* $\varphi \notin \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$.
         B. *Suppose* $u^s \in \varphi^{\mathcal{F}}$ *and* $u^s \notin \phi^{\mathcal{F}}$. *Thus* $\phi \in \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$ *such that* $\phi \notin \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$.
         C. *Suppose* $\phi^{\mathcal{F}} = \varphi^{\mathcal{F}}$. *Thus* $\phi^{\mathcal{CSF}} = \varphi^{\mathcal{CSF}}$ *and* $\gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}}) = \gamma(\varphi^{\mathcal{CSF}})$.
   (b) *Suppose* $\phi^{\mathcal{C}} = \varphi^{\mathcal{C}}$.
      i. *Suppose* $\phi^{\mathcal{C}}(u)[s] = f(t_1, \ldots, t_n)$ *and* $\varphi^{\mathcal{C}}(u)[s'] = f'(t'_1, \ldots, t'_{n'})$ *with* $f \neq f'$ *or* $n \neq n'$. *Thus* $\gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}}) \cap \gamma(\varphi^{\mathcal{CSF}}) = \emptyset$.
      ii. *Suppose* $\phi^{\mathcal{C}}(u)[s] \in Uvar$ *and* $\varphi^{\mathcal{C}}(u)[s'] = f(t_1, \ldots, t_n)$. *Thus* $\phi \in \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$ *such that* $\phi \notin \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$.
      iii. *Suppose* $\phi^{\mathcal{C}}(u)[s] = f(t_1, \ldots, t_n)$ *and* $\varphi^{\mathcal{C}}(u)[s'] \in Uvar$. *Thus* $\varphi \in \gamma^{\mathcal{CSF}}(\varphi^{\mathcal{CSF}})$ *such that* $\varphi \notin \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$.

Proof 7.2 (for lemma 3.2) *For brevity put* $\varphi^{\mathcal{CSF}} = \sqcup^{\mathcal{CSF}}(\Phi^{\mathcal{CSF}})$.

1. *Suppose* $\Phi^{\mathcal{CSF}} = \emptyset$.
   (a) *To show* $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \varphi^{\mathcal{CSF}}$ *for all* $\phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}$. *Immediate since* $\Phi^{\mathcal{CSF}} = \emptyset$.
   (b) *Suppose* $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \theta^{\mathcal{CSF}}$ *for all* $\phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}$. *To show* $\varphi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \theta^{\mathcal{CSF}}$. *Since* $\varphi^{\mathcal{CSF}} = \perp^{\mathcal{CSF}}$, $\gamma^{\mathcal{CSF}}_{Pvar}(\perp^{\mathcal{CSF}}) = \emptyset$ *and thus* $\varphi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \theta^{\mathcal{CSF}}$.
2. *Suppose* $\Phi^{\mathcal{CSF}} \neq \emptyset$.
   (a) *To show* $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \varphi^{\mathcal{CSF}}$ *for all* $\phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}$. *Let* $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ *for* $\phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}$ *so that* $\phi \restriction Pvar = \phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar$. *Note that* $\varphi^{\mathcal{C}} \sqsubseteq \phi^{\mathcal{C}}$ *as required.*
      i. *Since* $\varphi^{\mathcal{C}} \sqsubseteq \phi^{\mathcal{C}}$ *there exists* $\varphi^{\mathcal{D}}$ *such that* $\varphi^{\mathcal{D}} \circ \varphi^{\mathcal{C}} \restriction Pvar = \phi \restriction Pvar$. *Let* $u \in Uvar$. *To show* $occ_{cod(\varphi^{\mathcal{C}})}(u, \varphi^{\mathcal{D}}) \in \varphi^{\mathcal{S}}$. *Now* $\{v^s \in cod(\phi^{\mathcal{C}}) \mid u \in var(\phi(v)[s])\}$ $= occ_{cod(\phi^{\mathcal{C}})}(u, \phi^{\mathcal{D}}) \in \phi^{\mathcal{S}}$. *Hence* $\widehat{\phi^{\mathcal{B}}}(\{v^s \in cod(\phi^{\mathcal{C}}) \mid u \in var(\phi(v)[s])\}) \in \varphi^{\mathcal{S}}$ *where* $\phi^{\mathcal{B}} \circ \varphi^{\mathcal{C}} \restriction Pvar = \phi^{\mathcal{C}}$. *Therefore* $\{v^s \in cod(\varphi^{\mathcal{C}}) \mid u \in var(\varphi(v)[s])\} = occ_{cod(\varphi^{\mathcal{C}})}(u, \varphi^{\mathcal{D}}) \in \varphi^{\mathcal{S}}$.

ii. Suppose $u^s \in \varphi^{\mathcal{F}}$. But $u^s \in \phi^{\mathcal{F}}$ and thus $\phi(u)[s] \in Uvar$ as required.

Hence $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\varphi^{\mathcal{CSF}})$.

(b) Suppose $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \theta^{\mathcal{CSF}}$ for all $\phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}$ and $\theta^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \varphi^{\mathcal{CSF}}$. To show $\varphi^{\mathcal{CSF}} = \theta^{\mathcal{CSF}}$. Let $\phi^{\mathcal{CSF}} \in \Phi^{\mathcal{CSF}}$. Since $\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}}) \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\theta^{\mathcal{CSF}})$, $\theta^{\mathcal{C}} \sqsubseteq \phi^{\mathcal{C}}$. Hence $\theta^{\mathcal{C}} \sqsubseteq \varphi^{\mathcal{C}}$. But since $\gamma^{\mathcal{CSF}}_{Pvar}(\theta^{\mathcal{CSF}}) \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\varphi^{\mathcal{CSF}})$, $\varphi^{\mathcal{C}} \sqsubseteq \theta^{\mathcal{C}}$. Therefore $\theta^{\mathcal{C}} = \phi^{\mathcal{C}}$. Thus $\theta^{\mathcal{S}} \subseteq \varphi^{\mathcal{S}}$ and $\varphi^{\mathcal{F}} \subseteq \theta^{\mathcal{F}}$ since $\gamma^{\mathcal{CSF}}_{Pvar}(\theta^{\mathcal{CSF}}) \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\varphi^{\mathcal{CSF}})$.

i. To show $\varphi^{\mathcal{S}} \subseteq \theta^{\mathcal{S}}$. Let $occ_{cod(\varphi^{\mathcal{C}})}(u, \varphi^{\mathcal{D}}) \in \varphi^{\mathcal{S}}$ for some $u \in Uvar$ with $\varphi^{\mathcal{D}} \circ \varphi^{\mathcal{C}} \restriction Pvar \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$. Since $\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}}) \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\theta^{\mathcal{CSF}})$ and $\phi^{\mathcal{C}} = \theta^{\mathcal{C}}$ it follows that $occ_{cod(\theta^{\mathcal{C}})}(u, \varphi^{\mathcal{D}}) \in \theta^{\mathcal{S}}$.

ii. To show $\theta^{\mathcal{F}} \subseteq \varphi^{\mathcal{F}}$. Let $u^s \in \theta^{\mathcal{F}}$. Thus $\theta^{\mathcal{D}} \circ \theta^{\mathcal{C}}(u)[s] \in Uvar$ for all $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\theta^{\mathcal{CSF}})$ with $\phi \restriction Pvar = \theta^{\mathcal{D}} \circ \theta^{\mathcal{C}} \restriction Pvar$. Since $\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}}) \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\theta^{\mathcal{CSF}})$ and $\phi^{\mathcal{C}} = \theta^{\mathcal{C}}$ it follows that $\theta^{\mathcal{D}} \circ \phi^{\mathcal{C}}(u)[s] \in Uvar$ for all $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ with $\phi \restriction Pvar = \theta^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar$. Therefore $u^s \in \phi^{\mathcal{F}}$.

Hence $\varphi^{\mathcal{CSF}} = \theta^{\mathcal{CSF}}$.

Proof 7.3 (for corollary 3.1) *Put* $\sqcap^{\mathcal{CSF}}(\emptyset) = \top^{\mathcal{CSF}}$. *Since* $Type_{Pvar}$ $(\sqsubseteq^{\mathcal{CSF}})$ *is a poset and* $\sqcup^{\mathcal{CSF}}$ *is defined for every subset of* $Type_{Pvar}$ *then, by theorem 3 of chapter 5 of [1], it follows that* $Type_{Pvar}$ $(\sqsubseteq^{\mathcal{CSF}})$ *is a complete lattice.*

Proof 7.4 (for lemma 3.4) *Suppose* $\Phi \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$. *For conciseness let* $\psi^{\mathcal{CSF}}$ *and* $\varphi^{\mathcal{CSF}}$ *respectively denote the right-hand-sides of definition 3.12 and lemma 3.4. Observe that* $\psi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \varphi^{\mathcal{CSF}}$. *Note that* $\phi^{\mathcal{C}} \sqsubseteq \varphi^{\mathcal{C}}$ *since* $\varphi^{\mathcal{C}} = \alpha^{\mathcal{C}}_{Pvar}(\Phi)$. *Moreover if* $\phi^{\mathcal{C}} = \varphi^{\mathcal{C}}$ *then* $\varphi^{\mathcal{S}} \subseteq \phi^{\mathcal{S}}$ *and* $\phi^{\mathcal{F}} \subseteq \varphi^{\mathcal{F}}$. *Thus* $\varphi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \psi^{\mathcal{CSF}}$.

Proof 7.5 (for lemma 3.3) *Immediate since* $\alpha^{\mathcal{CSF}}_{Pvar}$ *is total and monotonically increasing as is* $\gamma^{\mathcal{CSF}}_{Pvar}$; $\Phi \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\alpha^{\mathcal{CSF}}_{Pvar}(\Phi))$ *for all* $\Phi \in \wp(Sub)$; *and* $\alpha^{\mathcal{CSF}}_{Pvar}(\gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})) \sqsubseteq^{\mathcal{CSF}} \phi^{\mathcal{CSF}}$ *for all* $\phi^{\mathcal{CSF}} \in Type_{Pvar}$. *The last point is evident because if* $\Phi = \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ *then* $\alpha^{\mathcal{CSF}}_{Pvar}(\Phi) = \sqcap^{\mathcal{CSF}}(\{\phi^{\mathcal{CSF}} \mid \Phi \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})\})$ *and* $\Phi \subseteq \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$.

Proof 7.6 (for lemma 3.5) *Put* $\varphi^{\mathcal{CSF}} = \bigtriangledown^{\mathcal{CSF}}_k(\phi^{\mathcal{CSF}})$. *Note that* $\varphi^{\mathcal{C}} \sqsubseteq \phi^{\mathcal{C}}$ *so that* $\phi^{\mathcal{CSF}} \sqsubseteq^{\mathcal{CSF}} \varphi^{\mathcal{CSF}}$ *follows like cases 2(a)i and 2(a)ii of proof 7.2.*

Proof 7.7 (for lemma 4.1) *Let* $\phi \in \gamma^{\mathcal{S}}_{Pvar}(\phi^{\mathcal{S}})$, $\varphi \in mgu(\{\phi(u) = \phi(t)\})$, $\{u\} \cup var(t) \subseteq Pvar$ *and* $u \notin var(t)$. *Let* $v \in Uvar$. *To show* $occ_{Pvar}(v, \varphi \circ \phi) \in mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$.

1. Suppose $v \notin cod(\varphi \circ \phi)$. Thus $v \notin var(\varphi \circ \phi(w))$ for all $w \in dom(\varphi \circ \phi)$.

(a) Suppose $v \notin dom(\varphi \circ \phi)$, that is, $\varphi \circ \phi(v) = v$. Thus $\phi(v) = v'$ and $\varphi(v') = v$. Suppose $v \neq v'$. Hence $v \in var(\phi(u)) \cup var(\phi(t))$. Thus there exists $w \in \{u\} \cup var(t)$ such that $v \in var(\phi(w))$. But since $\phi(v) = v'$, $v \neq w$ and because $dom(\varphi) \cap cod(\varphi) = \emptyset$, $\varphi(v) = v$ and therefore $v \in var(\varphi \circ \phi(w))$. Hence $v \in cod(\varphi \circ \phi)$ which is a contradiction. Thus $v = v'$.

i. Suppose $v \notin var(\phi(u))$ and $v \notin var(\phi(t))$. Hence $v \notin cod(\varphi)$ and therefore $occ_{Pvar}(v, \varphi \circ \phi) = occ_{Pvar}(v, \phi)$. But $u \notin var(occ_{Pvar}(v, \phi))$ and $var(t) \cap var(occ_{Pvar}(v, \phi)) = \emptyset$ and therefore $u \notin var(occ_{Pvar}(v, \phi))$ and $var(t) \cap var(occ_{Pvar}(v, \phi)) = \emptyset$. Hence $occ_{Pvar}(v, \varphi \circ \phi) \in mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$.

ii. *Suppose* $v \in var(\phi(u))$ *and* $v \notin var(\phi(t))$. *Since* $\varphi \in mgv(\{\phi(u) = \phi(t)\})$, $v \in dom(\varphi)$ *or* $v \in cod(\varphi)$. *Since* $\varphi(v) = v$, $v \notin dom(\varphi)$ *and thus* $v \in cod(\varphi)$. *Thus* $v \in var(\varphi \circ \phi(u))$ *and therefore* $v \in var(\varphi \circ \phi(t))$. *Since* $v \notin var(\phi(t)$, *there exists* $w \in var(\phi(t))$ *such that* $v \in var(\varphi(w))$. *Thus* $v \in var(\varphi \circ \phi(t))$ *and since* $v \notin cod(\varphi \circ \phi)$, $v = t$ *so that* $\phi(t) = v$ *which is a contradiction.*

iii. *Suppose* $v \notin var(\phi(u))$ *and* $v \in var(\phi(t))$. *Like case 1(a)ii.*

iv. *Suppose* $v \in var(\phi(u))$ *and* $v \in var(\phi(t))$. *Since* $\varphi(v) = v$ *and* $v \notin cod(\varphi \circ \phi)$, $v \notin cod(\phi)$. *Thus* $\phi(u) = v$ *and therefore* $\varphi \in mgv(\{v = \phi(t)\})$ *with* $v \in var(\phi(t))$ *which is a contradiction.*

(b) *Suppose* $v \in dom(\varphi \circ \phi)$. *Since* $v \notin cod(\varphi \circ \phi)$, $occ_{Pvar}(v, \varphi \circ \phi) = \emptyset \in mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$.

2. *Suppose* $v \in cod(\varphi \circ \phi) \setminus var(\varphi \circ \phi(u))$. *Let* $w \in Pvar$. *Suppose* $v \in var(\varphi \circ \phi(w))$ *but* $v \notin var(\phi(w))$. *Thus* $v \in cod(\varphi)$ *and hence* $v \in var(\varphi \circ \phi(u))$ *which is a contradiction. Suppose* $v \in var(\phi(w))$ *but* $v \notin var(\varphi \circ \phi(w))$. *Thus* $v \in dom(\varphi)$ *and* $v \notin cod(\varphi)$ *so that* $v \notin cod(\varphi \circ \phi)$ *which is a contradiction. Hence* $occ_{Pvar}(v, \varphi \circ \phi) = occ_{Pvar}(v, \phi) \in \phi^{\mathcal{S}}$. *Suppose* $v \in var(\phi(u)) \cup var(\phi(t))$. *Since* $v \notin var(\varphi \circ \phi(u))$, $v \in dom(\varphi)$ *and therefore* $v \notin cod(\varphi)$. *Hence* $v \notin cod(\varphi \circ \phi)$ *which is a contradiction. Thus* $u \notin var(occ_{Pvar}(v, \phi))$ *and* $var(t) \cap var(occ_{Pvar}(v, \phi)) = \emptyset$ *and therefore* $occ_{Pvar}(v, \varphi \circ \phi) \in mgu^{\mathcal{SF}}(u, t, \phi^{\mathcal{SF}})$.

3. *Suppose* $v \in cod(\varphi \circ \phi) \cap var(\varphi \circ \phi(u))$. *Note that* $occ_{Pvar}(v, \varphi \circ \phi) = \cup_{v \in var(\varphi(w))} occ_{Pvar}(w, \phi)$.

   (a) *Suppose* $u \in \phi^{\mathcal{F}}$ *with* $\phi(t) = v_u$.

   i. *Suppose* $\varphi = \{v_u \mapsto \phi(t)\}$. *Since* $v \in var(\varphi \circ \phi(u))$, $v \in var(\phi(t))$. *Thus* $\{w \mid v \in var(\varphi(w)\} = \{v_u, v\}$. *Hence* $occ_{Pvar}(v, \varphi \circ \phi) \in rel(u, \phi^{\mathcal{S}}) \,\square\, rel(t, \phi^{\mathcal{S}}) \subseteq mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$.

   ii. *Suppose* $\varphi = \{v_t \mapsto v_u\}$ *with* $\phi(t) = v_t$. *Since* $v \in var(\varphi \circ \phi(u))$, $v = v_u$. *Thus* $\{w \mid v \in var(\varphi(w)\} = \{v_u, v_t\}$. *Hence* $occ_{Pvar}(v, \varphi \circ \phi) \in rel(u, \phi^{\mathcal{S}}) \,\square\, rel(t, \phi^{\mathcal{S}}) \subseteq mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$.

   (b) *Suppose* $t \in \phi^{\mathcal{F}}$. *Like case 3a.*

   (c) *Suppose* $u \notin \phi^{\mathcal{F}}$ *and* $t \notin \phi^{\mathcal{F}}$. *There exists* $W_u \subseteq var(\phi(u))$ *and* $W_t \subseteq var(\phi(t))$ *such that* $occ_{Pvar}(v, \varphi \circ \phi) = \cup_{w \in W_u \cup W_t} occ_{Pvar}(w, \phi)$. *Since* $v \in var(\varphi \circ \phi(u))$, $W_u \neq \emptyset$ *and thus* $W_t \neq \emptyset$. *Thus* $occ_{Pvar}(u, \varphi \circ \phi) \in rel(u, \phi^{\mathcal{S}})^{\star} \,\square\, rel(t, \phi^{\mathcal{S}})^{\star} \subseteq mgu^{\mathcal{S}}(u, t, \phi^{\mathcal{SF}})$.

Proof 7.8 (for lemma 4.2) *Let* $\phi \in \gamma^{\mathcal{F}}_{Pvar}(\phi^{\mathcal{F}})$, $\varphi \in mgu(\{\phi(u) = \phi(t)\})$, $\{u\} \cup var(t) \subseteq Pvar$ *and* $u \notin var(t)$. *To show* $fr_{Pvar}(\varphi \circ \phi) \supseteq mgu^{\mathcal{F}}(u, t, \phi^{\mathcal{SF}})$. *Let* $v \in mgu^{\mathcal{F}}(u, t, \phi^{\mathcal{SF}})$.

1. *Suppose* $u \in \phi^{\mathcal{F}}$ *and* $t \in \phi^{\mathcal{F}}$ *where* $\phi(u) = v_u$ *and* $\phi(t) = v_t$.

   (a) *If* $\varphi = \{v_u \mapsto v_t\}$ *then* $v \in fr_{Pvar}(\varphi \circ \phi)$ *since* $v \in fr_{Pvar}(\phi)$.

   (b) *If* $\varphi = \{v_t \mapsto v_u\}$ *then* $v \in fr_{Pvar}(\varphi \circ \phi)$ *since* $v \in fr_{Pvar}(\phi)$.

2. *Suppose* $u \in \phi^{\mathcal{F}}$.

   (a) *Suppose* $\varphi = \{v_u \mapsto \phi(t)\}$ *where* $\phi(u) = v_u$. *Since* $v \notin var(rel(u, \phi^{\mathcal{S}}))$, $v_u \notin var(\phi(v))$. *Hence* $\varphi \circ \phi(v) = \phi(v)$. *Thus, since* $v \in fr_{Pvar}(\phi)$, $v \in fr_{Pvar}(\varphi \circ \phi)$.

(b) *Suppose* $\varphi = \{v_t \mapsto v_u\}$ *where* $\phi(u) = v_u$ *and* $\phi(t) = v_t$. *Thus* $v \in fr_{Pvar}(\varphi \circ \phi)$ *since* $v \in fr_{Pvar}(\phi)$.

3. *Suppose* $t \in \phi^{\mathcal{F}}$. *Like case 2.*

4. *Suppose* $u \notin \phi^{\mathcal{F}}$ *and* $t \notin \phi^{\mathcal{F}}$. *Since* $v \notin var(rel(u, \phi^{\mathcal{S}}))$ *and* $v \notin var(rel(t, \phi^{\mathcal{S}}))$, $var(\phi(v)) \cap var(\phi(u)) = \emptyset$ *and* $var(\phi(v)) \cap var(\phi(t)) = \emptyset$ *and hence* $\varphi \circ \phi(v) = \phi(v) \in fr_{Pvar}(\phi)$.

Proof 7.9 (for theorem 4.1) *Let* $\phi \in \gamma^{\mathcal{SF}}_{Pvar}(\phi^{\mathcal{SF}})$, $\varphi \in mgu(\phi(E))$ *and* $mgu^{\mathcal{SF}}(E, \phi^{\mathcal{SF}}, \mu^{\mathcal{SF}})$ *with* $var(E) \subseteq Pvar$. *By induction on the steps of* $mgu^{\mathcal{SF}}$ *and by lemmas 4.1 and 4.2, there exists* $\theta \in mgu(\phi(E))$ *such that* $\theta \circ \phi \in \gamma^{\mathcal{SF}}_{Pvar}(\psi^{\mathcal{SF}})$. *But* $\theta \approx \varphi$ *[17] and thus* $\theta \circ \phi \approx \varphi \circ \phi$. *Hence* $\varphi \circ \phi \in \gamma^{\mathcal{SF}}_{Pvar}(\psi^{\mathcal{SF}})$.

Proof 7.10 (for lemma 4.3) *Proof by induction. Suppose* $\phi \restriction Pvar = \phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar$; $\varphi^{\mathcal{B}} \in mgu(\phi^{\mathcal{C}}(E))$; $\psi^{\mathcal{B}} = sub_{Pvar}(\varphi^{\mathcal{B}})$; $\psi^{\mathcal{B}}_n = \{u^s_i \mapsto t_i\}^n_{i=1}$ *so that* $\psi^{\mathcal{B}}_N = \psi^{\mathcal{B}}$; $\vartheta_n \in mgu(\psi^{\mathcal{B}}_n \cup \phi^{\mathcal{D}})$ *so that* $\vartheta_N = \vartheta$; *and that by the inductive hypothesis* $\vartheta_n \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta^{\mathcal{SF}}_n)$. *Put* $\vartheta^{\mathcal{S}}_{n+1} = mgu^{\mathcal{BS}}(u^s_{n+1}, t_{n+1}, \vartheta^{\mathcal{SF}}_n)$ *and* $\vartheta^{\mathcal{F}}_{n+1} = mgu^{\mathcal{BF}}(u^s_{n+1}, t_{n+1}, \vartheta^{\mathcal{SF}}_n)$. *To show* $sh_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta) \subseteq \vartheta^{\mathcal{S}}_{n+1}$ *and* $fr_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta) \supseteq \vartheta^{\mathcal{F}}_{n+1}$.

Note that $cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar) \setminus cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar) = \{u^s_{n+1}\}$ *and that* $cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar) \setminus cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar) = var(t_{n+1})$. *Let* $w \in Uvar$.

1. *Suppose* $w = u^s_{n+1}$. *Now* $\vartheta_{n+1} = mgu(\{u^s_{n+1} \mapsto t_{n+1}\}(\vartheta_n)) \circ \{u^s_{n+1} \mapsto t_{n+1}\}$ *and thus* $w \in dom(\vartheta_{n+1})$ *so that* $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_{n+1}) = \emptyset \in \vartheta^{\mathcal{S}}_{n+1}$. *If* $w \in \phi^{\mathcal{F}}_n$ *then* $w \in var(rel(u^s_{n+1}, \vartheta^{\mathcal{S}}_n))$ *and thus* $w \notin \vartheta^{\mathcal{F}}_{n+1}$.

2. *Suppose* $w \in var(t_{n+1})$.

   (a) *Suppose* $u^s_{n+1} \in \vartheta^{\mathcal{F}}_n$. *Thus* $\vartheta_n(u^s_{n+1}) = v \in Uvar$. *Hence* $\vartheta_{n+1} = mgu(\{u^s_{n+1} \mapsto t_{n+1}\} \cup \vartheta_n) = \zeta \circ \vartheta_n$ *where* $\zeta = \{v \mapsto t_{n+1}\}$. *Now* $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(w, \zeta \circ \vartheta_n) = \{x \in cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar) \,|\, w \in var(\zeta(y)) \wedge y \in var(\vartheta_n(x))\} = \cup_{w \in var(\zeta(y))} occ_{\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar}(y, \vartheta_n)$. *Because* $\zeta = \{v \mapsto t_{n+1}\}$ *and* $w \in var(t_{n+1})$, $\{y \,|\, w \in var(\zeta(y))\} = \{v, w\}$. *Hence* $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(w, \zeta \circ \vartheta_n) = occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(v, \vartheta_n) \cup occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(w, \vartheta_n)$. *Now* $w \notin cod(\vartheta_n)$ *and thus* $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(w, \vartheta_n) = \{w\}$ *since* $w \in cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)$. *Also* $v \in var(\vartheta_n(t_{n+1})) = var(t_{n+1})$ *and therefore* $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(w, \vartheta_n) = occ_{cod(\phi^{\mathcal{C}})}(w, \vartheta_n) \setminus \{u^s_{n+1}\}$. *Hence* $occ_{cod(\phi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(w, \vartheta_{n+1}) \subseteq \mathcal{U}_u \,\square\, U_t \subseteq \vartheta^{\mathcal{S}}_{n+1}$. *Since* $w \in \vartheta^{\mathcal{F}}_{n+1}$ *to show* $w \in fr_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(\vartheta_{n+1})$. *First, note that* $w \in var(\phi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)$. *Second, observe that* $w \in cod(\zeta \circ \vartheta_n)$ *and thus* $w \in fr_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Uvar)}(\vartheta_{n+1})$.

   (b) *Suppose* $u^s_{n+1} \notin \vartheta^{\mathcal{F}}_n$. *Note that* $\vartheta_{n+1} = mgu(\{\vartheta_n(u^s_{n+1}) = t_{n+1}\}) \circ \vartheta_n$ *since* $dom(\vartheta_n) \cap t_{n+1} = \emptyset$. *Let* $\zeta = mgu(\{\vartheta_n(u^s_{n+1}) = t_{n+1}\})$. *Now* $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \zeta \circ \vartheta_n) = \cup_{w \in var(\zeta(y))} occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(y, \vartheta_n)$. *Since* $\{y \,|\, w \in var(\zeta(y))\} \subseteq var(\vartheta_n(u^s_{n+1})) \cup var(t_{n+1})$ *there exists* $Y_u \subseteq var(\vartheta_n(u^s_{n+1}))$ *and* $Y_t \subseteq var(t_{n+1})$ *such that* $\{y \,|\, w \in var(\zeta(y))\} = Y_u \cup Y_t$. *But because* $\chi(t_{n+1}) = 1$, *by lemma 2.1,* $Y_u \subseteq \{y_u\}$.

i. If $Y_u = \emptyset$ then $w \notin var(\zeta(\vartheta_n(u^s_{n+1})))$ and thus $w \notin var(\zeta(t_{n+1}))$ so that $Y_t = \emptyset$. Thus $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_{n+1}) = \emptyset \in \vartheta^{\mathcal{S}}$.

ii. If $Y_u = \{y_u\}$ then $Y_t \neq \emptyset$. Because $u^s_{n+1} \in dom(\psi^{\mathcal{B}})$, $u^s_{n+1} \in cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)$ and thus $occ_{cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)}(y_u, \vartheta_n) \in rel(u^s_{n+1}, \vartheta^{\mathcal{S}}_n)$. Now $y_u \in var(\vartheta_n(u^s_{n+1})) \subseteq \{u^s_{n+1}\} \cup cod(\vartheta_n)$. But $(\{u^s_{n+1}\} \cup cod(\vartheta_n)) \cap var(t_{n+1}) = \emptyset$ and thus $var(t_{n+1}) = var(\vartheta_n(t_{n+1}))$. Thus $y_u \notin var(\vartheta_n(t_{n+1}))$. Hence $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(y_u, \vartheta_n) \in \mathcal{U}_u$. If $y_t \in Y_t$, $y_t \notin cod(\vartheta_n)$ and thus $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(y_t, \vartheta_n) = \{y_t\}$. Hence $\cup_{w \in var(\zeta(y))}\, occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(y, \vartheta_n) = Y_t \in \mathcal{U}_t$ since $Y_t \neq \emptyset$. Thus $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_{n+1}) \in \mathcal{U}_u \,\square\, \mathcal{U}_t$.

Note that $w \notin \vartheta^{\mathcal{F}}_n$.

3. Suppose $w \notin \{u^s_{n+1}\} \cup var(t_{n+1})$. Let $\zeta = mgu(\{\vartheta_n(u^s_{n+1}) = \vartheta_n(t_{n+1})\})$ so that $\vartheta_{n+1} = \zeta \circ \vartheta_n$. Hence $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \zeta \circ \vartheta_n) = \cup_{w \in var(\zeta(y))}\, occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(y, \vartheta_n)$. Since $w \notin \{u^s_{n+1}\} \cup var(t_{n+1})$ and $w \notin cod(\vartheta_n)$, $\zeta(w) = w$ and hence $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \zeta \circ \vartheta_n) = occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_n)$. But since $w \notin var(\vartheta_n(u^s_{n+1}))$ and $w \notin var(\vartheta_n(t_{n+1}))$, $occ_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_n) = occ_{cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_n)$. Because $occ_{cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_n) \notin rel(u^s_{n+1}, \vartheta^{\mathcal{S}}_n)$ and $occ_{cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_n) \notin rel(t_{n+1}, \vartheta^{\mathcal{S}}_n)$, $occ_{cod(\phi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(w, \vartheta_{n+1}) \in \vartheta^{\mathcal{S}}_{n+1}$. If $w \in \vartheta^{\mathcal{F}}_n$ then $w \in cod(\psi^{\mathcal{B}}_n \circ \phi^{\mathcal{C}} \restriction Pvar)$ and $\vartheta_n(w) = v \in Uvar$. Since $w \neq u^s_{n+1}$, $w \in cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)$. Also, because $\zeta(w) = w$, $\zeta \circ \vartheta_n(w) = v$ and hence $w \in \vartheta^{\mathcal{F}}_{n+1}$ only if $w \in cod(\phi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)$ and $\vartheta_{n+1}(w) \in Uvar$.

Thus $sh_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta_{n+1}) \subseteq \vartheta^{\mathcal{S}}_{n+1}$ and $fr_{cod(\psi^{\mathcal{B}}_{n+1} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta_{n+1}) \supseteq \vartheta^{\mathcal{F}}_{n+1}$.

Proof 7.11 (for theorem 4.2) *Let* $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$, $\varphi \in mgu(\phi(E))$ *and* $mgu^{\mathcal{CSF}}(E, \phi^{\mathcal{CSF}}, \mu^{\mathcal{CSF}})$ *with* $var(E) \subseteq Pvar$. *Thus* $\mu^{\mathcal{CSF}} = \langle \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar,\ \mu^{\mathcal{S}},\ \mu^{\mathcal{F}} \rangle$ *where* $\varphi^{\mathcal{B}} \in mgu(\phi^{\mathcal{C}}(E))$ *and* $\psi^{\mathcal{B}} = sub_{Pvar}(\varphi^{\mathcal{B}})$. *To show that there exists* $\psi^{\mathcal{D}}$ *such that* $[\psi^{\mathcal{D}} \circ (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)] \restriction Pvar = \varphi \circ \phi$ *with* $\psi^{\mathcal{D}} \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\mu^{\mathcal{SF}})$.

Since $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ *there exists* $\phi^{\mathcal{D}}$ *such that* $\phi = \phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar$. *Now* $\varphi \circ \phi = mgu(E \cup \phi) = mgu(E \cup (\phi^{\mathcal{D}} \circ \phi^{\mathcal{C}} \restriction Pvar)) = mgu(E \cup mgu(\phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar) = mgu(E \cup \phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar = mgu(\phi^{\mathcal{C}}(E) \cup \phi^{\mathcal{C}}(\phi^{\mathcal{D}}) \cup \phi^{\mathcal{C}}) \restriction Pvar = mgu(\phi^{\mathcal{C}}(E) \cup \phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar$ *because* $dom(\phi^{\mathcal{C}}) \cap dom(\phi^{\mathcal{D}}) = \emptyset$ *and* $dom(\phi^{\mathcal{C}}) \cap cod(\phi^{\mathcal{D}}) = \emptyset$. *But* $mgu(\phi^{\mathcal{C}}(E) \cup \phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar = mgu(\phi^{\mathcal{C}}(E) \cup \phi^{\mathcal{C}}(E) \cup \phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar = mgu(\phi^{\mathcal{C}}(E) \cup \psi^{\mathcal{B}} \cup \phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar$ *since* $\psi^{\mathcal{B}} \sqsubseteq \zeta$ *for all* $\zeta \in mgu(\phi^{\mathcal{C}}(E))$. *But* $mgu(\phi^{\mathcal{C}}(E) \cup \psi^{\mathcal{B}} \cup \phi^{\mathcal{D}} \cup \phi^{\mathcal{C}}) \restriction Pvar = mgu(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \psi^{\mathcal{B}} \cup \psi^{\mathcal{B}}(\phi^{\mathcal{D}}) \cup (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar) \restriction Pvar$ *because* $dom(\psi^{\mathcal{B}}) \cap dom(\phi^{\mathcal{C}}) = \emptyset$. *But* $mgu(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \psi^{\mathcal{B}} \cup \psi^{\mathcal{B}}(\phi^{\mathcal{D}}) \cup (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar) \restriction Pvar = mgu(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta \cup (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar) \restriction Pvar$ *since* $\vartheta = mgu(\phi^{\mathcal{D}} \cup \psi^{\mathcal{B}})$. *But* $mgu(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta \cup (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar) \restriction Pvar = mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta \cup \vartheta[(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar]) \restriction Pvar = mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta \cup (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar) \restriction Pvar = mgu(mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta) \cup (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar) \restriction Pvar = [mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta) \circ (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar] \restriction Pvar$ *because* $dom(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar) \cap var(mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta)) = \emptyset$.

Thus put $\psi^{\mathcal{D}} = mgu(\vartheta \circ \psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E) \cup \vartheta)$. *Hence* $\varphi \circ \phi = [\psi^{\mathcal{D}} \circ (\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}) \restriction Pvar] \restriction Pvar$. *But by lemma 4.3,* $\vartheta \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\vartheta^{\mathcal{SF}})$ *for* $\vartheta^{\mathcal{SF}} = mgu^{\mathcal{SF}}(\psi^{\mathcal{B}}, \phi^{\mathcal{SF}})$ *and thus by theorem 4.1,* $\psi^{\mathcal{D}} \in \gamma^{\mathcal{SF}}_{cod(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}} \restriction Pvar)}(\mu^{\mathcal{SF}})$ *where* $mgu^{\mathcal{SF}}(\psi^{\mathcal{B}} \circ \phi^{\mathcal{C}}(E), \vartheta^{\mathcal{SF}}, \mu^{\mathcal{SF}})$.

Proof 7.12 (for theorem 4.3) *Let* $\Phi \subseteq \gamma^{\mathcal{CSF}}(\phi^{\mathcal{CSF}})$, $\Psi \subseteq \gamma^{\mathcal{CSF}}(\psi^{\mathcal{CSF}})$ *and* $\theta \in unify^c(a, \Phi, b, \Psi)$. *Thus* $\theta = (\varphi \circ \phi) \upharpoonright Pvar$ *where* $\varphi \in mgu(\{\phi(a) = \Upsilon(\psi(b))\})$, $\phi \in \Phi$ *and* $\psi \in \Psi$. *Observe that* $\varphi \in mgu(\{\phi(a) = \Upsilon(\psi(\Upsilon^{-1}(\Upsilon(b))))\})$ *and thus putting* $\sigma = \phi \cup (\Upsilon \circ \psi \circ \Upsilon^{-1})$, $\varphi \in mgu(\sigma(\{a = \Upsilon(b)\}))$. *Note that* $\phi \in \gamma^{\mathcal{CSF}}_{Pvar}(\phi^{\mathcal{CSF}})$ *and* $\psi \circ \Upsilon^{-1} \in \gamma^{\mathcal{CSF}}_{\Upsilon(Pvar)}(\Upsilon(\psi^{\mathcal{CSF}}))$ *and hence* $\Upsilon \circ \psi \circ \Upsilon^{-1} \in \gamma^{\mathcal{CSF}}_{\Upsilon(Pvar)}(\Upsilon(\psi^{\mathcal{CSF}}))$. *Since* $var(\phi) \cap var(\Upsilon \circ \psi \circ \Upsilon^{-1}) = \emptyset$, $\sigma \in \gamma_{Pvar \cup \Upsilon(Pvar)}(\phi^{\mathcal{CSF}} \cup \Upsilon(\psi^{\mathcal{CSF}}))$ *Thus, by theorem 4.2, since* $var(a) \cup var(\Upsilon(b)) \subseteq Pvar \cup \Upsilon(Pvar)$, $\varphi \circ \sigma \in \gamma^{\mathcal{CSF}}(mgu^{\mathcal{CSF}}(\{a = \Upsilon(b)\}, \phi^{\mathcal{CSF}} \cup \Upsilon(\psi^{\mathcal{CSF}})))$. *But* $(\varphi \circ \sigma) \upharpoonright Pvar = (\varphi \circ [\sigma \upharpoonright Pvar]) \upharpoonright Pvar = (\varphi \circ \phi) \upharpoonright Pvar$ *and* $(\varphi \circ \sigma) \upharpoonright Pvar \in \gamma^{\mathcal{CSF}}_{Pvar \cup \Upsilon(Pvar)}(mgu^{\mathcal{CSF}}(\{a = \Upsilon(b)\}, \phi^{\mathcal{CSF}} \cup \Upsilon(\psi^{\mathcal{CSF}})) \upharpoonright^{\mathcal{CSF}} Pvar$ *and therefore* $\theta \in \gamma^{\mathcal{CSF}}_{Pvar}(unify^{\mathcal{CSF}}(a,\ \phi^{\mathcal{CSF}},\ b,\ \psi^{\mathcal{CSF}}))$.