The Symbolic Term Abstract Domain

Patrick Cousot  
Coutant Institute of Mathematical Studies, Computer Science Department  
New York University  
New York, NY, USA  
p cousot@cims.nyu.edu

Abstract—We construct the abstract domain of symbolic terms ordered by subsumption by abstraction of the sets of ground terms ordered by inclusion.

Index Terms—Abstract interpretation, Abstract domain, Symbolic term, Subsumption.

I. INTRODUCTION

In mathematical logic, Jacques Herbrand introduced ground terms [1, Ch. 1] to denote a basic mathematical object (for example, 0) or operation on objects (such as +(1, 2)) as well as symbolic terms that is terms with variables (where the variables x are unknowns standing for any ground term [1, Ch. 2] (for example, +(1, x)).

Gordon Plotkin [3], [4] and John Reynolds [5] proved that the set of symbolic terms form a complete lattice with the less general/subsumption partial order \( \preceq \) on terms. For example, +(1, 2) \( \preceq \) +(1, y) \( \preceq \) +(x, y) \( \preceq \) z.

Symbolic terms are of interest in various areas of Computer Science such as refutation theorem-proving based on the resolution rule of inference [6], [7], satisfiability modulo theories [8], symbolic execution [9], type inference [10], [11], logic and constraint programming [12]–[16], [17]–[19], pointer analysis in imperative [20] or logic languages [21], and so on.

In [22], we showed that Hindley’s monotypes with variables [23] as well as Milner’s polymorphic types [10] are abstractions of sets of Church’s monotypes [24]. Thanks to restrictions on types (for example, no union type) and on the language (for example, the two branches of a conditional must have the same type), the set of monotypes of a lambda-expression is exactly represented by a monotype with variables. In that case the abstraction is exact. This is no longer the case for polymorphic types, for which a widening is needed (all recursive calls must have the same type).

Generalizing this initial point view, our objective is to study the complete lattice of symbolic terms by abstraction of the powerset of ground terms.

This material is based upon work supported by DARPA under Agreement No. HR00112020022. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the United States Government or DARPA.

1 English translation in [2].

II. THE COMPLETE LATTICE OF GROUND TERMS

The signature \( F \) defines a set of function symbols \( f \}\) (for brevity), each one with an arity \( n \), that is, a fixed number of parameters (0 for constants). The round parentheses \( (",", "\) and comma \( ",\) do not belong to \( F \).

\[
\begin{align*}
  f \in F & \quad \text{constants of arity 0} \\
  f(n(t_1, \ldots, t_n)) & \quad \text{term of arity } n \in \mathbb{N}^+
\end{align*}
\]

We assume that the signature \( F \) has at least two different function symbols. Ground terms denote uninterpreted functional expressions.

\[
\begin{align*}
  t & \in T ::= \text{ground terms} \\
  f \in F & \quad \text{constants of arity 0} \\
  f(n(t_1, \ldots, t_n)) & \quad \text{term of arity } n \in \mathbb{N}^+
\end{align*}
\]

The set \( T \) of all ground terms is called the Herbrand universe with signature \( F \).

Sets of ground terms form a complete lattice partially ordered by inclusion

\[
(\wp(T), \subseteq, \emptyset, T, \cup, \cap) \quad \text{sets of ground terms} \quad (1)
\]

III. TERMS WITH VARIABLES

A term with variables (also called symbolic term) abstracts a set of terms. For example the set of ground terms \( \{+(0,1),+(0,+,(1,1)),+(0,+,(1,1)),+(0,+,(1,1)),\ldots\} \) can be abstracted by the term \( +(0,\alpha) \) with variable \( \alpha \). The abstraction can be very imprecise. For example \( \{0,+,(0,1)\} \) would be abstracted by variable \( \alpha \) which concretization is the set of all ground terms. So the abstraction is precise enough only for set of terms with adequate regularity properties.

\[
\begin{align*}
  \alpha, \beta, \gamma & \in \mathbb{V}_k \quad \text{term variables} \\
  \tau & \in T^\nu ::= \text{terms with variables} \quad (2) \\
  f \in F & \quad \text{constants} \\
  f(n(\tau_1, \ldots, \tau_n)) & \quad \text{term of arity } n \in \mathbb{N} \\
  \alpha & \quad \text{term variable}
\end{align*}
\]

The round parentheses \( ",", ",\) and comma \( ",\) and variables \( \alpha \in \mathbb{V}_k \) do not belong to \( F \).

We write \( \text{vars}[\tau] \) for the free variables of a term \( \tau \).

\[
\begin{align*}
  \text{vars}[\alpha] & \triangleq \{\alpha\} & \alpha & \in \mathbb{V}_k \quad (3) \\
  \text{vars}[f(\tau_1, \ldots, \tau_n)] & \triangleq \bigcup_{i=1}^n \text{vars}[\tau_i]
\end{align*}
\]
Proof of lemma 1. By structural induction on \( \tau \).

- If \( \tau = \alpha \) then, by (4), \( \rho(\alpha \leftarrow \tau') = \rho(\alpha \leftarrow \rho'(\tau'))(\alpha) = \rho'(\tau) \) by (6).
- If \( \tau = \beta \neq \alpha \) then, by (4), \( \rho(\tau)(\alpha \leftarrow \beta) = \rho(\alpha \leftarrow \rho'(\tau))(\alpha) = \rho'(\tau) \) by (6).
- Otherwise, \( \tau = f(\tau_1, \ldots, \tau_n) \) so that, by (4), (6), induction hypothesis, and (6) again, \( \rho(\tau)(\alpha \leftarrow \tau) = f(\rho(\tau_1)(\alpha \leftarrow \tau), \ldots, \rho(\tau_n)(\alpha \leftarrow \tau)) = f(\rho(\tau_1)(\alpha \leftarrow \tau'), \ldots, \rho(\tau_n)(\alpha \leftarrow \alpha)) = \rho(\tau')(\tau_1, \ldots, \tau_n) \).

Let us call \( \rho(\tau) \) the ground instance of \( \tau \) for the assignment \( \rho \). Unless it is reduced to a variable, a term with variables cannot have the same instance as any one of its variables (this is known as occur-check).

**Lemma 2.** For all variables \( \alpha \in \text{vars}(\tau) \) of a term with variables \( \tau \in \mathbb{P} \setminus \emptyset \), there is no assignment \( \rho \in \mathbb{P} \) with \( \rho(\alpha) \neq \rho(\tau) \).

**Proof of lemma 2.** Since \( \tau \in \mathbb{P} \setminus \emptyset \), we have \( \tau = f^1(\tau_1, \ldots, \tau_n) \). Let \( \rho(\tau) = f^1(\rho(\tau_1), \ldots, \rho(\tau_n)) \). Therefore, \( \rho(\tau) \) is a ground instance of \( \tau \). Since \( \rho \in \mathbb{P} \), all instances of \( \tau \) have the same concretization. Therefore, the names attributed to the same instances of variables in terms with variables do not matter. For example, \( \text{ground}(f(\alpha, \alpha)) = \text{ground}(f(\beta, \beta)) = \{ f(t, t) \mid t \in \mathbb{T} \} \).

V. THE SYMBOLIC ABSTRACTION

The symbolic abstraction abstracts a set of ground terms into a term with variables. The symbolic abstraction is easily defined by its concretization, that is, it’s set of ground instances.

\[
\text{ground}(\rho) \triangleq \{ \rho(\tau) \mid \rho \in \mathbb{P} \}
\]

Since all terms with variables \( \tau \in \mathbb{P} \) have a nonempty concretization \( \text{ground}(\tau) \), we add the empty term \( \emptyset \notin \mathbb{P} \) to denote the empty set \( \emptyset \) with \( \rho(\emptyset) = \emptyset \).

**Remark 1** (the symbolic abstraction is relational). An important remark on the definition of ground in (8) is that all instances \( \rho(\alpha) \) of a variable \( \alpha \) in a term with variables \( \tau \) are the same in a given instance \( \rho(\tau) \) of the term. For example, \( f(a, b) \notin \text{ground}(f(\alpha, \alpha)) \) when \( a \neq b \). However, two terms with variables equal up to variable renaming have the same concretization. Therefore, the names attributed to the same instances of variables in terms with variables do not matter. For example, \( \text{ground}(f(\alpha, \alpha)) = \text{ground}(f(\beta, \beta)) = \{ f(t, t) \mid t \in \mathbb{T} \} \).

VI. THE HERBRAND SYMBOLIC ABSTRACT DOMAIN

The abstract symbolic domain is a lattice of ground term properties (1).

A. The subsumption partial order

We define the preorder \( \preceq^v \) on terms with variables, called subsumption, as the inclusion of sets of their ground instances. (This will be shown to be equivalent to the classical definition in theorem 2.)

\[
(\tau \preceq^v \tau') \triangleq (\text{ground}(\tau) \subseteq \text{ground}(\tau'))
\]

This is a preorder \( (\mathbb{T}^v \cup \{ \emptyset \}, \preceq^v) \) with inﬁmum \( \emptyset \). For example \( f(a, b) \preceq^v f(a, b) \preceq^v f(\alpha, \beta) \preceq^v \gamma \).

**Lemma 3.** Observe that for all terms with variables \( \tau, \tau' \in \mathbb{T}^v \), we have \( \tau \preceq^v \tau' \) if and only if \( \forall \rho \in \mathbb{P} \) \( \exists \rho' \in \mathbb{P} \) \( \rho(\tau) = \rho'(\tau') \).

**Proof of lemma 3.** The case of \( \emptyset \) is trivial. Otherwise,
t \subseteq^\nu \tau' \\
\iff \text{ground}(t) \subseteq \text{ground}(\tau') \quad \text{(def. (9) of } \subseteq^\nu) \\
\iff \{\phi(t) | \phi \in \mathbb{P}^\nu\} \subseteq \{\phi'(\tau') | \phi' \in \mathbb{P}^\nu\} \\
\text{(def. (8) of } \subseteq) \\
\iff \forall \phi \in \mathbb{P}^\nu. \exists \phi' \in \mathbb{P}^\nu. \phi(t) = \phi'(\tau') \quad \text{(def. } \subseteq) \square

The corresponding equivalence relation is \( \sim^\nu \). The quotient is a partial order \( \langle \mathbb{P}^H, \subseteq^\nu \rangle \) where

\[
\tau \sim^\nu \tau' \iff \tau \subseteq^\nu \tau \land \tau' \subseteq^\nu \tau
\]

\[
\mathbb{P}^H \triangleq (\mathbb{P}^\nu \cup \{\emptyset\})^\nu
\]

\[
[\tau]_{\sim^\nu} \triangleq \{[\tau']_{\sim^\nu} | \tau \in \mathbb{P}^\nu \cup \{\emptyset\}\}
\]

For example, \( f(\alpha, \alpha) \sim^\nu f(\beta, \beta) \) and \( f(\alpha, \alpha)_{\sim^\nu} = \{f(\gamma, \gamma) | \gamma \in \mathcal{V}\} \). More generally, equivalent terms are equal up to variable renaming.

**Lemma 4.** A renaming is an assignment \( \rho \in \mathcal{V}_2 \rightarrow \mathcal{V}_2 \) between variables extended to terms with variables by (6), that is, \( \rho(f(t_1, \ldots, t_n)) = f(\rho(t_1), \ldots, \rho(t_n)) \). Equivalent terms have a bijective renaming of their variables and reciprocally, that is, \( \forall t, t' \in \mathbb{P}^\nu. (t \sim^\nu t') \iff (\exists \rho \in \text{vars}[t] \rightarrow \text{vars}[t']). \rho(t) = t' \).

**Proof of lemma 4.** (\( \Rightarrow \)) Assume \( t \sim^\nu t' \) are equivalent so that, by def. \( \sim^\nu \) and lemma 3, \( \forall \phi \in \mathbb{P}^\nu. \exists \phi' \in \mathbb{P}^\nu. \phi(t) = \phi'(t') \) and \( \forall \phi', \exists \rho \in \text{vars}[t]. \rho(t) = t' \). Let us define a relation \( \rho \in \mathcal{V}(\text{vars}[t] \times \text{vars}[t']) \), starting from \( \emptyset \) as follows.

1. If \( t \) is a variable \( \alpha \) and \( t' \) is not then \( t \sim^\nu t' \) so \( t' \) must be a variable \( \beta \) and we let \( \{\alpha, \beta\} \in \rho \).
2. If \( t = f(t_1, \ldots, t_n) \), then \( t \sim^\nu t' \) implies that \( \forall n \in [1, n]. t_k \sim^\nu t'_k \), so by structural induction, there is a relation \( \rho_k \in \mathcal{V}(\text{vars}[t_k] \times \text{vars}[t'_k]) \) so we take \( \rho = \bigsqcup_{k=1}^n \rho_k \).

We have to prove that \( \rho \) is a function. By contradiction, if \( \rho \) is not a function then there is a variable \( \alpha \) of \( t \) and two variables \( \beta \) and \( \gamma \) of \( t' \) with at least one which is not \( \alpha \), say \( \gamma \). Then instances of \( \gamma \) in \( t' \) cannot be replicated with \( \alpha \) in \( t \) so the two terms cannot be equivalent. Now if \( \rho \) is not injective there are two variables \( \beta \) and \( \gamma \) of \( t \) with only one correspondent \( \alpha \) of \( t \), so again the instances of \( \beta \) and \( \gamma \) cannot be matched with \( \alpha \). Finally, if \( \rho \) is not surjective, then there is a variable \( \gamma \) with arbitrary instantiations in \( t' \) with no correspondent in \( t \), which again prevents equivalence. In conclusion, \( \rho \) is a bijection.

(\( \Leftarrow \)) Conversely let \( \rho \in \mathcal{V}(\text{vars}[t] \rightarrow \text{vars}[t']) \) be such \( \rho(t) = t' \). Given any \( \phi \in \mathbb{P}^\nu \), define \( \phi'(\alpha) \triangleq \phi(\rho^{-1}(\alpha)) \), \( \alpha \in \mathcal{V}_2 \). Let us show, by structural induction on \( \alpha \), that \( \phi(t) = \phi'(t') \).

1. If \( t = \alpha \) then \( \rho \in \mathcal{V}(\text{vars}[t] \rightarrow \text{vars}[t']) \) and \( \rho(t) = t' \) imply that \( t' = \beta \) is the variable \( \beta = \rho(\alpha) \). It follows that \( \phi'(t') = \phi'(\beta) \triangleq \phi(\rho^{-1}(\beta)) = \phi(\alpha) = \phi(t) \); 
2. Otherwise, \( t = f(t_1, \ldots, t_n) \) so that \( \rho(t) = \rho(f(t_1, \ldots, t_n)) = f(\rho(t_1), \ldots, \rho(t_n)) \) and \( t' \) implies that \( \phi(t) = \phi(t_1, \ldots, t_n) \) with \( \rho(t_i) = \rho(t_i), i \in [1, n] \). It follows, by (6) and ind. hyp., that \( \phi(t') = \phi'(f(t_1', \ldots, t_n')) = f(\phi(t'_1), \ldots, \phi(t'_n)) = f(\phi(t_1), \ldots, \phi(t_n)) = \phi(t). \square \)

The comparison of equivalence classes is equivalent to the comparison of the representatives of these classes.

**Lemma 5.** \( [t_1]_{\sim^\nu} \leq^\nu [t_2]_{\sim^\nu} \iff t_1 \leq^\nu t_2. \)

**Proof of lemma 5.**

\[
[t_1]_{\sim^\nu} \leq^\nu [t_2]_{\sim^\nu} \\
\iff \exists \rho_1 \in [t_1]_{\sim^\nu}, \rho_2 \in [t_2]_{\sim^\nu}. \rho_1 \leq^\nu \rho_2 \quad \text{(def. (10) of } \leq^\nu) \\
\iff \exists \rho_1', \rho_2'. \rho_1 \equiv^\nu \rho_1 \land \rho_2 \equiv^\nu \rho_2 \iff \rho_1 \equiv^\nu \rho_2 \quad \text{(def. } \sim^\nu) \\
\iff \rho_1 \equiv^\nu \rho_2
\]

\( \{\Rightarrow\} \quad \text{(transitivity) \quad \text{(\( \Leftarrow \)) choosing } \rho_1 = \rho_1', \rho_2 = \rho_2, \text{ and reflexivity}\} \)

**B. The symbolic abstraction function**

The abstraction of \( \{f(\alpha, a), f(b, b), f(c, c)\} \) is \( f(a, \alpha) \) since the parameters of \( f \) are equal while \( \{f(a, b), f(b, a), f(a, a)\} \) is \( f(\beta, \gamma) \) since the parameters of \( f \) are not always related. The abstraction function must select variables so as to identify equal parameters on all instances of \( f \). For this purpose, we encode sets as families, for example, sequences \( \{f(a, a), f(b, b), f(c, c)\} \). In the first case, the subterms all yield \( (a, b, c) \) which is abstracted by a variable \( \alpha \). In the second case we get \( (a, b, a) \) encoded by \( \beta \) and \( (b, a, a) \) which is different so is encoded by a different variable \( \gamma \). Notice that the variable name does not matter and that the order in the sequences does not matter either (so sets of ground terms encoded differently as index families will have the same abstraction, up to variable renaming via a bijection between variables; see lemma 6).

We arbitrarily define a scheme to name sets of ground terms by a unique variable thanks to an injective function

\[
\nu \in (\Delta \rightarrow \mathcal{T}) \rightarrow \mathcal{V}_2 \quad \text{(naming scheme)}
\]

assigning a variable \( \nu(\{t_i \mid i \in \Delta\}) \) to any arbitrary family of ground terms \( \{t_i \mid i \in \Delta\} \). Injectivity ensures uniqueness, that is, different families of terms are abstracted by different variables.

The abstraction is called the least common generalization (lcg).

\[
\text{lcg}[\emptyset] \triangleq \emptyset^\nu
\]

\[
\text{lcg}[\nu](\{t_i \mid i \in \Delta\}) \triangleq \text{lcf}[\nu](\{t_i \mid i \in \Delta\})
\]

if \( \forall i, j \in \Delta. f_i = f_j = f \land n_i = n_j = n \) then let \( T_k = \text{lcg}[\nu](\{t_i \mid i \in \Delta\}), k = 1, \ldots, n \) in

\[
f(T_1, \ldots, T_n)
\]

else \( \nu(\{f_i(t_1, \ldots, t_n) \mid i \in \Delta\}) \)

If all the terms in the family have the same structure then the abstraction proceeds recursively else the family is
Lemma 6. The definition (12) of the symbolic abstraction $lcg_v$ is independent of the naming scheme $v$. If $w \vDash v' \in (\Delta \rightarrow T) \Rightarrow v'_0$ then $\forall T \in \phi(T) \cdot lcg_v(T) \sim v' \cdot lcg_v(T')$. Proof of lemma 6. Since $v \in (\Delta \rightarrow T) \Rightarrow v'_0$ is injective, it has a left inverse (improperly denoted $v^{-1}$) such that $v^{-1} \circ \nu \equiv \nu$. Define $\phi \equiv \nu \circ v^{-1}$. Given $T \in \phi(T)$, let us show that $\rho(lcg_v(T)) = lcg_v(T')$, by structural induction and case analysis on the def. (12) of $lcg_v$. 1) If $T = \emptyset$ then $\rho(lcg_v(T)) = \rho(lcg_v(\emptyset)) = \emptyset \circ v$ = $lcg_v(\emptyset) = lcg_v(T)$; 2) Else, if $lcg_v(T) = T$ then $lcg_v(T) = T'$, so that $\rho(lcg_v(T)) = \rho(T) = T' \circ v^{-1} \circ v = v'$; 3) Otherwise, $T = \{ f(t_1, \ldots, t^n) \mid i \in \Delta \}$, so that by ind., $\rho(T_
u^k) = \rho(lcg_v(T \{ f(t_1, \ldots, t^n) \mid i \in \Delta \})) = lcg_v(T \{ f(t_1, \ldots, t^n) \mid i \in \Delta \}) = f(T^k_1, \ldots, T^k_n) = T^k_1 \circ \ldots \circ T^k_n$. Therefore, by (6), $\rho(lcg_v(T)) = \rho(f(T^k_1, \ldots, T^k_n)) = f(T^k_1, \ldots, T^k_n) = \nu(T) = lcg_v(T')$. If $\alpha \in \text{vars}[lcg_v(T)]$ then case 1. of the above proof shows that $\alpha = lcg_v(T') = v'(T')$ for some $T' \in \phi(T)$ and therefore $\rho(lcg_v(T)) | \alpha = v'(T) = \nu \circ v^{-1} \circ v = v'$. Similarly, $v'(T') = \nu \circ v^{-1} \circ v = v'(T)$. Thus, proving that $\rho \in \text{vars}[lcg_v(T)] \Rightarrow \text{vars}[lcg_v(T)]$ is surjective. If $\alpha_1, \alpha_2 \in \text{vars}[lcg_v(T)]$ then case 1. of the above proof shows that $\alpha_1 = \text{lcg}_v(T) \in \alpha_1$ and $\alpha_2 = \text{lcg}_v(T) \in \alpha_2$ for some $\alpha_1, \alpha_2 \in \phi(T)$. Assume that $\rho(\alpha_1) = \rho(\alpha_2)$. Then we have $v' \circ v^{-1}(\alpha_1) = v' \circ v^{-1}(\alpha_2)$ that is $v' \circ v^{-1}(v(\alpha_1)) = v' \circ v^{-1}(v(\alpha_2))$, which implies $\nu(\alpha_1) = \nu(\alpha_2)$ since $v^{-1} = 1_{\nu(T)}$. It follows that $\alpha_1 = \alpha_2$ since $\nu'$ is injective. Therefore $\alpha_1 = \nu(\alpha_1) = \nu(\alpha_2) = \alpha_2$, proving that $\rho$ is injective. It follows that $\rho \in \text{vars}[lcg_v(T)] \Rightarrow \text{vars}[lcg_v(T)]$ is bijective so that $lcg_v(T) \sim v' \cdot lcg_v(T')$ by lemma 4. □

We now want to identify a Galois connection with abstraction $lcg_v$ and concretization ground. Several preliminary results are needed. First, the symbolic abstraction $lcg_v$ is $\subseteq$-increasing.
The concretization of a term with variables loses no reason on the quotient partial order of terms \( \langle P, \subseteq \rangle \) with variables up to variable renaming (see lemma \( C \)). The symbolic term Galois connection remains to prove that

\[
\text{ground} \vdash \ \text{Corollary 1.} \quad \text{If } \Delta \text{ is a nonempty set and } \{ t_i | i \in \Delta \} \subseteq \Delta \to T, \text{ then } \{ t_i | i \in \Delta \} \subseteq \text{ground}(\text{lcg})[\{ t_i | i \in \Delta \}].
\]

Proof of corollary 1. \( \quad \{ t_i | i \in \Delta \} \subseteq \text{ground}(\text{lcg})[\{ t_i | i \in \Delta \}] \)

\[
\text{Proof of corollary 2.} \quad \text{For all } \tau \in T\nu. \quad \text{ground} \circ \text{lcg}[\tau] = \text{ground}(\tau).
\]

Corollary 2. \( \quad \text{For all } \tau \in T\nu. \quad \text{ground} \circ \text{lcg}[\tau] = \text{ground}(\tau).
\]

The following corollary shows that the abstraction of the concretization of a term with variables loses no information.

\[
\text{Theorem 1.} \quad \text{For any naming scheme } \nu \in (\Delta \to T) \mapsto \forall \nu,
\]

\[
\langle \nu(T), \subseteq \rangle \ocircle \text{ground}_{\nu} \circ \text{lcg}[\nu] \Rightarrow \langle \nu(H), \preceq \rangle \quad (14)
\]

\[
\text{This definition of the Galois retraction is independent of the choice of the naming scheme } \nu. \quad \square
\]

Proof of theorem 1. By def. of a Galois connection, we must prove that for all families of terms \( \{ t_i | i \in \Delta \} \in \varphi(T) \) and term with variables \( \tau \in T\nu \vee \{ 0 \}, \)

\[
\text{lcg}_{\nu}[\{ t_i | i \in \Delta \}] \preceq \nu \circ \text{ground}_{\nu}[\{ t_i | i \in \Delta \}] \quad (\Rightarrow)
\]

\[
\text{by corollary 1 and transitivity;}
\]

\[
(\Leftarrow) \quad \text{By def. 7, } \text{lcg}[\nu] \text{ is increasing. By def.}
\]

\[
\text{ground}(\nu) \text{ is increasing. Their composition is increasing so } \text{ground} \circ \text{lcg}[\nu][\{ t_i | i \in \Delta \}] \subseteq \text{ground} \circ \text{lcg}[\nu] \circ \text{ground}(\nu) = \text{ground}(\nu) \quad \text{by corollary 2.}
\]

\[
\text{Moreover } \text{ground}_{\nu} \circ \text{lcg}[\nu] \text{ is injective so (14) is a Galois retraction (also called Galois insertion).}
\]

By lemma 6, if \( \nu, \nu' \in (\Delta \to T) \mapsto \forall \nu \text{ then } \forall T \in \varphi(T) \), \( \text{lcg}[\nu](T) \equiv \nu \circ \text{lcg}[\nu]'(T) \) so that this definition of the Galois retraction (14) is independent of the choice of the naming scheme \( \nu. \quad \square
\]

In a Galois connection \( \langle C, \subseteq \rangle \ocircle \text{ground}_{\nu} \circ \text{lcg}[\nu] \Rightarrow \langle A, \sqsubseteq \rangle, \alpha = \gamma \circ \alpha \circ \alpha. \) It follows immediately that \( \text{ground}_{\nu}(\text{lcg}[\nu]) \quad \subseteq \equiv \text{ground}_{\nu} \circ \text{lcg}[\nu] \Rightarrow \langle \text{lcg}[\nu], \preceq \rangle \) is a Galois isomorphism, an essential remark for completeness in typing [22].

D. The symbolic abstract domain is a complete lattice

By [26, THEOREM 4.1], the image of a complete lattice by an upper closure operator is a complete lattice. This extends to a Galois retraction \( \langle C, \subseteq \rangle \ocircle \text{ground}_{\nu} \circ \text{lcg}[\nu] \Rightarrow \langle A, \sqsubseteq \rangle, \gamma = \alpha \circ \gamma \circ \alpha. \) It follows immediately that \( \text{ground}_{\nu}(\text{lcg}[\nu]) \quad \subseteq \equiv \text{ground}_{\nu} \circ \text{lcg}[\nu] \Rightarrow \langle \text{lcg}[\nu], \preceq \rangle \) is a Galois isomorphism, an essential remark for completeness in typing [22].
Corollary 3 (symbolic abstract domain). For any naming scheme $\nu \in (\Delta \rightarrow T) \Rightarrow V_\nu$, $(P^H, \trianglelefteq, \exists^\nu, \exists^\nu)$ is a complete lattice where $\alpha \in V_\nu$, the least upper bound is $LGC_{\forall^\nu}(S) \triangleq \exists^\nu \{ \nu(\bigcup ground^\nu(S)) \}$ (binary $lgc$ for symbolic terms and $lgc_{\forall^\nu}$ for term classes), and the greatest lower bound is $GCI_{\forall^\nu}(S) \triangleq \exists^\nu \{ \nu(\bigcap ground^\nu(S)) \}$ (binary $gci$ and $gci_{\forall^\nu}$). This characterization of the lattice operations is independent of the naming scheme $\nu$ which is used.

Proof of corollary 3. Since $(\nu(T), \subseteq, \emptyset, T, \bigcup, \bigcap)$ is a complete lattice and (14) is a Galois retraction, it follows that, for any naming scheme $\nu \in (\Delta \rightarrow T) \Rightarrow V_\nu$, its image $lgc_{\forall^\nu}[\nu](\nu(T)) \in P^H$ by $lgc_{\forall^\nu}[\nu]$ is also a complete lattice $(P^H, \subseteq, [\exists^\nu]^\nu, [\alpha]^\nu, LGC_{\forall^\nu}, GCI_{\forall^\nu})$ where the supremum is $lgc_{\forall^\nu}[\nu](\{\emptyset\}) = [\exists^\nu]^\nu$, the supremum is $lgc_{\forall^\nu}[\nu](T) = [\nu(\{t \in T\})]^\nu \triangleright \sim^\nu [\alpha]^\nu, \alpha \in V_\nu$, the least upper bound is $LGC_{\forall^\nu}(S) \triangleq \exists^\nu \{ \nu(\bigcup ground^\nu(S)) \}$ and the greatest lower bound is $GCI_{\forall^\nu}(S) \triangleq \exists^\nu \{ \nu(\bigcap ground^\nu(S)) \}$.

By lemma 6, if $\nu, \nu' \in (\Delta \rightarrow T) \Rightarrow V_\nu$ then $\forall T \in \nu(T) \Rightarrow \nu'[T] \blacktriangleright \sim \nu[\nu(T)]$ so that this characterization of the lattice operations is independent of the choice of the naming scheme $\nu$. □

We use $lgc_{\forall^\nu}$ (respectively $gci_{\forall^\nu}$) for the binary version of $LGC_{\forall^\nu}$ (respect. $GCI_{\forall^\nu}$).

Observe that ground terms $\{t\}^\nu \in P^H$ belongs to the abstract domain and abstract the concrete property \{t\} of being that ground term. Then $lgc_{\forall^\nu}[\nu](\{t\}) = LGC_{\forall^\nu}(\{t\})$, because, by (14), we have

$LGC_{\forall^\nu}(\{t\}) \triangleq \exists^\nu \{ \nu(\bigcup ground^\nu(\{t\})) \}
= \exists^\nu \{ \nu(\bigcup ground^\nu(\{t\})) \}
= \exists^\nu \{ \nu(\bigcup ground^\nu(\{t\})) \}
= \exists^\nu \{ \nu(\bigcup \{t\}) \}
= \exists^\nu \{ \nu(t) \}.

This explains why the abstraction and the lub in the complete lattice have been given the same name.

VII. The classical definition of the subsumption partial order using substitutions

The subsumption preorder $\preceq^\nu$ is classically defined syntactically, using substitutions [7, pp. 180–188] (instead of (9)) [3–5]. We show that this classical syntactic definition is equivalent to the semantic definition (9) based on the interpretation of terms with variables as properties of ground terms.

A. Substitutions

The same way that assignments (5) record ground values of variables, we use substitutions to record symbolic values of some variables, so substitutions are partial functions

$$\vartheta \in \Sigma \triangleq V_\vartheta \Rightarrow T^\nu \tag{15}$$

mapping variables $\alpha$ in its domain $\text{dom}(\vartheta)$ to terms with variables $\vartheta(\alpha)$.

A substitution is extended to a total function $\vartheta \in V_\vartheta \Rightarrow T^\nu$ and homomorphically to terms with variables, as follows

$$\vartheta(\alpha) \triangleq \alpha \text{ when } \alpha \not\in \text{dom}(\vartheta) \quad (16)$$

$$\vartheta(f(\tau_1, \ldots, \tau_n)) \triangleq f(\vartheta(\tau_1), \ldots, \vartheta(\tau_n))$$

Observe that the substitution is carried out simultaneously on all variable occurrences.

The empty substitution $\varepsilon$ is totally undefined, that is $\text{dom}(\varepsilon) = \emptyset$. Its total extension is the identity $\forall \alpha \in V_\vartheta$. $\varepsilon(\alpha) = \alpha$. By structural induction on terms with variables, we have $\forall \tau \in T^\nu \Rightarrow \varepsilon(\tau) = \tau$.

B. The classical characterization of the subsumption preorder using substitutions

The following theorem 2 shows that the syntactic and semantic definitions of subsumption are equivalent. It follows that the subsumption lattice of [3–5], [27] is the complete lattice considered in corollary 3 since the partial order is the same (although defined differently).

Theorem 2. $\forall \tau_1, \tau_2 \in T^\nu \, [\tau_1 \preceq^\nu \tau_2] \Leftrightarrow \exists \vartheta \in \Sigma \, \vartheta(\tau_2) = \tau_1.$

Proof of theorem 2. Let us first show that for all $\tau_1, \tau_2 \in T^\nu$,

$$(\forall \varphi \in P^\nu \Rightarrow \exists \varphi' \in P^\nu \Rightarrow \varphi(\tau_1) = \varphi'(\tau_2)) \Leftrightarrow \exists \vartheta \in \Sigma \, \vartheta(\tau_2) = \tau_1. \tag{17}$$

(⇒) Choose $\varphi' = \lambda \alpha \cdot \varphi(\vartheta(\alpha))$ so that, by structural induction on terms with variables, $\forall \tau \in T^\nu \Rightarrow \varphi'(\tau) = \varphi(\vartheta(\tau))$. Then $\vartheta(\tau_2) = \tau_1$ implies $\varphi(\vartheta(\tau_2)) = \varphi(\tau_1)$ and so $\varphi'(\tau_2) = \varphi'(\tau_1)$.

(⇐) We assume that $\forall \varphi \in P^\nu \Rightarrow \exists \varphi' \in P^\nu \Rightarrow \varphi(\tau_1) = \varphi'(\tau_2)$. The structural proof is by cases on the pair $(\tau_1, \tau_2)$ ordered lexicographically. It consists in constructing $\vartheta$ given $\varphi$ and $\varphi'$.

• $\tau_1 = a \in F_0$,
  - $\tau_2 = b \in F_0$. If $b \neq a$, the hypothesis is false and the implication is true. Otherwise $b = a$ and any substitution has $\vartheta(\tau_1) = \vartheta(a) = a = \vartheta(\tau_2)$.
  - $\tau_2 = \beta \in V_\vartheta$. Any substitution s.t. $\vartheta(\beta) = a$ has $\vartheta(\tau_2) = \vartheta(\beta) = a = \tau_1$.
• $\tau_2 = g(\tau_1', \ldots, \tau_m'), g \in F_m$. $\forall \varphi, \varphi' \Rightarrow \varphi(\tau_1) = \varphi'(\tau_2)$ so that the hypothesis is false and the implication is true.

• $\tau_1 = \alpha \in V_\vartheta$,
  - $\tau_2 = b \in F_0$. If $\varphi(\alpha) \neq b$, the hypothesis is false and the implication is true. Otherwise $\varphi(\alpha) = b$.
  - $\tau_2 = \beta \in V_\vartheta$. Any substitution such that $\vartheta(\beta) = \alpha$ will do.
  - $\tau_2 = g(\tau_1', \ldots, \tau_m')$, $g \in F_m$. Since we assume that the signature $F$ has at least two different function symbols, there is a term $\tau$ with this different symbol at
the root. For \( \varphi \) such that \( \varphi(\alpha) = \tau \) there is no \( \varphi' \) such that \( \varphi(\alpha) = \tau = \varphi'(\tau_2) = g(\varphi'(\tau_j'), \ldots, \varphi'(\tau_n')) \). In that case, the hypothesis is false and the implication is true.

- \( \tau_1 = f(\tau_1', \ldots, \tau_n') \), \( f \in F_n \)
- \( \tau_2 = b \in F_0 \). \( \forall \varphi, \varphi' : \varphi(\tau_1) \neq \varphi'(\tau_2) \) so the hypothesis is false and the implication is true.
- \( \tau_2 = \beta \in V_2 \). Simply choose \( \theta(\beta) = \tau_1 \).
- \( \tau_2 = \varphi^0(\tau, \ldots, \tau_n, \mathcal{g}) \in F_m \). The hypothesis that \( \varphi(\tau_1) = \varphi'(\tau_2) \), that is, \( g(\varphi(\tau_1), \ldots, \varphi(\tau_n)) = g(\varphi'(\tau_1'), \ldots, \varphi'(\tau_n')) \), implies that \( g = g, m = n, \) and \( \forall i \in [1, n] \). If \( \varphi(\tau_i) = \varphi'(\tau_i') \). So, by structural ind. \( \exists \varphi_j \). \( \varphi_j(\tau_j') = \tau_j' \) that is \( \tau_1 = f(\tau_1', \ldots, \tau_n') = f(\varphi(\tau_1'), \ldots, \varphi_n'(
(\mathcal{g})) \). We now have to define \( \theta \) such that \( \varphi(\tau_2) = \tau_1 \).

* For the variables \( \alpha \) that do not occur in \( \tau_2 \), we choose \( \theta(\alpha) = \alpha \); 
* For variables \( \alpha \) that do occur more or in \( \tau_2 \), say in \( \tau_j' \) and \( \tau_n' \), \( j, k \in [1, n] \), we have 

\[
\begin{align*}
\varphi(\tau_1') = \varphi(\tau_2) & = f(\varphi(\tau_1'), \ldots, \varphi(\tau_n)) \\
& = f(\varphi(\tau_1'), \ldots, \varphi(\tau_1')') = \varphi(\tau_1') \\
& = \varphi(\theta(\tau_1')') = \varphi(\tau_1) \quad \text{(by hypothesis,} \exists \varphi'). \\
\end{align*}
\]

There are two subcases:

- If for all occurrences, the substitutions are identical, we choose \( \theta(\alpha) = \varphi_j(\alpha) = \varphi_k(\alpha) \); 
- Otherwise, \( \exists \alpha, j \neq k \). \( \varphi_j(\alpha) \neq \varphi_k(\alpha) \) and so \( \theta(\alpha) = \varphi_j(\alpha) \). 

\[ \varphi(\tau_1) = \varphi(\tau_2) \] implies that \( \exists \theta \in \mathcal{g} \). \( \theta(\tau_2) = \tau_1 \). It follows that 

\[
[\tau_1]_{\mathcal{g}} \succeq [\tau_2]_{\mathcal{g}} \tag{17}
\]

\[ \implies \quad \text{VIII. Conclusion} \]

We have shown that a concrete program property represented by a set of ground terms can be over-approximated by a term with variables. Of course a concrete property \( P \) represented by a set of terms with variables can be overapproximated by the term with variables \( LCG_P(X) \) (this is Galois connection of the homomorphic abstraction \( \alpha(X) \triangleq \{ h(x) \mid x \in X \} \) of a set \( X \), where \( h \) is the identity). The complete lattice structure follows from the fact that the image of a complete lattice by a Galois retraction is a complete lattice [26, Theorem 4.1]. This approach yields algorithms together with their soundness proof by abstraction preservation [28, section 48.8].

We have shown that this semantic construction yields the same subsumption partial order defined syntactically by Gordon Plotkin [3, 4] and John Reynolds [5].

One can avoid variables by representing a term with variables as a rooted directed acyclic graph (DAG) that is the term syntax tree where the leaf nodes that have the same variable are joined [29] (mathematically represented by a tree and an equivalence relation between leaves that have the same variable). The lattice structure of symbolic terms generalizes to rooted order-sorted feature (OSF) graphs [30, 31].

\[ \text{REFERENCES} \]


