Property Checking Driven Abstract Interpretation-Based Static Analysis[⋆]

Damien Massé

LIX, École Polytechnique, Palaiseau, France,
masse@lix.polytechnique.fr,
http://www.lix.polytechnique.fr/~masse/

Abstract. Concrete semantics used for abstract interpretation analyses are generally expressed as fixpoints. Checking a property on this kind of semantics can be done by intersecting the fixpoint with a specification related to the property. In this paper, we show how to produce a new, "reverse" analysis from this specification. The result of this analysis, expressed as a lower closure operator, is then used to guide the initial analysis. With this approach, we can refine the result given by the direct abstract analysis. We show that this method enables to deduce forward analyses from backward analyses (and *vice-versa*), and to combine them iteratively in a way similar to the forward-backward combination of analyses.

1 Introduction

The main idea of abstract interpretation [1–3] is to derive an abstract semantics of a program from its concrete semantics. When the goal is to prove a property (e.g. temporal property), the abstraction must be precise enough to express this property. However, the computation of the abstract semantics often induces losses of information, and gives only an approximation of the concrete semantics, which may not be sufficient to prove the property.

Refining the abstract domain to a complete one, as presented by Ranzato and *al.* [6], is a method to reduce the loss of information. But the refined abstract domain may be not representable (or very complex), moreover this method is not applicable when widenings [3] are required.

This paper exposes a different approach, based on the property we want to check. In general, the fixpoint defining the concrete semantics is a description of the "behaviors" of the program. This fixpoint computed (or approximated), the property is checked by "intersecting" the result with the specification of the property. From this operation, described as a lower closure operator, we construct another lower closure operator which can be included in the concrete semantics. The derived abstract semantics is then more precise than the initial abstract

⋆ This work was supported in part by the RTD project IST-1999-20527 DAEDALUS of the European IST FP5 program.

semantics. We prove that we can abstract this construction (which is then seen as a new analysis) while preserving the correctness of the method.

To complete the approach, we show how the result of this new abstract analysis can be used to refine the (approximated) lower closure operator, from which we derive another abstract semantics, and so on.

This approach gives results which are similar to the combination of backward and forward analysis used in abstract interpretation [1, 8], but we start from only one analysis: the second analysis is derived from the first. As an example, we show that, by applying this approach, we can obtain constructions similar to previous such examples of combination.

This paper remains mainly theoretical. Though it is not very hard to implement analyzers based on this approach, getting the full advantage of it requires efficient abstractions on the set of lower closure operators on a lattice (and efficient abstract operators). We present in this paper a non-trivial class of abstractions on this set as a starting point for such abstractions, but more work is needed to make real analyzers.

2 Preliminaries

In order to present the new approach, we need to define some notations, and recall some known results on lower closure operators.

2.1 Notations

Let $\mathbb{Z}$ be the set of integers, and $\mathbb{Z}^\infty = \mathbb{Z} \cup \{-\infty, +\infty\}$.

Let τ be a transition relation on the set Σ, we recall the four predicate transformers in $\wp(\Sigma) \rightarrow \wp(\Sigma)$:

$$\begin{aligned}
\mathrm{post}(X) &= \{\sigma' \in \Sigma \mid \exists \sigma \in X, (\sigma, \sigma') \in \tau\} \\
\widetilde{\mathrm{post}}(X) &= \{\sigma' \in \Sigma \mid \forall \sigma \in \Sigma, (\sigma, \sigma') \in \tau \Rightarrow \sigma \in X\} \\
\mathrm{pre}(Y) &= \{\sigma \in \Sigma \mid \exists \sigma' \in Y, (\sigma, \sigma') \in \tau\} \\
\widetilde{\mathrm{pre}}(Y) &= \{\sigma \in \Sigma \mid \forall \sigma' \in \Sigma, (\sigma, \sigma') \in \tau \Rightarrow \sigma' \in Y\}
\end{aligned}$$

If $(D, \leq)$ is a partially ordered set and $a \in D$, the principal ideal generated by a is denoted $(\uparrow_{\leq} a) = \{b \in D \mid a \leq b\}$. When there is no ambiguity, we will note $(\uparrow a)$ instead of $(\uparrow_{\leq} a)$. We let $D \overset{m}{\rightarrow} D$ be the set of monotone operators from D into D.

ϕ being a monotone operator on a complete lattice $\langle D, \leq, \vee, \wedge, \bot, \top \rangle$, we denote by $\mathrm{lfp}\,\phi$ (resp. $\mathrm{gfp}\,\phi$) the least fixpoint (resp. the greatest fixpoint) of ϕ. $\mathrm{lgfp}\,\phi$ will denote either $\mathrm{lfp}\,\phi$ or $\mathrm{gfp}\,\phi$.

We recall that ϕ is continuous (resp. co-continuous) iff for all increasing (resp. decreasing) chain A in D, $\phi(\bigvee A) = \bigvee_{a \in A} \phi(a)$ (resp. $\phi(\bigwedge A) = \bigwedge_{a \in A} \phi(a)$).

When ϕ is extensive and a is an element of D, we denote by $\mathrm{luis}\,(\phi, a)$ the limit of the upper iteration sequence of ϕ starting by a.

2.2 (Lower) closure operators

Closure operators and Moore families are widely used in abstract interpretation [1, 3], to represent abstractions without the use of an abstract domain. Here, we will use lower closure operators to "restrict" the functions of the analysis independantly of the abstractions, which will be represented as upper closure operators.

In this section, we recall the results on lower closure operators.

General results In the following section, $\langle D, \leq, \vee, \wedge, \bot, \top \rangle$ is a complete lattice.

We recall that an operator ρ on D is a lower (resp. upper) closure operator if it is monotone, idempotent and reductive (resp. extensive).

Proposition 1. *The set $lco\,(D)$ of lower closure operators on D is a complete lattice $\langle lco\,(D), \sqsubseteq, \sqcup, \sqcap, \lambda x.x, \lambda x.\bot \rangle$, with:*

$$\begin{aligned} \rho \sqsubseteq \rho' &\iff \forall x \in D, \rho(x) \leq \rho'(x), \\ \textstyle\bigsqcup_{i\in\Delta} \rho_i &= \lambda x. \textstyle\bigvee_{i\in\Delta} \rho_i(x), \\ \textstyle\bigsqcap_{i\in\Delta} \rho_i &= \lambda x.\mathrm{lfp}\, \lambda y.(x \wedge \textstyle\bigwedge_{i\in\Delta} \rho_i(y)). \end{aligned}$$

For upper closure operators, we will denote by $\langle uco\,(D), \sqsubseteq, \sqcup, \sqcap, \lambda x.\top, \lambda x.x \rangle$ the complete lattice of upper closure operators on D.

Proposition 2 (Moore families). *Any lower closure operator ρ is uniquely determined by the set of its fixpoints $\rho(D)$, which is an (upper) Moore family (i.e. $\rho(D) = \mathcal{M}\,(\rho(D)) = \{\vee X \mid X \subseteq \rho(D)\}$). The correspondence between $lco\,(D)$ and the set of upper Moore families in D is described by the following equations:*

$$\begin{aligned} \rho \sqsubseteq \rho' &\iff \rho(D) \subseteq \rho'(D) \\ \textstyle\bigsqcup_{i\in\Delta} \rho_i &= \mathcal{M}\left(\textstyle\bigcup_{i\in\Delta} \rho(D)\right) \\ \textstyle\bigsqcap_{i\in\Delta} \rho_i &= \textstyle\bigcap_{i\in\Delta} \rho(D) \end{aligned}$$

We will use indifferently lower closure operators and upper Moore families. When there is no ambiguity, we will denote $\rho(D)$ by ρ itself (e.g. $\rho \sqsubseteq \rho' \iff \rho \subseteq \rho'$). An example of upper Moore family is given Fig. (3a).

Proposition 3 (Soundness [3]). *For all $\phi \in D \overset{m}{\to} D$ and $\rho \in lco\,(D)$, we have:*

$$\begin{aligned} &\rho \circ \phi \circ \rho \leq \rho \circ \phi \\ &\rho(\mathrm{lfp}\,\phi) \geq \mathrm{lfp}\,(\rho \circ \phi) \\ &\rho(\mathrm{gfp}\,\phi) \geq \mathrm{gfp}\,(\rho \circ \phi). \end{aligned}$$

Completeness *Completeness* (or *exactness*) is the inverse property of soundness:

Definition 1 (Completeness [3]). *$\rho \in lco(D)$ is said to be* complete *for a monotone operator ϕ iff $\rho \circ \phi \circ \rho \geq \rho \circ \phi$ (and so $\rho \circ \phi \circ \rho = \rho \circ \phi$).*

Whereas the soundness is always satisfied for all monotone operator ϕ, completeness is always relative to an operator (or a set of operators). When we do not state the operator, completeness will be for ϕ.

The following proposition gives the relation between *completeness* and *fixpoint completeness.*

Proposition 4 (Fixpoint completeness).

1. *If ρ is complete, then ρ is* gfp-complete *(i.e. $\rho(\mathrm{gfp}\,\phi) = \mathrm{gfp}\,\rho \circ \phi$).*
2. *If ρ is complete* and *continuous, then ρ is* lfp-complete *(i.e. $\rho(\mathrm{lfp}\,\phi) = \mathrm{lfp}\,\rho \circ \phi$).*

For upper closures, it is well-known that completeness implies lfp-completeness (called *fixpoint completeness* in [6]). This result was first presented by Cousot and Cousot [3]. However, completeness (for upper closures) does *not* imply gfp-completeness. By duality, for lower closure operators, completeness ensures gfp-completeness but not lfp-completeness.

Example 1. We choose $D = \wp(\mathbb{Z})$ and $\phi = \lambda X.\{0\} \cup \{x+1 \mid x \in X\}$. We know that $\mathrm{lfp}\,\phi = [0, +\infty[$. Let $\rho = \{\emptyset, [0, +\infty[\}$. Since ρ is an upper Moore family, ρ defines a lower closure operator[1], and one can check easily that ρ is complete for ϕ. However, $\mathrm{lfp}\,\rho \circ \phi = \emptyset$, and ρ is not lfp-complete.

Construction of complete closures The construction of complete closure operators was studied by Giacobazzi, Ranzato and Scozzari [6]. Here is the result we will use in this paper:

Theorem 1 (Complete closures (dual of [6, Thm. 5.10])). *Let ρ_0 be a lower closure operator on D. If ϕ is co-continuous, then:*

$$\begin{aligned} R_\phi &= \lambda\eta.\ \mathcal{M}(\cup_{a\in\eta} \min(\phi^{-1}(\uparrow a))) \\ \textit{and}\ \mathcal{R}_\phi(\rho_0) &= \mathrm{lfp}_{\sqsubseteq}\ \lambda\eta.(\rho_0 \sqcup R_\phi(\rho_0)) \end{aligned} \tag{1}$$

are well defined and $\mathcal{R}_\phi(\rho_0)$ is the lowest complete lower closure operator greater than ${\rho_0}$[2].

[1]
$$\rho(X) = \begin{cases} [0, +infty[& \text{if } X \supseteq [0, +infty[, \\ \emptyset & \text{otherwise.} \end{cases}$$

[2] $\mathcal{R}_\phi(\rho_0)$ is called as the complete shell of ρ_0.

When ϕ is not co-continuous, this theorem does not hold because the min operator does not satisfy $\forall(a,x),\ x \in \phi^{-1}(\uparrow a) \Rightarrow (\exists y \in \min(\phi^{-1}(\uparrow a)), x \geq y)$. However, with $\min' : \wp(D) \to \wp(D)$ satisfying:

- $\forall X \subseteq D, \min'(X) \subseteq X$,
- $\forall X \subseteq D, \forall a \in X, \exists a' \in \min'(X)$ s.t. $a' \leq a$,

we can defined R_ϕ and $\mathcal{R}_\phi$ as:

$$\begin{aligned} R_\phi &= \lambda\eta.\ \mathcal{M}(\cup_{a\in\eta} \min'(\phi^{-1}(\uparrow a))) \\ \text{and } \mathcal{R}_\phi(\rho_0) &= \mathrm{lfp}_{\sqsubseteq}\ \lambda\eta.(\rho_0 \sqcup R_\phi(\rho_0)) \end{aligned} \qquad (2)$$

Then $\mathcal{R}_\phi(\rho_0)$ is a complete lower closure operator greater than ρ_0 (though it may not be minimal).

From this result, we can construct gfp-complete operators. To construct lfp-complete operator, we need a complete and continuous operator. We can achieve this goal by using an extensive operator C on $lco\,(D)$ such that $C(lco\,(D))$ includes only continuous operators.

Proposition 5. *Let C be an extensive operator on $lco\,(D)$ such that, for all lower closure operator η, $C(\eta)$ is $\leq$-continuous. Then $C \circ \mathcal{R}_\phi$ (with $\mathcal{R}_\phi$ defined either by equation (1) or by equation (2) is an extensive operator, and luis$(C \circ \mathcal{R}_\phi, \rho_0)$ is a continuous complete lower closure operator greater than ρ_0 (thus, it is* lfp*-continuous).*

Example 2. We give here two simple examples for C.

1. As $\lambda a.a$ is continuous, $C = \lambda\eta.(\lambda a.a)$ satisfies the conditions of the proposition. This examples is worthless, but it proves the existence of at least one possible operator.
2. When $D = \wp(\Sigma)$, $\lambda a.(a \cap X)$ for $X \subseteq \Sigma$ is a continuous lower closure operator. Therefore, $C = \lambda\eta.(\lambda a.a \cap (\eta(\Sigma)))$ satisfies the conditions. This example is interesting, as C is an upper closure operator and all elements of $C(lco\,(D))$ can be defined by a subset of Σ. Thus, we will be able to use abstractions of $\wp(\Sigma)$ to abstract lower closure operators.

3 Using lower closure operators

In the abstract interpretation framework, lower closure operators are seen as lower approximations. However, this usage is hardly seen in practice[3], and does not present any additional theoretical interest, as a lower analysis is merely the dual of an upper analysis.

Our approach does not intend to use lower approximations. On the contrary, all abstractions will be upper abstractions. Lower closure operators are used to "reduce" the "range" of the analysis, to restrict it to significant parts. In this section we formalize this approach.

[3] It seems that useful lower abstractions are harder to find than upper abstractions.

3.1 Concrete description

The first step of static analyses is the description of the concrete semantics of a program. The semantics is often described as a fixpoint $\mathcal{S} = \text{lgfp}\,\phi$ on a *cpo* D. We suppose that D is a complete lattice, and, for simplicity, that $D = \wp(\Sigma)$, e.g. Σ is a set of states, traces, trees, etc.

Our main hypothesis is that the property we want to prove is expressed by the inclusion of $\mathcal{S}$ in a subset $\mathcal{P}$ of Σ. This hypothesis may seem too strong, but we can modify the concrete semantics in order to satisfy it.

Example 3. We can describe the program as a transition system τ, Σ being the set of states. I are the initial states of the program.

1. To express that the program will never reach error states E, we can write either:
$$(\text{lfp}\lambda X.I \cup \text{post}(X)) \subseteq \Sigma \backslash E,$$
or
$$(\text{lfp}\lambda X.E \cup \text{pre}(X)) \subseteq \Sigma \backslash I.$$
2. To express that for all initial state, the program may not go wrong (with the CTL formalism, $\forall i \in I, i \models \mathbf{EG}(\neg \mathit{error})$), we can only use a backward approach:
$$(\text{lfp}\lambda X.E \cup \widetilde{\text{pre}}(X)) \subseteq \Sigma \backslash I \qquad \text{where } E \text{ are the error states.}$$
3. For complex CTL properties, with more than one fixpoint, we may need to change the concrete domain in order to keep one general fixpoint for the computation. The backward semantics expressed in [9] is an example.

Proving $\mathcal{S} \subseteq \mathcal{P}$ is equivalent to prove $\mathcal{S} \cap (\Sigma \backslash \mathcal{P}) = \emptyset$. This kind of property may be checked in the framework of abstract interpretation: an upper approximation of $\mathcal{S} \cap \mathcal{Q}$ (with $\mathcal{Q} = \Sigma \backslash \mathcal{P}$) is computed and compared with $\emptyset$.

The main idea of the approach is to consider $\rho_0 = \lambda X.X \cap \mathcal{Q}$ as a *lower closure operator* on $\wp(\Sigma)$, and to exploit the results on the construction of fixpoint complete lower closure operators. With the constructions described in section 2.2, we can construct a lower closure ρ greater than ρ_0 which is lgfp-complete for ϕ. Then:
$$\rho_0\,(\text{lgfp}\,\rho \circ \phi) = \rho_0 \circ \rho\,(\text{lgfp}\,\phi) = \mathcal{S} \cap \mathcal{Q}$$
The figure (1) is an illustration of this result.

Thus, instead of computing an over-approximation of lgfp ϕ, we can compute an over-approximation of lgfp $\rho \circ \phi$.

Example 4. With $\phi = \lambda X.F \cup \widetilde{\text{pre}}(X)$, we can use the equation (1) since ϕ is co-continuous. With $a \subseteq \Sigma$, we have:
$$\mathit{min}(\phi^{-1}(\uparrow a)) = \{\text{post}(a \backslash F)\}$$

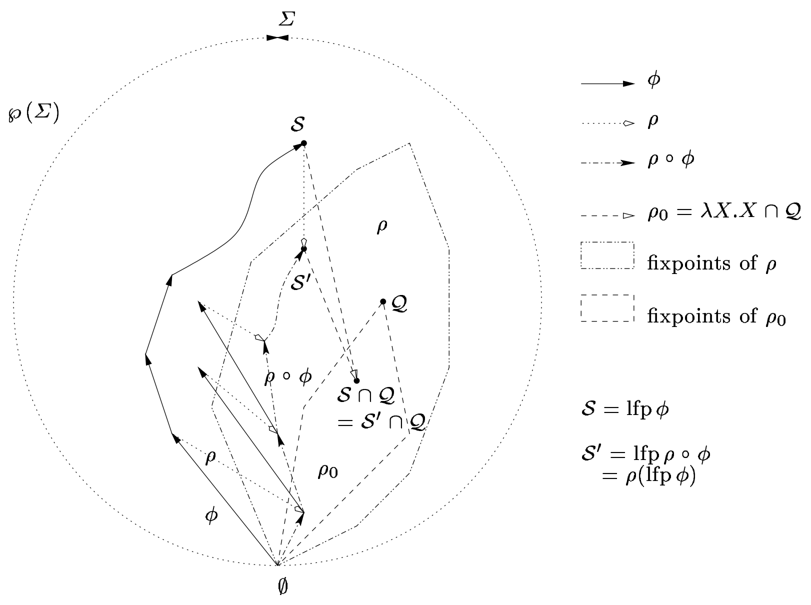
Σ
$\wp(\Sigma)$
$\mathcal{S}$
$\mathcal{S}'$
$\mathcal{Q}$
ρ
$\rho \circ \phi$
$\mathcal{S} \cap \mathcal{Q}$
$= \mathcal{S}' \cap \mathcal{Q}$
ρ_0
ϕ
$\emptyset$
ϕ
ρ
$\rho \circ \phi$
$\rho_0 = \lambda X.X \cap \mathcal{Q}$
fixpoints of ρ
fixpoints of ρ_0
$\mathcal{S} = \mathrm{lfp}\,\phi$
$\mathcal{S}' = \mathrm{lfp}\,\rho \circ \phi$
$= \rho(\mathrm{lfp}\,\phi)$

Fig. 1. Description of the combination, without abstractions. We want to compute $\mathcal{Q} \cap \mathrm{lfp}\,\phi = \rho_0(\mathrm{lfp}\,\phi)$. We design $\rho \sqsupseteq \rho_0$ such that ρ is lfp-complete for ϕ. Then $\mathcal{Q} \cap \mathrm{lfp}\,\rho \circ \phi = \mathcal{Q} \cap \mathrm{lfp}\,\phi$.

For example, we can choose $\Sigma = \{1,2,3,4,5\}$, $\mathcal{Q} = \{1\}$, $F = \{5\}$ and $\tau = \{(1,2),(1,3),(2,4),(3,5),(4,3),(4,2)\}$.

Then $\mathcal{R}_\phi(\rho_0) = \mathcal{M}(\{\{1\},\{2,3\},\{4,5\}\})$ (cf. Fig. 2). The derived analysis $\mathrm{lfp}\,(\mathcal{R}_\phi(\rho_0)) \circ \phi$ gives $\emptyset$ at the first iteration. Therefore, the computation of $\mathcal{R}_\phi(\rho_0)$ is a kind of “forward analysis” which carry informations to prove the property we want to check.

3.2 Results with abstractions

The results given until now were on the concrete domain, without abstractions. Of course, the lfp-complete lower closure operator is, in general, not computable. However, the goal of our approach is to get a new, computable analysis. Thus we need abstractions.

In this section, we introduce them, both in $\wp(\Sigma)$ (to compute sound approximations in the first analysis) and in *lco* $(\wp(\Sigma))$ (to compute sound approximations of the complete lower closure operators). We will show that the previous results are still correct with these abstractions, thanks to the usage of lower closure operators[4]. This will prove the correctness of our method.

[4] Specifying the property with upper closures would require to use a lower abstraction on the closure domain *and* on the initial concrete domain.

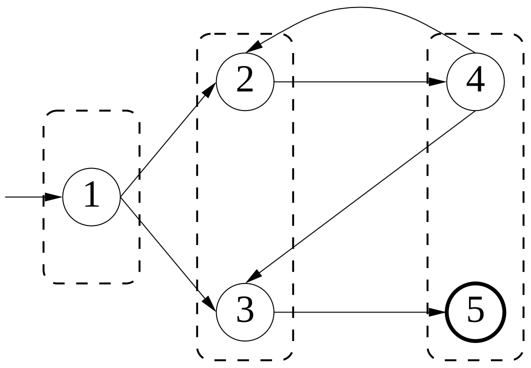

Fig. 2. Transition system described in example 4. $\{1\}$, $\{2,3\}$ and $\{4,5\}$ generates $\mathcal{R}_\phi(\rho_0)$

In the following propositions, R_ϕ is defined with the equation (1) if ϕ is co-continuous or with the equation (2) otherwise.

The case of a greatest fixpoint is easier, so we present it first.

Proposition 6 (gfp **analysis**). *Let ϕ be a monotone operator on $\wp(\Sigma)$, and $\mathcal{Q}$ be a subset of Σ. We define $\rho_0 = \lambda X.X \cap \mathcal{Q}$. Let $\nu \in uco(\wp(\Sigma))$ and $\Upsilon \in uco(lco(\wp(\Sigma)))$ be (upper) abstractions of $\wp(\Sigma)$ and $lco(\wp(\Sigma))$, respectively. Then, with:*

$$\rho = \text{lfp}\,\lambda\eta.\Upsilon(\rho_0 \sqcup R_\phi(\eta)),$$

we have $\mathcal{Q} \cap \text{gfp}\,\phi \subseteq \text{gfp}\,\nu \circ \rho \circ \phi$.

For lfp analysis, we need to construct a continuous operator. We use the method presented in proposition 5.

Proposition 7 (lfp **analysis**). *Let ϕ be a monotone operator on $\wp(\Sigma)$, and $\mathcal{Q}$ be a subset of Σ. We define $\rho_0 = \lambda X.X \cap \mathcal{Q}$. Let C be an extensive operator on $lco(\wp(\Sigma))$ such that for all η, $C(\eta)$ is continuous. Let $\nu \in uco(\wp(\Sigma))$ and $\Upsilon \in uco(lco(\wp(\Sigma)))$ be (upper) abstractions of $\wp(\Sigma)$ and $lco(\wp(\Sigma))$, respectively. Then, with:*

$$\begin{aligned} R &= \lambda\eta.\Upsilon \circ C(\text{lfp}\,\lambda\eta'.\Upsilon(\eta \sqcup R_\phi(\eta'))) \\ \rho &= luis(R, \rho_0), \end{aligned}$$

we have $\mathcal{Q} \cap \text{lfp}\,\phi \subseteq \text{lfp}\,\nu \circ \rho \circ \phi$.

It may seem that ρ would be long to compute, as there are two imbricated fixpoints. In practical applications, however, it is probable that we do not need to apply C many times.

These theorems give a method to define a new, “reverse” analysis (since R_ϕ depends on ϕ^{-1}) which can be used to “guide” the first analysis, and thus to enhance its result.

Expressing $R_\phi(\eta)$ is not so hard in practice. When η is generated by a set $\mathcal{A}$ of subsets of Σ (that is, $\eta = \mathcal{M}(A)$), we need only to know that $R_\phi(\eta)$ is generated by $\cup_{X\in\mathcal{A}}\min^?(\phi^{-1}(\uparrow X))$ ($\min^?$ being either min or min$'$). Therefore, we do not need to keep a representation of the whole Moore family, just of a set of generators.

Example 5. Starting from an abstraction ν of $\wp(\Sigma)$, we can use $\Upsilon = C = \lambda\eta.(\lambda X.X \cap \nu \circ \eta(\Sigma))$ (which can be represented as an element of ν). Then the result is the same for lfp and gfp analysis. To examine the result of this abstraction, we take $\nu = \lambda x.x$.

Then, for all $\eta \in \Upsilon(\mathit{lco}(\wp(\Sigma)))$, η satisfies $\eta = \mathcal{M}(\{\{x\} \mid x \in \eta(\Sigma)\})$, so we only have to express $\min'(\phi^{-1}(\uparrow \{x\}))$ to get $R_\phi(\eta)$.

With Σ being a set of states and $\phi = \lambda X.A \cap (F \cup \mathrm{pre}(X))$ (that is, lgfp ϕ are the states which can go to F or, for the gfp, loop indefinitely in A), we can use:

$$\min'(\phi^{-1}(\uparrow \{x\})) = \begin{cases} \emptyset & \text{if } x \notin A \\ \{\emptyset\} & \text{if } x \in F \\ \{\{y\} \mid y \in \mathrm{post}(\{x\})\} & \text{otherwise} \end{cases}$$

Then, with $\eta = \lambda X.X \cap Y$ and $\rho_0 = \lambda X.X \cap \mathcal{Q}$,

$$\begin{aligned} &\Upsilon(\rho_0 \sqcup R_\phi(\eta)) = \lambda X.X \cap Y' \\ &\qquad \text{with } Y' = \mathcal{Q} \cup \mathrm{post}(Y \cap (A \backslash F)) \end{aligned}$$

Thus $\rho = \lambda X.X \cap (\mathrm{lfp}\, \lambda Y.\mathcal{Q} \cup \mathrm{post}(Y \cap (A\backslash F)))$. We got the reachability analysis in A (with a slight modification due to F). Using it before the backward analysis is a well-known idea both in abstract interpretation and in model-checking [4].

With this abstraction, this is also the best result we can get with $\phi = \lambda X.A \cap (F \cup \widetilde{\mathrm{pre}}(X))$.

3.3 The combination

Until now, we just show how to construct a reverse analysis from an initial one, and with this reverse analysis restrict the range of the first analysis. However, the backward-forward combination used in abstract interpretation works in both ways: the result of the first analysis is used to get a better reverse result, which we can use in the first analysis, and so on.

Our approach is not symmetrical: the first analysis is on $\wp(\Sigma)$, whereas the second one is on $\mathit{lco}(\wp(\Sigma))$. But we can still use the result of the initial analysis, even restricted, in the reverse analysis. This property is given by the following proposition for lfp analysis (the same result holds for gfp analysis):

Proposition 8. *Let ϕ be a monotone operator on $\wp(\Sigma)$, and $\mathcal{Q}$ be a subset of Σ. Let ρ_1 be a* lfp*-complete lower closure operator for ϕ such that $\mathcal{Q} \cap \mathrm{lfp}\,\phi \subseteq \mathrm{lfp}\,\rho_1 \circ \phi$, and T be a subset of Σ such that $T \supseteq \mathrm{lfp}\,\rho_1 \circ \phi$. We define $\varrho_T \in \mathit{lco}(\wp(\Sigma))$ as $\varrho_T = \lambda X.X \cap T$.*

Let C be an extensive and monotone operator on $\mathit{lco}(\wp(\Sigma))$ such that for all η, $C(\eta)$ is continuous.

Let $\nu \in uco(\wp(\Sigma))$ and $\Upsilon \in uco(lco(\wp(\Sigma)))$ be (upper) abstractions of $\wp(\Sigma)$ and $lco(\wp(\Sigma))$, respectively. Then, with:

$$\begin{aligned} R^T &= \lambda\eta.\Upsilon \circ C(\operatorname{lfp} \lambda\eta'.\Upsilon(\varrho_T \sqcap (\eta \sqcup R_\phi(\eta')))) \\ \rho^T &= \operatorname{lfp} \lambda\eta.(\varrho_T \sqcap (\rho_0 \sqcup R^T(\eta)) \end{aligned}$$

then it exists $\rho_2 \in lco(\wp(D))$ such that ρ_2 is lfp*-complete for ϕ, $\rho^T \sqsupseteq \rho_2$ and $\mathcal{Q} \cap \operatorname{lfp}\phi \subseteq \operatorname{lfp}\rho_2 \circ \phi$ (hence, $\mathcal{Q} \cap \operatorname{lfp}\phi \subseteq \operatorname{lfp}\nu \circ \rho^T \circ \phi$).*

With this proposition, we can construct a decreasing sequence (T_n) of elements of $\wp(\Sigma)$ greater than $\operatorname{lfp}\phi \cap \mathcal{Q}$ (along with a decreasing sequence of lower closure operators (ρ^{T_n}), each ρ^{T_n} being greater than a lfp-complete closure operator ρ_n which satisfies $\mathcal{Q} \cap \operatorname{lfp}\phi \subseteq \operatorname{lfp}\rho_n \circ \phi$):

$$\begin{aligned} T_0 &= \Sigma \\ T_{k+1} &= \operatorname{lfp}\nu \circ \rho^{T_k} \circ \phi \\ T_\omega &= \textstyle\bigcap_{k<\omega} T_k \qquad \text{for all limit ordinal } \omega \end{aligned}$$

Example 6. We continue the example (5). We have:

$$\rho^T = \lambda X.X \cap (\operatorname{lfp}\lambda Y.T \cap (\mathcal{Q} \cup \operatorname{post}(Y \cap (A \backslash F)))).$$

Then, in the sequence (T_n), T_{n+1} is constructed by doing a forward analysis restricted to T_n, then a backward analysis restricted to the result of the forward analysis. This result is similar to the backward-forward combination used in abstract interpretation [5], even if the goal is not the same.

Remark 1. This combination which uses lower closures may seem similar to Granger's local decreasing iterations [7]. However, in Granger's iterations, lower closures are constant and applied several times. Here lower closures are modified at each iteration.

4 Abstractions of $lco(\wp(\Sigma))$

The examples we presented until now gives mainly already known results. To find better results, we must develop efficient abstractions of $lco(\wp(\Sigma))$. The first approach is to find abstractions from existing abstractions of $\wp(\Sigma)$. As an upper Moore family is an element of $\wp(\wp(\Sigma))$, this work can be related to the search of abstractions of $\wp(\wp(\Sigma))$. Following this principle, we will express the abstractions Υ as operators on the lattice of upper Moore families.

4.1 "Upper" abstraction

The "upper" abstraction was already given in example (5): from an abstraction $\nu \in uco(\wp(\Sigma))$, we define $\Upsilon_\nu(\rho) = \{X \mid X \subseteq \nu \circ \rho(\Sigma)\}$.

Then we can make an equivalence between the elements of Υ_ν and the elements of ν, which helps the calculus.

All these abstractions are less precise than the "generic" upper abstraction, independent of ν: $\Upsilon_0(\rho) = \{X \mid X \subseteq \rho(\Sigma)\}$ This abstraction is represented on figure (3b).

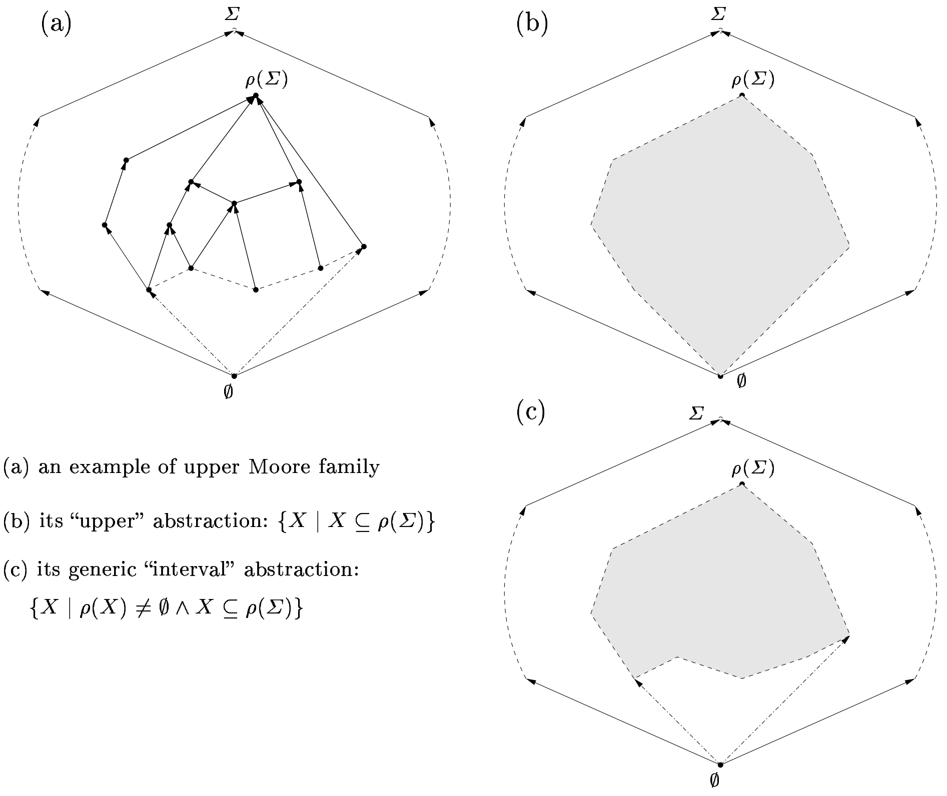
(a)
Σ
$\rho(\Sigma)$
$\emptyset$
(b)
Σ
$\rho(\Sigma)$
$\emptyset$
(c)
Σ
$\rho(\Sigma)$
$\emptyset$
(a) an example of upper Moore family
(b) its "upper" abstraction: $\{X \mid X \subseteq \rho(\Sigma)\}$
(c) its generic "interval" abstraction:
$\{X \mid \rho(X) \neq \emptyset \wedge X \subseteq \rho(\Sigma)\}$

Fig. 3. Illustration of a lower closure operator as a Moore family (a), its generic "upper" abstraction (b) and "interval" abstraction (c).

4.2 "Interval" abstraction

The "interval" abstraction is a combination of an "upper" abstraction and a "lower" one. The "upper" abstraction was presented before. The "lower" would give properties on the lowest elements of the lower closure operator ρ. However, the lowest element of ρ is simply $\emptyset$. Thus, we will try to give properties on the "lower part" of $\rho\backslash\{\emptyset\}$. On the other hand, we forget what is between this "lower part" and the maximum $\rho(\Sigma)$: all elements are possible fixpoints.

Our "generic" abstraction is then:

$$\Upsilon_0(\rho) = \{\emptyset\} \cup \{X \mid \exists (Y, Z) \in \rho^2, Y \neq \emptyset \wedge Y \subseteq X \subseteq Z\}$$

An illustration of this abstraction is given Fig. (3c).

Described with closure operators, we see that this abstraction keeps $\rho^{-1}(\{\emptyset\})$ (i.e. $\rho^{-1}(\{\emptyset\}) = (\Upsilon_0(\rho))^{-1}(\{\emptyset\})$):

$$\Upsilon_0(\rho) = \lambda X. \begin{cases} \emptyset & \text{if } X \notin \{Y \mid \rho(Y) \neq \emptyset\} \\ X \cap \rho(\Sigma) & \text{otherwise} \end{cases}$$

With this definition, $\Upsilon_0(\rho)$ is represented by an element of $\wp(\Sigma)$ and an element of $\wp(\wp(\Sigma))$. The first one, $\rho(\Sigma)$, can be easily abstracted. However, the second one ($\{Y \mid \rho(Y) \neq \emptyset\}$) is difficult to approximate. A possible approach would be to choose a (lower) approximation of $\bigcap\{Y \mid \rho(Y) \neq \emptyset\}$. In practical cases, this approach often gives $\emptyset$. Thus, we propose another alternative, which consists in abstracting each element of the set before intersecting them in the abstract domain. The more the abstract intersection keeps informations, the better this alternative will be. Following this principle, we will use a Galois connection for this abstraction: we do not require the abstraction function to be surjective.

Starting from an abstraction of $\wp(\Sigma)$ described as a Galois connection $\wp(\Sigma) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \Sigma^\sharp$ ($\Sigma^\sharp$ being a complete lattice), we define $\Upsilon_{\alpha,\gamma} \in \mathit{uco}(\mathit{lco}(\wp(\Sigma)))$ as:

$$\Upsilon_{\alpha,\gamma}(\rho) = \lambda X. \begin{cases} \emptyset & \text{if } \alpha(X) \sqsubset^\sharp \sqcap^\sharp\{\alpha(Y) \mid Y \in \rho\} \\ X \cap \gamma \circ \alpha(\rho(\Sigma)) & \text{otherwise} \end{cases} \tag{3}$$

We can remark that all elements of $\Upsilon_{\alpha,\gamma}$ can be represented by an element of $\Sigma^\sharp \times \Sigma^\sharp$:

Theorem 2. *$\Upsilon_{\alpha,\gamma} \in \mathit{uco}(\mathit{lco}(\wp(\Sigma)))$, and $\Upsilon_{\alpha,\gamma} \sqsubseteq \Upsilon_0$. Furthermore, $\Upsilon_{\alpha,\gamma}$ is associated to the Galois connection:*

$$(\mathit{lco}(\wp(\Sigma)), \sqsubseteq) \underset{\alpha^\bullet}{\overset{\gamma^\bullet}{\leftrightarrows}} (\Sigma^\sharp, \sqsupseteq^\sharp) \times (\Sigma^\sharp, \sqsubseteq^\sharp)$$
$$\alpha^\bullet(\rho) = (\sqcap^\sharp\{\alpha(Y) \mid Y \in \rho\}, \alpha(\rho(\Sigma)))$$
$$\gamma^\bullet(l,u) = \{X \mid l \sqsubseteq^\sharp \alpha(X) \sqsubseteq^\sharp u\}$$

Remark 2. If ρ is generated by $\mathcal{A}$, then

$$\alpha^\bullet(\rho) = (\sqcap^\sharp\{\alpha(Y) \mid Y \in \mathcal{A}\}, \alpha(\cup\mathcal{A}))$$

Thus we do not need to compute the whole Moore family to get the abstract element.

Example 7. We choose $\Sigma = \mathbb{Z}$ and $\Sigma^\sharp = \mathbb{Z}^\infty \times \mathbb{Z}^\infty$, with $\alpha(X) = (\min X, \max X)$ and $\gamma(m, M) = \{i \mid m \leq i \leq M\}$ (this is the numerical interval domain [2], except that we do not restrict this domain to "true" intervals, where $m \leq M$).

If ρ is generated by $\{\{n, n+3\} \mid n \geq 1\}$, then $\alpha^\bullet(\rho) = ((+\infty, 4), (1, +\infty))$, and

$$\Upsilon_{\alpha,\gamma}(\rho) = \lambda X. \begin{cases} \emptyset & \text{if } \max X < 4 \\ X \cap [1, +\infty[& \text{otherwise} \end{cases}$$

We can see that this approximation is more precise than the "upper-only" abstraction used in the example (5).

Remark 3. As we can see in the example above, using a Galois connection for the abstraction of $\wp(\Sigma)$ is very important: we can use a larger abstract domain $\Sigma^\sharp$, with α non-surjective, to obtain a better precision. On the contrary, using the classical interval domain for $\Sigma^\sharp$ (where all elements (m, M) with $m > M$ are collapsed into $\perp$) would give $(\perp, (1, +\infty))$ for $\alpha^\bullet(\rho)$, which is less precise. Since all closure operators induce a surjective Galois connection, we can not use this framework here.

5 Conclusion

We have shown how lower closure operators can be used in verification in the abstract interpretation framework. Starting from a condition on the final result, we derive a new analysis which can be combined with the initial one. The news semantics is a refinement of the original one, stable for the property we want to check, so the abstract semantics will be more precise. The work of Giacobazzi and *al.* about completion of abstractions is used to get the concrete description of the new analysis and to prove its general correctness.

The main point of our contribution is the possibility of abstracting this new analysis to compute automatically an abstract lower closure operator which can be used in the original abstract analysis. Lower closure operators are needed because they can be over-approximated while keeping the correctness of the (upper) analysis, something which would not work with upper closure operators. Thus, the efficiency of this approach relies mainly on the efficiency of the abstractions on $lco(D)$.

This paper showed examples of abstractions on $lco(D)$, but many other abstractions can be developed. For example, we can construct new abstractions from existing one by using the structure of D (as a Cartesian product, or as a lattice of functions). This work is essential for the design of a real analyzer which would use this approach.

Acknowledgments I wish to thank the anonymous referees, as well as Radhia and Patrick Cousot and Francesco Logozzo for their helpful comments and suggestions.

References

1. P. Cousot. *Méthodes itératives de construction et d'approximation de point fixes d'opérateurs monotones sur un treillis, analyse sémantique des programmes.* Thèse ès sciences mathématiques, University of Grenoble, March 1978.
2. P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. ACM Press, New York, NY.

3. P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Conference Record of the Sixth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 269–282, San Antonio, Texas, 1979. ACM Press, New York, NY.
4. P. Cousot and R. Cousot. Refining model checking by abstract interpretation. *Automated Software Engineering*, 6(1):69–95, 1999.
5. P. Cousot and R. Cousot. Software analysis and model checking. In E. Brinksma and K.G. Larsen, editors, *Proceedings of the 14th International Conference on Computer Aided Verification, CAV 2002*, Copenhagen, Denmark, LNCS 2404, pages 37–56. Springer-Verlag Berlin Heidelberg, 27–31 July 2002.
6. R. Giacobazzi, F. Ranzato, and F. Scozzari. Making abstract interpretations complete. *Journal of the ACM*, 47(2):361–416, 2000.
7. P. Granger. Improving the results of static analyses of programs by local decreasing iterations. In R. K. Shyamasundar, editor, *Foundations of Software Technology and Theoretical Computer Science, 12th conference, New Dehli, India*, volume 652 of *Lecture Notes in Computer Science*, pages 68–79. Springer-Verlag, 1992.
8. D. Massé. Combining backward and forward analyses of temporal properties. In O. Danvy and A. Filinski, editors, *Proceedings of the Second Symposium PADO'2001, Programs as Data Objects*, volume 2053 of *Lecture Notes in Computer Sciences*, pages 155–172, Århus, Denmark, 21 – 23 May 2001. Springer-Verlag, Berlin, Germany.
9. D. Massé. Semantics for abstract interpretation-based static analyzes of temporal properties. In M. Hermenegildo, editor, *Proceedings of the Ninth Static Analysis Symposium SAS'02*, volume 2477 of *Lecture Notes in Computer Sciences*, pages 428 – 443, Madrid, Spain, 17 – 20 September 2002. Springer-Verlag, Berlin, Germany.