Higher-order abstract interpretation (and application to comportment analysis generalizing strictness...

Conference Paper · June 1994

DOI: 10.1109/ICCL.1994.288389 · Source: IEEE Xplore

2 authors:

Patrick Cousot
New York University
172 PUBLICATIONS 14,800 CITATIONS
SEE PROFILE

Radhia Cousot
Ecole Normale Supérieure de Paris
99 PUBLICATIONS 11,606 CITATIONS
SEE PROFILE

Higher-Order Abstract Interpretation (and Application to Comportment Analysis Generalizing Strictness, Termination, Projection and PER Analysis of Functional Languages)

Invited paper

Patrick Cousot

LIENS — DMI — École Normale Supérieure
75230 Paris cedex 05 (France)
cousot@dmi.ens.fr

Radhia Cousot

LIX — École Polytechnique
91128 Palaiseau cedex (France)
radhia@poly.polytechnique.fr

abstract

The original formulation of abstract interpretation [12, 13, 14, 16] represents program properties by sets. A property is understood as the set of semantic values satisfying it. Strongest program properties are defined by the collecting semantics which extends the standard semantics to powersets of semantic values. The approximation relation corresponding to the logical implication of program properties is subset inclusion. This was expressed using set and lattice theory in the context of transition systems [16] that is of an operational semantics. This approach was applied to imperative programs [14], first-order procedures [15], communicating processes [17], parallel [18] and logic [19] programs.

Some applications of abstract interpretation, such as strictness analysis for lazy functional languages [10, 54], require infinite behaviors of higher-order functions to be taken into account. In this context denotational semantics is very natural (strictness is $f(\perp) = \perp$ where $\perp$ denotes non-termination). The set-theoretic approach to abstract interpretation was felt incompatible with denotational semantics. The attempts to express the collecting semantics in denotational form were unsuccessful [3] since properties of functions $f \in D^1 \mapsto D^2$ had to be expressed as continuous functions between powerdomains $F \in \mathsf{P}D^1 \mapsto \mathsf{P}D^2$ which is not expressive enough.

We solve the problem by returning to the sources of abstract interpretation, which consists in considering collecting semantics such that e.g. properties of functions $f \in D^1 \mapsto D^2$ are sets of functions $F \in \wp(D^1 \mapsto D^2)$. Various Galois connection based approximations of $F \in \wp(D^1 \mapsto D^2)$ can then be applied. By using Galois connections, properties of the standard semantics naturally transfer to the collecting and then to the abstract semantics.

This set-theoretic abstract interpretation framework is formulated in a way which is independent of both the programming language and the method used to specify its semantics. It is illustrated for a higher-order monomorphically typed lazy functional language starting from its standard denotational semantics. The chosen application is comportment analysis which generalizes strictness, termination, projection (including absence) [64], dual projection (including totality) and PER analysis [41] and is expressed in denotational style.

Part I : Higher-Order Abstract Interpretation

1: Principles of abstract interpretation

In the context of program analysis, abstract interpretation consists in answering questions about programs by approximation of a collecting semantics expressing program properties relative to a standard semantics [12, 13, 14, 16].

1.1: Collecting semantics

For example, the *collecting semantics* $\{\!|p|\!\} \in \wp(\mathcal{D})$ of program p is a set $\{[\![p]\!]\iota \mid \iota \in I\} \subseteq \mathcal{D}$ of possible output values (in the set $\mathcal{D}$ of *concrete values*) corresponding to a given set I of possible input values, as defined by the *standard semantics* $[\![p]\!]$.

1.2: Questions about programs

Concrete questions asked about program p have the form "$\{\!|p|\!\} \subseteq R$?" where the set $R \in \mathcal{P}$ of desired results is a *concrete property* of $\mathcal{P} \overset{\text{def}}{=} \wp(\mathcal{D})$ which is a complete lattice $\langle \mathcal{P};\ \subseteq,\ \emptyset,\ \Upsilon,\ \cup,\ \cap \rangle$ with $\Upsilon = \mathcal{D}$.

1.3: Approximation ordering

Question Q is said to be *more precise than* Q' or Q' *is an approximation of* Q if and only if $Q \subseteq Q'$. The partial order $\subseteq$ is called the *approximation ordering*. Observe that the collecting semantics $\{\!|p|\!\}$ is the most precise question which can be answered about program p. The approximation ordering is a logical ordering corresponding to implication which is totally unrelated with any relation between semantic values.

1074-8970/94 $3.00 © 1994 IEEE

1.4: Abstract semantics

The collecting semantics $\{\!|p|\!\}$ is not computable, so that an *abstract semantics* $(\!|p|\!) \in \mathcal{P}_a$ can be used instead. The set $\mathcal{P}_a$ of *abstract properties* is a complete lattice $\langle \mathcal{P}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a \rangle$.

1.5: Connecting the collecting and abstract semantics

The correspondence between concrete and abstract properties is given by means of a *Galois connection*[1]:

$$\langle \mathcal{P};\ \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \mathcal{P}_a;\ \subseteq_a \rangle$$

that is a pair of functions:

$$\alpha \quad \in \quad \mathcal{P} \mapsto \mathcal{P}_a \qquad\qquad \gamma \quad \in \quad \mathcal{P}_a \mapsto \mathcal{P}$$

satisfying:

$$\begin{array}{c} \forall Q \in \mathcal{P} : \forall Q_a \in \mathcal{P}_a : \\ \alpha(Q) \subseteq_a Q_a \iff Q \subseteq \gamma(Q_a) \end{array} \tag{1}$$

or equivalently:

$$\begin{array}{ll} \multicolumn{2}{l}{\forall Q,\ Q' :\in \mathcal{P},\ \forall Q_a,\ Q'_a \in \mathcal{P}_a :} \\ \alpha \text{ monotone:} & (Q \subseteq Q') \Rightarrow (\alpha(Q) \subseteq_a \alpha(Q')) \\ \gamma \text{ monotone:} & (Q_a \subseteq_a Q'_a) \Rightarrow (\gamma(Q_a) \subseteq \gamma(Q'_a)) \\ \gamma \circ \alpha \text{ extensive:} & Q \subseteq \gamma(\alpha(Q)) \\ \alpha \circ \gamma \text{ reductive:} & \alpha(\gamma(Q_a)) \subseteq_a Q_a \end{array} \tag{2}$$

1.6: Best approximation

The only considered properties are now of the form $\gamma(Q_a)$ where $Q_a \in \mathcal{P}_a$ is an *abstract property*. Q_a is said to be *more precise* than Q'_a if and only if $\gamma(Q_a) \subseteq \gamma(Q'_a)$. Let us call an *approximation* of a concrete property Q any abstract property Q_a such that $Q \subseteq \gamma(Q_a)$. The interest of Galois connections is that $\alpha(Q)$ is the *best approximation* of Q (it is an approximation by $Q \subseteq \gamma(\alpha(Q))$ in (2) and $\alpha(Q)$ is more precise than any other approximation Q_a since $Q \subseteq \gamma(Q_a)$ implies $\alpha(Q) \subseteq_a Q_a$ by (1) so that $\gamma(\alpha(Q)) \subseteq \gamma(Q_a)$ by monotony).

1.7: Correctness and completeness of the abstract interpretation

Questions are now answered in the abstract form "$(\!|p|\!) \subseteq_a Q_a$?". This approach is *correct* whenever:

$$\forall Q_a \in \mathcal{P}_a : (\!|p|\!) \subseteq_a Q_a \Rightarrow \{\!|p|\!\} \subseteq \gamma(Q_a)$$

and *complete* whenever:

$$\forall Q_a \in \mathcal{P}_a : \{\!|p|\!\} \subseteq \gamma(Q_a) \Rightarrow (\!|p|\!) \subseteq_a Q_a$$

By the Galois connection property (1), any choice of $(\!|p|\!)$ such that $\alpha(\{\!|p|\!\}) \subseteq_a (\!|p|\!)$ is correct while $(\!|p|\!) \subseteq_a \alpha(\{\!|p|\!\})$ is complete.

1.8: Higher-order abstract interpretation

In order to lift this approach to higher-order, we have to provide methods for approximating a set of functions (corresponding e.g. to the collecting semantics of a function type) and a relation (corresponding e.g. to the collecting semantics of a pair type or e.g. to an ordering on values).

2: Abstraction of a set of functions

We now consider abstract interpretations of sets of functions in $\wp(\mathcal{D}^1 \mapsto \mathcal{D}^2)$ where $\mathcal{D}^1$ and $\mathcal{D}^2$ are sets for which abstract interpretations are available:

$$\begin{array}{l} \langle \wp(\mathcal{D}^i);\ \subseteq,\ \emptyset,\ \mathcal{D}^i,\ \cup,\ \cap \rangle \\ \qquad\qquad \underset{\alpha^i}{\overset{\gamma^i}{\leftrightarrows}} \\ \langle \mathcal{D}^i_a;\ \subseteq^i_a,\ \emptyset^i_a,\ \Upsilon^i_a,\ \cup^i_a,\ \cap^i_a \rangle \qquad i = 1,\ 2 \end{array} \tag{3}$$

2.1: Abstraction of a set of functions by a binary relation

A first abstraction consists in approximating a set F of functions $\{f_i \mid i \in \Delta\}$ by a relation r relating elements $\langle x,\ y \rangle$ which can be mapped by some function f_i in the set F: $f_i(x) = y$. Precisely which function f_i is ignored. We write $\mathcal{D}^1 \leftrightarrow \mathcal{D}^2$ for $\wp(\mathcal{D}^1 \times \mathcal{D}^2) = \{\langle x,\ y \rangle \mid x \in \mathcal{D}^1 \wedge y \in \mathcal{D}^2\}$. We define:

$$\begin{array}{rcl} \alpha^\varrho(F) & \stackrel{\text{def}}{=} & \{\langle x,\ f(x) \rangle \mid x \in \mathcal{D}^1 \wedge f \in F\} \\ \gamma^\varrho(r) & \stackrel{\text{def}}{=} & \{f \in D^1 \mapsto D^2 \mid \forall x \in \mathcal{D}^1 : \langle x,\ f(x) \rangle \in r\} \end{array}$$

so that we have the Galois connection:

$$\begin{array}{c} \langle \wp(\mathcal{D}^1 \mapsto \mathcal{D}^2);\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \mapsto \mathcal{D}^2,\ \cup,\ \cap \rangle \\ \underset{\alpha^\varrho}{\overset{\gamma^\varrho}{\leftrightarrows}} \\ \langle \mathcal{D}^1 \leftrightarrow \mathcal{D}^2;\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \times \mathcal{D}^2,\ \cup,\ \cap \rangle \end{array}$$

2.2: Binary relations as set-valued functions

Once a set of functions has been approximated by a binary relation, we are left with the problem of approximating this relation with respect to the approximation ordering. We first consider two isomorphic representations of binary relation by functions and then their approximation.

[1] Évariste Galois introduced such "correspondences" as the basis of his criterion for solvability of a polynomial equation of degree ≥ 5 by radicals and for the constructibility by straight-edge and compass. If E is a given field then let $\text{Inv } G \stackrel{\text{def}}{=} \{a \in E \mid \exists \eta \in G : \eta(a) = a\}$ for a group G of automorphisms in E. The *Galois group* Gal E/F of E over a subfield F is the set of automorphisms η of E such that $\eta(a) = a$ for every $a \in F$. The maps $\alpha(F) = \text{Gal } E/F$ and $\gamma(F) = \text{Gal } E/F$ are such that:

$$\begin{array}{l} (F_1 \subseteq F_2) \Rightarrow (\alpha(F_1) \supseteq \alpha(F_2)) \\ (G_1 \supseteq G_2) \Rightarrow (\gamma(G_1) \subseteq \gamma(G_2)) \\ F \subseteq \gamma(\alpha(F)) \\ \alpha(\gamma(G)) \supseteq G \end{array}$$

which, as remarked in [16], corresponds to (2), but for the use of the dual ordering $\supseteq\ =\ \subseteq_a$, hence more precisely to the residuated mappings of P. Dubreuil and R. Croisot [23, 28]. The idea of using Galois connection in the context of order theory is in [31, 61] and, implicitly, in [6].

Pointwise coding: There are many possible codings of a relation by a function. A first one is the *pointwise coding* into a function mapping elements to their images under the relation:

$$\begin{aligned}\alpha^{\varpi}(r) &\stackrel{\text{def}}{=} \lambda x \bullet \{y \mid \langle x,\ y\rangle \in r\}\\ \gamma^{\varpi}(\phi) &\stackrel{\text{def}}{=} \{\langle x,\ y\rangle \mid y \in \phi(x)\}\end{aligned}$$

$$\begin{gathered}\langle \mathcal{D}^1 \leftrightarrow \mathcal{D}^2;\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \times \mathcal{D}^2,\ \cup,\ \cap\rangle\\ \underset{\alpha^{\varpi}}{\overset{\gamma^{\varpi}}{\substack{\twoheadleftarrow\\ \twoheadrightarrow}}}\\ \langle \mathcal{D}^1 \mapsto \wp(\mathcal{D}^2);\ \dot{\subseteq},\ \lambda x \bullet \emptyset,\ \lambda x \bullet \mathcal{D}^2,\ \dot{\cup},\ \dot{\cap}\rangle\end{gathered}$$

The arrow $\twoheadleftarrow$ indicates that in the Galois connection γ^{ϖ} is surjective or equivalently that α^{ϖ} is injective. The arrow $\twoheadrightarrow$ indicates that α^{ϖ} is surjective or equivalently that α^{ϖ} is injective. Here we have an order isomorphism which is a special case of Galois connection ($\alpha^{\varpi} \circ \gamma^{\varpi}$ and $\gamma^{\varpi} \circ \alpha^{\varpi}$ are the identity). Another *inverse pointwise coding* would consist in using the pointwise coding for the inverse relation.

Set-transformer coding: A second equivalent coding is *set-transformer coding*. The relation is coded by a set-transformer mapping sets to their images under the relation. Such set-transformers are *complete union-morphisms* i.e. $f \in \wp(\mathcal{D}^1) \xmapsto{\cup} \wp(\mathcal{D}^2)$ such that $\bigcup_{x\in X} f(\{x\}) = f(\bigcup_{x\in X}\{x\})\ (= f(X))$:

$$\alpha^{\varsigma}(r) \stackrel{\text{def}}{=} \lambda X \bullet \{y \mid \exists x \in X : \langle x,\ y\rangle \in r\} \tag{4}$$

$$\gamma^{\varsigma}(\Phi) \stackrel{\text{def}}{=} \{\langle x,\ y\rangle \mid y \in \Phi(\{x\})\} \tag{5}$$

$$\begin{gathered}\langle \mathcal{D}^1 \leftrightarrow \mathcal{D}^2;\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \times \mathcal{D}^2,\ \cup,\ \cap\rangle\\ \underset{\alpha^{\varsigma}}{\overset{\gamma^{\varsigma}}{\substack{\twoheadleftarrow\\ \twoheadrightarrow}}}\\ \langle \wp(\mathcal{D}^1) \xmapsto{\cup} \wp(\mathcal{D}^2);\ \dot{\subseteq},\ \lambda X \bullet \emptyset,\ \lambda X \bullet \mathcal{D}^2,\ \dot{\cup},\ \dot{\cap}\rangle\end{gathered}$$

Observe that this coding is familiar when the relation r is a function f (in which case $\langle x,\ y\rangle \in r$ and $\langle x,\ y'\rangle \in r$ imply $y = y' = f(x)$), since $\alpha^{\varsigma}(r) = \lambda X \bullet \{f(x) \mid x \in X\}$ is the usual extension of functions on elements to functions on sets of elements. Another *inverse set-transformer coding* would be relative to the inverse relation.

2.3: Abstraction of a set-valued function

Pointwise abstraction of a set-valued function: The approximation of a set-valued function in $\mathcal{D}^1 \mapsto \wp(\mathcal{D}^2)$ can be done using a *pointwise abstraction* (with no loss of information on $\mathcal{D}^1$ and approximation on $\wp(\mathcal{D}^2)$ only), as follows:

$$\begin{aligned}\alpha^{\pi}(\phi) &\stackrel{\text{def}}{=} \lambda x \bullet \alpha^2(\phi(x))\\ \gamma^{\pi}(\Phi) &\stackrel{\text{def}}{=} \lambda x \bullet \{y \mid y \in \gamma^2(\Phi(x))\}\end{aligned}$$

$$\begin{gathered}\langle \mathcal{D}^1 \mapsto \wp(\mathcal{D}^2);\ \dot{\subseteq},\ \lambda x \bullet \emptyset,\ \lambda x \bullet \mathcal{D}^2,\ \dot{\cup},\ \dot{\cap}\rangle\\ \underset{\alpha^{\pi}}{\overset{\gamma^{\pi}}{\leftrightarrows}}\\ \langle \mathcal{D}^1 \mapsto \mathcal{D}^2_a;\ \dot{\subseteq}^2_a,\ \lambda x \bullet \emptyset^2_a,\ \lambda x \bullet \Upsilon^2_a,\ \dot{\cup}^2_a,\ \dot{\cap}^2_a\rangle\end{gathered}$$

Functional abstraction of a set-transformer: A set-transformer in $\wp(\mathcal{D}^1) \xmapsto{\cup} \wp(\mathcal{D}^2)$, which is a complete union-morphism hence $\emptyset$-strict ($f(\emptyset) = \emptyset$) and set-inclusion monotonic ($X \subseteq Y \Rightarrow f(X) \subseteq f(Y)$), can be approximated by a $\emptyset$-strict and monotonic function on abstract values (with loss of information both on $\wp(\mathcal{D}^1)$ and $\wp(\mathcal{D}^2)$) using the following *set-transformer abstraction* [12, 13, 14, 16]:

$$\begin{aligned}\alpha^{\varphi}(\Phi) &\stackrel{\text{def}}{=} \alpha^2 \circ \Phi \circ \gamma^1\\ \gamma^{\varphi}(\Psi) &\stackrel{\text{def}}{=} \gamma^2 \circ \Psi \circ \alpha^1\end{aligned}$$

$$\begin{gathered}\langle \wp(\mathcal{D}^1) \xmapsto{\emptyset,\subseteq} \wp(\mathcal{D}^2);\ \dot{\subseteq},\ \lambda X \bullet \emptyset,\ \lambda X \bullet \mathcal{D}^2,\ \dot{\cup},\ \dot{\cap}\rangle\\ \underset{\alpha^{\varphi}}{\overset{\gamma^{\varphi}}{\leftrightarrows}}\\ \langle \mathcal{D}^1_a \xmapsto{\emptyset,\subseteq} \mathcal{D}^2_a;\ \dot{\subseteq}^2_a,\ \lambda A \bullet \emptyset^2_a,\ \lambda A \bullet \Upsilon^2_a,\ \dot{\cup}^2_a,\ \dot{\cap}^2_a\rangle\end{gathered} \tag{6}$$

3: Compositional abstraction

The composition of Galois connections $\langle\alpha^a,\ \gamma^a\rangle$:

$$\langle \wp(\mathcal{D});\ \subseteq,\ \emptyset,\ \mathcal{D},\ \cup,\ \cap\rangle \underset{\alpha^a}{\overset{\gamma^a}{\leftrightarrows}} \langle \mathcal{D}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a\rangle$$

and $\langle\alpha^b,\ \gamma^b\rangle$:

$$\langle \mathcal{D}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a\rangle \underset{\alpha^b}{\overset{\gamma^b}{\leftrightarrows}} \langle \mathcal{D}_b;\ \subseteq_b,\ \emptyset_b,\ \Upsilon_b,\ \cup_b,\ \cap_b\rangle$$

is a Galois connection $\langle\alpha^b,\ \gamma^b\rangle \circ \langle\alpha^a,\ \gamma^a\rangle$:

$$\begin{gathered}\langle \wp(\mathcal{D});\ \subseteq,\ \emptyset,\ \mathcal{D},\ \cup,\ \cap\rangle\\ \underset{\alpha^b \circ \alpha^a}{\overset{\gamma^a \circ \gamma^b}{\leftrightarrows}}\\ \langle \mathcal{D}_b;\ \subseteq_b,\ \emptyset_b,\ \Upsilon_b,\ \cup_b,\ \cap_b\rangle\end{gathered} \tag{7}$$

It follows that an abstract interpretation can be designed *compositionally* by composition of successive abstractions. For example we consider two possible abstractions of sets of functions by an abstract function.

Pointwise abstraction of a set of functions: A set of functions in $\wp(\mathcal{D}^1 \mapsto \mathcal{D}^2)$ can be approximated pointwise without loss of information on the domain $\mathcal{D}^1$ and abstraction on the co-domain $\mathcal{D}^2$ only:

$$\begin{aligned}\alpha^{\Pi} &\stackrel{\text{def}}{=} \alpha^{\pi} \circ \alpha^{\varpi} \circ \alpha^{\varrho}\\ &= \lambda F \bullet \lambda x \bullet \alpha^2(\{f(x) \mid x \in \mathcal{D}^1 \wedge f \in F\})\\ \gamma^{\Pi} &\stackrel{\text{def}}{=} \gamma^{\varrho} \circ \gamma^{\varpi} \circ \gamma^{\pi}\\ &= \lambda F \bullet \{f \mid \forall x : f(x) \in \gamma^2(\Phi(x))\}\end{aligned}$$

$$\begin{gathered}\langle \wp(\mathcal{D}^1 \mapsto \mathcal{D}^2);\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \mapsto \mathcal{D}^2,\ \cup,\ \cap\rangle\\ \underset{\alpha^{\Pi}}{\overset{\gamma^{\Pi}}{\leftrightarrows}}\\ \langle \mathcal{D}^1 \mapsto \mathcal{D}^2_a;\ \dot{\subseteq}^2_a,\ \dot{\emptyset}^2_a,\ \dot{\Upsilon}^2_a,\ \dot{\cup}^2_a,\ \dot{\cap}^2_a\rangle\end{gathered} \tag{8}$$

Functional abstraction of a set of functions: A coarser approximation of a set of functions in $\wp(\mathcal{D}^1 \mapsto \mathcal{D}^2)$ is by abstraction as a set transformer and then on both the domain $\mathcal{D}^1$ and on the co-domain $\mathcal{D}^2$:

$$\begin{array}{rcl}
\alpha^{\phi} & \stackrel{\mathrm{def}}{=} & \alpha^{\varphi} \circ \alpha^{\varsigma} \circ \alpha^{\varrho} \\
 & = & \lambda F \bullet \lambda X \bullet \alpha^2(\{f(x) \mid x \in \gamma^1(X) \wedge f \in F\}) \\
\gamma^{\phi} & \stackrel{\mathrm{def}}{=} & \gamma^{\varrho} \circ \gamma^{\varsigma} \circ \gamma^{\varphi} \\
 & = & \lambda \Psi \bullet \{f \mid \forall x : f(x) \in \gamma^2 \circ \Psi \circ \alpha^1(\{x\})\}
\end{array}$$

$$\begin{array}{c}
\langle \wp(\mathcal{D}^1 \mapsto \mathcal{D}^2);\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \mapsto \mathcal{D}^2,\ \cup,\ \cap\rangle \\
\underset{\alpha^{\phi}}{\overset{\gamma^{\phi}}{\leftrightarrows}} \\
\langle \mathcal{D}^1_a \xrightarrow{\emptyset,\subseteq} \mathcal{D}^2_a;\ \dot{\subseteq}^2_a,\ \dot{\emptyset}^2_a,\ \dot{\Upsilon}^2_a,\ \dot{\cup}^2_a,\ \dot{\cap}^2_a\rangle
\end{array} \qquad (9)$$

4: Abstraction of a binary relation

We now consider abstract interpretations of relations in $\mathcal{D}^1 \leftrightarrow \mathcal{D}^2$ where $\mathcal{D}^1$ and $\mathcal{D}^2$ are sets for which abstract interpretations (3) are available. Observe that by the isomorphisms between binary relations and set-valued functions (Sect. 2.2) and set-transformers (Sect. 2.2), we can already use the abstractions given in Sect. 2.

4.1: Relations on elements as relations on sets

Corresponding to the extension of a function on elements to a function on sets of elements (by the functional set-transformer of Sect. 2.2), a relation on elements can be coded by a relation on sets of elements:

$$\begin{array}{rcl}
\downarrow^r Y & \stackrel{\mathrm{def}}{=} & \{x \in \mathcal{D}^1 \mid \exists y \in Y : \langle x,\ y\rangle \in r\} \\
\uparrow^r X & \stackrel{\mathrm{def}}{=} & \{y \in \mathcal{D}^2 \mid \exists x \in X : \langle x,\ y\rangle \in r\} \\
\alpha^{\rightharpoonup}(r) & \stackrel{\mathrm{def}}{=} & \{\langle X,\ Y\rangle \in \wp(\mathcal{D}^1) \leftrightarrow \wp(\mathcal{D}^2) \mid X \subseteq \downarrow^r Y\} \\
\alpha^{\leftharpoonup}(r) & \stackrel{\mathrm{def}}{=} & \{\langle X,\ Y\rangle \in \wp(\mathcal{D}^1) \leftrightarrow \wp(\mathcal{D}^2) \mid Y \subseteq \uparrow^r X\} \\
\alpha^{\star}(r) & \stackrel{\mathrm{def}}{=} & \alpha^{\leftharpoonup}(r)\ \cap\ \alpha^{\rightharpoonup}(r) \\
\gamma^{\star}(R) & \stackrel{\mathrm{def}}{=} & \{\langle x,\ y\rangle \mid \langle\{x\},\ \{y\}\rangle \in R\}
\end{array}$$

The same way that not all functions on sets are set-transformers (they must be complete union-morphisms hence $\emptyset$-strict), not all relations between sets are set relators. Therefore we define:

$$\begin{array}{l}
\wp(\mathcal{D}^1) \overset{\emptyset}{\leftrightarrow} \wp(\mathcal{D}^2) \stackrel{\mathrm{def}}{=} \\
\quad \{R \in \wp(\mathcal{D}^1) \leftrightarrow \wp(\mathcal{D}^2) \mid \\
\qquad \forall X \in \wp(\mathcal{D}^1) : (\langle X,\ \emptyset\rangle \in R) \iff (X = \emptyset) \wedge \\
\qquad \forall Y \in \wp(\mathcal{D}^2) : (\langle \emptyset,\ Y\rangle \in R) \iff (Y = \emptyset)\}
\end{array}$$

$$\begin{array}{l}
\wp(\mathcal{D}^1) \overset{\cup}{\leftrightarrow} \wp(\mathcal{D}^2) \stackrel{\mathrm{def}}{=} \\
\quad \{R \in \wp(\mathcal{D}^1) \leftrightarrow \wp(\mathcal{D}^2) \mid \\
\qquad \forall \{\langle X_i,\ Y_i\rangle \mid i \in \Delta\} \subseteq R : \langle \bigcup\limits_{i \in \Delta} X_i,\ \bigcup\limits_{i \in \Delta} Y_i\rangle \in R\}
\end{array}$$

$$\begin{array}{l}
\wp(\mathcal{D}^1) \overset{\emptyset,\cup}{\leftrightarrow} \wp(\mathcal{D}^2) \stackrel{\mathrm{def}}{=} \\
\quad \wp(\mathcal{D}^1) \overset{\emptyset}{\leftrightarrow} \wp(\mathcal{D}^2)\ \cap\ \wp(\mathcal{D}^1) \overset{\cup}{\leftrightarrow} \wp(\mathcal{D}^2)
\end{array}$$

so that we have the Galois connection:

$$\begin{array}{c}
\langle \mathcal{D}^1 \leftrightarrow \mathcal{D}^2;\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \times \mathcal{D}^2,\ \cup,\ \cap\rangle \\
\underset{\alpha^{\star}}{\overset{\gamma^{\star}}{\leftrightarrows}} \\
\langle \wp(\mathcal{D}^1) \overset{\emptyset,\cup}{\leftrightarrow} \wp(\mathcal{D}^2);\ \subseteq,\ \emptyset^{\star},\ \Upsilon^{\star},\ \cup,\ \cap\rangle
\end{array} \qquad (10)$$

where:

$$\begin{array}{rcl}
\emptyset^{\star} & \stackrel{\mathrm{def}}{=} & \{\langle \emptyset,\ \emptyset\rangle\} \\
\Upsilon^{\star} & \stackrel{\mathrm{def}}{=} & \emptyset^{\star} \cup (\wp(\mathcal{D}^1) \setminus \{\emptyset\}) \times (\wp(\mathcal{D}^2) \setminus \{\emptyset\})
\end{array}$$

The above connection is useful in conjunction with (4) to extend a relation defined for the standard semantics to a corresponding relation for the collecting semantics:

$$\begin{array}{l}
\forall f \in \mathcal{D} \mapsto \mathcal{D} : \forall r \in \mathcal{D} \leftrightarrow \mathcal{D} : \\
\quad \forall \langle x,\ y\rangle \in \mathcal{D} \times \mathcal{D} : \langle x,\ y\rangle \in r \Rightarrow \langle f(x),\ f(y)\rangle \in r \\
\Leftrightarrow \\
\quad \forall \langle X,\ Y\rangle \in \wp(\mathcal{D}) \times \wp(\mathcal{D}) : \\
\qquad \langle X,\ Y\rangle \in \alpha^{\star}(r) \Rightarrow \langle \alpha^{\varsigma}(f)(X),\ \alpha^{\varsigma}(f)(Y)\rangle \in \alpha^{\star}(r)
\end{array} \qquad (11)$$

Example 1 (Fixpoint inducing) $f \in \mathcal{D}^{\tau \mapsto \tau}$ is $\sqsubseteq^{\tau}$-monotonic whence by (11), $f^{\star} \stackrel{\mathrm{def}}{=} \alpha^{\varsigma}(f)$ is $\alpha^{\star}(\sqsubseteq^{\tau})$-preserving. $\langle \mathcal{D}^{\tau};\ \sqsubseteq^{\tau},\ \bot^{\tau},\ \sqcup^{\tau}\rangle$ is a poset so that $\langle \wp(\mathcal{D}^{\tau});\ \sqsubseteq^{\tau\star},\ \bot^{\tau\star},\ \sqcup^{\tau\star}\rangle$ is a preorder where $\sqsubseteq^{\tau\star} \stackrel{\mathrm{def}}{=} \alpha^{\star}(\sqsubseteq^{\tau})$, $\bot^{\tau\star} \stackrel{\mathrm{def}}{=} \{\bot^{\tau}\}$ and $\bigsqcup\limits_{i \in \Delta}^{\tau\star} X_i \stackrel{\mathrm{def}}{=} \{\bigsqcup\limits_{i \in \Delta}^{\tau} x_i \mid \forall i \in \Delta : x_i \in X_i\}$. $\bot^{\tau\star}$ is an infimum on $\wp(\mathcal{D}^{\tau}) \setminus \{\emptyset\}$. We have:

$$\mathrm{lfp}^{\star}\ f^{\star} \stackrel{\mathrm{def}}{=} \bigsqcup_{n \in \mathbb{N}}^{\tau\star} f^{\star n}(\bot^{\tau\star}) = \{\mathrm{lfp}\, f\} \qquad (12)$$

which is the least fixpoint on the poset $\langle \wp^{\star}(\mathcal{D}^{\tau});\ \sqsubseteq^{\tau\star}\rangle$ where $\wp^{\star}(\mathcal{D}^{\tau}) \stackrel{\mathrm{def}}{=} \wp(\mathcal{D}^{\tau})/{\equiv^{\star}} \stackrel{\mathrm{def}}{=} \{[X]^{\star}_{\equiv} \mid X \in \wp(\mathcal{D}^{\tau}) \setminus \{\emptyset\}\}$, $[X]^{\star}_{\equiv} \stackrel{\mathrm{def}}{=} \{Y \mid X \equiv^{\star} Y\}$ is the equivalence class of X for the equivalence relation $X \equiv^{\star} Y \stackrel{\mathrm{def}}{=} X \sqsubseteq^{\tau\star} Y \wedge Y \sqsubseteq^{\tau\star} X$ and $[X]^{\star}_{\equiv} \sqsubseteq^{\tau\star} [Y]^{\star}_{\equiv} \stackrel{\mathrm{def}}{=} X \sqsubseteq^{\tau\star} Y$. □

4.2: Abstraction of a relation on sets by a relation on abstract values

Using the abstractions (3) of sets of values in $\mathcal{D}^1$ and $\mathcal{D}^2$, one can abstract a set relator in $\wp(\mathcal{D}^1) \overset{\emptyset,\cup}{\leftrightarrow} \wp(\mathcal{D}^2)$:

$$\begin{array}{rcl}
\alpha^{\rho}(R) & \stackrel{\mathrm{def}}{=} & \{\langle x,\ y\rangle \mid \langle \gamma^1(x),\ \gamma^2(y)\rangle \in R\} \\
\gamma^{\rho}(r) & \stackrel{\mathrm{def}}{=} & \{\langle X,\ Y\rangle \mid \forall x : (X \in \gamma^1(x)) \Rightarrow \\
 & & \qquad (\exists y : Y \in \gamma^2(y) \wedge \langle x,\ y\rangle \in r)\}
\end{array}$$

$$\begin{array}{c}
\langle \wp(\mathcal{D}^1) \overset{\emptyset,\cup}{\leftrightarrow} \wp(\mathcal{D}^2);\ \subseteq,\ \emptyset^{\star},\ \Upsilon^{\star},\ \cup,\ \cap\rangle \\
\underset{\alpha^{\rho}}{\overset{\gamma^{\rho}}{\leftrightarrows}} \\
\langle \mathcal{D}^1_a \leftrightarrow \mathcal{D}^2_a;\ \subseteq,\ \emptyset^{\star}_a,\ \Upsilon^{\star}_a,\ \cup,\ \cap\rangle
\end{array} \qquad (13)$$

where:

$$\begin{array}{rcl}
\emptyset^{\star}_a & \stackrel{\mathrm{def}}{=} & \{\langle \emptyset^1_a,\ \emptyset^2_a\rangle\} \\
\Upsilon^{\star}_a & \stackrel{\mathrm{def}}{=} & \emptyset^{\star}_a \cup (\mathcal{D}^1_a \setminus \{\emptyset^1_a\}) \times (\mathcal{D}^2_a \setminus \{\emptyset^2_a\})
\end{array}$$

so that relator preserving set-transformers are approximated by abstract relation preserving abstract transformers:

$$\begin{array}{l}
\forall F \in \wp(\mathcal{D}) \xrightarrow{\emptyset,\subseteq} \wp(\mathcal{D} : \forall R \in \wp(\mathcal{D}) \overset{\emptyset,\cup}{\leftrightarrow} \wp(\mathcal{D}) : \\
\quad \forall \langle X,\ Y\rangle \in \wp(\mathcal{D}) \times \wp(\mathcal{D}) : \\
\qquad \langle X,\ Y\rangle \in R \Rightarrow \langle F(X),\ F(Y)\rangle \in R \\
\Leftrightarrow \\
\quad \forall \langle x,\ y\rangle \in \mathcal{D}_a \times \mathcal{D}_a : \\
\qquad \langle x,\ y\rangle \in \alpha^{\rho}(R) \Rightarrow \langle \alpha^{\varphi}(F)(x),\ \alpha^{\varphi}(F)(y)\rangle \in \alpha^{\rho}(R)
\end{array} \qquad (14)$$

Example 2 (Fixpoint inducing) Going on with Ex. 1, $\langle \wp(\mathcal{D}^\tau);\ \sqsubseteq^{\tau\star},\ \bot^{\tau\star},\ \sqcup^{\tau\star}\rangle$ is a pre-order so that $\langle \mathcal{D}^\tau_\mathcal{B};\ \sqsubseteq^\tau_\mathcal{B},\ \bot^\tau_\mathcal{B},\ \sqcup^\tau_\mathcal{B}\rangle$ is also a pre-order where $\sqsubseteq^\tau_\mathcal{B} \stackrel{\text{def}}{=} \alpha^\rho(\sqsubseteq^{\tau\star})$, $\bot^\tau_\mathcal{B} \stackrel{\text{def}}{=} \alpha^\tau(\bot^{\tau\star})$ and $\sqcup^\tau_{\mathcal{B}i\in\Delta}\ x_i \stackrel{\text{def}}{=} \alpha^\tau(\sqcup^{\tau\star}{}_{i\in\Delta}\,\gamma^\tau(x_i))$. By (11), $f_\mathcal{B} \stackrel{\text{def}}{=} \alpha^\varphi(f^\star)$ is $\sqsubseteq^{\tau\star}$-preserving. It has a least fixpoint (unique up to equivalence classes):

$$\mathrm{lfp}^\tau_\mathcal{B} f_\mathcal{B} \ \stackrel{\text{def}}{=}\ \bigsqcup^\tau_{\substack{\mathcal{B}\\ n\in\mathbb{N}}} {f_\mathcal{B}}^n(\bot^\tau_\mathcal{B}) \ =\ \{\mathrm{lfp}\, f\} \qquad (15)$$

□

4.3: Abstraction of a binary relation by a pair of sets

A relation can be approximated componentwise:

$$\begin{array}{rcl}
\alpha^\times(r) & \stackrel{\text{def}}{=} & \langle \Pi_1\, r,\ \Pi_2\, r\rangle \\
\Pi_1\, r & \stackrel{\text{def}}{=} & \{x \mid \exists y : \langle x,\ y\rangle \in r\} \\
\Pi_2\, r & \stackrel{\text{def}}{=} & \{y \mid \exists x : \langle x,\ y\rangle \in r\} \\
\gamma^\times(\langle X,\ Y\rangle) & \stackrel{\text{def}}{=} & X \times Y
\end{array}$$

$$\begin{array}{c}
\langle \mathcal{D}^1 \leftrightarrow \mathcal{D}^2;\ \subseteq,\ \emptyset,\ \mathcal{D}^1 \times \mathcal{D}^2,\ \cup,\ \cap\rangle \\
\underset{\alpha^\times}{\overset{\gamma^\times}{\leftrightarrows}} \\
\langle \wp(\mathcal{D}^1) \times \wp(\mathcal{D}^2);\ \subseteq^\times,\ \emptyset^\times,\ \Upsilon^\times,\ \cup^\times,\ \cap^\times\rangle
\end{array} \qquad (16)$$

where $\subseteq^\times \stackrel{\text{def}}{=} \subseteq \times \subseteq$, $\emptyset^\times \stackrel{\text{def}}{=} \langle\emptyset,\ \emptyset\rangle$, $\Upsilon^\times \stackrel{\text{def}}{=} \langle \mathcal{D}^1,\ \mathcal{D}^2\rangle$, $\cup^\times \stackrel{\text{def}}{=} \cup \times \cup$ and $\cap^\times \stackrel{\text{def}}{=} \cap \times \cap$.

4.4: Abstraction of a pair of sets by an abstract pair

In turn a pair $\langle X,\ Y\rangle \in \wp(\mathcal{D}^1) \times \wp(\mathcal{D}^2)$ of sets can be approximated by a pair of corresponding abstract values:

$$\begin{array}{rcl}
\alpha^\otimes(\langle X,\ Y\rangle) & \stackrel{\text{def}}{=} & \langle \alpha^1(X),\ \alpha^2(Y)\rangle \\
\gamma^\otimes(\langle x,\ y\rangle) & \stackrel{\text{def}}{=} & \langle \gamma^1(x),\ \gamma^2(y)\rangle
\end{array}$$

$$\begin{array}{c}
\langle \wp(\mathcal{D}^1) \times \wp(\mathcal{D}^2);\ \subseteq^\times,\ \emptyset^\times,\ \Upsilon^\times,\ \cup^\times,\ \cap^\times\rangle \\
\underset{\alpha^\otimes}{\overset{\gamma^\otimes}{\leftrightarrows}} \\
\langle \mathcal{D}^1_a \times \mathcal{D}^2_a);\ \subseteq^\otimes,\ \emptyset^\otimes,\ \Upsilon^\otimes,\ \cup^\otimes,\ \cap^\otimes\rangle
\end{array} \qquad (17)$$

where $\subseteq^\otimes \stackrel{\text{def}}{=} \subseteq^1_a \times \subseteq^2_a$, $\emptyset^\otimes \stackrel{\text{def}}{=} \langle \emptyset^1_a,\ \emptyset^2_a\rangle$, $\Upsilon^\otimes \stackrel{\text{def}}{=} \langle \Upsilon^1_a,\ \Upsilon^2_a\rangle$, $\cup^\otimes \stackrel{\text{def}}{=} \cup^1_a \times \cup^2_a$ and $\cap^\otimes \stackrel{\text{def}}{=} \cap^1_a \times \cap^2_a$.

5: Abstraction by partitioning

A common way of abstracting elements of $\wp(\mathcal{D})$ is by partitioning $\mathcal{D}$. A partition $\mathcal{P} \in \wp(\wp(\mathcal{D}))$ satisfies $\forall A, B \in \mathcal{P} : A \cap B = \emptyset$ and $\mathcal{D} = \cup_{B\in\mathcal{P}} B$. It can be defined e.g. by an equivalence relation $\equiv$ as $\mathcal{P} = \{[x]_\equiv \mid x \in \mathcal{D}\}$. A subset S of $\mathcal{D}$ can then be approximately described by the list of blocks (equivalence classes) in which S has elements:

$$\begin{array}{rcl}
\alpha^\mathcal{P}(X) & \stackrel{\text{def}}{=} & \{B \in \mathcal{P} \mid B \cap X \neq \emptyset\} \\
\gamma^\mathcal{P}(L) & \stackrel{\text{def}}{=} & \cup\{S \mid S \in L\}
\end{array}$$

$$\langle \wp(\mathcal{D});\ \subseteq,\ \emptyset,\ \mathcal{D},\ \cup,\ \cap\rangle \ \underset{\alpha^\mathcal{P}}{\overset{\gamma^\mathcal{P}}{\leftrightarrows}}\ \langle \wp(\mathcal{P});\ \subseteq,\ \emptyset,\ \mathcal{D},\ \cup,\ \cap\rangle$$

In practice a coding of $\wp(\mathcal{P})$ by an $\subseteq$-isomorphic set may be used.

6: Reduction of an abstraction

If α^i is not surjective in (3), then there exists different abstract values $x \in \mathcal{D}^i_a$ and $y \in \mathcal{D}^i_a$ with the same meaning $\gamma^i(x) = \gamma^i(y)$. Hence one of them can be eliminated from $\mathcal{D}^i_a$ without loss of expressiveness of the abstract interpretation, since (3) implies:

$$\begin{array}{c}
\langle \wp(\mathcal{D}^i);\ \subseteq,\ \emptyset,\ \mathcal{D}^i,\ \cup,\ \cap\rangle \\
\underset{\alpha^i}{\overset{\gamma^i}{\leftrightarrows}}\!\!\twoheadrightarrow \\
\langle \Re^{\gamma^i}_{\alpha^i}(\mathcal{D}^i_a);\ \subseteq^i_a,\ \alpha^i \circ \gamma^i(\emptyset^i_a),\ \Upsilon^i_a,\ \cup^i_a,\ \lambda X\bullet \alpha^i \circ \gamma^i(\cap^i_a X)\rangle
\end{array} \qquad (18)$$

where $\Re^{\gamma^i}_{\alpha^i}(\mathcal{D}^i_a) \stackrel{\text{def}}{=} \{\alpha^i \circ \gamma^i(x) \mid x \in \mathcal{D}^i_a\}$ and $\longrightarrow\!\!\!\!\twoheadrightarrow$ indicates that α^i is surjective. For example, two abstract interpretations where $\gamma^i(\emptyset) = \emptyset$, $i = 1,\ 2$ can be extended to pairs with $\gamma^\oplus(\langle x,\ y\rangle) = \gamma^1(x) \cap \gamma^2(y)$ in which case all pairs with an empty component denote the empty set and can be eliminated in favor of $\langle\emptyset,\ \emptyset\rangle$. Our later examples are (implicitly) reduced.

7: Completions of lattices of properties

We now recall the disjunctive completion of a lattice of properties, a technique we introduced in [16] to prove that merge-over-paths (MOP) dataflow analyses can be equivalently expressed in fixpoint form. More generally, we consider the complete lattice of completions of the lattice of properties and exhibit a few interesting members which we present in various equivalent forms. Concrete and abstract properties are assumed to correspond, as follows:

$$\begin{array}{c}
\langle \wp(\mathcal{D});\ \subseteq,\ \emptyset,\ \mathcal{D},\ \cup,\ \cap\rangle \\
\underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \\
\langle \mathcal{D}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a\rangle
\end{array} \qquad (19)$$

7.1: Disjunctive completion

Disjunctive completion consists in enriching approximate disjunctions in the lattice of properties by exact ones.

Definition of the disjunctive completion: Define the preorder $\subseteq^\vee_a$ on $\wp(\mathcal{D}_a)$ by $X \subseteq^\vee_a Y \stackrel{\text{def}}{=} \forall x \in X : \exists y \in Y : x \subseteq_a y$. By considering the equivalence classes $[X]^\vee_{\equiv_a} \stackrel{\text{def}}{=} \{Y \mid X \equiv^\vee_a Y\}$ of $X \in \wp(\mathcal{D}_a)$ for the equivalence relation $X \equiv^\vee_a Y \stackrel{\text{def}}{=} X \subseteq^\vee_a Y \wedge Y \subseteq^\vee_a X$, $\wp^\vee(\mathcal{D}_a) \stackrel{\text{def}}{=} \wp(\mathcal{D}_a)/\equiv^\vee_a \stackrel{\text{def}}{=} \{[X]^\vee_{\equiv_a} \mid X \in \wp(\mathcal{D}_a) \setminus \{\emptyset\}\}$, is a complete lattice $\langle \wp^\vee(\mathcal{D}_a);\ \subseteq^\vee_a,\ \emptyset^\vee_a,\ \mathcal{D}^\vee_a,\ \cup^\vee_a,\ \cap^\vee_a\rangle$ where $[X]^\vee_{\equiv_a} \subseteq^\vee_a [Y]^\vee_{\equiv_a} \stackrel{\text{def}}{=} X \subseteq^\vee_a Y$, $\emptyset^\vee_a \stackrel{\text{def}}{=} [\{\emptyset_a\}]^\vee_{\equiv_a}$, $\mathcal{D}^\vee_a \stackrel{\text{def}}{=} [\mathcal{D}_a]^\vee_{\equiv_a}$, $\bigcup^\vee_{a\,i\in\Delta}[X_i]^\vee_{\equiv_a} \stackrel{\text{def}}{=} [\bigcup_{i\in\Delta} X_i]^\vee_{\equiv_a}$, $\bigcap^\vee_{a\,i\in\Delta}[X_i]^\vee_{\equiv_a} \stackrel{\text{def}}{=}$

$[\bigcap_{i\in\Delta} \downarrow^{\subseteq_a} X_i]^{\vee}_{\equiv_a}$ and $\downarrow^{\subseteq_a} X \stackrel{\text{def}}{=} \{x \in \mathcal{D}_a \mid \exists y \in X : x \subseteq_a y\}$.

Completion of the lattice of concrete properties: When $\langle \mathcal{D}_a; \subseteq_a\rangle$ is $\langle \wp(\mathcal{D}); \subseteq\rangle$ we obtain the lattice $\langle \wp^{\vee}(\wp(\mathcal{D})); \subseteq^{\vee}, \emptyset^{\vee}, \wp(\mathcal{D})^{\vee}, \cup^{\vee}, \cap^{\vee}\rangle$ of disjunctive concrete properties. By eliminating disjunctions, using:

$$\begin{array}{rcl} \alpha^{\vee}([X]^{\vee}_{\equiv}) & \stackrel{\text{def}}{=} & \bigcup_{y\in X} y \\ \gamma^{\vee}(X) & \stackrel{\text{def}}{=} & [\{\{x\} \mid x \in X\}]^{\vee}_{\equiv} \end{array}$$

we obtain a Galois connection with the original (non-disjunctive) properties:

$$\begin{array}{c} \langle \wp^{\vee}(\wp(\mathcal{D})); \subseteq^{\vee}, \emptyset^{\vee}, \wp(\mathcal{D})^{\vee}, \cup^{\vee}, \cap^{\vee}\rangle \\ \overset{\gamma^{\vee}}{\underset{\alpha^{\vee}}{\leftrightarrows}} \\ \langle \wp(\mathcal{D}); \subseteq, \emptyset, \mathcal{D}, \cup, \cap\rangle \end{array} \qquad (20)$$

Completion of the lattice of abstract properties: When completing both the lattice of concrete and abstract properties, the abstraction:

$$\begin{array}{rcl} \alpha^{\vee}_a(X) & \stackrel{\text{def}}{=} & [\{\alpha(x) \mid x \in X\}]^{\vee}_{\equiv_a} \\ \gamma^{\vee}_a([X]^{\vee}_{\equiv_a}) & \stackrel{\text{def}}{=} & [\{\gamma(y) \mid y \in X\}]^{\vee}_{\equiv} \end{array}$$

is, by (19), a Galois connection:

$$\begin{array}{c} \langle \wp^{\vee}(\wp(\mathcal{D})); \subseteq^{\vee}, \emptyset^{\vee}, \wp(\mathcal{D})^{\vee}, \cup^{\vee}, \cap^{\vee}\rangle \\ \overset{\gamma^{\vee}_a}{\underset{\alpha^{\vee}_a}{\leftrightarrows}} \\ \langle \wp^{\vee}(\mathcal{D}_a); \subseteq^{\vee}_a, \emptyset^{\vee}_a, \mathcal{D}^{\vee}_a, \cup^{\vee}_a, \cap^{\vee}_a\rangle \end{array} \qquad (21)$$

This disjunctive abstract interpretation is more precise than the original one, since:

$$\alpha^{\triangledown}_a([X]^{\vee}_{\equiv_a}) \stackrel{\text{def}}{=} \cup_a X \qquad \gamma^{\triangledown}_a(x) \stackrel{\text{def}}{=} [\{x\}]^{\vee}_{\equiv_a}$$

is a Galois connection:

$$\begin{array}{c} \langle \wp^{\vee}(\mathcal{D}_a); \subseteq^{\vee}_a, \emptyset^{\vee}_a, \mathcal{D}^{\vee}_a, \cup^{\vee}_a, \cap^{\vee}_a\rangle \\ \overset{\gamma^{\triangledown}_a}{\underset{\alpha^{\triangledown}_a}{\leftrightarrows}} \\ \langle \mathcal{D}_a; \subseteq_a, \emptyset_a, \Upsilon_a, \cup_a, \cap_a\rangle \end{array}$$

When $\langle \alpha^{\vee}_a, \gamma^{\vee}_a\rangle$ is strictly more precise than the original abstract interpretation, this original abstract interpretation $\langle \alpha, \gamma\rangle$ is said to be "non-disjunctive" else it is "disjunctive".

The meaning of the completion of the abstract properties is defined by (21) with respect to the completion of the concrete properties. By composing with (20), as defined in (7), the meaning can be expressed with respect to the original (non-disjunctive) lattice of the concrete properties.

7.2: Order ideal completion

The order ideal completion consists in considering order ideals to represent the equivalence classes $[X]^{\vee}_{\equiv_a}$ of $\wp^{\vee}(\mathcal{D}_a)$. An *order ideal* of the complete lattice $\langle \mathcal{D}_a; \subseteq_a, \emptyset_a, \Upsilon_a, \cup_a, \cap_a\rangle$ is $I \subseteq \mathcal{D}_a$ such that $I = \downarrow^{\subseteq_a} I$. The *order ideal completion* of $\mathcal{D}_a$ is the complete lattice $\langle \wp^{\downarrow}(\mathcal{D}_a); \subseteq, \{\emptyset_a\}, \mathcal{D}_a, \cup, \cap\rangle$ where $\wp^{\downarrow}(\mathcal{D}_a) = \{I \subseteq \mathcal{D}_a \mid I = \downarrow^{\subseteq_a} I \wedge I \neq \emptyset\}$. The disjunctive and order ideal completions are isomorphic:

$$\alpha^{\downarrow}_a([X]^{\vee}_{\equiv_a}) \stackrel{\text{def}}{=} \downarrow^{\subseteq_a} X \qquad \gamma^{\downarrow}_a(I) \stackrel{\text{def}}{=} [I]^{\vee}_{\equiv_a}$$

$$\begin{array}{c} \langle \wp^{\vee}(\mathcal{D}_a); \subseteq^{\vee}_a, \emptyset^{\vee}_a, \mathcal{D}^{\vee}_a, \cup^{\vee}_a, \cap^{\vee}_a\rangle \\ \overset{\gamma^{\downarrow}_a}{\underset{\alpha^{\downarrow}_a}{\twoheadleftarrow\!\!\twoheadrightarrow}} \\ \langle \wp^{\downarrow}(\mathcal{D}_a); \subseteq, \{\emptyset_a\}, \mathcal{D}_a, \cup, \cap\rangle \end{array}$$

7.3: Scott closed ideal completion

Considering Scott closed ideals (containing lubs of increasing chains) leads to a less precise completion. The *Scott closed ideal of* $X \in \wp(\mathcal{D}_a)$ is $\Downarrow^{\subseteq_a} X \stackrel{\text{def}}{=} \downarrow^{\subseteq_a} X \cup \hbar(\downarrow^{\subseteq_a} X)$ where $\hbar(X) \stackrel{\text{def}}{=} \{\cup_{a\, i\in\mathbb{N}}\, x_i \mid \forall i \in \mathbb{N} : x_i \in X \wedge x_i \subseteq_a x_{i+1}\}$ is the *adherence* of X. The *lower power domain* of $\mathcal{D}_a$ is $\wp^{\Downarrow}(\mathcal{D}_a) \stackrel{\text{def}}{=} \{I \subseteq \mathcal{D}_a \mid I = \Downarrow^{\subseteq_a} I \wedge I \neq \emptyset\}$. It is a complete lattice $\langle \wp^{\Downarrow}(\mathcal{D}_a); \subseteq, \{\emptyset_a\}, \mathcal{D}_a, \lambda X \bullet \hbar(\cup X), \cap\rangle$. By defining the Galois connection:

$$\alpha^{\Downarrow}(I) \stackrel{\text{def}}{=} \hbar(I) \qquad \gamma^{\Downarrow}(J) \stackrel{\text{def}}{=} J$$

$$\begin{array}{c} \langle \wp^{\vee}(\mathcal{D}_a); \subseteq^{\vee}_a, \emptyset^{\vee}_a, \mathcal{D}^{\vee}_a, \cup^{\vee}_a, \cap^{\vee}_a\rangle \\ \overset{\gamma^{\Downarrow}}{\underset{\alpha^{\Downarrow}}{\leftrightarrows}} \\ \langle \wp^{\Downarrow}(\mathcal{D}_a); \subseteq, \{\emptyset_a\}, \mathcal{D}_a, \lambda X \bullet \hbar(\cup X), \cap\rangle \end{array}$$

we see that the Scott closed ideal completion of $\wp(\mathcal{D}_a)$ is an abstract interpretation of order ideal completion $\wp^{\vee}(\mathcal{D}_a)$, hence, by (7), an abstract interpretation of $\wp(\mathcal{D}_a)$. It is in general less precise since for all $X \in \wp(\mathcal{D}_a)$, $\downarrow^{\subseteq_a} X = \Downarrow^{\subseteq_a} X$ if and only if $\mathcal{D}_a$ satisfies the ascending chain condition, in which case the order and Scott closed ideal completions coincide:

$$\begin{array}{cl} \langle \wp^{\vee}(\mathcal{D}_a); \subseteq^{\vee}_a, \emptyset^{\vee}_a, \mathcal{D}^{\vee}_a, \cup^{\vee}_a, \cap^{\vee}_a\rangle & \mathcal{D}_a \text{ satisfies} \\ \overset{\gamma^{\Downarrow}}{\underset{\alpha^{\Downarrow}}{\twoheadleftarrow\!\!\twoheadrightarrow}} & \text{the ascend-} \\ & \text{ing chain} \\ \langle \wp^{\Downarrow}(\mathcal{D}_a); \subseteq, \{\emptyset_a\}, \mathcal{D}_a, \lambda X \bullet \hbar(\cup X), \cap\rangle & \text{condition.} \end{array}$$

7.4: Anti-chain completion

Scott closed ideals can be represented by their maximal elements [7]. The *crown* $\mathbb{W}(X) \stackrel{\text{def}}{=} \{m \in X \mid \forall x \in X : m \subseteq_a x \Rightarrow m = x\}$ of X is the set of its maximal elements. It is an *anti-chain* since no two elements are comparable. The *crown completion* $\wp^{\mathbb{W}}(\mathcal{D}_a) \stackrel{\text{def}}{=} \{C \subseteq \mathcal{D}_a \mid C = \mathbb{W}(C) \wedge C \neq \emptyset\}$ of $\mathcal{D}_a$ is a complete lattice $\langle \wp^{\mathbb{W}}(\mathcal{D}_a); \subseteq^{\mathbb{W}}_a, \{\emptyset_a\}, \{\Upsilon_a\}, \cup^{\mathbb{W}}, \cap^{\mathbb{W}}\rangle$, where $C \subseteq^{\mathbb{W}}_a C' \stackrel{\text{def}}{=} \forall x \in C : \exists y \in C' : x \subseteq_a y$, $\cup^{\mathbb{W}} \stackrel{\text{def}}{=} \lambda X \bullet \mathbb{W}(\bigcup X)$ and $\cap^{\mathbb{W}} \stackrel{\text{def}}{=} \lambda X \bullet \mathbb{W}(\bigcap_{C\in X} \downarrow^{\subseteq_a} C)$. The crown and Scott closed ideal completions coincide:

$$\alpha^{\mathbb{W}}(J) \stackrel{\text{def}}{=} \mathbb{W}(J) \qquad \gamma^{\mathbb{W}}(C) \stackrel{\text{def}}{=} \downarrow^{\subseteq_a} C$$

$$\langle \wp^{\Downarrow}(\mathcal{D}_a);\ \subseteq,\ \{\emptyset_a\},\ \mathcal{D}_a,\ \lambda X \bullet \hbar(\cup X),\ \cap\rangle \qquad (22)$$
$$\underset{\alpha^{\overline{\mathrm{W}}}}{\overset{\gamma^{\overline{\mathrm{W}}}}{\leftrightarrows}}$$
$$\langle \wp^{\overline{\mathrm{W}}}(\mathcal{D}_a);\ \subseteq_a^{\overline{\mathrm{W}}},\ \{\emptyset_a\},\ \{\Upsilon_a\},\ \cup^{\overline{\mathrm{W}}},\ \cap^{\overline{\mathrm{W}}}\rangle$$

Again, the crown and order ideal completions coincide if and only if $\mathcal{D}_a$ satisfies the ascending chain condition.

7.5: The complete lattice of join completions

More generally, a *join completion* is any subset $\wp^{\downarrow^{\cup}}(\mathcal{D}_a)$ of $\wp^{\downarrow}(\mathcal{D}_a)$ which is a *Moore family* (i.e. contains the supremum $\mathcal{D}_a$ and $\cap X$ with any $X \subseteq \wp^{\downarrow^{\cup}}(\mathcal{D}_a)$) and contains all *principal ideals* of $\mathcal{D}_a$ (i.e. $\wp^{\downarrow^{\mathrm{P}}}(\mathcal{D}_a) \stackrel{\text{def}}{=} \{\downarrow^{\subseteq_a}\{x\} \mid x \in \mathcal{D}_a\}$) is a complete lattice which is an approximation of the disjunctive completion:

$$\begin{aligned}\alpha_a^{\downarrow^{\cup}}([X]^{\vee}_{\equiv_a}) &\stackrel{\text{def}}{=} \cap\{I \in \wp^{\downarrow^{\cup}}(\mathcal{D}_a) \mid X \subseteq I\}\\ \gamma_a^{\downarrow^{\cup}}(I) &\stackrel{\text{def}}{=} [I]^{\vee}_{\equiv_a}\end{aligned}$$
$$\langle \wp^{\vee}(\mathcal{D}_a);\ \subseteq_a^{\vee},\ \emptyset_a^{\vee},\ \mathcal{D}_a^{\vee},\ \cup_a^{\vee},\ \cap_a^{\vee}\rangle$$
$$\underset{\alpha_a^{\downarrow^{\cup}}}{\overset{\gamma_a^{\downarrow^{\cup}}}{\leftrightarrows}}$$
$$\langle \wp^{\downarrow^{\cup}}(\mathcal{D}_a);\ \subseteq,\ \{\emptyset_a\},\ \mathcal{D}_a,\ \cup,\ \cap\rangle$$

Up to isomorphism, the complete lattice of all $\wp^{\downarrow^{\cup}}(\mathcal{D}_a)$ for $\subseteq$ has infimum $\wp^{\downarrow^{\mathrm{P}}}(\mathcal{D}_a)$ and supremum $\wp^{\downarrow}(\mathcal{D}_a)$. The *principal completion* $\wp^{\downarrow^{\mathrm{P}}}(\mathcal{D}_a)$ (i.e. the Moore family corresponding to the intersection of principal ideals) is isomorphic with $\mathcal{D}_a$ while the *disjunctive completion* $\wp^{\downarrow}(\mathcal{D}_a)$ corresponds to the most precise properties obtained by completing missing disjunctions in $\mathcal{D}_a$.

7.6: Order filter completion

We now examine the dual situation and observe that in the abstract lattice $\mathcal{D}_a$, conjunctions are exact with respect to $\wp(\mathcal{D})$ (while disjunctions are approximate).

An *order filter* of a complete lattice $\langle \mathcal{D}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a\rangle$ is $F \subseteq \mathcal{D}_a$ such that $F = \uparrow^{\subseteq_a} F$ where the *order filter of* X is $\uparrow^{\subseteq_a} X \stackrel{\text{def}}{=} \{y \mid \exists x \in X : x \subseteq_a y\}$. The *order filter completion* of $\mathcal{D}_a$ is the complete lattice $\langle \wp^{\uparrow}(\mathcal{D}_a);\ \supseteq,\ \mathcal{D}_a,\ \{\Upsilon_a\},\ \cap,\ \cup\rangle$ where $\wp^{\uparrow}(\mathcal{D}_a) = \{F \subseteq \mathcal{D}_a \mid F = \uparrow^{\subseteq_a} F \wedge F \neq \emptyset\}$. If (19) then:

$$\begin{aligned}\alpha^{\uparrow}(X) &\stackrel{\text{def}}{=} \uparrow^{\subseteq_a}\{\alpha(\{x\}) \mid x \in X\}\\ \gamma^{\uparrow}(F) &\stackrel{\text{def}}{=} \cap\{\gamma(y) \mid y \in F\}\end{aligned}$$
$$\langle \wp(\mathcal{D});\ \subseteq,\ \emptyset,\ \mathcal{D},\ \cup,\ \cap\rangle$$
$$\underset{\alpha^{\uparrow}}{\overset{\gamma^{\uparrow}}{\leftrightarrows}}$$
$$\langle \wp^{\uparrow}(\mathcal{D}_a);\ \supseteq,\ \mathcal{D}_a,\ \{\Upsilon_a\},\ \cap,\ \cup\rangle$$

The original abstract interpretation is an abstraction of its order filter completion:

$$\alpha_a^{\uparrow}(F) \stackrel{\text{def}}{=} \cap_a F \qquad \gamma_a^{\uparrow}(x) \stackrel{\text{def}}{=} \uparrow^{\subseteq_a}\{x\}$$

$$\langle \wp^{\uparrow}(\mathcal{D}_a);\ \supseteq,\ \mathcal{D}_a,\ \{\Upsilon_a\},\ \cap,\ \cup\rangle$$
$$\underset{\alpha_a^{\uparrow}}{\overset{\gamma_a^{\uparrow}}{\leftrightarrows}}$$
$$\langle \mathcal{D}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a\rangle$$

However the order filter completion is not more expressive than the original abstract interpretation, since its reduction is isomorphic with the original abstract interpretation:

$$\begin{aligned}\langle \Re_{\alpha^{\uparrow}}^{\gamma^{\uparrow}}(\wp^{\uparrow}(\mathcal{D}_a));\ \supseteq,\ \alpha^{\uparrow} \circ \gamma^{\uparrow}(\mathcal{D}_a),\ \{\Upsilon_a\},\ \cap,&\\ \lambda X \bullet \alpha^{\uparrow} \circ \gamma^{\uparrow}(\cup X)\rangle&\end{aligned}$$
$$\underset{\alpha_a^{\uparrow}}{\overset{\gamma_a^{\uparrow}}{\leftrightarrows}}$$
$$\langle \mathcal{D}_a;\ \subseteq_a,\ \emptyset_a,\ \Upsilon_a,\ \cup_a,\ \cap_a\rangle$$

Otherwise stated, the intersections which are introduced by the filters where already present in the original abstract lattice $\mathcal{D}_a$. Hence order filter completion as well as conjunctive, dual Scott closed and dual crown completion of $\mathcal{D}_a$ are useless – with respect to $\wp(\mathcal{D})$ – in the context of abstraction by Galois connections (which ensures the existence of a best approximation)..

Part II : Application to Comportment Analysis Generalizing Strictness, Termination, Projection and PER Analysis of Functional Languages

To get faster implementations of lazy functional languages on sequential or parallel machines, optimizing compilers transform call-by-need into call-by-value when the program meaning is not altered (up to the reason for divergence or run-time errors). Four program analysis techniques are mainly used in order to determine when this transformation is safe:

- — *Strictness* and *termination analysis* introduced by Mycroft [54, 55];
- — *Projection analysis* introduced by Hughes [38] and Wadler [64];
- — *PER analysis* introduced by Hunt [42].

We introduce a new application of higher-order abstract interpretation, called *comportment analysis*, which unifies and generalizes all four methods into a single abstract interpretation framework.

8: Background on the analysis of lazy functional languages

8.1: Strictness analysis

Strictness analysis is used to answer the question of knowing if $f(\bot) = \bot$ where $\bot$ denotes divergence (and run-time errors), as usual in denotational semantics. This shows that function f either does not terminate or needs its argument. Strictness analysis is based on *abstract interpretation* [14, 16]. The approximation of $\emptyset$ and of $\{\bot\}$ is $\mathbf{0}^{\sharp}$. The approximation of any other nonempty subset of values is $\mathbf{1}^\sharp$. Therefore the meaning of these abstract values is $\gamma(\mathbf{0}^\sharp) \stackrel{\text{def}}{=} \{\bot\}$ and $\gamma(\mathbf{1}^\sharp) \stackrel{\text{def}}{=} D_\bot$ where $D_\bot \stackrel{\text{def}}{=} D \cup \{\bot\}$ and D is the domain of values. The denotational semantics of functions f on $D_\bot$ is approximated by an abstract semantics $f^\sharp$ on $\{\mathbf{0}^\sharp, \mathbf{1}^\sharp\}$ such that $f^\sharp(\mathbf{0}^\sharp) = \mathbf{0}^\sharp$ implies $f(\bot) = \bot$ and $f^\sharp(\mathbf{1}^\sharp) = \mathbf{0}^\sharp$ implies $\forall x \in D_\bot : f(x) = \bot$ whereas $f^\sharp(a) = \mathbf{1}^\sharp$ represents an unknown behavior:

$\lambda x \bullet \mathbf{1}$	tt	truth
$\lambda x \bullet x$	$f(\bot) = \bot$	strictness
$\lambda x \bullet \mathbf{0}$	$\forall x \in D_\bot : f(x) = \bot$	divergence

When considering functions with multiple arguments, Mycroft's strictness analysis [55] is *disjunctive* ("relational analysis" in [47]). It can express that a function is jointly strict in its arguments when $f^\sharp(\mathbf{0}^\sharp, \mathbf{0}^\sharp) = \mathbf{0}^\sharp$ but neither $f^\sharp(\mathbf{1}^\sharp, \mathbf{0}^\sharp) = \mathbf{0}^\sharp$ nor $f^\sharp(\mathbf{0}^\sharp, \mathbf{1}^\sharp) = \mathbf{0}^\sharp$. Johnsson's strictness analysis [46] is *non-disjunctive* ("independent attribute" in [47]) whence less expensive but also less precise. The strictness is expressed independently for each argument by $f_1^\times(\mathbf{0}^\sharp) = \mathbf{0}^\sharp$ for $\forall y \in D_\bot : f(\bot, y) = \bot$ and $f_2^\times(\mathbf{0}^\sharp) = \mathbf{0}^\sharp$ for $\forall x \in D_\bot : f(x, \bot) = \bot$. In all cases disjunctive analyses are more powerful than non-disjunctive ones.

Strictness analysis is a *forward* analysis in that the abstract result is computed knowing the abstract arguments representing the past history of the computation. By observing that the knowledge of the inverse image $f^{\sharp -1}(\mathbf{0}^\sharp)$ of $\mathbf{0}^\sharp$ is equivalent to that of $f^\sharp$, one obtains ideal-based *backward* strictness analyses [29, 30] where the abstract arguments are computed using the abstract result representing the future history of the computation. Relating forward and backward analyses is not so easy in denotational semantics since the inverse of function may not be a function and continuity may be obtained only by restriction to finite abstract domains [35, 36]. For a practical example, the fact that forward and backward disjunctive strictness analyses are isomorphic and that, if no useless approximation is done, the same holds for non-disjunctive strictness analyses seems to have escaped from the attention of [26]. The cases when forward analysis is equivalent to backward analysis should now be well-understood [36].

Most approaches to strictness analysis use denotational semantics as *standard semantics* [59], but can also be formalized with an operational semantics [27]. One difficulty with denotational semantics is that the *collecting semantics* uses powerdomains [57]. When considering nondeterministic functional languages one should consider powerdomains of powerdomains which becomes complicated.

Strictness analysis has been extended to higher-order [9, 10, 34], to lazy data structures [33, 63], to polymorphism [1, 4, 5] and can be mixed with type inference [11, 49, 65] using the equivalence between logical rule-based and fixpoint presentations [44].

8.2: Termination analysis

Mycroft's termination analysis [2, 55] is used to answer the question of knowing if function f terminates for all arguments that is $\forall x \in D_\bot : f(x) \in D$. The evaluation of an always terminating call-by-need argument can be safely anticipated.

Termination analysis is an abstract interpretation of the denotational semantics on the abstract domain $\{\mathbf{1}^\flat, \mathbf{0}^\flat\}$ with interpretation $\gamma(\mathbf{1}^\flat) \stackrel{\text{def}}{=} D$ and $\gamma(\mathbf{0}^\flat) \stackrel{\text{def}}{=} D_\bot$. It follows that $f^\flat(\mathbf{1}^\flat) = \mathbf{1}^\flat$ implies totality (convergence for converging argument): $\forall x \in D : f(x) \in D$ and $f^\flat(\mathbf{0}^\flat) = \mathbf{1}^\flat$ implies convergence: $\forall x \in D_\bot : f(x) \in D$:

$\lambda x \bullet \mathbf{0}$	tt	truth
$\lambda x \bullet x$	$\forall x \in D : f(x) \in D$	totality
$\lambda x \bullet \mathbf{1}$	$\forall x \in D_\bot : f(x) \in D$	convergence

Observe that termination analysis is a very crude form of constant propagation [48] where the value of constants is simply ignored. This may explain why it has not be much studied [2].

8.3: Projection analysis

Projection analysis [38, 25, 64] uses *projections* $\beta, \delta \in D_\bot \mapsto D_\bot$ which are reductive ($\beta \sqsubseteq id$ where id is the identity function: $\forall x \in D_\bot : id(x) = x$) and idempotent ($\beta \circ \beta = \beta$) continuous functions on $D_\bot$. Here $\sqsubseteq$ is Scott's partial ordering: $\forall x \in D : \bot \sqsubseteq \bot \sqsubset x \sqsubseteq x$. A projection β represents a safe loss of information. For example abs such that $\forall x \in D_\bot : abs(x) = \bot$ specifies that a value x can be replaced by $\bot$ without changing the meaning of the program since this value is not used. The equivalent relations:

$$\beta \circ f = \beta \circ f \circ \delta \quad \Longleftrightarrow \quad \beta \circ f \sqsubseteq f \circ \delta$$

are denoted by Hughes/Wadler's backward notation $f : \beta \Rightarrow \delta$ (to get β's worth about the result we only need to know δ's worth about the argument to f) or by Launchbury's forward notation $f : \delta \rightarrow \beta$ (if we know δ's worth about the argument to f then we know β's worth about the result). For example *absence* $f : abs \rightarrow id$ means that replacing the argument by $\bot$ does not change the result, that is $id \circ f = id \circ f \circ abs$ whence $\forall x \in D_\bot : f(x) = f(\bot)$. Unfortunately there are no projections δ and β on $D_\bot$ such that strictness $f(\bot) = \bot$ can be expressed as $f : \delta \rightarrow \beta$. To do so $D_\bot$ must be lifted into $D_{\bot_\lightning}$ with a new infimum $\lightning$ called *abort*: $\forall x \in D_\bot : \lightning \sqsubseteq \lightning \sqsubset x$. If b is true then $b \,?\, e_t : e_f$ is e_t else e_f. By defining:

id	$\stackrel{\text{def}}{=}$	$\lambda x \bullet x$	str	$\stackrel{\text{def}}{=}$	$\lambda x \bullet (x \in \{\lightning, \bot\} \,?\, \lightning : x)$
$fail$	$\stackrel{\text{def}}{=}$	$\lambda x \bullet \lightning$	abs	$\stackrel{\text{def}}{=}$	$\lambda x \bullet (x = \lightning \,?\, \lightning : \bot)$

and considering that all functions are $\lightning$-strict ($f(\lightning) \stackrel{\text{def}}{=} \lightning$), one can express:

$f : str \rightarrow str$	$f(\bot) = \bot$	strictness
$f : abs \rightarrow str$, $f : abs \rightarrow id$	$\forall x \in D_\bot : f(x) = f(\bot)$	absence
$f : fail \rightarrow str$	$\forall x \in D_\bot : f(x) = \bot$	divergence
$f : id \rightarrow id$, $f : id \rightarrow str$, $f : id \rightarrow abs$, $f : id \rightarrow fail$, $f : str \rightarrow abs$, $f : str \rightarrow fail$, $f : abs \rightarrow abs$, $f : abs \rightarrow fail$, $f : fail \rightarrow fail$	tt	truth
$f : str \rightarrow id$, $f : fail \rightarrow id$, $f : fail \rightarrow abs$	ff	falsity

Observe than there may be many ways to express the same property.

For functions with multiple arguments, traditional projection analysis is non-disjunctive. $f : [\delta^1, \ldots, \delta^n] \rightarrow \beta$ means that $\forall i \in [1, n] : \beta(f(x^1, \ldots, x^n)) \sqsubseteq f(x^1, \ldots, \delta^i(x^i), \ldots, x^n)$. A disjunctive form $\bigvee_{i \in \Delta} f : [\delta_i^1, \ldots, \delta_i^n] \rightarrow \beta_i$ can also be used [58].

Projection analysis has been used for time complexity analysis [62], binding-time analysis [50, 51] and extended to higher-order [24], to lazy data structures [37, 64] and to polymorphism [39, 40].

Burn has observed that projection analysis, which can express strictness and divergence, encompasses strictness analysis. For strict functions (for which absence is equivalent to divergence), the projection and strictness results are equivalent [8]. Neuberger and Mishra [58] have shown that when considering a disjunctive version of projection analysis but with projections *fail*, *str* and *id* only, one obtain results isomorphic with Mycroft's non-disjunctive strictness analysis. In fact not only the results but the iterative computations themselves are isomorphic and this also holds for the non-disjunctive versions.

The overall informal impression when comparing projection analysis and strictness analysis is that projection analysis is more precise. However, the comparisons found in the literature are confusing since they proceed by restricting abstract interpretation (to *bottom-reflecting* abstraction maps in [8]: $\forall x \in D_\bot : \alpha(x) = \bot \Rightarrow x = \bot$) or projection analysis (to *smash projections* in [58]: $\forall x \in D_\bot : \beta(x) \neq \lightning \Rightarrow \beta(x) = x$).

8.4: Dual projection analysis

As noticed by Launchbury (private communication), by inverting the order relation, one can define a dual projection analysis:

$$f : \delta \stackrel{\sim}{\rightarrow} \beta \stackrel{\text{def}}{=} f \circ \delta \sqsubseteq \beta \circ f$$

so as to express the following properties:

$f : id \stackrel{\sim}{\rightarrow} str$, $f : abs \stackrel{\sim}{\rightarrow} str$	$\forall x \in D_\bot : f(x) \in D$	convergence
$f : str \stackrel{\sim}{\rightarrow} str$	$\forall x \in D : f(x) \in D$	totality
$f : id \stackrel{\sim}{\rightarrow} abs$, $f : abs \stackrel{\sim}{\rightarrow} abs$, $f : str \stackrel{\sim}{\rightarrow} abs$	$\forall x \in D_\bot : f(x) = \bot$	divergence
$f : id \stackrel{\sim}{\rightarrow} id$, $f : str \stackrel{\sim}{\rightarrow} id$, $f : abs \stackrel{\sim}{\rightarrow} id$, $f : fail \stackrel{\sim}{\rightarrow} id$, $f : fail \stackrel{\sim}{\rightarrow} fail$, $f : fail \stackrel{\sim}{\rightarrow} str$, $f : fail \stackrel{\sim}{\rightarrow} abs$	tt	truth
$f : id \stackrel{\sim}{\rightarrow} fail$, $f : str \stackrel{\sim}{\rightarrow} fail$, $f : abs \stackrel{\sim}{\rightarrow} fail$	ff	falsity

Dual projection analysis is definitely more expressive than termination analysis.

8.5: Per analysis

Hunt's PER analysis [42] can express program properties of the form:

$$f(\textsc{a}) = \textsc{b} \quad \stackrel{\text{def}}{=} \quad \forall x, y \in D_\bot : x \equiv_{\textsc{a}} y \Rightarrow f(x) \equiv_{\textsc{b}} f(y)$$

where $\equiv_{\textsc{a}}$ and $\equiv_{\textsc{b}}$ are partial equivalence relations (PERs) that is transitive and symmetric binary relations on $D_\bot$. Hunt's PER analysis generalizes projection analysis by defining:

$\gamma(\textsc{bot}) \stackrel{\text{def}}{=} \equiv_{\textsc{bot}}$
$\gamma(\textsc{id}) \stackrel{\text{def}}{=} \equiv_{\textsc{id}}$
$\gamma(\textsc{abs}) \stackrel{\text{def}}{=} \equiv_{\textsc{abs}}$

$\equiv_{\textsc{bot}} \stackrel{\text{def}}{=} \{\langle\bot, \bot\rangle\}$
$\equiv_{\textsc{id}} \stackrel{\text{def}}{=} \{\langle x, x\rangle \mid x \in D_\bot\}$
$\equiv_{\textsc{abs}} \stackrel{\text{def}}{=} D_\bot \times D_\bot$

so as to express the following properties:

$f(\textsc{bot}) = \textsc{bot}$	$f(\bot) = \bot$	strictness
$f(\textsc{abs}) = \textsc{id}$	$\forall x, y \in D_\bot : f(x) = f(y)$	absence
$f(\textsc{abs}) = \textsc{bot}$	$\forall x \in D_\bot : f(x) = \bot$	divergence
$f(\textsc{bot}) = \textsc{abs}$, $f(\textsc{bot}) = \textsc{id}$, $f(\textsc{id}) = \textsc{bot}$, $f(\textsc{id}) = \textsc{abs}$, $f(\textsc{id}) = \textsc{id}$, $f(\textsc{abs}) = \textsc{abs}$	tt	truth

PER-based abstract interpretations have been introduced as a generalization of projection analysis for strictness [42] and binding-time properties [43]. In fact the generalization is not so obvious since no method is given for constructing the abstract domain of PERs corresponding to a given set of projections. For example, [42] passes over abort $\lightning$ in silence.

In order to express totality as in dual projection analysis, one can introduce the PER $\textsc{val}$ such that $\gamma(\textsc{val}) \stackrel{\text{def}}{=} \equiv_{\textsc{val}}$ where $\equiv_{\textsc{val}} \stackrel{\text{def}}{=} D \times D$. Totality is then

$f(\textsc{val}) = \textsc{val}$ and convergence is $f(\textsc{abs}) = \textsc{val}$. Observe that both [42] and [43] use totally ordered domains of PERs whereas $\textsc{bot}$ and $\textsc{val}$ are incomparable. Since PERs are required to be closed under intersection, it is also necessary to introduce the empty PER $\textsc{emp}$ such that $\gamma(\textsc{emp}) \overset{\text{def}}{=} \equiv_{\text{EMP}}$ where $\equiv_{\text{EMP}} \overset{\text{def}}{=} \emptyset$. We can now express falsity as $f(P) = \textsc{emp}$ for all $P \neq \textsc{emp}$. Then PER analysis generalizes both projection and dual projection analysis. We can even express properties that can neither be expressed by projection nor by dual projection analysis such as $f(\textsc{bot}) = \textsc{bot} \wedge f(\textsc{val}) = \textsc{val}$, that is "the function diverges if and only if its argument diverges" ($f(\bot) = \bot \wedge \forall x \in D : f(x) \in D$). However, PER analysis cannot express properties of the form $f(\textsc{bot}) = \textsc{bot} \wedge (f(\textsc{val}) = \textsc{bot} \vee f(\textsc{val}) = \textsc{val})$ and this excludes functions that terminate for some but not all terminating values of their parameters. The problem with PERs here is that disjunctions are missing.

9: A simply typed lambda calculus

To illustrate higher-order abstract interpretation, we consider a simply typed lambda calculus, as the core of a functional language with *basic types* β (such as `bool`, `num`, etc.) and *types* τ including basic types β, pairs $\tau \times \tau$ and functions $\tau \mapsto \tau'$:

$$\tau ::= \beta \mid \tau \times \tau \mid \tau \mapsto \tau'$$

The syntax of expressions e of type τ (written e^τ) is:

$$\begin{array}{lll}
e^\tau \;::= & x^\tau & \text{variables, } x^\tau \in \mathcal{V}, \\
\mid & c^\tau & \text{constants,} \\
\mid & e_1^{\texttt{bool}} \;?\; e_2^\tau : e_3^\tau & \text{conditional,} \\
\mid & \langle e_1^{\tau'},\, e_2^{\tau''} \rangle & \text{pair } (\tau = \tau' \times \tau''), \\
\mid & \texttt{fst}\, e^{\tau \times \tau'} & \text{first projection,} \\
\mid & \texttt{snd}\, e^{\tau' \times \tau} & \text{second projection,} \\
\mid & \lambda x^{\tau'} \bullet e^{\tau''} & \text{abstraction } (\tau = \tau' \mapsto \tau''), \\
\mid & e_1^{\tau' \mapsto \tau} e_2^{\tau'} & \text{application,} \\
\mid & \mu x^\tau \bullet e^\tau & \text{fixpoint.}
\end{array}$$

10: Standard denotational semantics of the simply typed lambda calculus

A Scott domain $\langle \mathcal{D};\, \sqsubseteq,\, \bot,\, \sqcup \rangle$ is a bounded-complete ω-algebraic complete partial order where $\sqsubseteq$ is the partial ordering, $\bot$ is the infimum and countable chains $\{x_n \mid n \in \mathbb{N}\}$ of elements of $\mathcal{D}$ (such that $\forall n \in \mathbb{N} : x_n \sqsubseteq x_{n+1}$) have a least upper bound (lub) $\sqcup_{n \in \mathbb{N}} x_n$ [32].

The set $\mathcal{D}^\tau$ of values of type τ is a Scott domain $\langle \mathcal{D}^\tau;\, \sqsubseteq^\tau,\, \bot^\tau,\, \sqcup^\tau \rangle$ which is given for basic types β (for example as a *flat domain* such that $\forall x \in \mathcal{D}^\beta : \bot^\beta \sqsubseteq^\beta \bot^\beta \sqsubset^\beta x \sqsubseteq^\beta x$). For pairs $\langle \mathcal{D}^{\tau \times \tau'};\, \sqsubseteq^{\tau \times \tau'},\, \bot^{\tau \times \tau'},\, \sqcup^{\tau \times \tau'} \rangle$ is defined componentwise as $\mathcal{D}^{\tau \times \tau'} \overset{\text{def}}{=} \mathcal{D}^\tau \times \mathcal{D}^{\tau'}$, $\langle x,\, y \rangle \sqsubseteq^{\tau \times \tau'} \langle x',\, y' \rangle \overset{\text{def}}{=} x \sqsubseteq^\tau x' \wedge y \sqsubseteq^{\tau'} y'$, $\bot^{\tau \times \tau'} \overset{\text{def}}{=} \langle \bot^\tau,\, \bot^{\tau'} \rangle$ and $\sqcup_{n \in \Delta}^{\tau \times \tau'} \langle x_n,\, y_n \rangle \overset{\text{def}}{=} \langle \sqcup_{n \in \Delta}^\tau x_n,\, \sqcup_{n \in \Delta}^{\tau'} y_n \rangle$. For functions $\langle \mathcal{D}^{\tau \mapsto \tau'};\, \sqsubseteq^{\tau \mapsto \tau'},\, \bot^{\tau \mapsto \tau'},\, \sqcup^{\tau \mapsto \tau'} \rangle$ is defined pointwise as $\mathcal{D}^{\tau \mapsto \tau'} \overset{\text{def}}{=} \mathcal{D}^\tau \mapsto \mathcal{D}^{\tau'}$, $f \sqsubseteq^{\tau \mapsto \tau'} g \overset{\text{def}}{=} \forall x \in \mathcal{D}^\tau : f(x) \sqsubseteq^{\tau'} g(x)$, $\bot^{\tau \mapsto \tau'} \overset{\text{def}}{=} \lambda x \bullet \bot^{\tau'}$ and $\sqcup_{n \in \Delta}^{\tau \mapsto \tau'} f_n \overset{\text{def}}{=} \lambda x \bullet \sqcup_{n \in \Delta}^{\tau'} f_n(x)$. Functions $f \in \mathcal{D}^{\tau \mapsto \tau}$ have a least fixpoint $\operatorname{lfp} f$ such that $f(\operatorname{lfp} f) = \operatorname{lfp} f$ and for all $x \in \mathcal{D}^\tau$, $f(x) = x$ implies $\operatorname{lfp} f \sqsubseteq^\tau x$. $\operatorname{lfp} f = \sqcup_{n \in \mathbb{N}}^\tau f^n(\bot^\tau)$ where $f^{n+1}(x) \overset{\text{def}}{=} f(f^n(x))$ and $f^0(x) \overset{\text{def}}{=} x$.

$$\begin{array}{rcll}
[\![x^\tau]\!]\rho & \overset{\text{def}}{=} & \rho(x^\tau) & \\
[\![c^\tau]\!]\rho & \overset{\text{def}}{=} & \underline{c}^\tau & \\
[\![e_1^{\texttt{bool}} \;?\; e_2^\tau : e_3^\tau]\!]\rho & \overset{\text{def}}{=} & \bot^\tau & \text{if } [\![e_1^{\texttt{bool}}]\!]\rho = \bot^{\texttt{bool}} \\
 & \overset{\text{def}}{=} & [\![e_2^\tau]\!]\rho & \text{if } [\![e_1^{\texttt{bool}}]\!]\rho = \mathit{tt} \\
 & \overset{\text{def}}{=} & [\![e_3^\tau]\!]\rho & \text{if } [\![e_1^{\texttt{bool}}]\!]\rho = \mathit{ff} \\
[\![\langle e_1^{\tau'},\, e_2^{\tau''} \rangle]\!]\rho & \overset{\text{def}}{=} & \langle [\![e_1^{\tau'}]\!]\rho,\, [\![e_2^{\tau''}]\!]\rho \rangle & \\
[\![\texttt{fst}\, e^{\tau \times \tau'}]\!]\rho & \overset{\text{def}}{=} & \pi^1 \circ [\![e^{\tau \times \tau'}]\!]\rho & \\
[\![\texttt{snd}\, e^{\tau' \times \tau}]\!]\rho & \overset{\text{def}}{=} & \pi^2 \circ [\![e^{\tau \times \tau'}]\!]\rho & \\
[\![\lambda x^{\tau'} \bullet e^{\tau''}]\!]\rho & \overset{\text{def}}{=} & \lambda v \in \mathcal{D}^{\tau'} \bullet [\![e^{\tau''}]\!]\rho[x^{\tau'} \leftarrow v] & \\
[\![e_1^{\tau' \mapsto \tau} e_2^{\tau'}]\!]\rho & \overset{\text{def}}{=} & \operatorname{app}([\![e_1^{\tau' \mapsto \tau}]\!]\rho,\, [\![e_2^{\tau'}]\!]\rho) & \\
[\![\mu x^\tau \bullet e^\tau]\!]\rho & \overset{\text{def}}{=} & \operatorname{lfp} \lambda v \in \mathcal{D}^\tau \bullet [\![e^\tau]\!]\rho[x^\tau \leftarrow v] &
\end{array}$$

where:

$$\begin{array}{rcl}
\underline{c}^\tau \in \mathcal{D}^\tau & & \text{is the value of } c^\tau \\
\pi^1(\langle x,\, y \rangle) & \overset{\text{def}}{=} & x \\
\pi^2(\langle x,\, y \rangle) & \overset{\text{def}}{=} & y \\
\operatorname{app}(f,\, x) & \overset{\text{def}}{=} & f(x)
\end{array}$$

Figure 1: Synopsis of the denotational semantics $[\![e^\tau]\!]$

An *environment* $\rho \in \mathcal{E}$ is a map of variables $x^\tau \in \mathcal{V}$ to values $\rho(x^\tau) \in \mathcal{D}^\tau$. If $v \in \mathcal{D}^\tau$ then we write $\rho[x^\tau \leftarrow v]$ for $\rho' \in \mathcal{E}$ such that $\rho'(x^\tau) = v$ and $\rho'(y^{\tau'}) = \rho(y^{\tau'})$ whenever $x^\tau \neq y^{\tau'}$. The set of environments is $\mathcal{E} \overset{\text{def}}{=} \mathcal{V} \mapsto \mathcal{D}$ where $\mathcal{D} \overset{\text{def}}{=} \cup_\tau \mathcal{D}^\tau$ is the set of values for all types.

The denotational semantics $[\![e^\tau]\!]\rho \in \mathcal{D}^\tau$ of expression e of type τ in environment ρ is defined in fig. 1.

11: Collecting semantics of the simply typed lambda calculus

11.1: Basic collecting semantics

Basic concrete questions asked about expressions e^τ have the form *"Does $[\![e^\tau]\!]\rho$ belong to R for all $\rho \in \theta$?"* for given sets of environments $\theta \in \wp(\mathcal{E})$ and for given sets of possible results $R \in \wp(\mathcal{D}^\tau)$ that is "$\forall \rho \in \theta : [\![e^\tau]\!]\rho \in R$". For example the strictness question in variable $x^{\tau'}$ corresponds to $\theta = \{\rho[x^{\tau'} \leftarrow \bot^{\tau'}] \mid \rho \in \mathcal{E}\}$ and $R = \{\bot^\tau\}$ where $\bot^\tau \in \mathcal{D}^\tau$ denotes non-termination for objects of type τ. By defining the collecting semantics $\{\!| e^\tau |\!\}$ as:

$$\begin{array}{rcll}
\{\!| e^\tau |\!\} & \in & \langle \wp(\mathcal{E}) \mapsto \wp(\mathcal{D}^\tau);\, \dot{\subseteq} \rangle & (23) \\
\{\!| e^\tau |\!\}\theta & \overset{\text{def}}{=} & \{ [\![e^\tau]\!]\rho \mid \rho \in \theta \} &
\end{array}$$

or equivalently $\{\!|e^\tau|\!\} = \alpha^\varsigma([\![e^\tau]\!])$ the question can be reformulated in the form considered in Sect. 1, that is "$\{\!|e^\tau|\!\}\theta \subseteq R$?" or equivalently "$\forall \vartheta \in \wp(\mathcal{E}) : \{\!|e^\tau|\!\}\vartheta \subseteq Q(\vartheta)$?" that is "$\{\!|e^\tau|\!\} \;\dot{\subseteq}^\tau\; Q$?" where $\{\!|e^\tau|\!\}$ and $Q = \lambda\vartheta\bullet(\vartheta = \theta \;?\; R : \mathcal{D}^\tau)$ belong to the *domain of concrete properties* $\mathcal{P}^\tau \stackrel{\text{def}}{=} \wp(\mathcal{E}) \mapsto \wp(\mathcal{D}^\tau)$ which is a complete lattice $\langle \mathcal{P}^\tau;\ \dot{\subseteq}^\tau,\ \dot{\emptyset}^\tau,\ \dot{\Upsilon}^\tau,\ \dot{\cup}^\tau,\ \dot{\cap}^\tau\rangle$ with pointwise subset inclusion partial ordering $\dot{\subseteq}^\tau$, infimum $\dot{\emptyset}^\tau = \lambda X \bullet \emptyset$, supremum $\dot{\Upsilon}^\tau = \lambda X \bullet \mathcal{D}^\tau$, pointwise union $\dot{\cup}^\tau$ and pointwise intersection $\dot{\cup}^\tau$.

11.2: More general collecting semantics

The notion of collecting semantics is relative to a set of questions. It defines exactly which questions can be answered about programs. These questions can take numerous forms. Different forms of questions usually correspond to different forms of collecting semantics. For example, another collecting semantics for the standard semantics $[\![e^\tau]\!]$ would be:

$$\begin{array}{rcl} \{\!|e^\tau|\!\} & \in & \langle \wp(\mathcal{E} \mapsto \mathcal{D}^\tau);\ \subseteq\rangle \\ \{\!|e^\tau|\!\} & \stackrel{\text{def}}{=} & \{[\![e^\tau]\!]\} \end{array} \qquad (24)$$

With (24), the absence property *"the value of e^τ does not depend upon the variable $x^{\tau'}$"* can be formulated in the form considered in Sect. 1 as:

$$\{\!|e^\tau|\!\} \subseteq \{\varphi \mid \forall \rho \in \mathcal{E} : \forall v,\ v' \in \mathcal{D}^\tau : \varphi(\rho[x^{\tau'} \leftarrow v]) = \varphi(\rho[x^{\tau'} \leftarrow v'])\}$$

Yet another form of collecting semantics would be:

$$\begin{array}{rcl} \{\!|e^\tau|\!\} & \in & \langle \wp(\wp(\mathcal{E})) \mapsto \wp(\wp(\mathcal{D}^\tau));\ \dot{\subseteq}^{\mathbb{W}}\rangle \\ \{\!|e^\tau|\!\} & \stackrel{\text{def}}{=} & \lambda\Theta\bullet\{\{[\![e^\tau]\!]\rho \mid \rho \in C\} \mid C \in \Theta\} \end{array} \qquad (25)$$

(25) is well suited for PER analysis (and avoids resorting to sets of pairs of values [41, 42]) since "$\{\!|E^\tau|\!\} \;\dot{\subseteq}^{\mathbb{W}}\; \varphi$" corresponds to the question:

$$\forall \Theta : \forall C \in \Theta : \exists C' \in \varphi(\Theta) : \forall \rho \in C : [\![e^\tau]\!]\rho \in C'$$

12: Abstraction of the basic collecting semantics

As in Sect. 11.1, we consider an abstract semantics $(\!|e^\tau|\!)$ belonging to the complete lattice $\langle \mathcal{P}^\tau_a;\ \dot{\subseteq}^\tau_a,\ \dot{\emptyset}^\tau_a,\ \dot{\Upsilon}^\tau_a,\ \dot{\cup}^\tau_a,\ \dot{\cap}^\tau_a\rangle$. The correspondence between concrete and abstract properties is given by means of a Galois connection:

$$\langle \mathcal{P}^\tau;\ \dot{\subseteq}^\tau\rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \mathcal{P}^\tau_a;\ \dot{\subseteq}^\tau_a\rangle$$

where $\mathcal{P}^\tau = \wp(\mathcal{E}) \mapsto \wp(\mathcal{D}^\tau)$ is defined as in (23).

12.1: Pointwise abstraction of sets of environments

For each type τ, let be given an abstraction of sets of values of type τ:

$$\begin{array}{c} \langle \wp(\mathcal{D}^\tau);\ \subseteq,\ \emptyset,\ \mathcal{D}^\tau,\ \cup,\ \cap\rangle \\ \underset{\alpha^\tau}{\overset{\gamma^\tau}{\leftrightarrows}} \\ \langle \mathcal{D}^\tau_a;\ \subseteq^\tau,\ \emptyset^\tau,\ \Upsilon^\tau,\ \cup^\tau,\ \cap^\tau\rangle \end{array} \qquad (26)$$

The abstraction of sets $\theta \in \wp(\mathcal{E})$ of environments, that is of sets of functions in $\wp(\mathcal{V} \mapsto \mathcal{D})$, can be done pointwise as in (8), that is by means of an abstract environment $\Theta \in \mathcal{E}_a$, associating an abstract value $\Theta(x^\tau) \in \mathcal{D}^\tau_a$ to each variable $x^\tau \in \mathcal{V}$:

$$\begin{array}{rcl} \mathcal{E}_a & \stackrel{\text{def}}{=} & \{\Theta \mid \forall x^\tau \in \mathcal{V} : \Theta(x^\tau) \in \mathcal{D}^\tau_a\} \qquad (27) \\ \alpha^{\mathcal{E}}(\theta) & \stackrel{\text{def}}{=} & \lambda x^\tau \bullet \alpha^\tau(\{\rho(x^\tau) \mid \rho \in \theta\}) \\ \gamma^{\mathcal{E}}(\Theta) & \stackrel{\text{def}}{=} & \{\rho \in \mathcal{E} \mid \forall x^\tau \in \mathcal{V} : \rho(x^\tau) \in \gamma^\tau(\Theta(x^\tau))\} \end{array}$$

$$\langle \wp(\mathcal{E});\ \subseteq,\ \emptyset,\ \mathcal{E},\ \cup,\ \cap\rangle \underset{\alpha^{\mathcal{E}}}{\overset{\gamma^{\mathcal{E}}}{\leftrightarrows}} \langle \mathcal{E}_a;\ \subseteq^{\mathcal{E}}_a,\ \emptyset^{\mathcal{E}}_a,\ \Upsilon^{\mathcal{E}}_a,\ \cup^{\mathcal{E}}_a,\ \cap^{\mathcal{E}}_a\rangle$$

12.2: Functional abstraction of the basic collecting semantics

Following [15, 16] the abstraction of the collecting semantics is defined by induction on the structure of its domain of definition. For example since $\mathcal{P}^\tau = \wp(\mathcal{E}) \xrightarrow{\cup} \wp(\mathcal{D}^\tau)$ we can use abstract interpretations of environments (27) and of values (26) and use the functional abstraction of set-transformers of (6):

$$\alpha(\phi) \stackrel{\text{def}}{=} \alpha^\tau \circ \phi \circ \gamma^{\mathcal{E}} \qquad \gamma(\Phi) \stackrel{\text{def}}{=} \gamma^\tau \circ \Phi \circ \alpha^{\mathcal{E}} \qquad (28)$$

so as to obtain an abstract interpretation of collecting semantics in $\mathcal{P}^\tau = \wp(\mathcal{E}) \xrightarrow{\cup} \wp(\mathcal{D}^\tau)$ by an abstract semantics in $\mathcal{P}^\tau_a \stackrel{\text{def}}{=} \mathcal{E}_a \xrightarrow{\emptyset,\subseteq} \mathcal{D}^\tau_a$:

$$\langle \mathcal{P}^\tau; \dot{\subseteq}^\tau, \dot{\emptyset}^\tau, \dot{\Upsilon}^\tau, \dot{\cup}^\tau, \dot{\cap}^\tau\rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \mathcal{P}^\tau_a; \dot{\subseteq}^\tau_a, \dot{\emptyset}^\tau_a, \dot{\Upsilon}^\tau_a, \dot{\cup}^\tau_a, \dot{\cap}^\tau_a\rangle$$

The correctness proof can be done in one of the following forms:

$$\begin{array}{ll} & \alpha(\{\!|e^\tau|\!\}) \;\dot{\subseteq}^\tau_a\; (\!|e^\tau|\!) \\ \Longleftrightarrow & [\text{by Def. (28)}] \\ & \alpha^\tau \circ \{\!|e^\tau|\!\} \circ \gamma^{\mathcal{E}} \;\dot{\subseteq}^\tau_a\; (\!|e^\tau|\!) \\ \Longleftrightarrow & [\text{by def. of } \dot{\subseteq}^\tau_a] \\ & \forall \Theta \in \mathcal{E}_a : \alpha^\tau(\{\!|e^\tau|\!\}(\gamma^{\mathcal{E}}(\Theta))) \subseteq^\tau_a (\!|e^\tau|\!)\Theta \\ \Longleftrightarrow & [\text{by Def. (1)}] \\ & \forall \Theta \in \mathcal{E}_a : \{\!|e^\tau|\!\}(\gamma^{\mathcal{E}}(\Theta)) \subseteq \gamma^\tau((\!|e^\tau|\!)\Theta) \\ \Longleftrightarrow & [\text{by Def. (23)}] \\ & \forall \Theta \in \mathcal{E}_a : \{[\![e^\tau]\!]\rho \mid \rho \in \gamma^{\mathcal{E}}(\Theta)\} \subseteq \gamma^\tau((\!|e^\tau|\!)\Theta) \\ \Longleftrightarrow & [\text{by def. of } \subseteq] \\ & \forall \Theta \in \mathcal{E}_a : \forall \rho \in \gamma^{\mathcal{E}}(\Theta) : [\![e^\tau]\!]\rho \in \gamma^\tau((\!|e^\tau|\!)\Theta) \end{array}$$

We say that $(\!|e^\tau|\!)_1$ is *better* than $(\!|e^\tau|\!)_2$ if and only if $\gamma((\!|e^\tau|\!)_1) \;\dot{\subseteq}^\tau\; \gamma((\!|e^\tau|\!)_2)$. Observe that $\alpha(\{\!|e^\tau|\!\})$ is the *best* abstract interpretation with respect to the abstraction $\langle\alpha,\ \gamma\rangle$ whence provides a guideline for designing $(\!|e^\tau|\!)$, a definite advantage of the Galois connection approach to abstract interpretation [14, 16] over its variant formalization using logical relations [2, 56].

13: Basic comportment abstraction

13.1: Abstraction of basic types

In basic comportment analysis we partition $\mathcal{D}^\beta$ into two blocks $\{\perp^\beta\}$ and $\mathcal{D}^\beta \setminus \{\perp^\beta\}$. We use the isomorphic coding by $\mathcal{D}^\beta_\mathcal{B} = \{\varnothing, \perp, \not\perp, \top\}$ where $\varnothing \cong \emptyset$ is the abstraction of the empty set $\emptyset$, $\perp \cong \{\{\perp^\beta\}\}$ is the abstraction of infinite behaviors, $\not\perp \cong \{\mathcal{D}^\beta \setminus \{\perp^\beta\}\}$ is the abstraction of non-$\perp^\beta$ (usually finite) behaviors i.e. of any set of basic values not containing $\perp^\beta$ while $\top \cong \{\{\perp^\beta\}, \mathcal{D}^\beta \setminus \{\perp^\beta\}\}$ is the abstraction of all possible behaviors i.e. of any subset of $\mathcal{D}^\beta$. This is formalized by the abstraction $\alpha^\beta_\mathcal{B}$:

$$\begin{array}{rclrcl} \alpha^\beta_\mathcal{B}(\emptyset) & \stackrel{\text{def}}{=} & \varnothing & \alpha^\beta_\mathcal{B}(X) & \stackrel{\text{def}}{=} & \not\perp \text{ if } \perp^\beta \notin X \neq \emptyset \\ \alpha^\beta_\mathcal{B}(\{\perp^\beta\}) & \stackrel{\text{def}}{=} & \perp & \alpha^\beta_\mathcal{B}(X) & \stackrel{\text{def}}{=} & \top \text{ if } \{\perp^\beta\} \subset X \end{array}$$

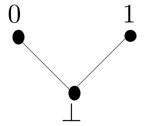
and concretization $\gamma^\beta_\mathcal{B}$:

$$\begin{array}{rclrcl} \gamma^\beta_\mathcal{B}(\varnothing) & \stackrel{\text{def}}{=} & \emptyset & \gamma^\beta_\mathcal{B}(\not\perp) & \stackrel{\text{def}}{=} & \mathcal{D}^\beta \setminus \{\perp^\beta\} \\ \gamma^\beta_\mathcal{B}(\perp) & \stackrel{\text{def}}{=} & \{\perp^\beta\} & \gamma^\beta_\mathcal{B}(\top) & \stackrel{\text{def}}{=} & \mathcal{D}^\beta \end{array}$$

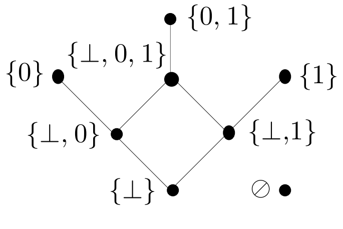
which form a Galois connection:

$$\langle \wp(\mathcal{D}^\beta); \subseteq, \emptyset, \mathcal{D}^\beta, \cup, \cap\rangle \underset{\alpha^\beta_\mathcal{B}}{\overset{\gamma^\beta_\mathcal{B}}{\leftrightarrows}} \langle \mathcal{D}^\beta_\mathcal{B}; \subseteq^\beta_\mathcal{B}, \varnothing, \top, \cup^\beta_\mathcal{B}, \cap^\beta_\mathcal{B}\rangle$$

for the *approximation ordering* $\subseteq^\beta_\mathcal{B}$. Scott ordering $\sqsubseteq$ on $\mathcal{D}^\beta$ is extended to the collecting semantics $\wp(\mathcal{D}^\beta)$ as in (10). For example, the extension of the flat ordering is Egli-Milner ordering (with $\emptyset$ isolated):

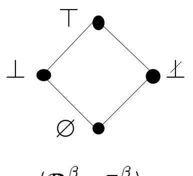


$\langle \mathcal{D}^\beta; \sqsubseteq\rangle$

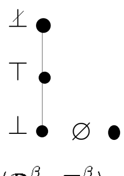


$\langle \wp(\mathcal{D}^\beta); \alpha^\rho(\sqsubseteq)\rangle$

This set relator is further abstracted by (13). Since Scott ordering is reflexive and $\perp^\beta$ is the infimum, the abstraction by $\langle \alpha^\beta_\mathcal{B}, \gamma^\beta_\mathcal{B}\rangle$ leads to the *computation ordering* which is a complete lattice $\langle \mathcal{D}^\beta_\mathcal{B}; \sqsubseteq^\beta_\mathcal{B}, \perp^\beta_\mathcal{B}, \top^\beta_\mathcal{B}, \sqcup^\beta_\mathcal{B}, \sqcap^\beta_\mathcal{B}\rangle$. The approximation and computation orderings are defined by the following Hasse diagrams:



$\langle \mathcal{D}^\beta_\mathcal{B}; \subseteq^\beta_\mathcal{B}\rangle$



$\langle \mathcal{D}^\beta_\mathcal{B}; \sqsubseteq^\beta_\mathcal{B}\rangle$

The computational ordering is an abstraction of Scott's ordering in the sense that $X \sqsubseteq^\beta_\mathcal{B} Y$ if and only if Y possibly describes more (finite) behaviors in $\mathcal{D}^\beta \setminus \{\perp^\beta\}$ and less infinite behaviors (in $\{\perp^\beta\}$) than X.

13.2: Abstraction of pair types

In the basic comportment abstract interpretation, the analysis of pairs is dependence-free. Given abstract interpretations for the components:

$$\begin{array}{c} \langle \wp(\mathcal{D}^\tau); \subseteq, \emptyset, \mathcal{D}^\tau, \cup, \cap\rangle \\ \underset{\alpha^\tau_\mathcal{B}}{\overset{\gamma^\tau_\mathcal{B}}{\leftrightarrows}} \\ \langle \mathcal{D}^\tau_\mathcal{B}; \subseteq^\tau_\mathcal{B}, \emptyset^\tau_\mathcal{B}, \Upsilon^\tau_\mathcal{B}, \cup^\tau_\mathcal{B}, \cap^\tau_\mathcal{B}\rangle \end{array}$$

$$\begin{array}{c} \langle \wp(\mathcal{D}^{\tau'}); \subseteq, \emptyset, \mathcal{D}^{\tau'}, \cup, \cap\rangle \\ \underset{\alpha^{\tau'}}{\overset{\gamma^{\tau'}}{\leftrightarrows}} \\ \langle \mathcal{D}^{\tau'}_\mathcal{B}; \subseteq^{\tau'}_\mathcal{B}, \emptyset^{\tau'}_\mathcal{B}, \Upsilon^{\tau'}_\mathcal{B}, \cup^{\tau'}_\mathcal{B}, \cap^{\tau'}_\mathcal{B}\rangle \end{array}$$

the abstract interpretation of pair types (i.e. sets of pairs i.e. relations):

$$\begin{array}{c} \langle \wp(\mathcal{D}^{\tau\times\tau'}); \subseteq, \emptyset, \mathcal{D}^{\tau\times\tau'}, \cup, \cap\rangle \\ \underset{\alpha^{\tau\times\tau'}_\mathcal{B}}{\overset{\gamma^{\tau\times\tau'}_\mathcal{B}}{\leftrightarrows}} \\ \langle \mathcal{D}^{\tau\times\tau'}_\mathcal{B}; \subseteq^{\tau\times\tau'}_\mathcal{B}, \emptyset^{\tau\times\tau'}_\mathcal{B}, \Upsilon^{\tau\times\tau'}_\mathcal{B}, \cup^{\tau\times\tau'}_\mathcal{B}, \cap^{\tau\times\tau'}_\mathcal{B}\rangle \end{array}$$

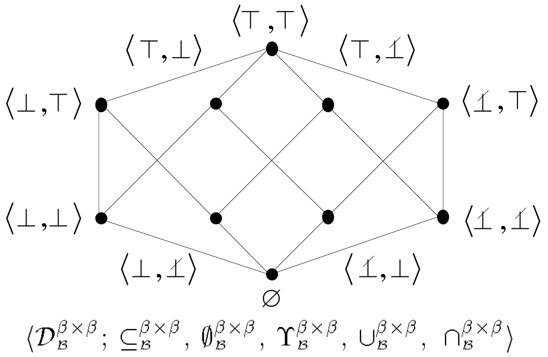
is defined componentwise as $\langle \alpha^\otimes, \gamma^\otimes\rangle \circ \langle \alpha^\times, \gamma^\times\rangle$ defined in (17) and (16):

$$\begin{array}{rcl} \langle x, y\rangle \subseteq^{\tau\times\tau'}_\mathcal{B} \langle x', y'\rangle & \stackrel{\text{def}}{=} & x \subseteq^\tau_\mathcal{B} x' \wedge y \subseteq^{\tau'}_\mathcal{B} y' \\ \alpha^{\tau\times\tau'}_\mathcal{B}(X) & \stackrel{\text{def}}{=} & \langle \alpha^\tau_\mathcal{B}(\Pi_1(X)), \alpha^{\tau'}(\Pi_2(X))\rangle \\ \gamma^{\tau\times\tau'}_\mathcal{B}(\langle x, y\rangle) & \stackrel{\text{def}}{=} & \gamma^\tau_\mathcal{B}(x) \times \gamma^{\tau'}(y) \end{array}$$

Since the abstract computation ordering $\sqsubseteq^{\tau\times\tau'}_\mathcal{B}$ is also defined componentwise:

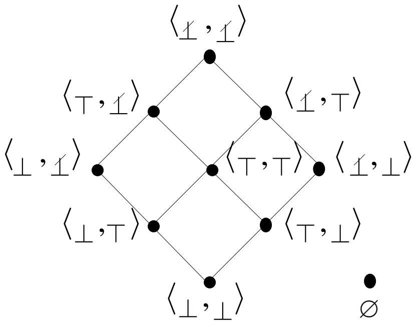
$$\langle x, y\rangle \sqsubseteq^{\tau\times\tau'}_\mathcal{B} \langle x', y'\rangle \quad \stackrel{\text{def}}{=} \quad x \sqsubseteq^\tau_\mathcal{B} x' \wedge y \sqsubseteq^{\tau'}_\mathcal{B} y'$$

the complete lattice structure $\langle \mathcal{D}^{\tau\times\tau'}_\mathcal{B}; \sqsubseteq^{\tau\times\tau'}_\mathcal{B}, \perp^{\tau\times\tau'}_\mathcal{B}, \top^{\tau\times\tau'}_\mathcal{B}, \sqcup^{\tau\times\tau'}_\mathcal{B}, \sqcap^{\tau\times\tau'}_\mathcal{B}\rangle$ is also preserved. For example, the abstraction of sets of pairs of values of basic types is:



$\langle \mathcal{D}^{\beta\times\beta}_\mathcal{B}; \subseteq^{\beta\times\beta}_\mathcal{B}, \emptyset^{\beta\times\beta}_\mathcal{B}, \Upsilon^{\beta\times\beta}_\mathcal{B}, \cup^{\beta\times\beta}_\mathcal{B}, \cap^{\beta\times\beta}_\mathcal{B}\rangle$

With the approximation ordering $\subseteq_{\mathcal{B}}^{\beta\times\beta}$, $\langle\perp,\ \mathsf{T}\rangle$ and $\langle\mathsf{T},\ \not\perp\rangle$ are not comparable since, for the first component, $\perp$ represents less possible values than T, while for the second component, T represents more possible values than $\not\perp$.


$\langle\not\perp,\not\perp\rangle$
$\langle\top,\not\perp\rangle$
$\langle\not\perp,\top\rangle$
$\langle\perp,\not\perp\rangle$
$\langle\top,\top\rangle$
$\langle\not\perp,\perp\rangle$
$\langle\perp,\top\rangle$
$\langle\top,\perp\rangle$
$\langle\perp,\perp\rangle$
$\varnothing$

$$\langle\mathcal{D}_{\mathcal{C}}^{\beta\times\beta};\ \sqsubseteq_{\mathcal{B}}^{\beta\times\beta},\ \perp_{\mathcal{B}}^{\beta\times\beta},\ \top_{\mathcal{B}}^{\beta\times\beta},\ \sqcup_{\mathcal{B}}^{\beta\times\beta},\ \sqcap_{\mathcal{B}}^{\beta\times\beta}\rangle$$

With the computation ordering $\subseteq_{\mathcal{B}}^{\beta\times\beta}$, $\langle\perp,\ \mathsf{T}\rangle$ and $\langle\mathsf{T},\ \not\perp\rangle$ are comparable since, for the first component, T represents more possible finite behaviors than $\perp$, while for the second component, $\not\perp$ represents less possible infinite behaviors than T.

13.3: Abstraction of function types

For function types $\mathcal{D}^{\tau\mapsto\tau'} = \mathcal{D}^{\tau} \xmapsto{\sqsubseteq} \mathcal{D}^{\tau'}$, we use the abstraction (9). By induction, the relations $\sqsubseteq^{\tau}\in \mathcal{D}^{\tau} \leftrightarrow \mathcal{D}^{\tau}$ and $\sqsubseteq^{\tau'}\in \mathcal{D}^{\tau'} \leftrightarrow \mathcal{D}^{\tau'}$ have been extended to the collecting semantics $\wp(\mathcal{D}^{\tau})$ and $\wp(\mathcal{D}^{\tau'})$ by (10) and then to their abstractions $\sqsubseteq_{\mathcal{B}}^{\tau} \in \mathcal{D}_{\mathcal{B}}^{\tau} \leftrightarrow \mathcal{D}_{\mathcal{B}}^{\tau}$ and $\sqsubseteq_{\mathcal{B}}^{\tau'} \in \mathcal{D}_{\mathcal{B}}^{\tau'} \leftrightarrow \mathcal{D}_{\mathcal{B}}^{\tau'}$ by (13), so that, by (11) and (14), abstract functions f must be pointwise monotonic:

$$\forall\langle x,\ y\rangle \in \mathcal{D}^{\tau'} \times \mathcal{D}^{\tau'} : (x \sqsubseteq_{\mathcal{B}}^{\tau} y) \Rightarrow (f(x) \sqsubseteq_{\mathcal{B}}^{\tau'} f(y))$$

Hence $\mathcal{D}_{\mathcal{B}}^{\tau\mapsto\tau} = \mathcal{D}_{\mathcal{B}}^{\tau} \xmapsto{\sqsubseteq,\emptyset,\subseteq} \mathcal{D}_{\mathcal{B}}^{\tau'}$. We get the following Galois connection:

$$\langle\wp(\mathcal{D}^{\tau\mapsto\tau'});\ \subseteq,\ \emptyset,\ \mathcal{D}^{\tau}\mapsto\mathcal{D}^{\tau'},\ \cup,\ \cap\rangle \qquad (29)$$
$$\underset{\alpha^{\tau\mapsto\tau'}}{\overset{\gamma_{\mathcal{B}}^{\tau\mapsto\tau'}}{\leftrightarrows}}$$
$$\langle\mathcal{D}_{\mathcal{B}}^{\tau\mapsto\tau};\ \dot{\subseteq}_{\mathcal{B}}^{\tau'},\ \dot{\emptyset}_{\mathcal{B}}^{\tau'},\ \dot{\Upsilon}_{\mathcal{B}}^{\tau'},\ \dot{\cup}_{\mathcal{B}}^{\tau'},\ \dot{\cap}_{\mathcal{B}}^{\tau'}\rangle$$

For example $\mathcal{D}_{\mathcal{B}}^{\beta\mapsto\beta}$ is given below. We see that $f = [\phi\mapsto\phi,\ \perp\mapsto\not\perp,\ \not\perp\mapsto\perp,\ \mathsf{T}\mapsto\mathsf{T}]$ is not in $\mathcal{D}_{\mathcal{B}}^{\beta\mapsto\beta}$ set since $\perp\sqsubseteq_{\mathcal{B}}\not\perp$ but not $f(\perp)\sqsubseteq_{\mathcal{B}} f(\not\perp))$. e_2 [] e_3 stands for the non-deterministic choice (or, in our deterministic language, for an expression e_1 ? e_2 : e_3 where the analysis of e_1 returns T):

divergence (e.g. $\mu f\bullet\lambda x\bullet f(x)$):

$$\begin{aligned}\textbf{div} &\overset{\text{def}}{=} [\phi\mapsto\phi,\ \perp\mapsto\perp,\ \not\perp\mapsto\perp,\ \mathsf{T}\mapsto\perp]\\ \gamma_{\mathcal{B}}^{\beta\mapsto\beta}(\textbf{div}) &= \{\varphi \mid \forall x\in\mathcal{D}^{\beta} : \varphi(x) = \perp\}\end{aligned}$$

identity (e.g. $\lambda x\bullet x$):

$$\begin{aligned}\textbf{ide} &\overset{\text{def}}{=} [\phi\mapsto\phi,\ \perp\mapsto\perp,\ \not\perp\mapsto\not\perp,\ \mathsf{T}\mapsto\mathsf{T}]\\ \gamma_{\mathcal{B}}^{\beta\mapsto\beta}(\textbf{ide}) &= \{\varphi \mid \forall x\in\mathcal{D}^{\beta} : \varphi(x) = \perp \Leftrightarrow x = \perp\}\end{aligned}$$

strictness (e.g. $\mu f\bullet\lambda x\bullet(x = 0\ ?\ 0 : f(x-1))$):

$$\begin{aligned}\textbf{str} &\overset{\text{def}}{=} [\phi\mapsto\phi,\ \perp\mapsto\perp,\ \not\perp\mapsto\mathsf{T},\ \mathsf{T}\mapsto\mathsf{T}]\\ \gamma_{\mathcal{B}}^{\beta\mapsto\beta}(\textbf{str}) &= \{\varphi \mid \varphi(\perp) = \perp\}\end{aligned}$$

convergence (e.g. $\lambda x\bullet 1$):

$$\begin{aligned}\textbf{con} &\overset{\text{def}}{=} [\phi\mapsto\phi,\ \perp\mapsto\not\perp,\ \not\perp\mapsto\not\perp,\ \mathsf{T}\mapsto\not\perp]\\ \gamma_{\mathcal{B}}^{\beta\mapsto\beta}(\textbf{con}) &= \{\varphi \mid \forall x\in\mathcal{D}^{\beta} : \varphi(x) \neq \perp\}\end{aligned}$$

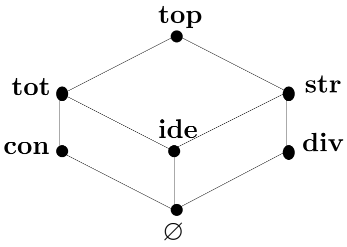
totality (e.g. $\lambda x\bullet(1\ []\ x)$):

$$\begin{aligned}\textbf{tot} &\overset{\text{def}}{=} [\phi\mapsto\phi,\ \perp\mapsto\mathsf{T},\ \not\perp\mapsto\not\perp,\ \mathsf{T}\mapsto\mathsf{T}]\\ \gamma_{\mathcal{B}}^{\beta\mapsto\beta}(\textbf{tot}) &= \{\varphi \mid \forall x\in\mathcal{D}^{\beta}\setminus\{\perp\} : \varphi(x) \neq \perp\}\end{aligned}$$

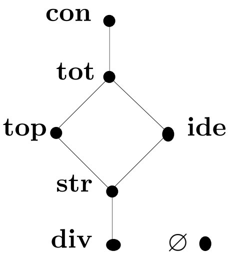
truth (e.g. $\mu f\bullet\lambda x\bullet(1\ []\ (x = 0\ ?\ 0 : f(x-1)))$):

$$\begin{aligned}\textbf{top} &\overset{\text{def}}{=} [\phi\mapsto\phi,\ \perp\mapsto\mathsf{T},\ \not\perp\mapsto\mathsf{T},\ \mathsf{T}\mapsto\mathsf{T}]\\ \gamma_{\mathcal{B}}^{\beta\mapsto\beta}(\textbf{top}) &= \mathcal{D}^{\beta\mapsto\beta}\end{aligned}$$

The approximation ordering is $\langle\mathcal{D}_{\mathcal{B}}^{\beta\mapsto\beta};\ \dot{\subseteq}_{\mathcal{B}}^{\beta}\rangle$:



Using (11) and (14) once again, the pointwise Scott-ordering $\dot{\sqsubseteq}'$ on $\mathcal{D}^{\tau\mapsto\tau'}$ is extended to the computation ordering $\langle\mathcal{D}_{\mathcal{B}}^{\tau\mapsto\tau'};\ \dot{\sqsubseteq}_{\mathcal{B}}^{\beta},\ \dot{\perp}_{\mathcal{B}}^{\tau'},\ \dot{\top}_{\mathcal{B}}^{\tau'},\ \dot{\sqcup}_{\mathcal{B}}^{\tau'},\ \dot{\sqcap}_{\mathcal{B}}^{\tau'}\rangle$. For basic types, $\langle\mathcal{D}_{\mathcal{B}}^{\beta\mapsto\beta};\ \dot{\sqsubseteq}_{\mathcal{B}}^{\beta}\rangle$ is:



13.4: Basic comportment semantics

The basic comportment semantics $(\!|e^{\tau}|\!)_{\mathcal{B}}\Theta \in \mathcal{D}_{\mathcal{B}}^{\tau}$ of expression e of type τ in abstract environment Θ is defined in Fig. 2.

Example 3 (absence) The basic comportment semantics of program $\mu f^{\text{num}\mapsto\text{num}}\bullet\lambda x^{\text{num}}\bullet\texttt{true}\ ?\ 1 : f x$ is:

$$\begin{aligned}&(\!|\mu f\bullet\lambda x\bullet\texttt{true}\ ?\ 1 : f p|\!)_{\mathcal{B}}\Theta =\\ &\qquad \text{lfp}_{\mathcal{B}}^{\text{num}\mapsto\text{num}}\ \lambda\varphi\bullet\lambda\chi\bullet\not\perp\cup_{\mathcal{B}}^{\text{num}}\varphi(\chi)\end{aligned}$$

The iterates are as follows:

$$\begin{aligned}\varphi^0 &= \lambda\chi\bullet\perp\\ \varphi^1 &= \lambda\chi\bullet\not\perp\cup_{\mathcal{B}}^{\text{num}}\varphi^0(\chi) = \lambda\chi\bullet\not\perp\cup_{\mathcal{B}}^{\text{num}}\perp = \lambda\chi\bullet\mathsf{T}\\ \varphi^2 &= \varphi^1\end{aligned}$$

$$
\begin{array}{rcl}
\llparenthesis x^{\tau} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \Theta(x^{\tau}) \\
\llparenthesis c^{\tau} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \alpha^{\tau}_{\mathcal{B}}(\{\underline{c}^{\tau}\}) \\
\llparenthesis e_1^{\mathtt{bool}} \mathbin{?} e_2^{\tau} : e_3^{\tau} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & (\llparenthesis e_1^{\mathtt{bool}} \rrparenthesis_{\mathcal{B}}\Theta = \bot \mathbin{?} \bot^{\tau}_{\mathcal{B}} : \\
 & & \quad (\llparenthesis e_1^{\mathtt{bool}} \rrparenthesis_{\mathcal{B}}\Theta = \top \mathbin{?} \bot^{\tau}_{\mathcal{B}} : \emptyset^{\tau}_{\mathcal{B}}) \\
 & & \quad \cup^{\tau}_{\mathcal{B}} \llparenthesis e_2^{\tau} \rrparenthesis_{\mathcal{B}}\Theta \cup^{\tau}_{\mathcal{B}} \llparenthesis e_3^{\tau} \rrparenthesis_{\mathcal{B}}\Theta) \\
\llparenthesis \langle e_1^{\tau'},\, e_2^{\tau''} \rangle \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \langle \llparenthesis e_1^{\tau'} \rrparenthesis_{\mathcal{B}}\Theta,\, \llparenthesis e_2^{\tau''} \rrparenthesis_{\mathcal{B}}\Theta \rangle \\
\llparenthesis \mathtt{fst}\, e^{\tau \times \tau'} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \pi^1_{\mathcal{B}}(\llparenthesis e^{\tau \times \tau'} \rrparenthesis_{\mathcal{B}}\Theta) \\
\llparenthesis \mathtt{snd}\, e^{\tau' \times \tau} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \pi^2_{\mathcal{B}}(\llparenthesis e^{\tau \times \tau'} \rrparenthesis_{\mathcal{B}}\Theta) \\
\llparenthesis \lambda x^{\tau'} \bullet e^{\tau''} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \lambda v \in \mathcal{D}^{\tau'}_{\mathcal{B}} \bullet \llparenthesis e^{\tau''} \rrparenthesis_{\mathcal{B}}\Theta[x^{\tau'} \leftarrow v] \\
\llparenthesis e_1^{\tau' \mapsto \tau} e_2^{\tau'} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \mathrm{app}(\llparenthesis e_1^{\tau' \mapsto \tau} \rrparenthesis_{\mathcal{B}}\Theta,\, \llparenthesis e_2^{\tau'} \rrparenthesis_{\mathcal{B}}\Theta) \\
\llparenthesis \mu x^{\tau} \bullet e^{\tau} \rrparenthesis_{\mathcal{B}}\Theta & \stackrel{\text{def}}{=} & \mathrm{lfp}^{\tau}_{\mathcal{B}}\, \lambda v \in \mathcal{D}^{\tau}_{\mathcal{B}} \bullet \llparenthesis e^{\tau} \rrparenthesis_{\mathcal{B}}\Theta[x^{\tau} \leftarrow v]
\end{array}
$$

where:

$$
\begin{array}{rcl}
\underline{c}^{\tau} \in \mathcal{D}^{\tau} & & \text{is the value of } c^{\tau} \\
\pi^1_{\mathcal{B}}(\langle x,\, y \rangle) & \stackrel{\text{def}}{=} & x \\
\pi^2_{\mathcal{B}}(\langle x,\, y \rangle) & \stackrel{\text{def}}{=} & y \\
\mathrm{app}(f,\, x) & \stackrel{\text{def}}{=} & f(x) \\
\mathrm{lfp}^{\tau}_{\mathcal{B}}\, \varphi & \stackrel{\text{def}}{=} & \bigsqcup^{\tau}_{\mathcal{B}\; n \in \mathbb{N}} \varphi^n(\bot^{\tau}_{\mathcal{B}})
\end{array}
$$

Figure 2: Synopsis of the basic comportment semantics $\llparenthesis e^{\tau} \rrparenthesis_{\mathcal{B}}$

Absence is not captured by basic comportment analysis. □

Proposition 1 (Correctness)

$$\forall \Theta \in \mathcal{E}_{\mathcal{B}} : \forall \rho \in \gamma^{\mathcal{E}}(\Theta) : [\![e^{\tau}]\!] \rho \in \gamma^{\tau}_{\mathcal{B}}(\llparenthesis e^{\tau} \rrparenthesis_{\mathcal{B}}\Theta) \qquad (30)$$

Observe that the collecting semantics is used as an intermediate step in the design of abstract interpretations for the formalization of program properties and the construction of the abstract semantics (e.g. of the computational ordering) but that no explicit formulation is required for the correctness proof. For example, $\varphi = \lambda v \in \mathcal{D}^{\tau}_{\mathcal{B}} \bullet [\![e^{\tau}]\!] \Theta[x^{\tau} \leftarrow v]$ is $\sqsubseteq$-monotonic in the denotational semantics so that, by (11) and (14), $\Phi = \lambda v \in \mathcal{D}^{\tau}_{\mathcal{B}} \bullet \llparenthesis e^{\tau} \rrparenthesis_{\mathcal{B}}\Theta[x^{\tau} \leftarrow v]$ is $\sqsubseteq^{\tau}_{\mathcal{B}}$-preserving in the abstract semantics. Moreover, by (12) and (15), $\mathrm{lfp}^{\tau}_{\mathcal{B}}\, \Phi$ exists in $\wp(\mathcal{D}^{\tau})$, hence in $\mathcal{D}^{\tau}_{\mathcal{B}}$, and is correct.

13.5: Comparing basic comportments and strictness

The abstraction of basic comportments to strictness properties only yields [10]. However the abstraction of abstract basic comportment properties into abstract strictness properties, as shown in Fig. 3 shows that in strictness analysis the approximation and computational orderings coincide. It follows that [10] do not distinguish between the approximation and the computational orderings, a point of view which is too restricted to make their framework of general scope.

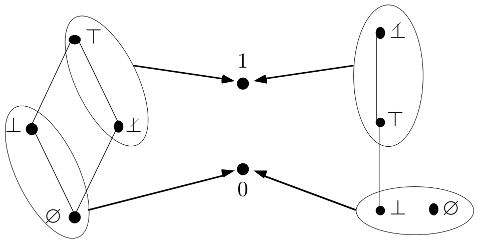


Figure 3: Abstraction of the basic comportment approximation and computation orderings into Mycroft's strictness ordering

13.6: Comparing basic comportments and smash projections

At this point comparison with smash projections $(\forall x \in D_{\bot} : \beta(x) \neq \lightning \Rightarrow \beta(x) = x)$ is easy. For smash projections, $f : \delta \rightarrow \beta$ is equivalent to $\forall x \in \delta^{-1}(\lightning) : f(x) \in \beta^{-1}(\lightning)$ where $\varphi^{-1}(y) = \{y \mid \varphi(x) = y\}$ denotes the inverse image of y by φ. By defining abstract values $\overline{\beta}$ with meaning $\gamma(\overline{\beta}) = \beta^{-1}(\lightning) - \{\lightning\}$, we can express a similar property in the abstract interpretation framework as $f^{\mathcal{B}}(\overline{\delta}) = \overline{\beta}$. For example $\overline{str} = \bot$ and $\overline{fail} = \top$ so that $f : str \rightarrow str$ corresponding to $f^{\mathcal{B}}(\bot) = \bot$ (satisfied by **div**, **ide** and **str**) expresses strictness whereas $f : fail \rightarrow str$ corresponding to $f^{\mathcal{B}}(\top) = \bot$ (satisfied by **div** only) expresses divergence.

The *abs* projection is much more difficult to understand. This may explain why it is excluded from all comparisons available in the literature between projection analysis and abstract interpretation. It cannot be described with basic comportments. For example, `f(x)` $\equiv$ `(f(x)` ▯ `1)` leads to divergence if the first alternative is always chosen or convergence if the second alternative is ever chosen. Its analysis $\mathbf{div} \cup \mathbf{con} = \mathbf{top}$ with basic comportments yields no information on the result of `f`. The problem here is that disjunctions are too approximate.

14: Comportment abstraction

Comportment properties are obtained by completion of basic comportment properties, as explained in Sect. 7.

14.1: The abstract domain of comportments

The collecting semantics of e^{τ} in comportement analysis is $\{\!| e^{\tau} |\!\} \in \wp(\mathcal{E} \mapsto \mathcal{D}^{\tau})$ as defined in (24). The corresponding abstract comportment semantics is:

$$\llparenthesis e^{\tau} \rrparenthesis_{\mathcal{C}} \in \wp^{\mathbb{W}}(\mathcal{E}_{\mathcal{B}} \mapsto \mathcal{D}^{\tau}_{\mathcal{B}})$$

with meaning given by:

$$\gamma^{\mathcal{E}\,\tau}_{\mathcal{C}}(\Gamma) \stackrel{\text{def}}{=} \cup\{\gamma^{\tau}_{\mathcal{B}}(\phi) \mid \phi \in \Gamma\}$$

$$
\begin{aligned}
\llparenthesis x^{\tau} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \{\lambda\Theta\bullet\, \Theta(x^{\tau})\}\\
\llparenthesis c^{\tau} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \{\lambda\Theta\bullet\, \alpha^{\tau}_{\mathcal{B}}(\{\underline{c}^{\tau}\})\}\\
\llparenthesis e_1^{\mathtt{bool}} \mathbin{?} e_2^{\tau} : e_3^{\tau} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\Gamma \mid \exists \Gamma_t \in \llparenthesis e_1^{\mathtt{bool}} \rrparenthesis_{\mathcal{C}} : \exists \Gamma_c \in \llparenthesis e_2^{\tau} \rrparenthesis_{\mathcal{C}}\\
&\qquad \cup\ \llparenthesis e_3^{\tau} \rrparenthesis_{\mathcal{C}} :\ \Gamma = \lambda\Theta\bullet(\Gamma_t(\Theta)\\
&\qquad =\ \bot \mathbin{?} \bot^{\tau}_{\mathcal{C}} : (\Gamma_t(\Theta) = \top \mathbin{?}\\
&\qquad \bot^{\tau}_{\mathcal{C}} : \emptyset^{\tau}_{\mathcal{C}}) \cup^{\tau}_{\mathcal{C}}\ \Gamma_c(\Theta))\})\\
\llparenthesis \langle e_1^{\tau'},\ e_2^{\tau''} \rangle \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\lambda\Theta\bullet\, \langle \Gamma_1(\Theta),\ \Gamma_2(\Theta)\rangle \mid\\
&\qquad \Gamma_1 \in \llparenthesis e_1^{\tau'} \rrparenthesis_{\mathcal{C}} \wedge \Gamma_2 \in \llparenthesis e_2^{\tau''} \rrparenthesis_{\mathcal{C}}\})\\
\llparenthesis \mathtt{fst}\, e^{\tau\times\tau'} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\lambda\Theta\bullet\, \pi^1_{\mathcal{C}}(\Gamma(\Theta)) \mid \Gamma \in \llparenthesis e^{\tau\times\tau'} \rrparenthesis_{\mathcal{C}}\})\\
\llparenthesis \mathtt{snd}\, e^{\tau'\times\tau} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\lambda\Theta\bullet\, \pi^2_{\mathcal{C}}(\Gamma(\Theta)) \mid \Gamma \in \llparenthesis e^{\tau'\times\tau} \rrparenthesis_{\mathcal{C}}\})\\
\llparenthesis \lambda x^{\tau'}\bullet\, e^{\tau''} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\lambda\Theta\bullet\, \lambda v \in \mathcal{D}^{\tau'}_{\mathcal{B}}\bullet\, \Gamma(\Theta[x^{\tau'} \leftarrow v]) \mid\\
&\qquad \Gamma \in \llparenthesis e^{\tau''} \rrparenthesis_{\mathcal{C}}\})\\
\llparenthesis e_1^{\tau'\mapsto\tau} e_2^{\tau'} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\lambda\Theta\bullet\, \mathrm{app}(\Gamma_1(\Theta),\ \Gamma_2(\Theta)) \mid\\
&\qquad \Gamma_1 \in \llparenthesis e_1^{\tau'\mapsto\tau} \rrparenthesis_{\mathcal{C}} \wedge \Gamma_2 \in \llparenthesis e_2^{\tau'} \rrparenthesis_{\mathcal{C}}\})\\
\llparenthesis \mu x^{\tau}\bullet\, e^{\tau} \rrparenthesis_{\mathcal{C}} &\stackrel{\text{def}}{=} \mathbb{W}(\{\lambda\Theta\bullet\, \mathrm{lfp}^{\tau}_{\mathcal{B}}\, \lambda v \in \mathcal{D}^{\tau}_{\mathcal{B}}\bullet\\
&\qquad \Gamma(\Theta[x^{\tau} \leftarrow v]) \mid \Gamma \in \llparenthesis e^{\tau} \rrparenthesis_{\mathcal{C}}\})
\end{aligned}
$$

Figure 4: Synopsis of the comportment semantics $\llparenthesis e^{\tau} \rrparenthesis_{\mathcal{C}}$

$$
\begin{aligned}
= \quad & \{\varphi \in \mathcal{E} \mapsto \mathcal{D}^{\tau} \mid \exists \phi \in \Gamma : \forall \Theta \in \mathcal{E}_{\mathcal{B}} :\\
& \forall \rho \in \gamma^{\mathcal{E}}(\Theta) : \varphi(\rho) \in \gamma^{\tau}_{\mathcal{B}}(\Gamma(\Theta))\}
\end{aligned}
$$

Comportment analysis is more precise than basic comportment analysis since:

$$
\langle \wp^{\mathbb{W}}(\mathcal{E}_{\mathcal{B}} \mapsto \mathcal{D}^{\tau}_{\mathcal{B}});\ \dot{\subseteq}^{\tau\,\mathbb{W}}_{\mathcal{B}} \rangle \ \underset{\alpha^{\tau}_{\mathcal{C}\mathcal{B}}}{\overset{\gamma^{\tau}_{\mathcal{C}\mathcal{B}}}{\leftrightarrows}}\ \langle \mathcal{E}_{\mathcal{B}} \mapsto \mathcal{D}^{\tau}_{\mathcal{B}};\ \dot{\subseteq}^{\tau}_{\mathcal{B}} \rangle
$$

where:

$$
\begin{aligned}
\alpha^{\tau}_{\mathcal{C}\mathcal{B}}(\Gamma) \quad &\stackrel{\text{def}}{=} \quad \lambda\Theta\bullet \cup^{\tau}_{\mathcal{B}}\{C(\Theta) \mid C \in \Gamma\}\\
\gamma^{\tau}_{\mathcal{C}\mathcal{B}}(\phi) \quad &\stackrel{\text{def}}{=} \quad \{\phi\}
\end{aligned}
$$

For expression e^{τ} without free variables, $\wp^{\mathbb{W}}(\mathcal{E}_{\mathcal{B}} \mapsto \mathcal{D}^{\tau}_{\mathcal{B}})$ is isomorphic with $\wp^{\mathbb{W}}(\mathcal{D}^{\tau}_{\mathcal{B}})$. Using (18), elements of $\wp^{\mathbb{W}}(\mathcal{D}^{\tau}_{\mathcal{B}})$ with the same meaning:

$$
\gamma^{\tau}_{\mathcal{C}}(\Xi) \quad \stackrel{\text{def}}{=} \quad \cup\{\gamma^{\tau}_{\mathcal{B}}(\chi) \mid \chi \in \Xi\}
$$

can be identified. In this way, for basic types, $\wp^{\mathbb{W}}(\mathcal{D}^{\beta}_{\mathcal{B}})$ is isomorphic with $\mathcal{D}^{\beta}_{\mathcal{B}}$. However, at higher order, $\wp^{\mathbb{W}}(\mathcal{D}^{\tau}_{\mathcal{B}})$ is more expressive than $\mathcal{D}^{\tau}_{\mathcal{B}}$. For example, the complete lattice $\langle \mathcal{D}^{\beta\mapsto\beta}_{\mathcal{C}};\ \subseteq^{\beta\mapsto\beta}_{\mathcal{C}},\ \emptyset^{\beta\mapsto\beta}_{\mathcal{C}},\ \Upsilon^{\beta\mapsto\beta}_{\mathcal{C}},\ \cup^{\beta\mapsto\beta}_{\mathcal{C}},\ \cap^{\beta\mapsto\beta}_{\mathcal{C}} \rangle$ resulting from the reduction of the crown completion of the lattice $\mathcal{D}^{\beta\mapsto\beta}_{\mathcal{B}}$ is given in Fig. 5. The corresponding computation ordering $\langle \mathcal{D}^{\beta\mapsto\beta}_{\mathcal{C}};\ \sqsubseteq^{\beta\mapsto\beta}_{\mathcal{C}},\ \bot^{\beta\mapsto\beta}_{\mathcal{C}},\ \top^{\beta\mapsto\beta}_{\mathcal{C}},\ \sqcup^{\beta\mapsto\beta}_{\mathcal{C}},\ \sqcap^{\beta\mapsto\beta}_{\mathcal{C}} \rangle$ is given in Fig. 6. For example in $\mathcal{D}^{\beta\mapsto\beta}_{\mathcal{C}}$, $\{\mathbf{con}, \mathbf{tot}\} = \{[\emptyset \mapsto \emptyset,\ \bot \mapsto \not\bot,\ \not\bot \mapsto \not\bot,\ \top \mapsto \not\bot],\ [\emptyset \mapsto \emptyset,\ \bot \mapsto \top,\ \not\bot \mapsto \not\bot,\ \top \mapsto \top]\}$ has the same meaning as $\{\mathbf{tot}\} = \{[\emptyset \mapsto \emptyset,\ \bot \mapsto \top,\ \not\bot \mapsto \not\bot,\ \top \mapsto \top]\}$ whereas $\{\mathbf{ide}, \mathbf{div}\} \neq \{\mathbf{str}\}$ since in the first case the behavior is the same for all the values of the parameter (as in `f(x)` $\equiv$ `(x` $\Box$ `f(x))`) whereas in the second case the behavior of the function may be different for different values of the parameter (as in `f(x)` $\equiv$ `(x = 0 ? x : f(x - 1))`).

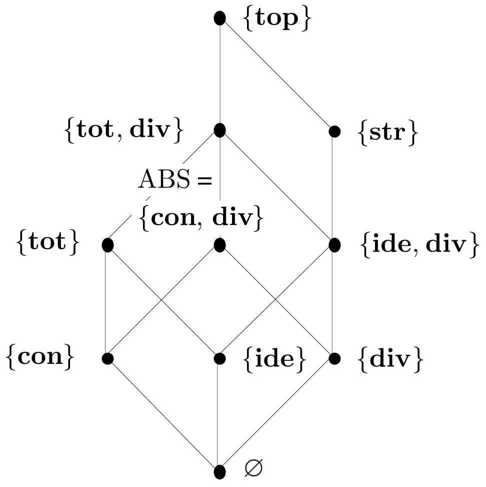


Figure 5: Approximation ordering of $\mathcal{D}^{\beta\mapsto\beta}_{\mathcal{C}}$

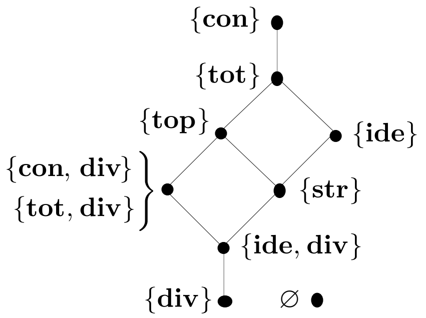


Figure 6: Computation ordering of $\mathcal{D}^{\beta\mapsto\beta}_{\mathcal{C}}$

14.2: Comportment semantics

The comportment semantics is defined in Fig. 4. Various approximations are possible to speed up the analysis at the cost of a loss of precision. For example, $\llparenthesis \mu x^{\tau}\bullet\, e^{\tau} \rrparenthesis_{\mathcal{C}} \stackrel{\text{def}}{=} \lambda\Theta\bullet\, \mathrm{lfp}^{\tau}_{\mathcal{B}}\, \lambda v \in \mathcal{D}^{\tau}_{\mathcal{B}}\bullet(\cup^{\tau}_{\mathcal{B}} \llparenthesis e^{\tau} \rrparenthesis_{\mathcal{C}})(\Theta[x^{\tau} \leftarrow v])$ would be correct but not optimal.

Example 4 (absence) The comportment semantics of program $\mu f^{\mathrm{num}\mapsto\mathrm{num}}\bullet\, \lambda x^{\mathrm{num}}\bullet\, \mathtt{true} \mathbin{?} 1 : fx$ is:

$$
\begin{aligned}
&\llparenthesis \mu f\bullet\, \lambda x\bullet\, \mathtt{true} \mathbin{?} 1 : fp \rrparenthesis_{\mathcal{C}} =\\
&\quad \{\lambda\Theta\bullet\, \mathrm{lfp}^{\mathrm{num}\mapsto\mathrm{num}}_{\mathcal{B}}\, \lambda\varphi \in \mathcal{D}^{\mathrm{num}\mapsto\mathrm{num}}_{\mathcal{B}}\bullet\, \Gamma(\Theta[f \leftarrow \varphi]) \mid\\
&\qquad\qquad \Gamma \in \{\lambda\varphi\bullet\, \lambda v\bullet\, \not\bot,\ \lambda\varphi\bullet\, \lambda v\bullet\, \varphi(v)\}\}
\end{aligned}
$$

that is $\{\lambda\Theta\bullet\, \lambda v\bullet\, \not\bot,\ \lambda\Theta\bullet\, \lambda v\bullet\, \bot\}$, so that absence is captured by comportment analysis. □

Proposition 2 (Correctness)

$$
\{\!|e^{\tau}|\!\} \quad \subseteq \quad \gamma^{\mathcal{E}\tau}_{\mathcal{C}}(\llparenthesis e^{\tau} \rrparenthesis_{\mathcal{C}})
$$

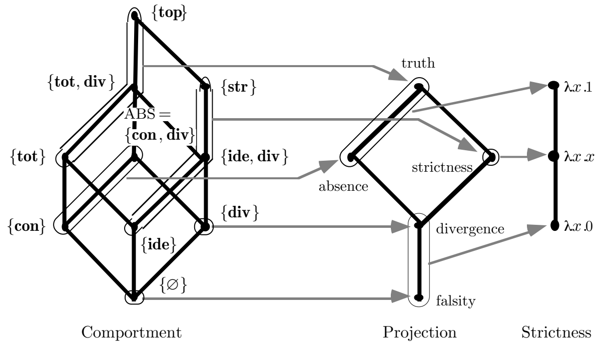


Figure 7: Abstraction of comportment analysis into projection and strictness analysis

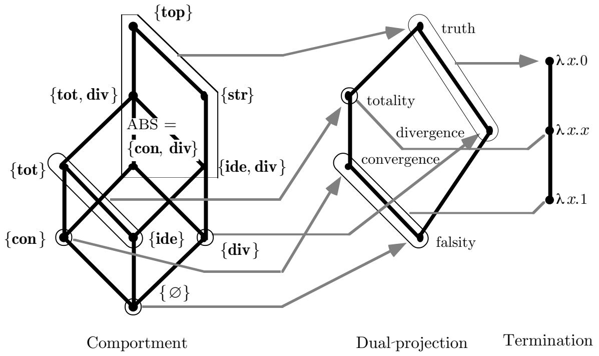


Figure 8: Abstraction of comportment analysis into dual projection and termination analysis

14.3: Projection analysis as an abstract interpretation

To show that comportment analysis generalizes projection analysis and that projection analysis can be done by abstract interpretation it is sufficient to exhibit an abstraction into the lattice of properties expressible by projections, as shown in Fig. 7. A further abstraction to strictness properties yields [44, 45]. Another abstraction, shown in Fig. 8, yields dual projections and termination analysis. By choosing a finer partition of $\mathcal{D}^{\beta}$, comportment analysis can easily be enriched, e.g. to take possible values of variables into account.

15: Summary and conclusions

We have shown that the abstract interpretation of a simply typed lambda calculus defined by its standard semantics can be defined by the method introduced in [14, 15, 16], that is by compositional abstraction of a collecting semantics using structured approximations based Galois connections defining a best approximation. This was possible in a set-theoretic framework since there is no necessity for providing a domain-based denotational definition of this collecting semantics[2] and indeed no explicit definition is needed in correctness proofs since the correctness of the standard semantics with respect to the (implicit) collecting semantics is a general result in the framework.

The application to comportment analysis generalizes strictness, termination, projection, dual-projection and PER-analysis. The abstract semantics leads to a system of equations which, in practice, must be solved efficiently. This would consist in using a compact representation of properties (using e.g. sets of generators of atoms for comportment analysis) and convergence acceleration methods [15]. Another problem beyond the scope of that paper is the usefulness of comportment analysis which can only be shown by practical experience.

As far as the methodological aspects are concerned, our approach is rather different from the other abstract interpretation frameworks based upon denotational semantics. In particular, we distinguish between the approximation and computation orderings and interpret them completely differently. The approximation ordering, does not exist in the standard semantics. It corresponds to logical implication of program properties which is fundamental in the definition of the approximation by Galois connections. The computation ordering happens to pre-exist in the standard semantics under the form of Scott's ordering. It is induced in the abstract domain through the Galois connections. Any other predicate, relation, etc. preexisting in the standard semantics could be abstracted in a similar way. Therefore our approach is tied up neither to a particular syntactical form of languages (or meta-languages [52, 60]), nor to a particular style for specifying the semantics such as denotational semantics, nor to a specific programming style such as functional programming, nor to a specific typing scheme, etc. It is directly applicable e.g. to a non-deterministic functional language with relational semantics [22] as well as to logic programming [19] with operational semantics.

This should be contrasted with the relational framework for abstract interpretation [56] which attempts to solve the problem of defining a collecting semantics in denotational style by completely evading the approximation ordering and overemphasizing the computation ordering, so that, e.g., the notion of best approximation completely disappears. Moreover, for logical relations [2], the approximation process is tied up with the standard computation ordering and the type system in the abstraction process. Application to logic programming with e.g. declarative semantics then becomes a bit tortuous. Moreover, it freezes approximation to a few paradigms (such as "approximate pairs by pairs", "approximate functions by functions") which should leave the place to a broader palette of possible choices, such as "approximate functions by pairs, functions, relations, ..., up to a Galois connection) as abundantly illustrated in this paper. For example an abstract interpretation framework should

[2] An explicit inductive definition of the collecting semantics could be given in G^{∞}SOS [21].

not enforce function properties to be necessarily of the form $\wp(\mathcal{D}_1) \mapsto \wp(\mathcal{D}_2)$ since we have seen that $\wp(\mathcal{D}_1 \mapsto \mathcal{D}_2)$ is more general and sometimes required. Choosing $\wp(\mathcal{D}_1) \mapsto \wp(\mathcal{D}_2)$, or a powerdomain form thereof [52, 57, 53, 60], introduces an initial approximation in the development of the abstract interpretation framework from which it is later very hard to recover. Galois connections themselves, which enforce the existence of a best approximation, can sometimes be too constraining. Such constraints can be lifted by using concretization functions only. However, by loosening up the connection too much, all fundamental theorems of abstract interpretation are lost. This problem of finding a reasonable balance between full generality and strong properties of abstract interpretation frameworks is discussed in [20].

16: Bibliographical references

[1] S. Abramsky. Strictness analysis and polymorphic invariance (extended abstract). In LNCS 217, pp. 1–23. Springer-Verlag, 1986.

[2] S. Abramsky. Abstract interpretation, logical relations and Kan extensions. *J. Logic and Comp.*, 1(1):5–40, 1990.

[3] S. Abramsky & C. Hankin, eds. *Abstract Interpretation of Declarative Languages.* Computers and their Applications. Ellis Horwood, 1987.

[4] S. Abramsky & P. J. Jensen. A relational approach to strictness analysis for higher-order polymorphic functions. In *18th POPL*, pp. 49–54, 1991. ACM Press.

[5] G. Baraki. A note on abstract interpretation of polymorphic functions. In LNCS 523, pp. 367–378. Springer-Verlag, 1991.

[6] G. Birkhoff. On the structure of abstract algebras. *Math. Proc. Cambridge Philos. Soc.*, 31:433–454, 1935.

[7] G. Birkhoff. *Lattice Theory*, vol. 25 of *Colloquium publications.* AMS, third ed., 1973.

[8] G. L. Burn. A relationship between abstract interpretation and projection analysis (extended abstract). In *17th POPL*, pp. 151–156, 1990. ACM Press.

[9] G. L. Burn. *Lazy Functional Languages: Abstract Interpretation and Compilation.* Research Monographs in Parallel and Distributed Computing. Pitman and MIT Press, 1991.

[10] G. L. Burn, C. L. Hankin, & S. Abramsky. Strictness analysis of higher-order functions. *Sci. Comput. Prog.*, 7:249–278, 1986.

[11] M. Coppo & A. Ferrari. Type inference, abstract interpretation and strictness analysis. *TCS*, 121:113–143, 1993.

[12] P. Cousot. *Méthodes itératives de construction et d'approximation de points fixes d'opérateurs monotones sur un treillis, analyse sémantique de programmes.* Thèse d'État ès sciences mathématiques, Université scientifique et médicale de Grenoble, Grenoble, (F), 21 Mar. 1978.

[13] P. Cousot. Semantic foundations of program analysis. In S. S. Muchnick & N. D. Jones, eds., *Program Flow Analysis: Theory and Applications*, ch. 10, pp. 303–342. Prentice-Hall, 1981.

[14] P. Cousot & R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *4th POPL*, pp. 238–252, 1977. ACM Press.

[15] P. Cousot & R. Cousot. Static determination of dynamic properties of recursive procedures. In E. Neuhold, ed., *IFIP Conference on Formal Description of Programming Concepts*, St-Andrews, N.B., Canada, pp. 237–277. North-Holland, 1977.

[16] P. Cousot & R. Cousot. Systematic design of program analysis frameworks. In *6th POPL*, pp. 269–282, 1979. ACM Press.

[17] P. Cousot & R. Cousot. Semantic analysis of communicating sequential processes. In LNCS 85, pp. 119–133. Springer-Verlag, July 1980.

[18] P. Cousot & R. Cousot. Invariance proof methods and analysis techniques for parallel programs. In A. W. Biermann, G. Guiho, & Y. Kodratoff, eds., *Automatic Program Construction Techniques*, ch. 12, pp. 243–271. Macmillan, 1984.

[19] P. Cousot & R. Cousot. Abstract interpretation and application to logic programs. *J. Logic Prog.*, 13(2–3):103–179, 1992.

[20] P. Cousot & R. Cousot. Abstract interpretation frameworks. *J. Logic and Comp.*, 2(4):511–547, 1992.

[21] P. Cousot & R. Cousot. Inductive definitions, semantics and abstract interpretation. In *19th POPL*, pp. 83–94, 1992. ACM Press.

[22] P. Cousot & R. Cousot. Galois connection based abstract interpretations for strictness analysis. In LNCS 735, pp. 98–127. Springer-Verlag, 1993.

[23] M.-R. Croisot. Applications résiduées. *Ann. Sci. École Norm. Sup. (4)*, 3ème série, 73, fasc. 4:453–474, 1956.

[24] K. Davis. Higher order binding time analysis. In *Proc. PEPM'93*, Copenhagen, (DK), 14–16 June 1993, pp. 80–87. ACM Press, 1993.

[25] K. Davis & P. Wadler. Backwards strictness analysis: Proved and improved. In Proc. *Functional Programming, Glasgow 1989*, Springer-Verlag and BCS, 1989.

[26] K. Davis & P. Wadler. Strictness analysis in 4D. In Proc. *Functional Programming, Glasgow 1990*, pp. 23–43. Springer-Verlag and BCS, 1990.

[27] A. Deutsch. An operational model of strictness properties and its abstraction. In In Proc. *Functional Programming, Glasgow 1991*, Springer-Verlag, 1991.

[28] P. Dubreuil & R. Croisot. Propriétés générales de la résiduation en liaison avec les correspondances de Galois. *Collect. Math.*, 7:193–203, 1954.

[29] P. Dybjer. Inverse image analysis generalizes strictness analysis. *Inf. & Comp.*, 90:194–216, 1991.

[30] C. Ernoult & A. Mycroft. Uniform ideals and strictness analysis. In LNCS 510, pp. 47–59. Springer-Verlag, July 1991.

[31] C. J. Everett. Closure operators and Galois theory in lattices. *Trans. Amer. Math. Soc.*, 55:514–525, 1944.

[32] C. A. Gunter & D. S. Scott. Semantic domains. In J. van Leeuwen, ed., *Formal Models and Semantics*,

vol. B of *Handbook of Theoretical Computer Science*, ch. 12, pp. 633–674. Elsevier, 1990.

[33] C. V. Hall & D. S. Wise. Compiling strictness into streams. In *14th POPL*, pp. 132–143, 1987. ACM Press.

[34] P. Hudak & J. Young. Higher-order strictness analysis in untyped lambda calculus. In *12th POPL*, pp. 97–109, 1986. ACM Press.

[35] J. Hughes & J. Launchbury. Relational reversal of abstract interpretations. *J. Logic and Comp.*, 2(4):465–509, 1992.

[36] J. Hughes & J. Launchbury. Reversing abstract interpretations. In LNCS 582, pp. 269–286. Springer-Verlag, 1992.

[37] R. J. M. Hughes. Strictness detection in non-flat domains. In LNCS 217, pp. 112–135. Springer-Verlag, 1986.

[38] R. J. M. Hughes. Backwards analysis of functional programs. In D. Bjørner, A. P. Ershov, & N. D. Jones, eds., *Partial Evaluation and Mixed Computation*, Proceedings IFIP TC2 Workshop, Gl Avernæs, Ebberup, 18–24 Oct. 1987, (DK), pp. 187–208. Elsevier, 1988.

[39] R. J. M. Hughes. Projections for polymorphic strictness analysis. In LNCS 389, pp. 82–100. Springer-Verlag, 1989.

[40] R. J. M. Hughes & J. Launchbury. Projections for polymorphic first-order strictness analysis. *MSCS*, 2:301–326, 1993.

[41] S. Hunt. PERs generalize projections for strictness analysis. Technical Report DOC 14/90, Department of Computing, Imperial College, London, U.K., 1990.

[42] S. Hunt. PERs generalize projections for strictness analysis. In Proc. *Functional Programming, Glasgow 1990*, Springer-Verlag and BCS, 1990.

[43] S. Hunt & D. Sands. Binding time analysis: A new PERspective. SIGPLAN Notices 26(9), pp. 154–165. ACM Press, 1991.

[44] T. P. Jensen. Strictness analysis in logical form. In LNCS 523, pp. 352–366. Springer-Verlag, 1991.

[45] T. P. Jensen. Abstract interpretation in logical form. PhD Thesis, Report 93/11, DIKU, University of Copenhagen (DK), 1993.

[46] T. Johnsson. Detecting when call-by-value can be used instead of call-by-need. Research Report LPM MEMO 14, Laboratory for Programming Methodology, Department of Computer Science, Chalmers University of Technology, S-412 96 Göteborg, (S), 1981.

[47] N. D. Jones & S. S. Muchnick. Complexity of flow analysis, inductive assertion synthesis and a language due to Dijkstra. In S. S. Muchnick & N. D. Jones, eds., *Program Flow Analysis: Theory and Applications*, ch. 12, pp. 380–393. Prentice-Hall, 1981.

[48] G. Kildall. A unified approach to global program optimization. In *1st POPL*, pp. 194–206, Boston, Mass., 1973. ACM Press.

[49] T. M. Kuo & P. Mishra. On strictness and its analysis. In *14th POPL*, pp. 144–155, 1987. ACM Press.

[50] J. Launchbury. Projections for specialization. In D. Bjørner, A. P. Ershov, & N. D. Jones, eds., *Partial Evaluation and Mixed Computation*, Proceedings IFIP TC2 Workshop, Gl Avernæs, Ebberup, 18–24 Oct. 1987, (DK), pp. 299–315. Elsevier, 1988.

[51] J. Launchbury. *Projection Factorizations in Partial Evaluation*, vol. 1 of *Distinguished Dissertations in Computer Science.* Cambridge U. Press, 1991.

[52] K. Marriott. Frameworks for abstract interpretation. *Acta Inf.*, 30:103–129, 1993.

[53] R. Muller & Y. Zhou. Abstract interpretation in weak powerdomains. *LISP Pointers*, 5(1):119–126, 1992.

[54] A. Mycroft. The theory and practice of transforming call-by-need into call-by-value. In LNCS 83, pp. 270–281. Springer-Verlag, 1980.

[55] A. Mycroft. *Abstract Interpretation and Optimising Transformations for Applicative Programs.* Ph.D. Dissertation, CST-15-81, Department of Computer Science, University of Edinburgh, Edinburgh, Scot., 1981.

[56] A. Mycroft & N. D. Jones. A relational framework for abstract interpretation. In LNCS 215, pp. 156–171. Springer-Verlag, 1986.

[57] A. Mycroft & F. Nielson. Strong abstract interpretation using power domains. In LNCS 154, pp. 536–547. Springer-Verlag, 1983.

[58] M. Neuberger & P. Mishra. A precise relationship between the deductive power of forward and backward strictness analysis. *LISP Pointers*, 5(1):127–138, 1992.

[59] F. Nielson. Strictness analysis and denotational abstract interpretation. *Inf. & Comp.*, 76(1):29–92, 1988.

[60] F. Nielson. Two-level semantics and abstract interpretation. *TCS — Fund. St.*, 69:117–242, 1989.

[61] O. Ore. Galois connexions. *Trans. Amer. Math. Soc.*, 55:493–513, 1944.

[62] P. Wadler. Strictness analysis aids time analysis. In *15th POPL*, pp. 119–132, 1988. ACM Press.

[63] P. L. Wadler. Strictness analysis on non-flat domains (by abstract interpretation over finite domains). In S. Abramsky & C. Hankin, eds., *Abstract Interpretation of Declarative Languages*, Computers and their Applications, ch. 12, pp. 266–275. Ellis Horwood, 1987.

[64] P. L. Wadler & R. J. M. Hughes. Projections for strictness analysis. In LNCS 274, pp. 385–407. Springer-Verlag, 1987.

[65] D. A. Wright. A new technique for strictness analysis. In LNCS 494, pp. 236–258. Springer-Verlag, 1991.

Proceedings of the 1994 International Conference on Computer Languages, May 16–19, 1994, Toulouse, France, IEEE Computer Society Press, Los Alamitos, California, pp. 95–112.