Sound up-to techniques and Complete abstract domains

Filippo Bonchi[1], Roberto Giacobazzi[2], and Dusko Pavlovic[3]

[1] University of Pisa, Italy
[2] INSERT AFFILIATION
[3] INSERT AFFILIATION

Abstract. Abstract interpretation has been introduced by Cousot as a method to automatically find invariants of programs or pieces of code whose semantics is given via least fixed-points. Up-to techniques have been introduced by Milner in the early 80ies as enhancements of coinduction, an abstract principle to prove properties expressed as greatest fixed-points.
While abstract interpretation is always sound by definition, the soundness of up-to techniques needs some ingenuity to be proven. For completeness, the setting in switched: up-to techniques are always complete, while abstract domains are not.
In this work we show that, under reasonable assumptions, there is an evident connection between sound up-to techniques and complete abstract domains.

Keywords: Confluence, DPO rewriting systems, adhesive categories, PROPs, string diagrams.

1 Introduction

2 Preliminaries and notation

We use $(L, \sqsubseteq)$, $(L_1, \sqsubseteq_1)$, $(L_2, \sqsubseteq_2)$ to range over complete lattices and x, y, z to range over their elements. We omit the ordering $\sqsubseteq$ whenever unecessary. As usual $\bigsqcup$ and $\bigsqcap$ denote greatest lower bound and least upper bound, $\sqcup$ and $\sqcap$ denote join and meet, $\top$ and $\bot$ top and bottom.

Hereafter we will consider just monotone maps, so we will usually omit to specify that they are monotone. Obviously, the identitiy map $id\colon L \to L$ and the composition $f \circ g\colon L_1 \to L_3$ of two monotone maps $g\colon L_1 \to L_2$ and $f\colon L_2 \to L_3$ are monotone. Given $l\colon L_1 \to L_2$ and $r\colon L_2 \to L_1$, we say that l is the *left adjoint* of r, or equivalently that r is the *right adjoint* of l, written $l \dashv r\colon L_1 \to L_2$, iff $x \sqsubseteq_1 rl(x)$ and $lr(y) \sqsubseteq_2 y$ for all $x \in L_1$ and $y \in L_2$.

Given a monotone map $f\colon L \to L$, x is said to be a *post-fixed point* iff $x \sqsubseteq f(x)$ and a *pre-fixed point* iff $f(x) \sqsubseteq x$[4]. A *fixed point* iff $x = f(x)$. Pre,

[4] It is also common to find in literature the reversed definitions of post and pre-fixed point. Here we adopted the terminology of [3]

post and fixed points form complete lattices, denoted by $Pre(f)$, $Post(f)$ and $Fix(f)$: we write μf and νf for the least and greatest fixed-point. Recall that if $l \dashv r$, then $x \sqsubseteq r(x)$ iff $l(x) \sqsubseteq l(r(x))$ iff $l(x) \sqsubseteq x$.

For a map $f \colon L \to L$, we inductively define $f^0 = id$ and $f^{n+1} = f \circ f^n$. We fix $f^{\uparrow} = \bigsqcup_{i\in\omega} f^i$ and $f^{\downarrow} = \bigsqcap_{i\in\omega} f^i$. A monotone map $f \colon L \to L$ is an *up-closure* operator if $x \sqsubseteq f(x)$ and $ff(x) \sqsubseteq x$. It is a *down-closure* operator if $f(x) \sqsubseteq x$ and $x \sqsubseteq ff(x)$. For any f, $f^{\uparrow}$ is an up-closure and $f^{\downarrow}$ is a down-closure. Closure operators and pairs of adjoints are in one to one correspondences: for a pair of adjoint $l \dashv r$, $r \circ l$ is an up-closure operator and $l \circ r$ a down-closure operator. Given an up-closure operator $f \colon L \to L$, the functions $l \colon L \to Pre(f)$, defined as $l(x) = \bigsqcap\{y \mid x \sqsubseteq y \sqsupseteq f(y)\}$ is the left adjoint of $r \colon Pre(f) \to L$, defined as $f(x) = x$.

For a monotone map $b \colon L \to L$ on a complete lattice L, the Knaster-Tarski fixed-point theorem characterises μb as the least upper bound of all pre-fixed points of b and νb as the greatest lower bound of all its post-fixed points:

$$\mu b = \bigsqcap\{x \mid b(x) \sqsubseteq x\} \qquad \nu b = \bigsqcup\{x \mid x \sqsubseteq b(x)\}.$$

This immediately leads to the *induction* and *coinduction* proof principles.

$$\frac{\exists x,\ b(x) \sqsubseteq x \sqsubseteq f}{\overline{\mu b \sqsubseteq f}} \qquad \frac{\exists x,\ i \sqsubseteq x \sqsubseteq b(x)}{\overline{i \sqsubseteq \nu b}} \tag{1}$$

Another fixed-point theorem, usually attributed to Kleene, plays an important role in our exposition. It characterises μb and νb as the least upper bound, respectively the greatest lower bound, of the chains

$$\bot \sqsubseteq b(\bot) \sqsubseteq bb(\bot) \sqsubseteq bbb(\bot)\dots \qquad \top \sqsupseteq b(\top) \sqsupseteq bb(\top) \sqsupseteq bbb(\top)\dots \tag{2}$$

In short,

$$\mu b = \bigsqcup_{i\in\omega} b^i(\bot) \qquad \nu b = \bigsqcap_{i\in\omega} b^i(\top)$$

The assumptions are stronger than for Knaster-Tarski: for the leftmost statement, it requires the map b to be *Scott-continuous* (i.e.., it preserves $\bigsqcup$ of directed chains) and, for the rightmost *Scott-cocontinuous* (similar but for $\bigsqcap$). Observe that every left adjoint is Scott-continuous and every right adjoint is Scott-cocontinuous.

3 Coinduction up-to

In order to motivate up-to techniques we find convenient to first illustrate how coinduction can be exploited to check language equivalence of automata.

3.1 Coinduction for Determinitic Automata

A deterministic automaton on the alphabet A is a triple (X, o, t), where X is a set of states, $o\colon X \to 2 = \{0, 1\}$ is the output function, determining if a state x is final ($o(x) = 1$) or not ($o(x) = 0$) and $t\colon X \to X^A$ is the transition function which returns the next state, for each letter $a \in A$.

Every automaton (X, o, t) induces a function $[\![-]\!]\colon X \to 2^{A^*}$ defined for all $x \in X$, $a \in A$ and $w \in A^*$ as $[\![x]\!](\varepsilon) = o(x)$ and $[\![x]\!](aw) = [\![t(x)(a)]\!](w)$. Two states $x, y \in X$ are said to be *language equivalent*, in symbols $x \sim y$, iff $[\![x]\!] = [\![y]\!]$. Alternatively, language equivalence can be defined *coinductively* as the greatest fixed-point of a map b on Rel_X, the lattice of relations over X. For all $R \subseteq X^2$, $b\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ is defined as

$$b(R) = \{(x, y) \mid o(x) = o(y) \text{ and for all } a \in A,\ (t(x)(a), t(y)(a)) \in R\}. \tag{3}$$

Indeed, one can check that b is monotone and that the greatest fixed-point of b, hereafter denoted by νb, coincides with $\sim$.

Thanks to this characterisation, one can prove $x \sim y$ by mean of the coinduction proof principle illustrated in (1). It is enough to provide a relation R that is a *b-simulation*, i.e., a post fixed-point of b, such that $\{(x, y)\} \subseteq R$.

For an example, consider the following deterministic automaton, where final states are overlined and the transition function is represented by labeled arrows. The relation consisting of dashed and dotted lines is a b-simulation witnessing, for instance, that $x \sim u$.

$$\begin{tikzcd}
x \arrow[r, "{a,b}"] \arrow[dd, dash, dashed] & \overline{y} \arrow[r, "{a,b}"] \arrow[d, dash, dashed] \arrow[dr, dash, dotted] & \overline{z} \arrow[loop right, "{a,b}"] \arrow[d, dash, dashed] \arrow[dl, dash, dashed] \\
& \overline{v} \arrow[r, "{a,b}"'] & \overline{w} \arrow[l, bend right, "{a,b}"'] \\
u \arrow[ur, "a"] \arrow[urr, bend right, "b"'] & &
\end{tikzcd} \tag{4}$$

Figure 1 illustrates an algorithm, called `Naive`, that takes in input a DA (X, o, t) and a pair of states (x_1, x_2). It attempts to build a bisimulation R containing (x_1, x_2): if it succeeds, then $x_1 \sim x_2$ and returns true, otherwise returns false.

The worst case complexity of the algorithm `Naive` is linear with the size of the computed bisimulation R. Therefore, it is quadratics with respect to the number of states in X. An optimised version of `Naive` can be defined via up-to techniques.

3.2 Up-to techniques

Coinduction allows to prove $i \sqsubseteq \nu b$ for a given map $b\colon C \to C$ on a complete lattice C and some $i \in C$. Up-to techniques have been introduced in [] as enhancement for coinduction. In a nutshell, an *up to technique* is a monotone map $a\colon C \to C$. A *b-simulation up to a* is a post-fixed point of ba, that is an x such

```
Naive ($x_1, x_2$)

(1) $R := \emptyset$;  $todo := \emptyset$
(2) insert $(x_1, x_2)$ into $todo$
(3) while $todo$ is not empty do
    (3.1)  extract $(x'_1, x'_2)$ from $todo$
    (3.2)  if $(x'_1, x'_2) \in R$ then continue
    (3.3)  if $o(x'_1) \neq o(x'_2)$ then return $false$
    (3.4)  for all $a \in A$,
               insert $(t_a(x'_1), t_a(x'_2))$ into $todo$
    (3.5)  insert $(x'_1, x'_2)$ into $R$
(4) return $true$
```

Fig. 1. Naive algorithm to check the equivalence of states $x_1, x_2 \in X$ for a deterministic automaton (X, o, t).

that $x \sqsubseteq ba(x)$. An up-to technique a is said to be *sound* for b if the following *coinduction up to* principle holds.

$$\frac{\exists x,\ i \sqsubseteq x \sqsubseteq ba(x)}{i \sqsubseteq \nu b} \tag{5}$$

Remark 1 (Completeness of up-to technique). Observe that, according to the above definition an up-to technique a might not be *complete*: it may exist an i such that $i \sqsubseteq \nu b$ for which there is no x satisfying $i \sqsubseteq x \sqsubseteq ba(x)$. However, if a is an up-closure operator, then $\nu b \sqsubseteq a(\nu b)$ and using monotonicity of b, one obtains that $i \sqsubseteq \nu b = b(\nu b) \sqsubseteq b(a(\nu b))$. The question of completeness for up-to techniques has never been raised because they have always been considered up-closure operators (e.g., up-to equivalence, up-to congruence). The main reason for considering arbitrary monotone maps rather than just up-closure operators, comes from the fact that the former allows for more modular proofs of their soundness. This is discussed in more details in Section **??**.

3.3 Hopcroft and Karp's algorithm

For an example of up-to technique, take the function $e\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ mapping every relation $R \subseteq X^2$ to its equivalence closure. We will see in Section **??**, that e is sound for the map b defined in (3). A *b-simulation up to e* is a relation R such that $R \subseteq b(e(R))$. Consider the automaton in (4) and the relation R containing only the dashed lines: since $t(x)(b) = y$, $t(u)(b) = w$ and $(y, w) \notin R$, then $(x, u) \notin b(R)$. This means that R is *not* a b-simulation; however it is a b-simulation up to e, since (y, w) belongs to $e(R)$ and (x, u) to $b(e(R))$.

This example shows that b-simulations up-to e can be smaller than plain b-simulations: this idea is implicitly exploited in the Hopcroft and Karp's algorithm [4] to check language equivalence of deterministic automata. This algorithm can be thought as an optimisation of `Naive`, where line `(3.2)` is replaced with the following.

```
(3.2)   if $(x'_1, x'_2) \in e(R)$ then continue
```

This optimised algorithm skips any pair which is in the equivalence closure of R: during the while loop `(3)`, it always holds that $R \subseteq be(R) \cup todo$. The algorithm returns true only if $R \subseteq be(R)$. This means that R is a b-simulation up to e containing (x_1, x_2).

This simple optimisation allows to reduce the worst case complexity of `Naive`: the size of the returned relation R cannot be larger than n (the number of states). The case of non-deterministic automata is even more impressive: another up-to technique, called *up-to congruence*, allows for an exponential improvement on the performance of algorithms for checking language equivalence [1].

Remark 2. The partition refinement algorithm by Hopcroft [5] computes language equivalence for deterministic automata by constructing the chain defined in (2). The crucial observation, for showing that this chain stabilises after n iterations is that every element of the chain is an equivalence relation. Somehow, the computation of the chain for the greatest fixed point is already up-to equivalence. This fact will be clarified in XXX

Finish here...

4 Abstract Interpretation

We introduce abstract interpretation by showing a simple problem of program analysis.

4.1 A toy program analysis

Consider the following piece of code, where x is an integer value.

```
$x := 5$; while $x > 0$ do { $x := x - 1$; }
```

We want to prove that after exiting the loop x has value 0. Our analysis works on the lattice of predicates over the integers, hereafter denoted by $Pred_Z$, and makes use of the function $\ominus 1 \colon Pred_Z \to Pred_Z$ defined as $P \ominus 1 = \{i - 1 \mid i \in P\}$, for all $P \in \mathsf{Pred}_Z$. We start by annotating the code so to make explicit its control flow.

```
$x := 5;^1$ while $^2 x > 0^3$ do { $x := x - 1;^4$ }$^5$
```

We then write the following system of equations where x^j contains the set of possible values that the variable x can have at the position j.

$$x^1 = \{5\}, \quad x^2 = x^1 \cup x_4, \quad x^3 = x^2 \cap [1, \infty), \quad x^4 = x^3 \ominus 1, \quad x^5 = x^2 \cap (-\infty, 0]$$

For x^2, we obtain the following equation $x^2 = \{5\} \cup ((x^2 \cap [1, \infty)) \ominus 1)$ that has as smallest solution μb where $b \colon Pred_Z \to Pred_Z$ is defined as

$$b(P) = \{5\} \cup ((P \cap [1, \infty)) \ominus 1) \tag{6}$$

for all predicates P. Our initial aim is to check whether $x^5 = x^2 \cap (-\infty, 0] = \mu(i \cup b^*) \cap (-\infty, 0] \subseteq \{0\}$. That is $\mu b \subseteq [0, \infty)$.

We proceed by computing μb as in (2):

$$\emptyset \subseteq \{5\} \subseteq \{5,4\} \subseteq \{5,4,3\} \subseteq \cdots \subseteq \{5,4,3,2,1\} \subseteq \{5,4,3,2,1,0\} \tag{7}$$

Since $\{5,4,3,2,1,0\} \subseteq [0, \infty)$, we have proved our conjecture.

Suppose now to modify the code above, by initialising x to -5, rather than 5. The above analysis rest untouched apart from b thats should be replaced by b' defined as $b'(P) = \{-5\} \cup ((P \cap [1, \infty)) \ominus 1)$. It is rather intuitive that the computation of b' diverges: $\emptyset \subseteq \{-5\} \subseteq \{-5, -6\} \subseteq \dots$

4.2 Abstract domains

The main idea of abstract interpretation is that to check whether $\mu b \sqsubseteq f$ for some $b \colon C \to C$ and $f \in C$, the computation of μb can be carried more effectively in some *abstract domain* A, namely a completely lattice with a pair of adjoints $(\alpha \dashv \gamma) \colon C \to A$. We will usually call abstract domain also the associated closure operator, hereafter denoted by $a \colon C \to C$.

Whenever $a(f) \sqsubseteq f$, that is $f \in A$, one prefers to check $\mu \bar{b} \sqsubseteq_A f$ where the map $\bar{b} \colon A \to A$ represents an approximation of the program b on the abstract domain A. The approximation $\bar{b}$ is said to be *sound* (w.r.t. b) if $\alpha(\mu b) \sqsubseteq_A \mu \bar{b}$ and *complete* if $\alpha(\mu b) = \mu \bar{b}$.

Proposition 1. *[[2], Corollary 7.2.0.4] Let $b, a \colon C \to C$ be a monotone map and a closure operator. The map $\bar{b}^a = \alpha \circ b \circ \gamma$ is* the best sound approximation, *that is: (1) $\bar{b}^a$ is sound and (2) if there exists a complete $\bar{b}$, then $\mu \bar{b} = \mu \bar{b}^a$.*

Is the assumption of Scott continuity necessary? In the paper Cousit say something about isotony but I cannot really understand

The above proposition allows to study soundness and completeness as a property of the abstract domain rather than of the approximation $\bar{b}$: a domain a is said to be *sound* (resp. *complete*) iff $\bar{b}^a$ is sound (complete). In this sense, the initial data for abstract interpretation are the same of those for coinduction up-to: a monotone map b and a up-closure a. But, while for up-to technique soundness should be proved and completeness is given, here the situation is reversed: soundness is given and completeness should be proved. In Section **??**, we will see that the problems of proving soundness of up-to techniques and completeness of abstract domains are intimately related. We first show an example of abstract interpretation at work.

4.3 Abstract Interpretation of a toy program

Consider the abstract domain of signs Sign_Z depicted on the left below, with right adjoint $\gamma\colon \mathsf{Sign}_Z \to \mathsf{Pred}_Z$ defined as on the right below.

$$\begin{tikzcd}
 & Z \arrow[dl, no head] \arrow[d, no head] \arrow[dr, no head] & \\
{\leq}0 \arrow[d, no head] \arrow[dr, no head] & {\neq}0 \arrow[dl, no head] \arrow[dr, no head] & {\geq}0 \arrow[d, no head] \arrow[dl, no head] \\
{<}0 \arrow[dr, no head] & 0 \arrow[d, no head] & {>}0 \arrow[dl, no head] \\
 & \emptyset &
\end{tikzcd}$$

$$\begin{array}{rrcl rrcl}
\gamma\colon & \mathsf{Sign}_Z & \to & \mathsf{Pred}_Z & \gamma\colon & \mathsf{Sign}_Z & \to & \mathsf{Pred}_Z \\
 & Z & \mapsto & Z & & {\neq}0 & \mapsto & (-\infty,1] \cup [1,\infty) \\
 & \emptyset & \mapsto & \emptyset & & 0 & \mapsto & \{0\} \\
 & {\leq}0 & \mapsto & (-\infty,0] & & {\geq}0 & \mapsto & [0,\infty) \\
 & {<}0 & \mapsto & (-\infty,1] & & {>}0 & \mapsto & [1,\infty)
\end{array}$$

For b defined as in (6), its best sound approximation is $\overline{b}^a(P) = {>}0 \sqcup ((P \sqcap {>}0)\overline{\ominus 1}^a)$ where $\overline{\ominus 1}^a$ is again defined as $\alpha \circ \ominus 1 \circ \gamma$, e.g., $\overline{\ominus 1}^a({>}\ 0) = {\geq}\ 0$. The computation of $\mu\overline{b}$ is shorter than the one of μb in (19): $\emptyset\ \sqsubseteq {>}0\ \sqsubseteq {\geq}0\ \sqsubseteq {\geq}0$. Observe that $\gamma(\mu\overline{b}) \subseteq [0,\infty)$. Since $\overline{b}^a$ is sound, we can conclude that $\mu b \subseteq [0,\infty)$.

Consider now the modified b': $\overline{b'}^a(P) = {<}0 \sqcup ((P \sqcap {>}0)\overline{\ominus 1})$. While the computation of $\mu b'$ diverges the one of $\mu\overline{b'}^a$ converges: $\emptyset\ \sqsubseteq {<}0\ \sqsubseteq {<}0$. Observe that $\gamma(\mu\overline{b'}) \not\subseteq [0,\infty)$. Since $\overline{b'}$ is complete, as we will show in Section **??**, we can conclude that $\mu b' \not\subseteq [0,\infty)$.

5 Proving soundness and completeness

In order to prove soundness of up-to techniques and completeness of abstract domains, there have been introduced sufficient conditions, that are of central interest for our work. Both for up-to techniques and abstract domains, these conditions enjoy several equivalent formulations which we report in the following lemma.

Lemma 1. *Let $b\colon C \to C$ be a monotone map and $a\colon C \to C$ an up-colure operator with $(\alpha \dashv \gamma)\colon C \to A$ as associated pair of adjoints.*
The followings are equivalent

1. *$ba = aba$;*
2. *$ab \sqsubseteq ba$ (EM law);*
3. *there exists a $\overline{b}\colon A \to A$ s.t. $\gamma\overline{b} = b\gamma$ (EM lifting).*

$$\begin{tikzcd}
A \arrow[r, "\overline{b}"] \arrow[d, "\gamma"'] & A \arrow[d, "\gamma"] \\
C \arrow[r, "b"'] & C
\end{tikzcd}$$

The followings are equivalent

1. *$ab = aba$;*
2. *$ba \sqsubseteq ab$ (Kl law);*
3. *there exists a $\overline{b}\colon A \to A$ s.t. $\overline{b}\alpha = \alpha b$ (Kl extension).*

$$\begin{tikzcd}
A \arrow[r, "\overline{b}"] & A \\
C \arrow[r, "b"'] \arrow[u, "\alpha"] & C \arrow[u, "\alpha"']
\end{tikzcd}$$

These facts are well known and appear in different places in literature: the reader can conveniently find a proof in Appendix C. If a enjoys the properties in the first part of Lemma 1, it is said to be *compatible* w.r.t. b, if it enjoys the properties in the second part is said *fully complete* (w.r.t. b). Compatibility entails soundness for up-to techniques, while full completeness entails completeness for abstract domains.

Theorem 1 ([7], Theorem 3.6.9). *If a is compatible w.r.t. b, then it is sound w.r.t. b.*

Theorem 2 ([2], Theorem 7.1.0.4). *If a is fully complete w.r.t. b, then it is complete w.r.t. b.*

Is the assumption of Scott continuity necessary? In the paper Cousit say something about isotony but I cannot really understand

It is important to remark here that both theorems state sufficient conditions that are not, in general, necessary. Indeed, both soundness and completeness require certain correspondences at the fixed points, that can happen to hold even when there is no step-wise correspondences between b and a. The name fully complete suggests indeed that the correspondence should hold at any step of the computations of μb and $\mu\bar{b}$. The name *compatibility*, instead have been introduced for up-to techniques to emphasise that these can be composed resulting in further compatible techniques.

Proposition 2 (modularity). *Let $g, h, g_1, g_2, h_1, h_2 \colon C \to C$ be monotone maps on some complete lattice C. Then:*

1. *$id \circ h \sqsubseteq h \circ id$;*
2. *if $g_1 \circ h \sqsubseteq h \circ g_1$ and $g_2 \circ h \sqsubseteq h \circ g_2$, then $(g_1 \circ g_2) \circ h \sqsubseteq h \circ (g_1 \circ g_2)$.*

Moreover:

3. *if $g_1 \circ h \sqsubseteq h \circ g_1$ and $g_2 \circ h \sqsubseteq h \circ g_2$, then $(g_1 \sqcup g_2) \circ h \sqsubseteq h \circ (g_1 \sqcup g_2)$;*
4. *if $g \circ h \sqsubseteq h \circ g$, then $g^{\uparrow} \circ h \sqsubseteq h \circ g^{\uparrow}$.*

Dually:

5. *if $g \circ h_1 \sqsubseteq h_1 \circ g$ and $g \circ h_2 \sqsubseteq h_2 \circ g$, then $g \circ (h_1 \sqcap h_2) \sqsubseteq (h_1 \sqcap h_2) \circ g$;*
6. *if $g \circ h \sqsubseteq h \circ g$, then $g \circ h^{\downarrow} \sqsubseteq h^{\downarrow} \circ g$.*

Proof. The proof is simple but should be written down. □

Remark 3. Observe that, by point 4. of Proposition 2, one can always obtain an up-closure operator. For instance suppose to have two up-closure operator a_1 and a_2 that are both forward complete. Then $a_1 \sqcup a_2$ is still forward complete, but it is not an up-closure. To have an up-closure, one takes $(a_1 \sqcup a_2)^{\uparrow}$ which is ensured to be forward complete by points 3. and 4. of Proposition 2.

to be rewritten and understood: maybe remove

Remark 4. HISTORICAL EXPLANATION

5.1 Proving soundness of equivalence closure

Recall the monotone map $b\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ defined in (3) and the up-closure $e\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ introduced in Section 3.3. In order to prove that the Hopcroft and Karp algorithm is sound one has to relies on the fact that e is sound w.r.t. b. Thanks to Theorem 1, one can prove soundness by showing that e is compatible w.r.t. b. The proof of compatibility can be simplified by using its formulation in point 2 of Lemma 1 and Proposition 2.

The map b can be decomposed as $b = b_* \sqcap f$ where $b_*, f\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ are defined for all relations R as

$$\begin{aligned} b_*(R) &= \{(x,y) \mid \text{ for all } a \in A,\ (t(x)(a), t(y)(a)) \in R\} \\ f(R) &= \{(x_1,x_2) \mid o(x_1) = o(x_2)\} \end{aligned} \tag{8}$$

and the equivalence closure as $e = (id \sqcup r \sqcup s \sqcup t)^{\uparrow}$ where $r, s, t\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ are defined as follows.

$$r(R) = \{(x,x) \mid x \in X\} \qquad s(R) = \{(y,x) \mid (x,y) \in R\}$$

$$t(R) = \{(x,z) \mid \exists y \text{ s.t. } (x,y) \in R \text{ and } (y,z) \in R\}$$

The proof of compatibility of e w.r.t. b can be decomposed by compatibility of e w.r.t. b_* and f and then use Proposition 2.5. Furthermore, to prove that e is compatible w.r.t. b_*, one can prove that r, s, t are compatible w.r.t. b_* and then use points 1,3 and 4 of Proposition 2. For f, it is immediate to check that $ef \subseteq f$.

5.2 Proving completeness of the domain of signs

Call s the up-closure associated with the domain of signs introduced in Section 4.3 and recall $b\colon \mathsf{Pred}_Z \to \mathsf{Pred}_Z$ defined in (6). Thanks to Theorem 2, one can prove s is complete w.r.t. b, by showing that it is fully complete. The proof of full completness can be simplified by using its formulation in point 2 of Lemma 1 and Proposition 2. Note that while for up-to techniques the map b plays the role of h, h_1, h_2 in Proposition 2, for abstract domains, b ranges over g, g_1, g_2.

The map b can be decomposed as $b = i \cup b^*$ where $i, b^*\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ are defined for all predicates P as follows.

$$\begin{aligned} i(P) &= \{5\} \\ b^*(P) &= (P \cap [1,\infty)) \ominus 1 \end{aligned} \tag{9}$$

To prove that s is fully complete w.r.t. b, one can prove that it is fully complete w.r.t. i and b^* and then use Proposition 2. For i, it is enough to observe that any abstract domain a is fully complete with any constant function c (that is $c \sqsubseteq a(c)$). Instead b^* can be further decomposed as $b^* = \ominus 1 \circ (id \cap [1,\infty))$

NO IT DOES NOT WORK: IT IS NOT TRUE THAT s is fully complete w.r.t. b,

6 Relating Abstract Interpretation and Coinduction up-to by adjointness

To relate abstract interpretation and coinduction up-to we need to assume that the monotone map $b\colon C \to C$ is part of an adjunction. Hereafter we denote the adjunction by $b^* \dashv b_* \colon C \to C$. Our idea is based on the observation in [] relating backward and forward completeness:

$$ab^* = ab^*a \quad \text{iff} \quad b_*a = ab_*a.$$

We find convenient to rephrase this as

$$b^*a \sqsubseteq ab^* \quad \text{iff} \quad ab_* \sqsubseteq b_*a. \tag{10}$$

Note however that since b^* is a left adjoint and b_* is a right adjoint one always has that $\mu b^* = \bot$ and $\nu b_* = \top$, meaning that the abstract interpretation and coinduction up-to are always rather trivial.

We then consider some *intial* and *final* conditions $i, f \in C$. With an abuse of notation, we will use this letter also to denote the constant functions $i\colon C \to C$ and $f\colon C \to C$ mapping all elements to, respectively, i and f. We consider the problem of completeness of a w.r.t. $b^* \sqcup i$ and the problem of soundness w.r.t. $b_* \sqcap f$.

The former is entailed by backward completeness

$$(b^* \sqcup i)a \sqsubseteq a(b^* \sqcup i)$$

which, by Proposition 2.3 is entailed by

$$b^*a \sqsubseteq ab^* \quad \text{and} \quad i \sqsubseteq ai. \tag{11}$$

Soundness is entailed by forward completeness (or compatibility)

$$a(b_* \sqcap f) \sqsubseteq (b_* \sqcap f)a.$$

which, by Proposition 2.5 is entailed by

$$ab_* \sqsubseteq b_*a \quad \text{and } af \sqsubseteq f. \tag{12}$$

Observe that there is a slight asymmetry in (11) and (12): the condition $i \sqsubseteq ai$ is guaranteed for any $i \in C$, since a is an up-closure. This is not the case for $af \sqsubseteq f$.

Therefore the data that allows to link abstract interpretation and coinduction up-to are:

- $b^* \dashv b_* \colon C \to C$,
- $a\colon C \to C$,
- $i \in C$,
- $f \in C$ such that $af \sqsubseteq f$.

In this setting, by mean of (10), a is forward complete w.r.t. b^* iff a is backward complete w.r.t. b_*. These entails both that a is forward complete w.r.t. $b^* \sqcup i$ and a is backward complete w.r.t. $b_* \sqcup f$.

Remark 5. Apparently, we do not have an iff between the conditions of a is forward complete w.r.t. $b^* \sqcup i$ and a is backward complete w.r.t. $b_* \sqcup f$.

6.1 Hopcroft and Karp as complete abstract interpretation

We now show how the Hopcroft and Karp algorithm can be seen as an instance of complete abstract interpretation, using the technology developed above.

First of all observe that $b\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ in (3) can be decomposed as

$$b = b_* \sqcap f$$

where $b_*\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ is defined for all relations R as

$$b_*(R) = \{(x,y) \mid \text{ for all } a \in A,\ (t(x)(a), t(y)(a)) \in R\} \tag{13}$$

and $f\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ maps everything to the relation f defined in (22). Now recall the up-closure $e\colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ mapping each relation in its equivalence closure. It is immediate to see that $f \in \mathsf{Rel}_X$ is an equivalence relation, that is $e(f) \subseteq f$.

The function b_* has a left adjoint, which is exactly b^* as defined in (21). We take i to be the singleton relation containing the pair of initial states (x_1, x_2).

These are all the data needed to link coinduction up-to and abstract interpretation. Indeed, the problem of language equivalence for x_1 and x_2 can be regarded both in an inductive and a coinductive way:

$$\mu(b^* \cup i) \subseteq f \qquad \text{iff} \qquad i \subseteq \nu(b_* \cap f).$$

We have shown that the righmost inclusion can be checked using coinduction up-to e. To prove soundness of this technique we needed to prove that e is compatible w.r.t. $b = (b_* \cap f)$. One can easily check that e is also compatible w.r.t. b_*.

Remark 6. Unfortunately here, we cannot say that the fact that e is compatible w.r.t. $b = (b_* \cap f)$ entails that e is compatible w.r.t. b_*. the other implication hold, but not this one. Indeed:

$$e(b_* \cap f) \subseteq (b_* \cap f)e \text{ iff } eb_* \cap ef \subseteq b_*e \cap f \text{ iff } eb_* \cap ef \subseteq b_*e \text{ and } eb_* \cap ef \subseteq f.$$

Now, since e is compatible w.r.t. b_*, we have by (10) that it is forward complete w.r.t. b^* and, by the fact that $i \subseteq e(i)$ and Proposition 2.5, also w.r.t. $b^* \cup i$. By Proposition 2, e is a complete domain for abstract interpretation.

This provides a novel perspective on the Hopcroft and Karp's algorithm. It can be though as the computation of the least fixed-point of the function $b^* \cup i$ abstracted to the lattice of equivalence relations $ERel_X$. We denote this function by $\overline{b^* \cup i}\colon ERel_X \to ERel_X$ and $\alpha \dashv \gamma\colon Rel_X \to ERel_X$ the pair of adjoint associated to e. The map α assign to every relation its equivalence closure; γ is just the obvious injection. The function $\overline{b^* \cup i}$ is just $\alpha \circ (b^* \cup i) \circ \gamma$: it basically takes an equivalence relation, computes $b^* \cup i$ and then makes its equivalence closure. The algorithm returns true iff $\mu(\overline{b^* \cup i}) \subseteq f$. Soundness is trivial: if $\mu(\overline{b^* \cup i}) \subseteq f$, then

$$\mu(b^* \cup i) \subseteq \gamma\alpha\mu(b^* \cup i) \subseteq \gamma\mu(\overline{b^* \cup i}) \subseteq \gamma f = f.$$

Completeness informs us that $\mu(\overline{b^* \cup i}) = \alpha\mu(b^* \cup i)$: if $\mu(b^* \cup i) \subseteq f$, then

$$\mu(\overline{b^* \cup i}) = \alpha\mu((b^* \cup i)) \subseteq \alpha(f) = f.$$

6.2 Fixed-point completeness

So far, we have established a link between the sufficient conditions (see Table **??**), namely between forward completeness of an abstract domain and compatibility of an up-to technique. Is it the case that one can establish a link also the final aims, that is completeness of an abstract domain and soundness of an up-to technique?

Hereafter, we reformulate the question in purely lattice-theoretic terms.

Given,

- an adjunction of monotone maps $b^* \dashv b_* : C \to C$,
- an up-closure $a: C \to C$, with corresponding pair of adjoints $\alpha \dashv \gamma$,
- an element $i \in C$,
- a pre-fixpoint $f \in C$, i.e., an element such that $af \sqsubseteq f$.

Does it hold that

$$\alpha(\mu(b^* \sqcup i)) = \mu(\alpha \circ (b^* \sqcup i) \circ \gamma) \qquad \text{iff} \qquad \nu((b \sqcap f)a) = \nu(b \sqcap f) \qquad ?$$

We have explored this in a rather extensive way without finding either a proof or a counterexample. Some hints could come by assuming the lattice C to be boolean. Some partial but interesting results are in the attached notes scanned by Roberto.

7 Most abstract Complete domain and companion

8 Side tracks

We conclude these notes with some ideas that emerged during the discussion, but that does not seem to play a pivotal role in the actual development.

8.1 Up-to techniques for induction and coinduction

Hereafter we compare soundness and completeness of induction up-to (on the left) and coinduction up-to (on the right) for two monotone maps $a, b: C \to C$.

$$\frac{\exists y,\ ba(y) \sqsubseteq y \sqsubseteq x}{\mu b \sqsubseteq x} \qquad\qquad \frac{\exists y,\ x \sqsubseteq y \sqsubseteq ba(y)}{x \sqsubseteq \nu b} \tag{14}$$

We distintiguish two important cases, when a is an up-closure and when it is a down-closure. Apart from the cases marked with ?, for which we have not found yet a simple condition (and probably we will never find), all the proofs are given in the lemmas below. The missing cases are obtained by duality.

a up-closure	Soundness	Trivial	$ab \sqsubseteq ba$
	Completeness	?	Trivial
a down-closure	Soundness	$ba \sqsubseteq ab$	Trivial
	Completeness	Trivial	?

Lemma 2. *Let a be an up-closure. Then coinduction up-to a is complete.*

Proof. Take $y = \nu b$. Since, a is an up-closure $\nu b \sqsubseteq a\nu b$. Then $x \sqsubseteq \nu b = b\nu b \sqsubseteq ba\nu b$. □

Lemma 3. *Let a be an up-closure. If $ab \sqsubseteq ba$, then coinduction up-to a is sound.*

Proof. Note that the statement is exactly the same of Proposition 1. We copied here once more for the convenience of the reader. □

Lemma 4. *Let a be an up-closure. Then induction up-to a is sound.*

Proof. Since, a is an up-closure, then $b(y) \sqsubseteq ba(y) \sqsubseteq y$. Therefore $\mu b \sqsubseteq y \sqsubseteq x$. □

8.2 A principle for abstract intepretation

The following principle, where $x', y' \in A$, coincides with abstract interpretation.

$$\frac{\exists y',\ \alpha b \gamma(y') \sqsubseteq y' \sqsubseteq x'}{\alpha(\mu b) \sqsubseteq x'} \tag{15}$$

We first prove that soundness of this principle holds iff $\alpha(\mu b) \sqsubseteq \mu(\alpha b \gamma)$. We have already seen that $\alpha(\mu b) \sqsubseteq \mu(\alpha b \gamma)$ is always true. We prove that from it, it follows the soundness of the proof principle observe that if the premises are true then $\alpha(\mu b) \sqsubseteq \mu(\alpha b \gamma) \sqsubseteq y' \sqsubseteq x'$.

The completeness is the interesting part. Assume that the rule is complete and take $x' = \alpha(\mu b)$. Then there exists $y' \sqsubseteq \alpha(\mu b)$ which is a prefix-point of $\alpha b \gamma$. Therefore $\mu(\alpha b \gamma) y' \sqsubseteq \alpha(\mu b)$. The other inclusion is always true. Now assume that $\mu(\alpha b \gamma) y' = \alpha(\mu b)$ and that the conclusion of the rule holds. Therefore $\alpha b \gamma(\mu(\alpha b \gamma)) = \mu(\alpha b \gamma) = \alpha(\mu b) x'$.

9 In Madrid

Proposition 3. *Let $b^* \dashv b_* \colon C \to C$ and $i, f, x \in C$. The following are equivalent:*

- *x is a pre-fixpoint of $(b^* \sqcup i)$ such that $x \sqsubseteq f$*
- *x is a post-fixpoint of $(b_* \sqcap f)$ such that $i \sqsubseteq x$.*

Proof. If x is a pre-fixpoint, then $i \sqsubseteq x$ and $b^*x \sqsubseteq x$. From the latter, it follows that $x \sqsubseteq b_*x$. Since $x \sqsubseteq f$, then $x \sqsubseteq b_*x \sqcap f$, that is $x \sqsubseteq (b_* \sqcap f)x$.

Conversely, $x \sqsubseteq (b_* \sqcap f)x$ entails that $x \sqsubseteq f$ and $x \sqsubseteq b_*x$. From the latter, it follows that $b^*x \sqsubseteq x$. Since $i \sqsubseteq x$, then $i \sqcup b^*x \sqsubseteq x$, that is $(i \sqcup b^*)x \sqsubseteq x$. □

The result below follows immediately by Knaster-Tarski.

Corollary 1. *$\mu(b^* \sqcup i) \sqsubseteq f$ iff $i \sqsubseteq \nu(b_* \sqcap f)$.*

This corollary has an interesting computational meaning. If the least-fixed point computed by the program b^* with input i is below f, then it is a bisimulation, witnessing that $i \sqsubseteq \nu(b_* \sqcap f)$. For an example consider the `Naive` algorithm: the loop `(3)` compute the least fixed point of $reach_i$, that is the pairs of all states reachable from i. If all of them are into f, then then these pairs form a relation that is a bisimulation witnessing that the pair of initial states in i are language equivalent.

We now move toward abstract domains and up-to techniques.

Proposition 4. *Let $b^* \dashv b_* \colon C \to C$ and $i, f, x \in C$. Let $a \colon C \to C$ be backward complete w.r.t. b^* or, equivalently, compatible w.r.t. b_*. Assume moreover that $a(f) \sqsubseteq f$. If x is a post-fixpoint of $(b_* \sqcap f)a$ such that $i \sqsubseteq x$, then $a(x)$ is a pre-fixpoint of $(b^* \sqcup i)$ such that $a(x) \sqsubseteq f$. In symbols,*

$$i \sqsubseteq x \sqsubseteq (b_* \sqcap f)a(x) \text{ entails } (b^* \sqcap i)a(x) \sqsubseteq a(x) \sqsubseteq f \tag{16}$$

Proof. If $x \sqsubseteq (b_* \sqcap f)a(x)$, then (1) $x \sqsubseteq f$ and (2) $x \sqsubseteq b_*a(x)$. From (1), $a(x) \sqsubseteq a(f) \sqsubseteq f$. From (2), $b^*(x) \sqsubseteq a(x)$ and using compatibility $b^*a(x) \sqsubseteq ab^*(x) \sqsubseteq aa(x) = a(x)$. Moreover $i \sqsubseteq x \sqsubseteq a(x)$ and thus $(b^* \sqcup i)a(x) = b^*a(x) \sqcup i \sqsubseteq a(x)$. □

Intuitively, the proposition is stating that every bisimulation up-to a is a prefixpoint of $(b^* \sqcap i)$ in the abstract domain. The other direction does not hold in general, since there is no way to deduce $i \sqsubseteq x$ from $(b^* \sqcap i)a(x) \sqsubseteq a(x)$. However, when $a(x)$ is the result of the abstract interpreter. To prove this, we do not need backward completeness, but simply fix point completeness: $a\mu(b^* \sqcup i) = \mu(a(b^* \sqcup i)a)$. In this case, the result of the abstract interpreter $\mu(a(b^* \sqcup i)a)$ is equal to $\mu a(b^* \sqcup i)$ (see below), and thus $(b^* \sqcup i)a\mu(b^* \sqcup i) \sqsubseteq a(b^* \sqcup i)a\mu(b^* \sqcup i) \sqsubseteq a\mu(b^* \sqcup i)$. This means that $a\mu(b^* \sqcup i)$ is a pre-fixpoint of $(b^* \sqcup i)$. By Proposition $a\mu(b^* \sqcup i)$ is a post-fixpoint for $(b_* \sqcap f)$ and thus $\mu(b^* \sqcup i)$ is a bisimulation up-to a. Moreover, it trivially holds that $i \sqsubseteq \mu(b^* \sqcup i)$.

Proposition 5. *Let $b^* \dashv b_* \colon C \to C$ and $i, f, x \in C$. Let $a \colon C \to C$ be fixpoint complete w.r.t. $b^* \sqcup i$, that is $a\mu(b^* \sqcup i) = \mu(a(b^* \sqcup i)a)$. Then $i \sqsubseteq \mu(b^* \sqcup i) \sqsubseteq (b^* \sqcup i)a(\mu(b^* \sqcup i))$.*

Proof. Copy the argument above! □

IMPORTANT TO REMEBER!!!

Lemma 5. *$\mu(aba) = \mu ab$.*

Proof. See Lemma 3.3 of Journal of ACM. □

Proposition 6. *a is fixpoint complete iff $a(\mu b) = aba(\mu b)$*

Proof. Theorem 4.12 in Journal of ACM □

9.1 An example of Abstract Interpretation seen as coinduction up-to.

Consider the following piece of code, where x is an integer value.

$x := 5;\ \texttt{while}\ \ x > 0\ \ \texttt{do}\ \{\ \ x := x - 1;\ \}$

We want to prove that after exiting the loop x has value 0. Our analysis works on the lattice of predictates over the integers, hereafter denoted by $Pred_Z$. We will make use of the operation $\ominus 1 \colon Pred_Z \to Pred_Z$ defined as $P \ominus 1 = \{i - 1 \text{ s.t. } i \in P\}$. We begin our analysis by annotating the code in order to make explicit its control flow.

$x := 5;^1\ \texttt{while}\ \ {}^2x > 0^3\ \ \texttt{do}\ \{\ \ x := x - 1;^4\ \}^5$

We then write the following system of equations where x^j contains the set of possible values that the variable x can have in the position j.

$$x^1 = \{5\},\ x^2 = x^1 \cup x_4,\ x^3 = x^2 \cap [1,\infty),\ x^4 = x^3 \ominus 1,\ x^5 = x^2 \cap (-\infty, 0] \quad (17)$$

For x^2, we obtain the following equation

$$x^2 = \{5\} \cup ((x^2 \cap [1,\infty)) \ominus 1) \quad (18)$$

that has as smallest solution $\mu(i \cup b^*)$ where $i = \{5\}$ and $b^* \colon Pred_Z \to Pred_Z$ is defined as $b^*(P) = ((P \cap [1,\infty)) \ominus 1)$ for all predicates P. Our initial aim is to check whether $x^5 = x^2 \cap (-\infty, 0] = \mu(i \cup b^*) \cap (-\infty, 0] \subseteq \{0\}$. That is $\mu(i \cup b^*) \subseteq [0,\infty)$. We fix $f = [0,\infty)$.

We proceed by computing $\nu(i \cup b^*)$ in the standard way:

$$\emptyset \subseteq \{5\} \subseteq \{5,4\} \subseteq \{5,4,3\} \subseteq \cdots \subseteq \{5,4,3,2,1\} \subseteq \{5,4,3,2,1,0\} \quad (19)$$

Form this computation, I would guess that definition of $\ominus 1$ is wrong. Please Roberto check it cerafully

Since $\{5,4,3,2,1,0\} \subseteq [0,\infty)$, we have proved our initial conjecture.

By using abstract interpretation, one can reduce the size of the chain in (19). Consider the abstract domains of signs depicted below.

$$\begin{tikzcd}
 & Z \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \\
\leq 0 \arrow[d, dash] \arrow[dr, dash] & \neq 0 \arrow[dl, dash] \arrow[dr, dash] & \geq 0 \arrow[dl, dash] \arrow[d, dash] \\
< 0 \arrow[dr, dash] & 0 \arrow[d, dash] & > 0 \arrow[dl, dash] \\
 & \emptyset &
\end{tikzcd}$$

We then have $\overline{b^*}(P) = (P \sqcap > 0) \ominus 1$, $\overline{f} = \geq 0$ and $\overline{i} = \geq 0$. The computation of the least fixpoint goes as follows: $\emptyset \sqsubseteq > 0 \sqsubseteq \geq 0 \sqsubseteq \geq 0$.

Another way of reasoning is provided by the greatest fixpoint. Take $b_* \colon Pred_Z \to Pred_Z$ to be the right adjoint of b^*, formally defined as

$$b_*(P) = \bigcup\{Q \text{ s.t. } ((P \cap [1, \infty)) \ominus 1)\} = ((-\infty, 0] \cup P) \oplus 1$$

$\oplus 1$ should be defined somewhere.

Since $\mu(i \cup b^*) \subseteq f$ iff $i \subseteq \nu(f \cap b_*)$, we can proceed also by computing $\nu(f \cap b_*)$ in the usual way:

$$Z \supseteq [0, \infty) \supseteq [0, \infty)$$

and clearly $\{5\} \subseteq [0, \infty)$. Observe that this computation is much shorter.

An interesting fact is that by reasoning with the greatest fixpoint all the elements computed by the chain are in the abstract domain, for any possible complete abstract domain.

Proposition 7. *Let $b^* \dashv b_* \colon C \to C$ and $i, f, x \in C$. Let $a \colon C \to C$ be backward complete w.r.t. b^* or, equivalently, compatible w.r.t. b_*. Assume moreover that $a(f) \sqsubseteq f$. For all k, $a(b_* \sqcap f)^k(\top) \sqsubseteq (b_* \sqcap f)^k(\top)$*

Proof. By induction on k. For $k = 0$, the above inequation amounts to $\top \sqsubseteq \top$, which trivially holds. For $k+1$, $a(b_* \sqcap f)^{k+1}(\top) = a(b_* \sqcap f)(b_* \sqcap f)^k(\top) = a(b_*(b_* \sqcap f))^k(\top) \sqcap a(f) \sqsubseteq b_*(a(b_* \sqcap f))^k(\top) \sqcap f = (b_* \sqcap f)(a(b_* \sqcap f))^k(\top)$. By induction hypothesis, the latter is equal to $(b_* \sqcap f)(b_* \sqcap f)^k(\top) = (b_* \sqcap f)^{k+1}(\top)$. $\square$

Note that the above does not mean in general that it is convenient to compute the greatest fixpoint. An example is provided by the algorithm `HKC` where it is performed a least fix point computation, rather than a greteast fixed point. For the greatest fixed point one should run the algorithm partition refinement on all the states of a determinised NFA, while with `HKC` one is able to explore just a small part. Another example is provided in the section below (INSERT EXAMPLE OF INTERVALS).

Making proofs by coinduction does not mean to compute the greatest fix point. In this example, it just means to find a predicate P such that $\{5\} \subseteq P \subseteq b_*(P) \cap [0, \infty)$. For instance, by taking $P = \{5, 4, 3\}$, one has $b_*(P) = (-\infty, 0] \cup \{6, 5, 4\}$ and $b_*(P) \cap [0, \infty) = \{6, 5, 4, 0\}$. Therefore the inclusion does not hold.

For a bisimulation P, one can take the least fixpoint computed in (19) $P = \{5, 4, 3, 2, 1, 0\}$.

One can also reason up-to the abstract domain of signs.

In this case, $\{5\}$ itself is a bisumulation up to. Indeed $a(\{5\}) = [1, \infty)$ and $b_*[1, \infty) \cap [0, \infty) = ((-\infty, 0] \cup ([1, \infty) \oplus 1)) \cap [0, \infty) = \{0\} \cup [1, \infty) = [0, \infty)$. Obviously $\{5\} \subseteq [0, \infty)$.

9.2 13/09/2017: Example of infinite descending chain

Consider the following piece of code, where `x` is an integer value.

$x := i$; `while` x `is even do {` $x := x/2$; `}`

We want to know whether at the end of the loop `x` has value 1. We proceed as before:

$x := i^1$; `while`^{2} x `is even`^{3} `do {` $x := x/2;^4$ `}`^{5}

$$x^1 = \{i\},\ x^2 = x^1 \cup x_4,\ x^3 = x^2 \cap 2Z,\ x^4 = x^3/2,\ x^5 = x^2 \cap (2Z+1) \qquad (20)$$

By solving the equation one has $x^2 = i \cup (x_2 \cap 2Z)/2$. The final condition amounts to $\mu(b^* \cup i) \cap (2Z+1) \subseteq \{1\}$ that is $\mu(b^* \cup i) \subseteq 2Z \cup \{1\}$. We fix $f = 2Z \cup \{1\}$. We then have $b_*(P) = (2Z+1) \cup 2P$ and $b_*(P) \cap f = 2P \cup \{1\}$. The computation of the final chain does not terminate:

$$Z \supseteq \{1\} \cup 2Z \supseteq \{1\} \cup \{2\} \cup 4Z \supseteq \dots$$

9.3 14/09/2017:Philosophy, speculations and, most of all, controversial-provocative arguments

1. Every proof generated by Abstract Interpretation is a proof by coinduction, without the need of up-to techniques;
2. Abstract Interpretation is always sound, while coinduction up-to is meaningfull only when Abstract Interpretation is complete;
3. Abstract Interpretation is complete iff Coinduction up-to is sound;
4. Whenever we have soundness and completeness, each proof by coinduction up-to can be transformed into a proof by induction in the abstract domain; viceversa the result of abstract interpretation can be seen as the witness of a proof by coinduction up-to.

9.4 15/09/2017: HK seen as a abstract interpreter

In the last day, we look at again HK as an abstract interpreter. This is shown in figure 2. The code is still a little rough and it should be better understood. Comparing with the coinductive version, this algorithm is closer to the original one presented by Hopcroft and Karp. Indeed they do the check at the end and they always store all equivalence classes (implemented in the algorithm by line `3.5`). Observe that the algorithm is parametric w.r.t. an arbitrary up-closure a. Proving the soundness of the algorithm should be trivial (TO BE CHECKED carefully). For proving its completeness, we should show that a is an complete domain. THE CODE IS STILL ROUGH SHOULD BE CHECKED MORE CAREFULLY... We also tried to make an abstract interpretation of the code of `HK` but it seems to be a real mess... SO we stopped...

```
                    AIHK$_a$ $(x_1, x_2)$
(1) $R := \emptyset$;  $todo := \emptyset$
(2) insert $(x_1, x_2)$ into $todo$
(3) while $todo$ is not empty do
   (3.1)  extract $(x_1', x_2')$ from $todo$
   (3.2)  if $(x_1', x_2') \in R$ then continue
   (3.3)  for all $a \in A$,
              insert $(t_a(x_1'), t_a(x_2'))$ into $todo$
   (3.4)  insert $(x_1', x_2')$ into $R$
   (3.5) $R := a(R)$;  $todo := a(todo)$;
(4) return $R \subseteq f$;
```

Fig. 2. Abstract Interpretation version of the algorithm by Hopcroft and Karp.

9.5 15/09/2017: Optimal Domain

The last day, we then asked the question whether e is the most abstract domain to be complete *for all* automata. It turns out that the question is, at some extent, weird.

The idea is to show that for all up-closure operators $a\colon C \to C$ such that $e \sqsubset a$, it holds that a is incomplete for some automata (X, o, t). The starting observation is that any such a can be constructed from e, by removing one of the co-atoms in the lattice of equivalence relations $ERel_X$. The coatoms are those elements which stay directly below the top, the relation equating everything. In $ERel_X$, the coatoms are exactly all those equivalence relations that can be represented as partitions of two blocks. For instance, for $X = \{1, 2, 3, 4\}$, the coatoms are the partitions $\{1\}\{2, 3, 4\}$, $\{2\}\{1, 3, 4\}$, $\{3\}\{1, 2, 4\}$, $\{4\}\{1, 2, 3\}$, $\{1, 2\}\{3, 4\}$, $\{1, 3\}\{2, 4\}$ and $\{1, 4\}\{2, 3\}$. Note that in general, by removing one of these elements – say k – and then taking the lattice generated by the remaining ones (via $\bigsqcap$), one obtain again the lattice $ERel_X$ but without k. The associated up-closure operator a will map every $x \neq k$ into x and k into $\top$.

Now, observe that the f, namely the relation equating those states having the same outputs (represented as partition $\{x \text{ s.t. } o(x) = 0\}, \{x \text{ s.t. } o(x) = 1\}$), is always a coatom. Now there are two cases: either $f = k$ or $f \neq k$. If $f = k$, then $a(f) = \top \not\sqsubseteq f$, that is the question of abstract inteepretation does not make any sense: f is not in the abstract domain. If $f \neq k$, and the least fixed point computation in $ERel_X$ is below f, then all the steps of the computations (the chain) in the novel abstract domain are below f. So the computation in $ERel_X$ and in the novel abstract domain always coincide.

Acknowledgment. TO INSERT ACK.

References

1. F. Bonchi and D. Pous. Checking NFA equivalence with bisimulations up to congruence. In *POPL*, pages 457–468. ACM, 2013.
2. Patrick Cousot and Radhia Cousot. Systematic design of program analysis frameworks. In *Proceedings of the 6th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 269–282. ACM, 1979.
3. B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2 edition, 2002.
4. J. E. Hopcroft and R. M. Karp. A linear algorithm for testing equivalence of finite automata. Technical Report 114, Cornell Univ., December 1971.
5. John Hopcroft. An n log n algorithm for minimizing states in a finite automaton. Technical report, STANFORD UNIV CALIF DEPT OF COMPUTER SCIENCE, 1971.
6. B. Klin. Bialgebras for structural operational semantics: An introduction. *TCS*, 412(38):5043–5069, 2011.
7. D. Pous and D. Sangiorgi. Enhancements of the bisimulation proof method. In *Advanced Topics in Bisimulation and Coinduction*, pages 233–289. Cambridge University Press, 2012.
8. Daniele Turi and Gordon Plotkin. Towards a mathematical operational semantics. In *Logic in Computer Science, 1997. LICS'97. Proceedings., 12th Annual IEEE Symposium on*, pages 280–291. IEEE, 1997.

A Induction (up to) for Deterministic Automata

TO BE WRITTEN PROPERLY

As already mentioned in Section 3.1, the `Naive` algorithm builds the smallest bisimulation containing the initial pair of states $\{(x_1, x_2)\}$. Rather than as the computation of a greatest fixed-point, `Naive` can be regarded as the computation of a least fixed-point. Let $b^* \colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ be the function mapping every relation $R \in \mathsf{Rel}_X$ into

$$b^*(R) = \{(x'_1, x'_2) \mid \exists (x_1, x_2) \in R, a \in A \text{ s.t. } t(x_1)(a) = x'_1 \text{ and } t(x_2)(a) = x'_2\} \tag{21}$$

and $reach_{x_1,x_2} \colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ be defined as

$$reach_{x_1,x_2}(R) = b^*_1(R) \cup \{(x_1, x_2)\}.$$

The least fixed-point of $reach_{x_1,x_2}$ is the set of all pairs of states reachable from (x_1, x_2). Let us call f the relation in Rel_X defined as

$$f = \{(x_1, x_2) \mid o(x_1) = o(x_2)\}. \tag{22}$$

Then the problem of language equivalence can be rephrased as a problem of induction:

$$x_1 \sim x_2 \text{ iff } \mu(reach_{x_1,x_2}) \subseteq f.$$

Now the soundness of `Naive` can be proved by means of induction: at any iteration of the while loop `(3)` the invariants

$$reach_{x_1,x_2}(R) \subseteq R \cup todo \qquad \text{and} \qquad R \subseteq f$$

hold. The algorithm returns true only if $todo$ is empty, so that $reach_{x_1,x_2}(R) \subseteq R \subseteq f$. By the induction proof principle (on the left of (1)) we have that $\mu(reach_{x_1,x_2}) \subseteq f$.

Question 1. Can we elegantly prove completeness using an inductive approach?

The story of induction for determinitic autoamaton ends here. Up-to techniques do not work for induction in this setting. The dual of the coinduction up to principle in (5), the *induction up to principle*, is given below.

$$\frac{\exists y,\ ba(y) \sqsubseteq y \sqsubseteq x}{\mu b \sqsubseteq x} \tag{23}$$

By taking $a = e$ and $b = reach_{x_1,x_2}$, the invariant

$$reach_{x_1,x_2}(e(R)) \subseteq R \cup todo$$

does *not* hold in the Hopcroft and Karp's algorithm. Intuitively, when using induction up to a, one would like a to shrink relations rather than enlarging. Indeed, as explained in the following remark, induction up to makes sense when a is a down-closure.

Remark 7 (Completeness of induction up-to). As for coinduction up-to, one may wonder about the completeness of induction up-to. Completeness holds whenever a is a down-closure. Indeed $a(\nu b) \sqsubseteq \nu(b)$ and, by monotonicity of b, one obtains $ba(\nu b) \sqsubseteq b\nu b = \nu b$.

Question 2. Researchers interested in coinduction up-to have been wondering for a while about induction up-to. As we will better explain later, coinduction up to a can be seen in a category of coalgebras over the category of Eilenberg-Moore algebras for the monad a. Its dual, induction up to a, can be regarded in a category of algebras over the category of Eilenberg-Moore coalgebras for the comonad a. While the former are common places in computer science (determinised automata, process calculi and so on), the latter does not show off much. What about physics? Can we find an interesting example? More particularly, can we make a nice (and reasonable) proof by means of induction up to?

B A categorical perspective

Most of the concepts discussed in this paper can be extended from lattices to categories: a lattice can be seen as a category, a monotone map as a functor, an up-closure operator as a monad and a down-closure operator as a comonad. Pre and post-fixed point as algebras and coalgebras, least and greatest fixed point as the initial algebra and the final coalgebra.

This perspective motivates the terminology Kleisli law and EM (Eilenberg Moore) law for the conditions $ba \sqsubseteq ab$ and $ab \sqsubseteq ba$ in Proposition **??**. Indeed, one can think to the problem of completeness of abstract interpretation and soundness of up-to techniques as the problem of *extending* and *lifting* the functor $b\colon C \to C$ to some functor $\bar{b}$ either on the Kleisli category $Kl(a)$ or to the Eilenberg-Moore category $EM(a)$ of algebras for the monad $a\colon C \to C$. In this case, since C is a lattice, one has that $Kl(a) = EM(a) = A$. In this perspective, completeness of full abstraction means that there is a functor $\alpha\colon Alg(b) \to Alg(\bar{b})$ preserving initial algebra (this is entailed by requiring α to be a left adjoint). Similarly, soundness of up-to techniques means that there is a functor $\gamma\colon Coalg(\bar{b}) \to Coalg(b)$ that preserves the final coalgebra (this is entailed by requiring γ to be a right adjoint). The latter is rather well-studied problem, which arise for instance with bialgebras (see e.g., [6,8]). The former instead is far less understood.

Abstract Interpretation	Coinduction up-to
Kleisli law $ba \sqsubseteq ab$	EM-law $ab \sqsubseteq ba$
Kleisli Extension $\bar{b}\colon Kl(a) \to Kl(a)$	EM lifting $\bar{b}\colon EM(a) \to EM(a)$
$\alpha\colon Alg(b) \to Alg(\bar{b})$ is a left adjoint	$\gamma\colon Coalg(\bar{b}) \to Coalg(b)$ is a right adjoint

C Proofs

Proof of Lemma1. First part.

(1 $\Rightarrow$ 2) $ab = aba \sqsupseteq ba$ since a is an up-closure. (2 $\Rightarrow$ 3) It always holds $(\alpha b\gamma)\alpha \sqsupseteq \alpha b$. For the other inclusion, observe that if $ba \sqsubseteq ab$, then $b\gamma\alpha \sqsubseteq \gamma\alpha b$ and $\alpha b\gamma\alpha \sqsubseteq \alpha\gamma\alpha b \sqsubseteq \alpha b$. (3 $\Rightarrow$ 1) Since $(\alpha b\gamma)\alpha = \alpha b$, then $\gamma(\alpha b\gamma)\alpha = \gamma\alpha b$, that is $aba = ab$.

Second part.

(1 $\Rightarrow$ 2) Since $b \circ a = a \circ b \circ a$ then $b \circ a = a \circ b \circ a \sqsupseteq a \circ b$. (2 $\Rightarrow$ 1) Since $a \circ b \sqsubseteq b \circ a$, then $a \circ b \circ a \sqsubseteq b \circ a \circ a \sqsubseteq b \circ a$. The other inclusion, $b \circ a \subseteq a \circ b \circ a$, holds since a is an up-closure. (2 $\Rightarrow$ 3) Observe that $A = Pre(a)$ by construction. For every pre fixed-point $a(x) \sqsubseteq x$, it holds that $ab(x) \sqsubseteq ba(x) \sqsubseteq b(x)$, namely $b(x)$ is a pre fixed-point. One can therefore define $\bar{b}(x) = b(x)$. The fact that $\gamma\bar{b} = b\gamma$ follows immediately by construction of γ. (3 $\Rightarrow$ 2) By construction of γ, for every pre fixed point x, $b(x)$ is forced to be a pre fixed-point of a: $ab(x) \sqsubseteq b(x)$. Therefore $ab(x) \sqsubseteq b(x) \sqsubseteq ba(x)$. □

Proof of Theorem 1. For each $y \sqsubseteq ba(y)$, $a(y) \sqsubseteq aba(y) \sqsubseteq baa(y) \sqsubseteq ba(y)$. Therefore $a(y) \sqsubseteq \nu b$. If $x \sqsubseteq y$, then $x \sqsubseteq a(y) \sqsubseteq \nu b$. □

Proof of Theorem 2. The assumption of Scott-continuity is necessary to characterise

$$\alpha(\mu b) = \alpha(\bigsqcup_n b^n(\bot_C)) \qquad \text{and} \qquad \mu(\alpha b\gamma) = \bigsqcup_n (\alpha b\gamma)^n(\bot_A).$$

Since α is a left adjoint we have that the leftmost is equivalent to $(\bigsqcup_n \alpha b^n(\bot_C)$.

By induction on n, we prove that $\alpha b^n(\bot_C) = (\alpha b\gamma)^n(\bot_A)$.

- For $n = 0$, $\alpha(\bot_C) = \bot_A$;
- For $n+1$, we have that $\alpha b\gamma(\alpha b\gamma)^n(\bot_A) = \alpha b\gamma\alpha b^n(\bot_C)$ by induction hypothesis. Using the property of Kl-lifting, the latter is equivalent to $\alpha bb^n(\bot_C) = \alpha b^{n+1}(\bot_C)$.

□

D Additional results

Proposition 8. *Let $a, b\colon C \to C$ be two monotone maps so that a is backward complete w.r.t. b. Let $\bar{b}\colon A \to A$ be the corresponding EM lifting. Then $\gamma(\nu\bar{b}) = \nu b$.*

Proof. Recall that $A = Pre(a)$. By the Knaster-Tarski fixed-point theorem $\gamma(\nu\bar{b}) = \bigsqcup\{x \mid a(x) \sqsubseteq x \sqsubseteq b(x)\}$ and $\nu b = \bigsqcup\{x \mid x \sqsubseteq b(x)\}$. Since $\{x \mid a(x) \sqsubseteq x \sqsubseteq b(x)\} \subseteq \{x \mid x \sqsubseteq b(x)\}$, then $\gamma(\nu\bar{b}) \sqsubseteq \nu b$. Observe that $a(\nu b) = ab(\nu b) \sqsubseteq ba(\nu b)$, namely $a(\nu b)$ is a post fixed-point of b. Therefore $a(\nu b) \sqsubseteq \nu b \sqsubseteq b(\nu b)$, which means $\nu b \in \{x \mid a(x) \sqsubseteq x \sqsubseteq b(x)\}$. This proves that $\nu b \sqsubseteq \gamma(\nu\bar{b})$. □