Building Complete Abstract Interpretations in a Linear Logic-based Setting

Roberto Giacobazzi * Francesco Ranzato† Francesca Scozzari‡

Abstract

Completeness is rather uncommon, although important, property of abstract interpretations, which arises especially in comparative semantics. Recently, the first two authors proved that in most cases, given any abstract domain A, there exists the most abstract domain, called least complete extension of A, which includes A and provides a complete abstract interpretation. In this paper we distinguish between the standard formulation of completeness, called full completeness, and a novel and particularly interesting one, called observation completeness. In particular we consider the problem of full and observation completeness in the context of quantales, i.e. models of linear logic, as concrete interpretations. We prove that various types of least complete extensions exist and, more importantly, we show that these extensions can all be specified in terms of an elegant linear logic-based formulation. As an application, we determine the least fully complete extension of a generic abstract domain w.r.t. a standard bottom-up concrete semantics characterizing computed answer substitutions. This general result is further instantiated to the case of groundness analysis.

Keywords: Abstract interpretation, completeness, quantale, linear logic, logic program analysis.

1 Introduction

It is widely held that the ideal goal of any semantic approximation methodology is to find sound and complete representations of concrete (actual) computations. Abstract interpretation is one such methodology, where soundness is always required, while completeness only rarely holds. Completeness issues in abstract interpretation have been studied since the Cousot and Cousot seminal paper [6]. The intuition is that a complete abstract interpretation induces an abstract semantics which is as precise as possible relatively to the underlying abstract domains. More technically, let us denote by $\mathcal{L}_C$ the so-called *lattice of abstract interpretations* of a concrete domain C [5, 6], where, for all $A, B \in \mathcal{L}_C$, $A \sqsubseteq B$ means that A is more precise (i.e. concrete) than B. Then, let us consider the simple case of an abstract interpretation $f^\sharp : A \to A$ of a semantic function

*Dipartimento di Informatica, Università di Pisa, Corso Italia 40, 56125 Pisa, Italy; Ph.: +39-050-887283; E-mail: giaco@di.unipi.it

†Dipartimento di Matematica Pura ed Applicata, Università di Padova, Via Belzoni 7, 35131 Padova, Italy; Ph.: +39-049-8275899; E-mail: franz@math.unipd.it

‡Dipartimento di Informatica, Università di Pisa, Corso Italia 40, 56125 Pisa, Italy; Ph.: +39-050-887248; E-mail: scozzari@di.unipi.it

$f : C \to C$, where the abstract domain $A \in \mathcal{L}_C$ is related with C by an adjoint pair of abstraction and concretization maps $\alpha : C \to A$ and $\gamma : A \to C$. Then, $f^\sharp$ is complete if $\alpha \circ f = f^\sharp \circ \alpha$. It is not too hard to see that $f^\sharp$ is complete if and only if the best correct approximation f^b of f in A, i.e. $f^b = \alpha \circ f \circ \gamma$, is complete, and, in such a case, $f^\sharp$ indeed coincides with f^b (cf. [10]). This key observation makes completeness an abstract domain property, namely a characteristic of the abstract domain A.

Although completeness may be sacrificed for efficiency – this is often the case in static program analysis – examples of complete abstract interpretations are very common in comparative semantics. For instance, thirteen different complete abstractions of algebraic polynomial systems have been studied in [8], and other complete abstractions appear in type systems [3] and comparative semantics [4, 7]. Moreover, complete abstract interpretations that are more concrete than a certain property of interest represent an absolute upper bound in precision for that property. These argumentations probably stimulated the recent trend of research on completeness in abstract interpretation [10, 12, 14, 16, 17]. It should be clear that a key problem consists in devising systematic methodologies for transforming abstract domains so that completeness is achieved and the resulting complete domains are as close as possible to the initial noncomplete ones. This problem has been first raised by Mycroft [12], who gave a methodology for deriving the most concrete domain which is complete and included in a given domain of properties. More recently, the first two authors proved in [10] that when the concrete semantic function f is continuous, for any given domain A, there always exists the most abstract domain, called *least complete extension* of A, which includes A and is complete for f. Analogously, [10] solved the aforementioned problem of Mycroft, by showing that, for any given domain A, the most concrete domain which is complete and included in a given domain A – the so-called *complete kernel* of A – exists for any monotone semantic function.

In this paper, we use quantales for solving a variety of completeness problems in abstract interpretation. Our goal is to solve completeness problems for binary operations. The solutions to these problems have interesting applications in static program analysis and comparative semantics, for instance in logic programming, where binary operations generalize unification which is the key computation step in the semantics of a logic program, or in functional programming by considering data constructors, such as *cons* for lists. Generic concrete semantic functions $\otimes$ of type $\otimes : C_1 \times C_2 \to C$ are considered, and our basic assumption consists of dealing with concrete interpretations which are *typed quantales* $\langle C, C_1, C_2, \otimes \rangle$, which is an algebraic structure with one binary operation which is additive on both arguments. Whenever $C = C_1 = C_2$, the above structure boils down to a *quantale* $\langle C, \otimes \rangle$. Quantales turn out to be the models of intuitionistic linear logic [1, 15, 18], and this argument will be the key point for providing explicit and elegant linear logic-based representations of our least complete extensions. The linear logic setting helps us in characterizing the objects of complete abstract domains. The main feature of (typed) quantales is that they always admit left and right linear implications between domain's objects. This allows us to introduce corresponding operators of linear implications between abstract domains, denoted by $A \overset{\curlywedge}{\multimap} D \in \mathcal{L}_{C_2}$ and $D \overset{\curlywedge}{\multimap} B \in \mathcal{L}_{C_1}$, for all abstractions $A \in \mathcal{L}_{C_1}$, $B \in \mathcal{L}_{C_2}$ and $D \in \mathcal{L}_C$. For instance, $A \overset{\curlywedge}{\multimap} D$ is defined to be the least abstract domain of C_2 containing all the linear implications between the objects of A and D, namely $\{\, a \multimap d \in C_2 \mid a \in A, d \in D \,\}$.

Our first result characterizes the least fully complete extension of an abstract domain A for a semantic operation $\otimes$ in terms of suitable combination of linear implication between domains.

More in general, we consider the following problem on typed quantales: Given a fixed abstraction $D \in \mathcal{L}_C$ and $\langle A, B\rangle \in \mathcal{L}_{C_1} \times \mathcal{L}_{C_2}$, does there exist the most abstract pair of domains $\langle A', B'\rangle \in \mathcal{L}_{C_1} \times \mathcal{L}_{C_2}$ such that $\langle A', B'\rangle \sqsubseteq \langle A, B\rangle$ and the triple of abstract domains $\langle D, A', B'\rangle$ is fully complete for $\otimes$? Here, an abstract domain of observation D is considered as fixed, and we look for the most abstract pair of domains in $\mathcal{L}_{C_1} \times \mathcal{L}_{C_2}$ which is more concrete than an initial pair $\langle A, B\rangle$, yet inducing full completeness. We call this an *observation completeness* problem. Again we construct the solutions to this problem in terms of linear implication of domains.

To illustrate our results, we first consider a simple but meaningful example involving abstract domains for detecting irredundant lists of natural numbers, where various (left and right) observation completeness problems are solved. Our results are then applied in the field of logic program analysis. We consider as concrete semantic function a T_P-like transformer, whose least fixpoint coincides with the s-semantics [9], i.e. characterizes computed answer substitutions. The T_P operator is shown to be defined in terms of three basic operations on (idempotent) substitutions: unification, union, and existential quantification over a set of variables. Of course, unification turns out to be the key operation. In fact, sets of substitutions and unification give rise to a unital commutative quantale, and it is shown how least fully complete extensions for this quantale naturally induce least fully complete extensions for T_P functions. We explicitly give, in terms of simple linear implications, the least fully complete extension for any T_P of a generic domain abstracting sets of substitutions. As an example, our result is applied to variable groundness analysis in order to characterize the least fully complete extension for the s-semantics.

2 Basic notions

Closure operators. Given any poset P, $\langle uco(P), \sqsubseteq\rangle$ denotes the poset of all (*upper*) *closure operators* (shortly uco's or closures) on P, i.e., monotone, idempotent and extensive operators on P, ordered pointwise. Let us recall that each $\rho \in uco(P)$ is uniquely determined by the set of its fixpoints, which is its image, i.e. $\rho(P) = \{x \in P \mid \rho(x) = x\}$, and that $\rho \sqsubseteq \eta$ iff $\eta(P) \subseteq \rho(P)$. When $\langle C, \leq, \vee, \wedge, \top, \bot\rangle$ is a complete lattice, $\langle uco(C), \sqsubseteq, \sqcup, \sqcap, \lambda x.\top, id\rangle$ is a complete lattice as well (here, $id = \lambda x.x$). A subset $X \subseteq C$ is the set of fixpoints of a uco iff X is meet-closed, i.e. $X = \curlywedge(X) = \{\wedge Y \mid Y \subseteq X\}$ (note that $\top \in X$). Moreover, given $\rho \in uco(C)$, $\langle \rho(C), \leq\rangle$ is a complete meet subsemilattice of C.

The lattice of abstract interpretations. In standard Cousot and Cousot's abstract interpretation theory, abstract domains can be equivalently specified either by Galois connections (GCs), i.e. adjunctions, or by closure operators (see [6]). In the first case, the concrete domain C and the abstract domain A (both assumed to be complete lattices) are related by a pair of adjoint functions of a GC (α, C, A, γ). Also, it is generally assumed that (α, C, A, γ) is a Galois insertion (GI), i.e. α is onto or, equivalently, γ is 1–1. In the second case instead, an abstract domain is specified as an uco on the concrete domain C. These two approaches are equivalent, modulo isomorphic representation of domain's objects. Hence, we will identify $uco(C)$ with the lattice $\mathcal{L}_C$ of abstract interpretations of C, i.e. the complete lattice of all possible abstract domains of the concrete domain C. For an abstract domain $A \in \mathcal{L}_C$, $\rho_A \in uco(C)$ will denote the corresponding uco on C, and whenever A is specified by a GI (α, C, A, γ) then $\rho = \gamma \circ \alpha$. Often, we will

identify closures with their sets of fixpoints, in this case sometimes denoted by capital Latin letters instead of lowercase Greek letters. This does not give rise to ambiguity, since one can distinguish their use as functions or sets according to the context. The ordering on $uco(C)$ corresponds precisely to the standard order used to compare abstract domains with regard to their precision: A_1 is *more precise* than A_2 (i.e., A_1 is more concrete than A_2 or A_2 is more abstract than A_1) iff $A_1 \sqsubseteq A_2$ in $uco(C)$. The *lub* and *glb* on $uco(C)$ have therefore the following meaning as operators on domains. Suppose $\{A_i\}_{i\in I} \subseteq uco(C)$: (i) $\sqcup_{i\in I} A_i$ is the most concrete among the domains which are abstractions of all the A_i's, i.e. it is their least common abstraction; (ii) $\sqcap_{i\in I} A_i$ is the most abstract among the domains (abstracting C) which are more concrete than every A_i; this domain is also known as *reduced product* of all the A_i's.

Quantales and linear logic. Quantales have been introduced for studying the foundations of the so-called quantum logic. More recently, they have been considered for the algebraic semantics of Girard's linear logic (see [15] for an introduction). A *typed quantale* is a multisorted algebra $\langle C, C_1, C_2, \otimes\rangle$, where: C, C_1, C_2 are complete lattices, $\otimes : C_1 \times C_2 \to C$ is a function such that $c_1 \otimes \bigvee x_i = \bigvee(c_1 \otimes x_i)$ and $\bigvee y_i \otimes c_2 = \bigvee(y_i \otimes c_2)$. In other terms, a typed quantale is a 3-sorted algebra endowed with a "product" $\otimes$ which distributes on both sides over arbitrary lub's. Thus, both functions $c_1 \otimes \_$ and $\_ \otimes c_2$ have right adjoints denoted, resp., by $c_1 \multimap \_$ and $\_ \multimap c_2$. Hence: $a \otimes c \leq b \Leftrightarrow c \leq a \multimap b$, and, dually, $c \otimes a \leq b \Leftrightarrow c \leq b \multimap a$. Thus, two functions $\multimap\colon C_1 \times C \to C_2$ and $\multimap\colon C \times C_2 \to C_1$ can be explicitly defined as:

$$a \multimap b = \bigvee \{\, c \in C_2 \mid a \otimes c \leq b \,\}; \qquad b \multimap a = \bigvee \{\, c \in C_1 \mid c \otimes a \leq b \,\}.$$

When $C = C_1 = C_2$ and $\otimes$ is associative, a typed quantale is called *quantale*. It is well-known that quantales turn out to be models of noncommutative intuitionistic linear logic [18]. A quantale $\langle C, \otimes\rangle$ is called *commutative* when $\otimes$ is commutative, and this is equivalent to require that, for all $a, b \in C$, $a \multimap b = b \multimap a$. Moreover, a commutative quantale $\langle C, \otimes\rangle$ is called *unitary* if there exists an object $1 \in C$ such that $1 \otimes a = a \otimes 1 = a$ for all $a \in C$. Any typed quantale $\langle C, C_1, C_2, \otimes\rangle$ enjoys the following main properties: For all $c_1, a, b \in C_1$, $c_2, d, e \in C_2$, $l \in C$ and $\{x_i\}_{i\in I} \subseteq C$:

1. $c_1 \otimes (c_1 \multimap l) \leq l$
2. $(l \multimap c_2) \otimes c_2 \leq l$
3. $c_1 \multimap \bigwedge_{i\in I} x_i = \bigwedge_{i\in I}(c_1 \multimap x_i)$
4. $(\bigwedge_{i\in I} x_i) \multimap c_2 = \bigwedge_{i\in I}(x_i \multimap c_2)$
5. $(a \vee b) \multimap l = (a \multimap l) \wedge (b \multimap l)$
6. $l \multimap (d \vee e) = (l \multimap d) \wedge (l \multimap e)$

For a quantale $\langle C, \otimes\rangle$ the following additional properties hold for all $a, b, c \in C$:

1. $a \multimap (c \multimap b) = (a \multimap c) \multimap b$
2. $b \multimap (a \multimap c) = (a \otimes b) \multimap c$
3. $(c \multimap b) \multimap a = c \multimap (a \otimes b)$

3 Completeness problems in abstract interpretations

Let $\langle C, C_1, C_2, \otimes\rangle$ be a concrete interpretation, i.e. C, C_1, C_2 are concrete semantic domains provided with an operation $\otimes : C_1 \times C_2 \to C$.[1] When $C = C_1 = C_2$, we simply

[1] The extension to generic n-ary semantic operations would be straightforward.

use the notation $\langle C, \otimes \rangle$. Given the abstractions $A_1 \in \mathcal{L}_{C_1}$, $A_2 \in \mathcal{L}_{C_2}$ and $A \in \mathcal{L}_C$, recall [6] that the best correct approximation $\otimes^b : A_1 \times A_2 \to A$ of $\otimes$ is defined as $\otimes^b = \alpha_{C,A} \circ \otimes \circ \langle \gamma_{A_1,C_1}, \gamma_{A_2,C_2} \rangle$. It has been shown in [10] that completeness for an abstract interpretation is a property depending only on the underlying abstract domains. This means that an abstract interpretation $\langle A, A_1, A_2, \otimes^\sharp \rangle$ is complete for $\langle C, C_1, C_2, \otimes \rangle$ iff $\langle A, A_1, A_2, \otimes^b \rangle$ is complete and $\otimes^\sharp = \otimes^b$. In other terms, the best correct approximation induced by the underlying abstract domains determines the property of being complete. Because of this it is more convenient and elegant, and, of course, completely equivalent, to reason using the closure operator approach.

Full completeness. Giacobazzi and Ranzato [10] stated the following full completeness problem: Given a concrete interpretation $\langle C, \otimes \rangle$ and an abstraction $A \in \mathcal{L}_C$, does the following system with variable ρ admit a most abstract solution?

$$\begin{cases} \rho \sqsubseteq A \\ \rho \circ \otimes \circ \langle \rho, \rho \rangle = \rho \circ \otimes \end{cases} \tag{3.1}$$

It is shown in [10] that if $\otimes$ is continuous, then the full completeness problem admits solution for any $A \in \mathcal{L}_C$, i.e. there exists the least fully complete extension of A. This most abstract solution represents the optimal domain which is complete for $\otimes$ and contains a given domain of basic properties A.

Observation completeness. The problem of full completeness clearly makes sense only in quantales. Whenever we deal with typed quantales having different concrete domains we have to change the above notion of full completeness. We study three different problems arising from this generalization. All these problems can still be defined in terms of the best correct approximation of $\otimes$, namely completeness is again an abstract domain property. An observation domain is any abstraction of the range of $\otimes$. Observation completeness problems arise when an observation domain is fixed. Given $\langle C, C_1, C_2, \otimes \rangle$, we consider a fixed observation abstraction $D \in \mathcal{L}_C$. Thus, the observation completeness problem for the pair $\langle A_1, A_2 \rangle \in \mathcal{L}_{C_1} \times \mathcal{L}_{C_2}$ admits solution when there exists the most abstract solution in $\mathcal{L}_{C_1} \times \mathcal{L}_{C_2}$ of the system below:

$$\begin{cases} \langle \eta, \mu \rangle \sqsubseteq \langle A_1, A_2 \rangle \\ \rho_D \circ \otimes \circ \langle \eta, \mu \rangle = \rho_D \circ \otimes \end{cases} \tag{3.2}$$

When in addition to the observation domain we also fix any of the abstractions in the argument domains, we obtain different completeness problems. Let us consider for instance the following left observation completeness problem. In this case, $D \in \mathcal{L}_C$ and $A_2 \in \mathcal{L}_{C_2}$ are fixed, and the solution to this problem for a given $A_1 \in \mathcal{L}_{C_1}$ exists when the following system admits a most abstract solution[2]:

$$\begin{cases} \eta \sqsubseteq A_1 \\ \rho_D \circ \otimes \circ \langle \eta, \rho_{A_2} \rangle = \rho_D \circ \otimes \circ \langle id, \rho_{A_2} \rangle \end{cases} \tag{3.3}$$

[2]By using adjunctions, the left observation completeness equation is equivalent to the following statement:

$$\forall x \in C_1. \forall y \in A_2.\ \alpha_{C_1,\eta}(x) \otimes^b y = \alpha_{C,D}(x \otimes \gamma_{A_2,C_2}(y)).$$

Of course, a dual formulation can be easily given for right observation completeness. Observation completeness characterizes those abstractions which, when applied to the arguments of $\otimes$, leave unchanged the result of a computation with respect to a given observable property ρ_D. This means that, with respect to ρ_D, complete abstractions behave like identity. This is particularly interesting when looking for the most precise abstract interpretation with respect to a given fixed property of interest.

4 Quantales and solutions to completeness problems

Quantales and typed quantales will be here the concrete semantic structures we deal with. With this assumption, solutions to the above completeness problems can be obtained explicitly and elegantly in terms of linear implications. This way, such complete abstract domains can be obtained with ease in static program analysis and comparative semantics.

Let $\langle C, C_1, C_2, \otimes \rangle$ be a typed quantale playing the role of concrete interpretation. To get the solutions, we will exploit two basic abstract domain transformers $\overset{\curlywedge}{\multimap}: uco(C_1) \times uco(C) \to uco(C_2)$ and $\overset{\curlywedge}{\multimapinv}: uco(C) \times uco(C_2) \to uco(C_1)$, defined by lifting left and right linear implications $\multimap$ and $\multimapinv$ to domains as follows:

$$A \overset{\curlywedge}{\multimap} D = \curlywedge(\{\, a \multimap d \in C_2 \mid a \in A, d \in D \,\});$$
$$D \overset{\curlywedge}{\multimapinv} B = \curlywedge(\{\, d \multimapinv b \in C_1 \mid d \in D, b \in B \,\}).$$

From the logic properties of linear implication, it is easy to derive the following distributivity laws for $\overset{\curlywedge}{\multimap}$ and $\overset{\curlywedge}{\multimapinv}$ over reduced product of abstract domains:

- $A \overset{\curlywedge}{\multimap} (\bigsqcap_{i \in I} B_i) = \bigsqcap_{i \in I} (A \overset{\curlywedge}{\multimap} B_i)$
- $(\bigsqcap_{i \in I} B_i) \overset{\curlywedge}{\multimapinv} A = \bigsqcap_{i \in I} (B_i \overset{\curlywedge}{\multimapinv} A)$

Solutions to full completeness problems. The following result characterizes the solution to full completeness problems in terms of linear implications among domain's objects.

Theorem 4.1 *$A \sqcap (C \overset{\curlywedge}{\multimap} A) \sqcap (A \overset{\curlywedge}{\multimapinv} C) \sqcap ((C \overset{\curlywedge}{\multimap} A) \overset{\curlywedge}{\multimapinv} C)$ is the most abstract solution of the system (3.1).*

Solutions to observation completeness problems. Let us first consider left observation completeness problems. The next result characterizes when a left observation completeness equation holds. We follow the notation introduced in the previous section.

Theorem 4.2 $\rho_D \circ \otimes \circ \langle \eta, \rho_{A_2} \rangle = \rho_D \circ \otimes \circ \langle id, \rho_{A_2} \rangle \;\Leftrightarrow\; \eta \sqsubseteq D \overset{\curlywedge}{\multimapinv} A_2$.

Therefore, as an immediate consequence, we get the solution to the left observation completeness problem.

Corollary 4.3 *$A_1 \sqcap (D \overset{\curlywedge}{\multimapinv} A_2)$ is the most abstract solution of the system (3.3).*

Of course, dual results hold for solving right observation completeness problems: In this case, the most abstract solution therefore is $A_2 \sqcap (A_1 \overset{\curlywedge}{\multimap} D)$.

Let us now turn to observation completeness problems. In this case we have the following results.

Theorem 4.4 $\rho_D \circ \otimes \circ \langle \eta, \mu \rangle = \rho_D \circ \otimes \circ \langle id, id \rangle \;\Leftrightarrow\; \langle \eta, \mu \rangle \sqsubseteq \langle D \overset{\curlywedge}{\leftarrow} C_2, C_1 \overset{\curlywedge}{\rightarrow} D \rangle$.

Corollary 4.5 *$\langle A_1 \sqcap D \overset{\curlywedge}{\leftarrow} C_2, A_2 \sqcap C_1 \overset{\curlywedge}{\rightarrow} D \rangle$ is the most abstract solution of the system (3.2).*

4.1 The case of unital commutative quantales

When dealing with unital and commutative quantales – the models of intuitionistic linear logic [1, 18, 15] – the solutions to full completeness problems can be simplified by exploiting their logical properties. The following are well-known properties of linear implication in a unital commutative quantale $\langle C, \otimes \rangle$. For all $a, b, c \in C$:

1.	$a \multimap (b \multimap c) = b \multimap (a \multimap c)$	2.	$1 \multimap a = a$
3.	$c \leq (c \multimap a) \multimap a$	4.	$((c \multimap a) \multimap a) \multimap a = c \multimap a$

In particular, from the above properties, it is not hard to check that for all $a \in C$, $\lambda c.(c \multimap a) \multimap a \in uco(C)$. By exploiting these remarkable properties, solutions to full completeness problems have the following more compact form.

Theorem 4.6 *If $\langle C, \otimes \rangle$ is a unital commutative quantale then $C \overset{\curlywedge}{\rightarrow} A$ is the most abstract solution of the system (3.1).*

Also, next result further simplifies such solution, by providing the following helpful characterization of the uco $C \overset{\curlywedge}{\rightarrow} A$.

Theorem 4.7 *If $\rho \in uco(C)$ is the uco associated with $C \overset{\curlywedge}{\rightarrow} A$, then, for all $c \in C$,*

$$\rho(c) = \bigwedge_{a \in A} (c \multimap a) \multimap a.$$

5 An application in data structure completeness

In this section we consider completeness problems in abstract interpretation of list data structures. We consider the case of lists of natural numbers, even if a similar construction can be applied in the analysis of lists of arbitrary objects.

Consider the typed quantale $\langle \wp(list(\mathbb{N})), \wp(\mathbb{N}), \wp(list(\mathbb{N})), :: \rangle$, where $list(\mathbb{N})$ is the set of all finite lists of natural numbers and $:: \; : \wp(\mathbb{N}) \times \wp(list(\mathbb{N})) \to \wp(list(\mathbb{N}))$ is defined as:

$$N :: L = \{\, [n|l] \mid n \in N \text{ and } l \in L \,\}.$$

where $\emptyset :: L = N :: \emptyset = \emptyset$. A list is *irredundant* if it does not contain twice the same element. Let us define an abstract domain for detecting irredundant lists. Let $\rho \in uco(\wp(list(\mathbb{N})))$ be defined as $\rho = \{\, list(\mathbb{N}), Irr \,\}$, where $Irr \subseteq list(\mathbb{N})$ is the set of irredundant lists over $\mathbb{N}$. We consider ρ as an abstract domain of observation and look for the most abstract solution $\langle X, Y \rangle \in uco(\wp(\mathbb{N})) \times uco(\wp(list(\mathbb{N})))$ to the observation completeness problem:

$$\rho \circ :: \circ \langle \rho_X, \rho_Y \rangle = \rho \circ ::$$

By Theorem 4.5, we get the following solutions:

$$\begin{aligned} X &= \rho \overset{\curlywedge}{\leftarrow} \wp(list(\mathbb{N})) \\ &= \curlywedge(\{\, L \leftarrow M \mid L \in \rho, M \in \wp(list(\mathbb{N}))\,\}) \\ &= \curlywedge(\{\, Irr \leftarrow M \mid M \in \wp(list(\mathbb{N}))\,\}) \\ &= \wp(\mathbb{N}) \\ Y &= \wp(\mathbb{N}) \overset{\curlywedge}{\rightarrow} \rho \\ &= \curlywedge(\{\, N \rightarrow L \mid N \in \wp(\mathbb{N}), L \in \rho\,\}) \\ &= \curlywedge(\{\, N \rightarrow Irr \mid N \in \wp(\mathbb{N})\,\}) \\ &= \bigcup_{N \subseteq \mathbb{N}} \{\, L \in \wp(list(\mathbb{N})) \mid l \in L \;\Leftrightarrow\; (l \in Irr \text{ and } \forall n \in N.\ n \text{ is not in } l)\,\} \end{aligned}$$

Thus, in order to be complete when observing irredundance, we need to consider all sets of natural numbers and only sets of (irreduntant) lists which do not contain a given set of natural numbers. Note that Y is the set of all irredundant lists closed by permutation of their objects.

Let us now consider a standard abstraction $\eta \in uco(\wp(\mathbb{N}))$ like parity analysis given by $\eta = \{\, \mathbb{N}, even, odd, \emptyset\,\}$. Here, we look for the most abstract domains $X \in uco(\wp(\mathbb{N}))$ and $Y \in uco(\wp(list(\mathbb{N})))$, which are, respectively, solutions of the following left and right observation completeness problems:

$$\text{(i)}\;\; \rho \circ :: \circ \langle \rho_X, \rho \rangle = \rho \circ :: \langle id, \rho \rangle \qquad \text{(ii)}\;\; \rho \circ :: \circ \langle \eta, \rho_Y \rangle = \rho \circ :: \langle \eta, id \rangle$$

By Corollary 4.3 (and its dual), we get the following solutions:

$$\begin{aligned} X &= \rho \overset{\curlywedge}{\leftarrow} \rho \\ &= \curlywedge(\{\, list(\mathbb{N}) \leftarrow list(\mathbb{N}), list(\mathbb{N}) \leftarrow Irr, Irr \leftarrow list(\mathbb{N}), Irr \leftarrow Irr\,\}) \\ &= \{\, \mathbb{N}, \emptyset\,\} \\ Y &= \eta \overset{\curlywedge}{\rightarrow} \rho \\ &= \curlywedge(\{\, \mathbb{N} \rightarrow Irr, even \rightarrow Irr, odd \rightarrow Irr, \emptyset \rightarrow Irr\,\}) \\ &= \{\, \{\, [\,]\,\}, Irr_{even}, Irr_{odd}, list(\mathbb{N})\,\} \end{aligned}$$

where $Irr_{even} = \{\, l \in list(\mathbb{N}) \mid l \in Irr$ and l does not contain even numbers $\}$ and $Irr_{odd} = \{\, l \in list(\mathbb{N}) \mid l \in Irr$ and l does not contain odd numbers $\}$. Thus, for problem (i), in order to obtain completeness it is enough to check whether a given set of numbers is empty or not, while, for problem (ii), we only need to consider sets of irreduntant lists which do not contain any even number (and, dually, any odd number).

6 Complete semantics for logic program analysis

In this section we characterize the optimal, i.e. most abstract, semantics for any logic program property abstracting computed answer substitutions. The idea is that the standard semantics of computed answer substitutions can be far too concrete for a given property of interest. We prove that "optimal" semantics for logic programs (i.e. semantics which are neither too concrete nor too abstract) for a given property can be obtained by solving a completeness problem relatively to the basic operation of unification.

6.1 Notation

Let $\mathcal{V}$ be an infinite, recursively enumerable (r.e.) set of variables, Σ be a set of function symbols and Π be a set of predicate symbols, that together determine a r.e. first-order

language $\mathcal{L}$. $Term$ denotes the set of terms of $\mathcal{L}$. For any syntactic object s, $vars(s)$ denotes the set of variables occurring in s. The set of idempotent substitutions modulo renaming on $\mathcal{L}$ is denoted by Sub. If $\sigma, \theta \in Sub$ then $\sigma \circ \theta$ denotes the usual composition of substitutions and $dom(\sigma) = \{ v \in \mathcal{V} \mid \sigma(v) \neq v \}$. Objects in Sub are partially ordered by instantiation, denoted by $\leq$. By adding to Sub an extra object τ as least element, one gets a complete lattice $\langle Sub^\tau, \leq, \vee, \wedge, \epsilon, \tau \rangle$, where $\vee$ is most general anti-instance, $\wedge$ is standard unification and ϵ is the empty substitution [13]. The set of most general atoms is as usual defined by $GAtom = \{ p(\bar{X}) \mid p \in \Pi \}$. We consider logic programs in normalized form, that is, a generic Horn clause is $p(\bar{X}) : -c, q_1(\bar{X}_1), \ldots, q_n(\bar{X}_n)$, where all the tuples of variables are distinct and $c \in Sub$ is the idempotent substitution binding variables to terms.

6.2 T_P-completeness

The basic semantic structure is the unital commutative quantale $\langle \wp(Sub), \otimes \rangle$, where $\wp(Sub)$ is a complete lattice w.r.t. set-inclusion, $\otimes : \wp(Sub) \times \wp(Sub) \to \wp(Sub)$ is the obvious lifting of unification to sets of substitutions: $X \otimes Y = \{ x \wedge y \mid x \in X, y \in Y \}$, and $1 = \{\epsilon\} \in \wp(Sub)$ is the unit. It is immediate to note that, being "collecting", $\langle \wp(Sub), \otimes \rangle$ actually is a unital commutative quantale [15, Example 10, p. 18].

We consider a least fixpoint semantics for logic programs characterizing computed answer substitutions, as usual resembling the s-semantics [2], and whose basic building blocks are given by the operations of unification, existential quantification (projection) over set of variables, and set-union. Our concrete semantic domain $CInt$ is the set of functions (as usual, called interpretations) which map most general atoms to sets of substitutions, that is $CInt = GAtom \to \wp(Sub)$, which ordered pointwise is trivially a complete lattice. Often, we will find convenient to denote an interpretation $I \in CInt$ by the set $\{p(\bar{X}), I(p(\bar{X})) \mid p \in \Pi\}$. The semantics $[\![P]\!]$ of a program P is a function mapping each most general atom into the set of its computed answer substitutions, closed by existential quantification. The result is a semantics which is more abstract than standard s-semantics and more concrete than Clark semantics (also known as c-semantics [9]). It is not a difficult task to prove that this semantics can be obtained as the least fixpoint of a continuous operator $T_P : CInt \to CInt$ defined as follows: For any $I \in CInt$,

$$T_P(I)(p(\bar{Y})) = \exists_{-\bar{Y}} \bigcup_{C \lll P} (c \otimes \bigotimes_{i=1..n} c_i \otimes (\bar{Y} = \bar{X})),$$

where $C = p(\bar{X}) : -c, q_1(\bar{X}_1), \ldots, q_n(\bar{X}_n)$ – by a slight abuse of notation, c is also thought of as a singleton set – and, for all $i \in [1..n]$, $\langle q_i(\bar{X}_i), c_i \rangle \in I$. Here, $\exists_{-\bar{X}}$ denotes the existential quantification of all variables but those in $\bar{X}$, $C \lll P$ denotes that the clause C of P is renamed apart with fresh variables, and $\langle p(\bar{X}), c \rangle \in I$ is intended modulo renaming.

Given an abstract domain on sets of substitutions $\rho \in uco(\wp(Sub))$, a corresponding abstraction on interpretations $\wr\rho\int \in uco(CInt)$ can be easily defined as follows: For any $I \in CInt$, $\wr\rho\int(I) = \{ \langle p(\bar{X}), \rho(c) \rangle \mid \langle p(\bar{X}), c \rangle \in I \}$. Note that for all $\rho, \eta \in uco(\wp(Sub))$ such that $\rho \sqsubseteq \eta$ it trivially holds $\wr\rho\int \sqsubseteq \wr\eta\int$.

Given a basic domain of properties of interest $\pi \in uco(\wp(Sub))$, our goal is therefore to find the most abstract domain containing π (more precisely, $\wr\pi\int$) and fully complete for any T_P function. The idea is that of designing a domain which is complete for the

basic operations involved in the T_P definition, and then to prove that, under weak hypothesis, this domain turns out to be the right one. Since every abstract domain is trivially fully complete for union, we only need to concentrate on unification and existential quantification. We first show that, when computing the semantics of a program, we can move existential quantification outside the fixpoint computation. Define the operator $\exists : CInt \to CInt$ as follows:

$$\exists(I) = \{\, \langle p(\bar{X}), \exists_{-\bar{X}} c\rangle \mid \langle p(\bar{X}), c\rangle \in I \,\}.$$

Since $\exists$ turns out to be a completely additive (i.e., preserving all lub's) closure operator on $CInt$, one easily gets that for any continuous function $f : CInt \to CInt$, $\exists(lfp(f)) = lfp(\exists \circ f)$. Following the notation above, if we define $T_P^\sharp : CInt \to CInt$ as follows:

$$T_P^\sharp(I)(p(\bar{Y})) = \bigcup_{C \lll P} (c \otimes \bigotimes_{i=1..n} c_i \otimes (\bar{Y} = \bar{X})),$$

we get that $[\![P]\!] = \exists(lfp(T_P^\sharp))$, i.e. the standard semantics of a program can be defined as an abstraction of a more concrete semantics[3] not involving existential quantification. As usual, all clauses are renamed apart, so that variables clashes are avoided. Obviously, by definition, $T_P = \exists \circ T_P^\sharp$ holds.

A reasonable assumption for a basic abstraction $\pi \in uco(CInt)$ is that it is strictly stronger than existential quantification, i.e. that $\pi \circ \exists = \pi$ holds. This means that $T_P = \exists \circ T_P^\sharp$ is a semantics concrete enough to represent the properties of π, and therefore it can be rightfully considered as a concrete semantics for π.

Proposition 6.1 *Let $\pi \in uco(CInt)$ such that $\pi \circ \exists = \pi$. Then, for any P, $\pi(lfp(T_P)) = \pi(lfp(T_P^\sharp))$.*

Next result shows that any abstract domain which is fully complete for unification $\otimes$ and includes the basic domain π, is fully complete for $T_P^\sharp$ as well.

Theorem 6.2 *Let $\rho \in uco(\wp(Sub))$. If ρ is fully complete for $\otimes$ then, for any P, $\wr\rho\wr \circ T_P^\sharp = \wr\rho\wr \circ T_P^\sharp \circ \wr\rho\wr$.*

As announced, the following key result shows that the least fully complete extension of a basic domain π for $T_P^\sharp$ can always be obtained, by means of the transformer $\wr\cdot\wr$, from the corresponding least fully complete extension $\wp(Sub) \xrightarrow{\curlywedge} \pi$ of π in the quantale $\langle \wp(Sub), \otimes\rangle$, as characterized by Theorem 4.6. Roughly, the idea of the proof is that, for any strict abstraction of $\wp(Sub) \xrightarrow{\curlywedge} \pi$, it is possible to build a program P for which this domain is not fully complete. This can be done when π is a *decidable* abstract domain, in the following natural sense.

Definition 6.3 $\pi \in uco(\wp(Sub))$ *is* decidable *if any $x \in \pi$ is a r.e. set.*

It should be clear that decidability is a reasonable requirement for most abstract domains used in program analysis: When dealing with a decidable abstraction, an effective procedure for checking whether a substitution belongs to (is approximated by) an abstract object is available.

[3]Being existential quantification $\exists$ a uco, it can be viewed as an abstraction of $CInt$.

Theorem 6.4 *Let $\pi \in uco(\wp(Sub))$ be decidable. Then, $\wr \wp(Sub) \stackrel{\wedge}{\rightarrow} \pi \int$ is the least fully complete extension of π for any $T_P^\sharp$.*

Thus, as a consequence, we readily gets the following main result which states that, for a wide family of analyses, the most abstract domains which are complete for the semantics of logic programs can be systematically derived by looking for the most abstract domains complete for unification.

Corollary 6.5 *Let $\rho \in uco(\wp(Sub))$ be decidable and $\exists \sqsubseteq \wr \rho \int$. Then, $\wr \wp(Sub) \stackrel{\wedge}{\rightarrow} \rho \int$ is the least fully complete extension of $\wr \rho \int$ for any T_P.*

6.3 Complete semantics for groundness analysis

Logic program groundness analysis aims to statically detecting whether variables will be bound to ground terms in successful derivations. Groundness analysis is arguably one of the most important analysis for logic-based languages. By exploiting the results above, if $\mathcal{G}$ denotes the basic abstract domain representing groundness information, we are here able to characterize the least fully complete extension of $\mathcal{G}$ for any fixpoint transformer T_P.

If $V \subseteq \mathcal{V}$ is a finite set of variables of interest, the simplest abstract domain for representing groundness information of variables in V is $\mathcal{G}_V = \langle \wp(V), \supseteq \rangle$, as first put forward by Jones and Søndergaard [11]. The intuition is that each $W \in \mathcal{G}_V$ represents the set of substitutions which ground every variable in W. $\mathcal{G}_V$ is related to the concrete domain $\wp(Sub)$ by the following concretization function: For each $W \in \mathcal{G}$, $\gamma_{\mathcal{G}_V}(W) = \{ \theta \in Sub \mid \forall v \in W.\, vars(\theta(v)) = \emptyset \}$. As usual, we shall abuse of notation and $\mathcal{G}_V$ will also denote the corresponding isomorphic image $\gamma_{\mathcal{G}_V}(\mathcal{G}_V)$.

A variable independent abstract domain $\mathcal{G} \in uco(CInt)$ for representing groundness information can be defined by:

$$\mathcal{G}(I) = \{ \langle p(\bar{X}), \mathcal{G}_{\bar{X}}(c) \rangle \mid \langle p(\bar{X}), c \rangle \in I \}.$$

It is easily seen that $\mathcal{G}$ is an uco on our concrete semantic domain $CInt$. Moreover, existential quantification turns out to be more concrete than $\mathcal{G}$ and $\mathcal{G}$ is clearly decidable.

Proposition 6.6 *$\mathcal{G} \circ \exists = \mathcal{G}$ and $\mathcal{G}$ is decidable.*

Thus, as an easy consequence of Theorem 6.4, we can prove the following result, where $V \subset_f \mathcal{V}$ means that V is a finite subset of $\mathcal{V}$.

Theorem 6.7 *$\bigsqcap_{V \subset_f \mathcal{V}} \wr \wp(Sub) \stackrel{\wedge}{\rightarrow} \mathcal{G}_V \int$ is the least fully complete extension of $\mathcal{G}$ for any T_P.*

Example 6.8 Consider the language $\mathcal{L} = \{ a/0, f/1 \} \cup \mathcal{V} \cup \Pi$, where $\Pi = \{ p/1 \}$ and $\mathcal{V} = \{X_i\}_{i \in \mathbb{N}}$. Then, let us consider the following program P built over $\mathcal{L}$:

$$p(X_0) : - X_0 = a.$$
$$p(X_0) : - X_0 = f(X_1).$$

Recall that the Clark semantics $[\![P]\!]^c$ (also known as c-semantics [9]) of a program P characterizes the correct answer substitutions for P, and can be obtained from the s-semantics $[\![P]\!]^s$ by the closure under instantiation $\lambda X.\downarrow X$: Thus, in our notation, for any predicate symbol p, $[\![P]\!]^c(p(\bar{X})) = \downarrow([\![P]\!]^s(p(\bar{X}))) = \downarrow([\![P]\!](p(\bar{X})))$. It is then an easy task to check that, for any variable $X_i \in \mathcal{V}$,

$$
\begin{array}{l}
[\![P]\!]^s(p(X_i)) = \{\{X_i \leftarrow a\}\} \cup \{\{X_i \leftarrow f(X_j)\} \mid X_j \in \mathcal{V}, X_i \neq X_j\} \\
[\![P]\!](p(X_i)) = \{\{X_i \leftarrow a\}\} \cup \{\{X_i \leftarrow f(t)\} \mid t \in Term, X_i \notin vars(t)\} \\
[\![P]\!]^c(p(X_i)) = \{\sigma \in Sub \mid \forall X_j.\sigma(X_i) \neq X_j\}.
\end{array}
$$

In this case, the least fully complete extension of Theorem 6.7, when applied to the s-semantics $[\![P]\!] \in CInt$, acts as follows:

$$((\bigcap_{V \subset_f \mathcal{V}} \wr \wp(Sub) \stackrel{\curlywedge}{\rightarrowtriangle} \mathcal{G}_V \wr)([\![P]\!]))(p(X_i)) = Sub.$$

Before proving this last equality, let us informally justify it. Intuitively, a fully complete semantics including groundness information should be able to observe both groundness and failures: Groundness, of course, because it is as concrete as the domain of observation, i.e. the groundness domain; failures because, being fully complete for unification, it is able to discriminate between succeeded and failed computations. Therefore, when considering a predicate whose computed answers include all possible ground substitutions and some nonground substitutions, this ensures us that we can have neither groundness nor failures, since for all possible goals, we have at least one answer for that goal. Thus, the most abstract fully complete semantics including groundness information does not need to distinguish between such a program and one whose computed answers are all possible substitutions, since both such programs have the same (negative) result for groundness analysis and do not fail for any goal.

Let us now prove the equality above by proving that, for all $V \subset_f \mathcal{V}$ and $g \in \mathcal{G}_V$, $[\![P]\!](p(X_i)) \rightarrowtriangle g = g$. It suffices to show that $[\![P]\!](p(X_i)) \rightarrowtriangle \emptyset = \emptyset$, and, for any $v \in \mathcal{V}$, $[\![P]\!](p(X_i)) \rightarrowtriangle \{v\} = \{v\}$. $\delta \in [\![P]\!](p(X_i)) \rightarrowtriangle \emptyset$ iff for all $\theta \in [\![P]\!](p(X_i))$ it holds $\delta \wedge \theta = \tau$. But note that, given any substitution δ, there always exists a $\sigma \in [\![P]\!](p(X_i))$ which unifies with δ, and therefore $[\![P]\!](p(X_i)) \rightarrowtriangle \emptyset = \emptyset$. On the other hand, for each $v \in \mathcal{V}$, $\delta \in [\![P]\!](p(X_i)) \rightarrowtriangle \{v\}$ iff $\forall \theta \in [\![P]\!](p(X_i))$ such that $\delta \wedge \theta \neq \tau$ it holds $\delta \wedge \theta \in \{v\}$. We have the following cases, where Z is a fresh variable not in δ:

$$
\begin{array}{llllll}
\delta(X_i) = a & \Rightarrow & \delta \wedge \{X_i \leftarrow a\} = \delta & \Rightarrow & \delta \in v; \\
\delta(X_i) = f(t) & \Rightarrow & \delta \wedge \{X_i \leftarrow f(t)\} = \delta & \Rightarrow & \delta \in v; \\
\delta(X_i) = X_j & \Rightarrow & \delta \wedge \{X_i \leftarrow f(Z)\} \in v \text{ iff } \delta \in v & \Rightarrow & \delta \in v.
\end{array}
$$

Hence, by Theorem 4.7,

$$
\begin{array}{ll}
 & (\bigcap_{V \subset_f \mathcal{V}} \wr \wp(Sub) \stackrel{\curlywedge}{\rightarrowtriangle} \mathcal{G}_V \wr)([\![P]\!])(p(X_i)) \\
= & \bigwedge_{V \subset_f \mathcal{V}} \bigwedge_{g \in \mathcal{G}_V} ([\![P]\!](p(X_i)) \rightarrowtriangle g) \rightarrowtriangle g \\
= & \bigwedge_{V \subset_f \mathcal{V}} \bigwedge_{g \in \mathcal{G}_V} g \rightarrowtriangle g \\
= & Sub.
\end{array}
$$

It is also worth noting that if we consider the equivalence relation on programs induced by the semantics $(\bigcap_{V \subset_f \mathcal{V}} \wr \wp(Sub) \stackrel{\curlywedge}{\rightarrowtriangle} \mathcal{G}_V \wr)([\![P]\!])$, the program P is in the same equivalence class of the program $\{p(X_0).\}$. ∎

7 Conclusion

Few applications are known of algebraic semantics of linear logic. We believe that the strong connection here given between completeness in abstract interpretation and linear implication might be a source for a number of applications of linear logic in static program analysis, and, more in general, in abstract interpretation.

References

[1] V.M. Abrusci. Non-commutative intuitionistic linear propositional logic. *Z. Math. Logik Grundlag. Math.*, 36:297-318, 1990.

[2] A. Bossi, M. Gabbrielli, G. Levi, and M. Martelli. The s-semantics approach: theory and applications. *J. Logic Program.*, 19-20:149–197, 1994.

[3] P. Cousot. Types as abstract interpretations (Invited Paper). In *Proc. 24th ACM POPL*, pages 316–331, 1997.

[4] P. Cousot. Constructive design of a hierarchy of semantics of a transition system by abstract interpretation (Invited Paper). In *Proc. of the 13th Int. Symp. on Math. Found. of Programming Semantics (MFPS'97)*, Electronic Notes in Theor. Comput. Sci., 1997.

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

[6] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Proc. 6th ACM POPL*, pages 269–282, 1979.

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

[8] P. Cousot and R. Cousot. Abstract interpretation of algebraic polynomial systems. In *Proc. 6th Int. Conf. on Algebraic Methodology and Software Technology (AMAST'97)*, LNCS 1349, pages 138–154, 1997.

[9] M. Falaschi, G. Levi, M. Martelli, and C. Palamidessi. Declarative modeling of the operational behavior of logic languages. *Theor. Comput. Sci.*, 69(3):289–318, 1989.

[10] R. Giacobazzi and F. Ranzato. Completeness in abstract interpretation: a domain perspective. In *Proc. 6th Int. Conf. on Algebraic Methodology and Software Technology (AMAST'97)*, LNCS 1349, pages 231–245, 1997.

[11] N.D. Jones and H. Søndergaard. A semantics-based framework for the abstract interpretation of Prolog. In S. Abramsky and C. Hankin, editors, *Abstract Interpretation of Declarative Languages*, pages 123–142. Ellis Horwood Ltd, 1987.

[12] A. Mycroft. Completeness and predicate-based abstract interpretation. In *Proc. ACM Conf. on Partial Evaluation and Program Manipulation (PEPM'93)*, pages 179–185, 1993.

[13] C. Palamidessi. Algebraic properties of idempotent substitutions. In *Proc. 17th Int. Colloq. on Automata, Languages and Programming (ICALP'90)*, LNCS 443, pages 386–399, 1990.

[14] U.S. Reddy and S.N. Kamin. On the power of abstract interpretation. *Computer Languages*, 19(2):79–89, 1993.

[15] K.I Rosenthal. *Quantales and their Applications.* Longman Scientific & Technical, 1990.

[16] R.C. Sekar, P. Mishra, and I.V. Ramakrishnan. On the power and limitation of strictness analysis. *J. ACM*, 44(3):505–525, 1997.

[17] B. Steffen. Optimal data flow analysis via observational equivalence. In *Proc. 14th Int. Symp. on Math. Found. of Comp. Sci. (MFCS'89)*, LNCS 379, pages 492–502, 1989.

[18] D. Yetter. Quantales and (noncommutative) linear logic. *J. Symbolic Logic*, 55(1):41–64, 1990.