Completeness and predicate-based abstract interpretation[*]

Alan Mycroft
Computer Laboratory, Cambridge University
New Museums Site, Pembroke Street
Cambridge CB2 3QG, United Kingdom
E-mail: `Alan.Mycroft@cl.cam.ac.uk`

Abstract

Traditionally, the theory of abstract interpretation has concentrated on the study of when one interpretation is *sound* (also *safe* or *correct*) with respect to another. We consider the dual notion of when one interpretation is *complete* with respect to another. Under the usual formulation of abstract interpretation, undecidability in general implies that a finitely computable sound abstraction of the standard interpretation is not complete. (For example, if we simplify $643 * (-192)$ to $(+) * (-)$ using the "rule of signs" we cannot expect to retrieve -123456 from the resulting $(-)$, even though we are *certain* that the result is negative.) Based on the idea that compilers can only depend on a finite number of program properties, we augment interpretations with predicate symbols specifying properties of interest (thereby replacing algebraic interpretations with logic interpretations). Interpretation J being sound (resp. complete) with respect to I is now phrased as "all questions (formulae) yielding true for J (resp. I) also yield true for I (resp. J)".

The traditional "rule of signs" turns out to be sound and complete for multiplication but only sound for addition.

Sometimes abstract interpretations have spurious domain elements. The state minimisation algorithm for finite deterministic automata can be used to produce a canonical (simplest) abstract interpretation which is sound and complete with respect to any given finite abstract interpretation but possibly simpler to compute.

A homomorphism always yields a sound and complete abstraction. Moreover, we show that a sound and complete abstraction map is not necessarily a homomorphism, but its composition with the natural map to the canonical interpretation is a homomorphism.

One side-effect of our formulation of abstract interpretation is that it de-emphasises the ordering on the abstract domain which is relegated to an (optional) proof basis.

[*]Research supported by (UK) SERC grant GR/H14465.

1 Introduction

Currently there are two main systematic frameworks for obtaining properties of programs. One is the technique of abstract interpretation created by the Cousots ([5] is a recent reference). Another is the technique of property inference which became well-known from the Hindley-Milner-Damas type reconstruction algorithm for ML.

Abstract interpretation rests on formulating and then evaluating a non-standard meaning (or approximation thereto) for a program. The result then enables one to make deductions concerning computation of the standard meaning—correctness is usually proved by the close connection between standard and non-standard computations. Abstract interpretation has an algebraic and domain-theoretic flavour and has been extensively developed in denotational form by the Nielsons ([12] is a definitive work). The Cousots have also been responsible for major developments including recent work [3] which extended the notion of operational semantics so as to enable denotational semantics to be exhibited as an abstraction of operational semantics.

Property inference is based on formulating a set of inference rules which specify the program property of interest. Standard program evaluation can be used to specify a notion of validity (in the logical sense) and correctness is then a matter of showing soundness of the inference rules (*i.e.* that provability implies validity).

Actual calculation of properties tends to differ: abstract interpretation by-and-large proceeds by symbolic evaluation with construction (or approximation) of fixpoints being used for loops or recursion. On the other hand property inference systems tends to use some form of unification as a computation method. These seemingly disparate notions of calculation can be reconciled by observing that the equation $x = f(x)$ can either be seen as a request either for a fixpoint iteration or alternatively for a unification (relative to a system of equalities).

Relatively few works have attempted to reconcile these two formulations. Mycroft and Jones [10] manage to exhibit the Hindley-Milner type system as an abstract interpretation of the untyped λ-calculus but the result is much more unwieldy to use than the inference rule formulation. Benton [2] and Jensen [9] independently formulate strictness properties as logical statements. Finally, the above work of the Cousots [3] can be seen as using abstract interpretation to relate inference systems.

The present work was developed as a position paper in which we argue for an extension of the concept of interpretation from algebraic to logical form. In abstract interpretation we traditionally see an interpretation as being of algebraic form consisting of a domain of discourse and functions operating thereon; we propose adding interpretable predicates as in the logical notion of interpretation. This has practical benefits in explaining various phenomena seen in the literature, some of which are given in section 1.1. Moreover it helps bridge the gap between abstract interpretation and property inference systems.

For simplicity and our desire to expose the connection between notions in abstract interpretation and algebraic concepts such as homomorphism, this version of the paper is formulated in a first-order algebraic meta-language. This suffices for first-order (independent attribute) abstract interpretation without a fixpoint operator. However, some examples are taken from a wider class of situations.

For uses in abstract interpretation, we assume an underlying object language whose phrases are given (denotational) semantics by primitive recursive translation into meta-language terms. These meta-language terms are then traditionally interpreted via *algebras* for standard and abstract semantics which provide a carrier set (possibly with additional domain structure) and functions on this set. Often this algebra is many-sorted to reflect multiple syntactic categories but this adds nothing but complication to the results herein.

For the rest of this paper we only study these meta-language terms, their interpretations and relationships between interpretations. Soundness results carry across to object language phrases by structural induction. A sufficient condition for completeness results to carry across is that the denotation function allows every meta-language term to be generated by some object-language phrase. (See section 1.3 for an informal connection with full abstractness.)

Section 2 deals with the traditional formulation of abstract interpretation and section 3 with the new predicate-based system. Sections 4–6 consider various examples—a collecting interpretation analogue, projection-based strictness and a group theoretic example.

One practical application of completeness is that of characterising whether two abstract properties of a program have any mutual dependence—remove the predicate identifying each of the properties in turn. The properties are *fully independent* or *orthogonal* if the number of values in the canonical forms of the two reduced interpretations factorise the number in the joint interpretation. Similarly, one is functionally dependent on another if removing the former does not affect the number of abstract values.

1.1 Examples

The work we describe can be used to explain the following examples.

Typically introductions to abstract interpretation start with the "rule of signs", whereby integer constants are abstracted by a set S of signs $\{(+), (-), (=)\}$. The operation of integer multiplication then induces an operation on S. However, the operation of addition requires S to contain a least upper bound value $(\pm)$ representing the uncertainty of $(+) + (-)$. Thus abstraction functions are often referred to as 'semi-homomorphisms'. This causes some initial surprise and a general feeling that this happens by the coincidence of abstraction being a homomorphism for multiplication but not for addition. Our framework shows this is not merely a coincidence, but a consequence of the *signs* interpretation being *complete* for questions involving the sign of expressions only involving multiplication, but incomplete for those involving addition. Section 6 contains a group-theoretic homologue.

Moreover, Ernoult and Mycroft [6] observed that the untyped higher-order strictness interpretation of Hudak and Young [7] contained spurious domain points. In particular, the former demonstrated that strictness domain $S_{EM} = 2 \times (S_{EM} \to S_{EM})$ could provide exactly the same information as the latter's domain $S_{HY} = (\mathcal{P}(X), \supseteq) \times (S_{HY} \to S_{HY})$ in spite of there not being a function $\gamma : S_{EM} \to S_{HY}$ which recreates HY-strictness from EM-strictness. The key to understanding this situation, which the current work addresses, is that the strictness information in either domain was only used in a testing predicate $p \subseteq S$, $(S \in \{S_{EM}, S_{HY}\})$, defined by $p(v, f) \equiv v = \bot$.

In this work we use abstraction *functions* which map from one interpretation's universe to another's, although the ideas appear to extend to abstraction *relations*. Given this framework it is standard first to define a *collecting interpretation.* This is derived by lifting the *standard interpretation* to operate on sets of standard values (representing properties). Here we do not wish to restrict abstraction functions to act between abstract interpretations including the collecting interpretation but also allow them to map directly from the standard semantics to an abstract interpretation. In this we follow Reddy and Kamin's recent work [13]. Section 4 shows that adjoining a Herbrand-indexed class of predicates to the set of function symbols can achieve a similar effect directly in the standard interpretation.

1.2 Related work

Recently, there have been classes of work which use the word completeness in association with abstract interpretation.

Sekar, Mishra and Ramakrishnan [14] defined a weaker notion of completeness from that discussed in this paper. They showed that Mycroft's original strictness analysis was complete with respect to a standard interpretation involving flat domains in the sense that Mycroft's strictness interpretation identified functions which remain strict under chaotic replacement of any integer value by any other during evaluation under the standard interpretation. Reddy and Kamin [13] develop this idea in a denotational form by defining two functions to be *similar* in the standard interpretation if, when applied to identical argument sequences, they have equal abstractions. This is a finer equivalence than "have equal abstractions". An abstract interpretation said to be complete if, letting f, g be functions and f^α, g^α their abstractions under abstraction function α, we have that any information loss in $\alpha \circ f \circ g \sqsubset f^\alpha \circ g^\alpha \circ \alpha$ can be removed by replacing f or g by a *similar* function.

Benton [2] and Jensen [9] independently formulated strictness properties as logical statements by adapting Abramsky's idea of 'domain logic'. Jensen provides strictness logic rules and shows these to be sound and complete with respect to Burn, Hankin and Abramsky strictness inference. Benton shows his strictness logic rules soundly and completely axiomatise strictness properties when interpreted as ideals of the standard semantics. Moreover he then gives rules for strictness formula assignment which are sound, but not complete, with respect to the standard semantics.

Steffen [15] and together with Jay and Mendler [16] define[1]

[1] I am grateful to an anonymous referee for these references.

a very similar notion of completeness to ours—instead of using interpretable predicates they use (algebraic-style) interpretations. Two such interpretations I_1 and I_2 are compared *via* an observation interpretation K whose values are observed with sound abstraction functions $\alpha_i : I_i \to K$. I_1 and I_2 are then K-observationally equivalent if, for all meta-language terms e, $\alpha_1(e^{I_1}) = \alpha_2(e^{I_2})$ where e^I yields the meaning of e under I. Our notion of sound and complete can be seen as essentially the same as their notion of observationally equivalent with our predicates replacing their observation domains.

However, some natural properties (such as projection-based strictness—see section 5) are not convex properties of the standard interpretation. This leads to a dilemma—either the abstraction function from the standard interpretation to the observation interpretation needs to be generalised to be possibly non-monotonic; or the problematic construction of a non-convex collecting interpretation needs to be resolved. Our use of predicates (which need not be convex-closed sets) sidesteps this issue.

1.3 Nomenclature

There is a large and overlapping variety of terminology, both in abstract interpretation and in related subject areas, used to describe notions corresponding to soundness and completeness. A good deal of this variety probably arose from the slow unification of ideas from logic and programming language semantics.

In abstract interpretation the words *correctness, soundness* and *safety* are used equivalently by various authors. Such words naturally evoke their duals *completeness* and *liveness.*

In logic, soundness and completeness are used to relate proof-based validity with semantics-based truth. This tradition is followed (possibly with relative completeness replacing completeness) in program logics (*e.g.* Hoare logic).

In abstract interpretation Nielson's terminology is probably the most careful: he uses *safety* for the relation between abstract interpretations and *correctness* for relations to the standard semantics. Indeed, it appears odd at first sight to claim one interpretation is sound with respect to another as logical soundness reflects a property holding over all interpretations. However, in type inference, the standard semantics of a programming language is used to set up a notion of truth, and other (type-valued) semantics (typically specified by rules) are then shown to be sound. Against this backdrop, it is then natural to say that one interpretation (or type system) is sound with respect to another.

The phrases *computationally adequate* and *fully abstract* are used to describe analogous ideas relating operational and denotational semantics *at the object-language level* where notions of substitution or composition may be lacking or may mismatch the corresponding semantic ideas.[2]

The concept of *conservative extension*, is used in logic to describe the situation when augmenting a set of inference rules R with further rules R' results in no change to the set of theorems. Indeed one could reasonably alternatively express that R is complete for $R \cup R'$.

[2]Note also the parallel between logical validity and denotational semantics and between provability and operational semantics.

2 Formalism

We assume a set of variables X, ranged over by x, and a set $\vec{F}$, ranged over by F_i, of function symbols each with their arity k_i. $\vec{F}$ is called the *functional signature.* The set of terms, E ranged over by e, is given by

$$e \quad ::= \quad x \mid F_i(e_1, \ldots, e_{k_i}).$$

Later, we will wish to consider logical formulae and hence assume a set $\vec{P}$, ranged over by P_k, of predicate symbols each with their arity r_i. $\vec{P}$ is called the *predicate signature.* The set of formulae, Φ ranged over by ϕ, is given by

$$\phi \quad ::= \quad P_i(e_1, \ldots, e_{r_i}).$$

For this paper, we only consider atomic formulae—the justification being that in abstract interpretation we are typically interested in safety (implicational) properties. There is a simple extension to monotonic logical connectives but non-monotonicity of (*e.g.*) implication rules out simple extension to all logical connectives. For example, ϕ (respectively ψ) may safely approximate ϕ' (resp. ψ') but $\phi \Rightarrow \psi$ not safely approximate $\phi' \Rightarrow \psi'$.

A classical *interpretation*, I, is a pair (U, f_i) which provides a set U (possibly with domain structure) as domain of discourse and functions $f_i \in U^{k_i} \to U$ interpreting the F_i. When we relate interpretations by abstraction, the more abstract interpretation requires a partial order structure on U and this to be preserved by the f_i. Although we start by considering such interpretations, we soon generalise by requiring an interpretation to be a triple (U, f_i, p_i) where $p_i \subseteq U^{r_i}$ interprets predicate symbol P_i. In this case we will not require any *a priori* ordering on U as implications between formulae play the rôle of orderings. We use $J = (V, g_i, q_i)$ as an alternative interpretation.

Given an interpretation $I = (U, f_i, p_i)$ we write U^X for the space of environments $X \to U$ so that a term e induces a function $e^I \in U^X \to U$. Similarly a formula ϕ induces a predicate $\phi^I \subseteq U^X$.

Given an (abstraction) map $\alpha : U \to V$ and $n \in \mathbb{N}$ we write $\alpha^n : U^n \to V^n$ and $\alpha^X : U^X \to V^X$ for its pointwise extension to tuples and environments. We sometimes write $\langle \alpha, \ldots, \alpha \rangle$ for either of these.

2.1 Abstract interpretation

Given two interpretations (U, f_i) and (V, g_i) we traditionally (see for example Abramsky and Hankin's book) say that $\alpha : U \to V$ is a (sound) abstraction function if, for all i, we have

$$\alpha \circ f_i \sqsubseteq g_i \circ \langle \alpha, \ldots, \alpha \rangle.$$

By induction, and the assumptions that V is partially ordered by $\sqsubseteq$ and the g_i are $\sqsubseteq$-monotonic, we have that, for all terms e,

$$\alpha \circ e^I \sqsubseteq e^J \circ \alpha^X$$

which is the traditional soundness result.

In the above we allow the possibility of I being the standard interpretation.[3] If we define $\Gamma : V \to \mathcal{P}(U)$ and

[3]In this case Nielson would use $\beta : I \to J$ for such an abstraction map, reserving α for A and γ for Γ above.

$\mathrm{A} : \mathcal{P}(U) \to V$ (here $\mathcal{P}(\cdot)$ denotes the *powerset*) by

$$\begin{aligned} \mathrm{A}(S) &= \bigsqcup \{\alpha(u) \mid u \in S\} \\ \Gamma(v) &= \{u \in U \mid \alpha(u) \sqsubseteq v\} \end{aligned}$$

then we have the traditional galois connection view, including the equivalent formulation of soundness via the collecting semantics:

$$\mathrm{A} \circ e^{COLL} \sqsubseteq e^J \circ \langle \mathrm{A}, \dots, \mathrm{A} \rangle.$$

Here $COLL$ is the (independent attribute) collecting interpretation for I given by

$$(\mathcal{P}(U); \lambda(S_1, \dots S_{k_i}).\{f_i(\vec{x}) \mid \vec{x} \in S_1 \times \dots \times S_{k_i}\}).$$

Note that the trivial interpretation $(\{*\}, \lambda\vec{x}.*)$ is a sound abstraction of any interpretation. In a sense we now start to make formal, we wish to say that it is *incomplete*.

2.2 An inadequate definition of completeness

It is tempting to define an abstract interpretation J to be complete for an interpretation I under an abstraction map $\alpha : I \to J$ if there is a function $\gamma : J \to I$ which satisfies, for all terms e, that

$$e^I = \gamma \circ e^J \circ \langle \alpha, \dots, \alpha \rangle.$$

Indeed, this ensures the collecting interpretation is a complete abstraction of the standard semantics *via* the embedding $\alpha(x) = \{x\}$.

However, this results in a trivial theory in that, in general, it requires γ to be a surjective map of J onto I. This requires that J be at least as large in cardinality as I which is unfortunately the opposite of what we expect of most other abstractions.

The rest of this paper builds a notion of abstraction (and hence completeness) induced from a notion of observability.

3 Abstract interpretation with predicates

We wish to criticise the above view of abstract interpretation as a little oversimplistic. The exact domain elements in V are unimportant: what matters is the information they give us for optimising a program.[4]

So here we augment the traditional view of abstract interpretation by requiring an interpretation to specify a meaning for each predicate symbol.[5] For example in the "rule of signs" we might typically have an abstract interpretation with lattice $\{(+), (-), (=), (\pm)\}$ with the obvious interpretations for meta-language function symbols (and constants). Moreover, we might further wish to have predicate symbols *mustbezero*, *maybepos*, whose truth sets are $\{0\}$, $\{1, 2, 3, \dots\}$ in the standard interpretation and $\{(=)\}$, $\{(+), (\pm)\}$ in the "rule of signs" interpretation. Soundness now does not depend on the absolute notion of set membership but on the relative notion of predicate interpretation which we now formalise.

[4]A fine analogy is that of state-reduction in finite state machines. The exact details of the internal states are unimportant: what matters is the observable output which is used to merge equivalent states.

[5]There is a slight infelicity here in that we might have two predicate symbols, one of which is implied by the other in all interpretations, thereby giving rise to only three rather than four possibilities. We might wish to avoid this by considering integer-valued predicates but here we merely ignore it as the theory is unaffected.

3.1 Soundness and completeness

Given predicate-based interpretations $I = (U, f_i, p_i)$ and $J = (V, g_i, q_i)$, we say an abstraction map $\alpha : I \to J$ is *sound* if, for all formulae ϕ, we have[6] that $\phi^J \circ \alpha^X \Rightarrow \phi^I$. Similarly we say it is *complete* if $\phi^I \Rightarrow \phi^J \circ \alpha^X$.

Observe that there are two trivial interpretation schemas $FF = (\{*\}, \lambda\vec{x}.*, \{\})$ and $TT = (\{*\}, \lambda\vec{x}.*, \{(*, \dots, *)\})$ with one value, functions which always return this value and predicates which constantly yield false (resp. true). Any abstraction map $\alpha : I \to FF$ is sound, but not complete in general whereas $\alpha : I \to TT$ is complete, but not sound in general.

Note that the previous definition of abstract interpretation fits neatly within this scheme and the embedding is illuminating—for each abstract value $v \in V$ we have a unary predicate symbol P_v with interpretations $p_v(x) \Leftrightarrow \alpha(x) \sqsubseteq v$ and $q_v(x) \Leftrightarrow x = v$. (Recall that traditional abstract interpretations require an ordering $\sqsubseteq$ on the abstract space reflecting property implication.) Our soundness criterion $(\forall\phi)\ (\phi^J \circ \alpha^X \Rightarrow \phi^I)$ now reduces to

$$(\forall v \in V)\ (e^J \circ \alpha^X = v \Rightarrow \alpha \circ e^I \sqsubseteq v)$$

for all terms e which simplifies to the traditional formulation $\alpha \circ e^I \sqsubseteq e^J \circ \alpha^X$.

The above soundness and completeness criteria are rather global and non-compositional so we look for local formulations.

First considering soundness, meta-language terms solely consisting of variables show it is necessary to have

1. $q_i \circ \alpha^{r_i} \Rightarrow p_i$.

Now, suppose we do in fact have a reflexive and transitive relation $\sqsubseteq$ on V, then a sufficiency condition for soundness is that, additionally, we have

2. The g_i are $\sqsubseteq$-monotonic.
3. $\alpha \circ f_i \sqsubseteq g_i \circ \langle \alpha, \dots, \alpha \rangle$. (†)
4. $(\forall \vec{x}, \vec{y} \in V^{r_i})\ (\vec{x} \sqsubseteq \vec{y} \Rightarrow q_i(\vec{y}) \Rightarrow q_i(\vec{x}))$.

(Conditions 2 and 3 ensure that $\alpha \circ e^I \sqsubseteq e^J \circ \alpha^X$ as before and the final point converts the order relation into implication.)

For completeness, a necessary condition is

1. $p_i \Rightarrow q_i \circ \alpha^{r_i}$.

Again, supposing we have a (possibly different) reflexive and transitive relation $\sqsubseteq$ on V, the following additional conditions suffice for completeness:

2. The g_i are $\sqsubseteq$-monotonic.
3. $g_i \circ \langle \alpha, \dots, \alpha \rangle \sqsubseteq \alpha \circ f_i$. (‡)
4. $(\forall \vec{x}, \vec{y} \in V^{r_i})\ (\vec{x} \sqsubseteq \vec{y} \Rightarrow q_i(\vec{y}) \Rightarrow q_i(\vec{x}))$.

There is a connection between α being a sound and complete abstraction and it being a homomorphism in that if the two $\sqsubseteq$ relations coincide and are antisymmetric then (†) and (‡) require α to be a homomorphism between (U, f_i)

[6]This notation is a natural abbreviatation for

$$(\forall \rho \in U^X)\ (\phi^J(\alpha^X \circ \rho) \Rightarrow \phi^I(\rho)).$$

and (V, g_i) seen as algebras. In this case α is a sound and complete abstraction.

The converse does not hold in that we can abstract $(\mathbb{Z}, \times)$ with a perverted "rule of signs" with $(\{(-), (=), (\epsilon), (+)\}, \otimes)$ where

$$\alpha \left\{ \begin{array}{cl} (-\infty, -1] & \mapsto (-), \\ \{0\} & \mapsto (=), \\ [1, 9] & \mapsto (\epsilon), \\ [10, \infty) & \mapsto (+) \end{array} \right.$$

and abstract multiplication $\otimes$ is given by the table

$\otimes$	$(-)$	$(=)$	(ϵ)	$(+)$
$(-)$	$(+)$	$(=)$	$(-)$	$(-)$
$(=)$	$(=)$	$(=)$	$(=)$	$(=)$
(ϵ)	$(-)$	$(=)$	$(+)$	$(+)$
$(+)$	$(-)$	$(=)$	$(+)$	$(+)$

Now, although α is not a homomorphism, it is sound and complete with respect to the question "is the result of a computation negative". In other words it is a homomorphism relative to predicate $P(\cdot)$ with interpretations $p(z) \Leftrightarrow z < 0$ and $q(z) \Leftrightarrow z = (-)$. The next section shows that this situation is quite general in that we can quotient this abstract interpretation by identifying (ϵ) and $(+)$ yielding a homomorphism which giving exactly the same information *via* P.

3.2 Canonical abstract interpretation

Taking our cue from Burstall's slogan that implementations of algebraic specifications should have "no junk and no confusion", given an abstract interpretation, we may wish to remove unreachable elements from our abstract domain and quotient together indistinguishable elements (the latter is strongly desirable so that concretisation maps can be injective).

Hence suppose that we have two interpretations, $I = (U, f_i, p_i)$ and $J = (V, g_i, q_i)$, as before and a sound abstraction map $\alpha : I \to J$. Define the equivalence relation $\sim$ on V by

$$v \sim v' \Leftrightarrow (\forall \phi \in \Phi, \forall \rho \in U^X) \quad (\phi^J(\alpha^X \circ \rho)[v/x] \Leftrightarrow \phi^J(\alpha^X \circ \rho)[v'/x]).$$

Informally, this construction states that two abstract values (v, v') are indistinguishable if no formula can tell them apart (by associating their value with a given variable $x \in X$) provided that any other free variables of ϕ are restricted to have values in the image $\alpha(U) \subseteq V$.

Almost by construction, the quotient algebra (actually logic interpretation) $K = J/{\sim} \triangleq (V/{\sim}, g_i/{\sim}, q_i/{\sim})$ with associated natural abstraction map $\eta : V \to V/{\sim}$ is sound and complete with respect to J. This generalises the usual construction for state-reducing an automaton.

Proposition If $\alpha : I \to J$ is a sound and complete abstraction then $\eta \circ \alpha : I \to J/{\sim}$ is a homomorphism.

Proof This is messy because of need to keep careful track of meta-language terms and their two interpretations as functions. Soundness and completeness enables us to write

$$(\forall \phi) \ (\phi^I \Leftrightarrow \phi^J \circ \alpha^X)$$

or, more precisely

$$(\forall \phi, \forall \rho \in U^X) \ (\phi^I(\rho) \Leftrightarrow \phi^J(\alpha^X \circ \rho)).$$

We now specialise this to the case $\phi \equiv \psi[F(y_1, \ldots, y_k)/x]$ substituting any occurrences of x in ψ with application of the function symbol F to distinct variables. Now, letting the interpretations of F in I be f and in J be g, we have

$$\begin{array}{rcl} \phi^I(\rho) & \Leftrightarrow & \psi^I(\rho[F(y_1, \ldots, y_k)^I \rho/x]) \\ & \Leftrightarrow & \psi^J(\alpha^X \circ (\rho[F(y_1, \ldots, y_k)^I \rho/x])) \\ & \Leftrightarrow & \psi^J(\alpha^X \circ \rho)[\alpha \circ f(\rho y_1, \ldots, \rho y_k)/x] \end{array}$$

Similarly we have

$$\begin{array}{rcl} \phi^J(\alpha^X \circ \rho) & \Leftrightarrow & \cdots \\ & \Leftrightarrow & \psi^J(\alpha^X \circ \rho)[g(\alpha(\rho y_1), \ldots, \alpha(\rho y_k))/x]. \end{array}$$

Now, since ψ is an arbitrary formula and the ρy_i are arbitrary values in U, this is exactly the criterion that

$$\alpha \circ f \sim g \circ \alpha^k$$

and $\sim$ is, of course, the equality relation on $J/{\sim}$.

Remark. If we define the *reachable* values V^* of V by $V^* = \{v \in V \mid (\exists \text{closed } e) e^J = v\}$, then the set of *relevant* values in V can be seen as the the smallest set W such that

$$W \supseteq V^* \cup \{\alpha(u) \mid u \in U\} \cup \{g_i(w_1, \ldots, w_{k_i}) \mid w_j \in W\}.$$

Removing irrelevant values corresponds to having "no junk" and quotienting away observably equivalent values as above corresponds to having "no confusion". Irrelevant values do not damage the theorem above, but should be removed to ensure the uniqueness of the canonical form of a given abstract interpretation.

4 Collecting interpretation

This section is more speculative than the rest of the paper and attempts to show how predicate-based interpretations form a bridge between abstract interpretation and rule-based property inference formulations of static analysis.

Given a function signature $\hat{F}$ of function symbols (consisting of at least one nullary function symbol), we can form the Herbrand Universe $H_{\hat{F}}$ of all variable-free meta-language terms built from function symbols in $\hat{F}$. We now define the Herbrand unary predicate signature $\hat{P}_1 = \{P_S \mid S \subseteq H_{\hat{F}}\}$. Given an (algebraic) interpretation $I = (U, f_i)$ for $\hat{F}$, we may extend it to a (logic) interpretation $I' = (U, f_i, p_S)$ for $\hat{F} \cup \hat{P}_1$ by defining

$$\begin{array}{rcl} p_S(x) & \Leftrightarrow & x \in \{e^I \mid e \in S\} \\ & \Leftrightarrow & (\exists e \in S) \ x = e^I. \end{array}$$

(We have taken the liberty of writing e^I for $e^I\langle\rangle$ since e is closed.) Taking I to be the standard interpretation results in I' becoming a natural correspondent of the independent attribute collecting interpretation in that we have one predicate for each property.

Given an interpretation function signature $\hat{F}$ and a standard interpretation $STD = (U, f_i)$, the independent attribute collecting interpretation $COLL$ is given by $(\mathcal{P}(U); \mathcal{P}(f_i))$ where

$$\mathcal{P}(f_i) = \lambda(S_1, \dots S_{k_i}).\{f_i(\vec{x}) \mid \vec{x} \in S_1 \times \cdots \times S_{k_i}\}$$

as in section 2.1. Observe that $\mathcal{P}(f_i)(S_1, \dots, S_{k_i}) \subseteq S_0$ holds precisely when the (Herbrant unary) sequent

$$P_{S_1}(x_1), \dots, P_{S_{k_i}}(x_{k_i}) \vdash P_{S_0}(F(x_1, \dots, x_{k_i}))$$

holds under STD' defined above. This result holds because there is a bijection between subsets (properties) in the collecting interpretation $COLL$ and predicates in the (Herbrand) predicate-augmented standard interpretation STD'. Recall that having descriptions for every property is the distinguishing feature of the collecting interpretation. The Cousots study similar notions under the name "predicate transformers" but the sequent form is more directly applicable as a link to rule-based property inference.

For the relational attribute collecting interpretation the natural correspondent is to use predicate signature $\hat{P}_\omega = \bigcup_n \{P^n_S \mid S \subseteq H^n_{\hat{F}}\}$. Again supposing $I = (U, f_i)$ is the standard algebraic interpretation, the logic interpretation is then $I'' = (U, f_i, p^n_S)$ where P^n_S is interpreted by

$$p^n_S(\vec{x}) \Leftrightarrow (\exists \vec{e} \in S)(\forall 1 \leq i \leq n)\ x_i = {e_i}^I.$$

5 Projection-based strictness

Wadler and Hughes [17] introduced the notion of projection-based strictness. Let D be a cpo. A subset S of a cpo D is convex if $(\forall s, s' \in S, \forall d \in D)\ (s \sqsubseteq d \sqsubseteq s' \Rightarrow d \in S)$.

A continuous map $\pi : D \to D$ is a projection if it is idempotent ($\pi \circ \pi = \pi$) and less than identity ($\pi \sqsubseteq 1_{D \to D}$). A function $f : D \to E$ is π-strict if $f \circ \pi = \pi$. Given a projection π, let S_π be the set of π-strict functions in $D \to D$. In general S_π is not a convex subset of $D \to D$ which results in problems in expressing it within a collecting interpretation. Hunt [8] showed how a ternary logical relation $\subseteq (STD \times STD \times I)$ generalising the abstraction relation approach to abstract interpretation could side-step this problem.

Predicate-based interpretations offer an attractive alternative possibility: given a prescribed set of projections, consider an interpretation whose predicate signature has one element for each projection. In the standard interpretation such a predicate P_π is interpreted by $p_\pi(f) \Leftrightarrow f \circ \pi = \pi$. In an abstract interpretation P_π may be interpreted by a computable q_π predicate, for example by an equality test on an abstract value a_π, *e.g.* $q_\pi(x) \Leftrightarrow x = a_\pi$.

6 A group-theoretic example

The following example is intended to show that the ideas of abstract interpretation and semi-homomorphism naturally appear in other algebraic models, here group theory. It generalises example 2.3 in [4].

Let $G = (G; 1, \cdot \times \cdot, \cdot^{-1})$ be a group. Let $S \subseteq G$ be a *subset*. Suppose we wish solve problems of the form "is $e \in S$" where e is a term involving elements of G with operators (functions) $\{1, \cdot \times \cdot, \cdot^{-1}\}$.

If S is a normal subgroup N, or indeed a coset $s \times N \triangleq \{s \times n \mid n \in N\}$ of a normal subgroup, then we have a short-cut calculation: replace each element $g \in G$ occurring in e with its coset $g \times N$ and perform the calculation in the quotient group G/N to yield a coset S'. The original problem "is $e \in S$" is then true iff $S' = S$. The abstraction map of G to G/N mapping elements their N-cosets is sound and complete.

Now suppose that S may now be an arbitrary subset of G, but instead we are only interested in one-way implications whereby the question "is $e \in S$" is answered by *no* or *maybe*. (The converse question for *yes* or *maybe* is obtained by replacing S with $G \setminus S$. The choice of *no* or *maybe* identifies $\sqsubseteq$ with $\subseteq$ rather that $\supseteq$ and so appears more natural.) Moreover, suppose computational or other reasons limit the size of calculation to be performed.

So, choose a suitable sublattice $L \subseteq \mathcal{P}(G)$, with l ranging over L, satisfying the requirement that

$$\alpha(g) = \text{least } l \in L \text{ such that } g \in l$$

and

$$l.l' = \text{least } l'' \in L \text{ such that } \{g \times g' \mid g \in l, g' \in l'\} \subseteq l''$$

be well-defined. There typically many such sublattices of varying levels of approximation. Let this interpretation be called J.

Now the question "is $e \in S$" can be answered as *maybe* or *no* according to whether $e^J \supseteq \alpha(S)$ or not.

If S is a coset of a normal subgroup N then such questions are precisely (*no* or *yes*) answered using the (discrete) sublattice of $\mathcal{P}(G)$ consisting of the set of all cosets of N—this is a consequence of the remark above about the abstraction map $G \to G/N$ being sound and complete.

7 Higher-order meta-language terms

The above description kept to a first-order meta-language so we could exploit the conventional notion of homomorphism.

Unfortunately, while the notion of homomorphism lifts from base types to sum and product types, it does not lift to function spaces. *I.e.*, given $\phi : A \to A'$ and $\psi : B \to B'$ we can define

$$\begin{array}{lll} (\phi \times \psi): & (A \times B) \to (A' \times B'), & (a, b) \mapsto (\phi a, \psi b) \\ (\phi + \psi): & (A + B) \to (A' + B'), & \mathit{in}_1 a \mapsto \mathit{in}_1(\phi a) \\ & & \mathit{in}_2 b \mapsto \mathit{in}_2(\psi b) \end{array}$$

but there is no corresponding definition for

$$(\phi \to \psi): \quad (A \to B) \to (A' \to B')$$

One solution (the relational approach, for example [11, 10, 1]) is to replace the concept of homomorphism in $A \to A'$ (structure-preserving function) with that of structure-preserving relation in $\mathcal{R}(A, A')$ and another to replace it with a pair of functions in $(A \to A') \times (A' \to A)$. In the former approach we define, at base types, $u \lhd v \Leftrightarrow \alpha(u) \sqsubseteq v$. In the latter we use pairs, *e.g.* $\phi = (\phi_1 : A \to A', \phi_2 : A' \to A)$ so that

$$(\phi \to \psi): \quad (A \to B) \to (A' \to B'), \quad f \mapsto (\psi_1 \circ f \circ \phi_2).$$

8 Conclusions and further work

We have shown the possibility of augmenting traditional 'universe-of-discourse and functions' abstract interpretations with an explicit 'predicates' component. This was used to define a notion of completeness between interpretations and then to relate sound complete abstractions to homomorphisms. Moreover, the separation of predicates (for observation) and functions (for manipulation) naturally allows continuous functions and discontinous predicates. Note that the separation of predicates and functions proves necessary in many logics (in very few logical systems is it possible to view predicates as functions to a two-valued space).

However, this work is very much a first step in exploring the idea of predicate-based interpretations and much remains to be done. The works of Steffen *et al.* and Kamin and Reddy on related notions of completeness need to be related with each other and with the current work.

High on the agenda is the addition of higher order functions (using logical relations instead of homomorphism) and the fixpoint operator. The Cousots' widening and narrowing operators need to be considered in this framework.

Moreover, the framework invites various ideas from logic to be used within abstract interpretation. Section 4 suggested that certain Herbrand-inspired predicates could model some facets of collecting semantics. Further developments of this idea could either use fragments of higher-order logic or use such sequents to represent assumptions. Variants of sequent forms might naturally express relational and independent attribute ideas.

Acknowledgments

Thanks are due to Nick Benton for useful discussions and to Patrick Cousot and the anonymous referees for their helpful comments on a draft version.

References

[1] Abramsky, S. Abstract interpretation, logical relations and Kan extensions. *Journal of logic and computation*, vol. 1(1), 1990.

[2] Benton, P.N. Strictness logic and polymorphic invariance. Proc. Logical foundations of computer science, LNCS vol. 620, Springer-Verlag, 1992.

[3] Cousot, P. and Cousot, R. Inductive definitions, semantics and abstract interpretation. Proc. 19th ACM symp. on Principles of Programming Languages, 1992.

[4] Cousot, P. and Cousot, R. Abstract interpretation and application to logic programs. *Journal of Logic Programming*, vol. 13, 1992.

[5] Cousot, P. and Cousot, R. Abstract Interpretation Frameworks. *Journal of Logic and Computation*, vol. 2:4, 1992.

[6] Ernoult, C. and Mycroft, A. Untyped strictness analysis, *Journal of Functional Programming*, to appear. Draft version appears as technical report 269, Cambridge University Computer Laboratory.

[7] Hudak, P. and Young, J. Higher order strictness analysis in untyped lambda calculus. Proc. 13th ACM symp. on Principles of Programming Languages, 1986.

[8] Hunt, L.S. PERs generalise projections for strictness analysis. Departmental report DOC 90/14, Dept. of Computing, Imperial College, London, 1990

[9] Jensen, T.P. Strictness analysis in logical form, Proc. Functional programming languages and computer architecture, LNCS vol. 523, Springer-Verlag, 1991.

[10] Mycroft, A. and Jones, N.D. A relational framework for abstract interpretation. In "Programs as Data Objects", Proc. of a Workshop, Copenhagen, LNCS vol. 215, Springer-Verlag, 1985.

[11] Nielson, F. Abstract interpretation using domain theory. Ph.D. thesis, Edinburgh University, 1984. Available as computer science report CST-31-84.

[12] Nielson, F. Two-level semantics and abstract interpretation. *Theoretical Computer Science*, vol. 69, 1989.

[13] Reddy, U.S. and Kamin, S.N. On the power of abstract interpretation. Proc. IEEE International Conference on Computer Languages, IEEE press, 1992.

[14] Sekar, R.C., Mishra, P. and Ramakrishnan, I.V. On the power and limitation of strictness analysis based on abstract interpretation. Proc. 18th ACM symp. on Principles of Programming Languages, 1991.

[15] Steffen, B. Optimal data flow analysis via observable equivalence. Proc. Mathematical Foundations of Computer Science, LNCS vol. 379, Springer-Verlag, 1989.

[16] Steffen, B., Jay, C.B. and Mendler, M. Compositional characterisation of program properties. *Informatique théorique et Applications* (AFCET), vol. 26, 1992.

[17] Wadler, P.L. and Hughes, R.J.M. Projections for strictness analysis. Proc. of the functional programming and computer architecture conference, LNCS vol. 274, Springer-Verlag, 1987.