Extracting program logics from abstract interpretations defined by logical relations

David Schmidt
Kansas State University

www.cis.ksu.edu/~schmidt
From where do programming logics originate?

Consider Hennessy-Milner logic:

- $c \models p$ is given, for primitive properties, $p$,
- $c \models [f]\phi$, if for all $c' \in f(c)$, $c' \models \phi$
- $c \models \langle f \rangle \phi$, if there exists $c' \in f(c)$ such that $c' \models \phi$

for $c \in C$, where $f : C \to \mathcal{P}(C)$ denotes a nondeterministic transition function/action.

Is "domain theory in logical form" [Abramsky02] hiding here?

We can deconstruct the logic to expose the underlying set-domains, for $S \subseteq C$:

- $S \models \forall \phi$, if for all $c \in S$, $c \models \phi$
- $S \models \exists \phi$, if there exists $c \in S$ such that $c \models \phi$
- $c \models f; \phi$, if $f(c) \models \phi$

(Read $[f]\phi$ as abbreviating $f; \forall \phi$, and read $\langle f \rangle \phi$ as abbreviating $f; \exists \phi$.)
The latter can be expanded into this logic, exposing lower- and upper-powerset constructions as well as function pre- and post-image:

\[ S \models_{L(\tau)} \forall (\bigvee_{i<k} \phi_i), \text{ if for all } c \in S, \text{ there exists } j < k \text{ such that } c \models_{\tau} \phi_j \]

\[ S \models_{U(\tau)} \bigwedge_{i<k} (\exists \phi_i), \text{ if for all } i < k, \text{ there exists } c \in S \text{ such that } c \models_{\tau} \phi_i \]

\[ c \models_{\tau_1} f; \phi, \text{ if } f(c) \models_{\tau_2} \phi, \text{ for } f : C_{\tau_1} \to C_{\tau_2} \]

\[ f(c) \models_{\tau_2} f(\phi), \text{ if } c \models \phi_{\tau_1}, \text{ for } f : C_{\tau_1} \to C_{\tau_2} \]

This judgement set is extracted from Plotkin-style *logical relations* for the types, \( P_{L}(\tau), P_{U}(\tau) \), and \( \tau_1 \to \tau_2 \) [Plotkin80], which generate a Cousot-Cousot-style *abstract interpretation* [Abramsky90, CousotCousot77].
Preview of the talk

1. We show how to define an abstract interpretation via an approximation relation on base type, lifted to compound types via logical relations, à la Abramsky90.

2. We show the coincidence between Galois-connection-based approximation and relational approximation regarding functional soundness and completeness.

3. We show that every abstract domain has an internal logic and show how the logical relations generate logical operators within the internal logic.

4. When there are logical operators that do not fall within an abstract domain’s internal logic, we show how to underapproximate them soundly by means of an external logic generated from the logical relations.
Abstract interpretation: computing on properties

readInt(x)
if x>0 :
    x:= pred(x)
x:= succ(x)
writeInt(x)

Q: Is output \textit{pos}?

\textbf{A:} abstractly interpret domain \textit{Int} by
\[ \text{Sign} = \{ \text{neg, zero, pos, any} \}: \]

\begin{align*}
\text{succ}^\#(\text{pos}) &= \text{pos} & \text{pred}^\#(\text{neg}) &= \text{neg} \\
\text{succ}^\#(\text{zero}) &= \text{pos} & \text{pred}^\#(\text{zero}) &= \text{neg} \\
\text{succ}^\#(\text{neg}) &= \text{any} & \text{pred}^\#(\text{pos}) &= \text{any} \\
\text{succ}^\#(\text{any}) &= \text{any} & \text{pred}^\#(\text{any}) &= \text{any}
\end{align*}

where and

\text{Calculate the \textit{static analysis}:}
\[ \{ \text{zero} \mapsto \text{pos}, \text{neg} \mapsto \text{any}, \text{pos} \mapsto \text{any}, \text{any} \mapsto \text{any} \} \]

The Question is decided only for \textit{zero} — the static analysis is \textit{sound} but \textit{incomplete}. 

\[ \]

readSign(x)
if isPos(x):
    x:= pred^\#(x)
x:= succ^\#(x)
writeSign(x)
The Galois connection underlying the analysis

\[ \gamma : \text{Sign} \rightarrow \mathcal{P}(\text{Int}) \]

- \( \gamma(\text{none}) = \{\} \)
- \( \gamma(\text{any}) = \text{Int} \)
- \( \gamma(\text{neg}) = \{\ldots, -3, -2, -1\} \)
- \( \gamma(\text{zero}) = \{0\} \)
- \( \gamma(\text{pos}) = \{1, 2, 3, \ldots\} \)

\[ \alpha : \mathcal{P}(\text{Int}) \rightarrow \text{Sign} \]

- \( \alpha(S) = \bigcap\{a \mid \gamma(a) \subseteq S\} \)
- e.g., \( \alpha\{2, 4, 6, 8, \ldots\} = \text{pos} \)
- \( \alpha\{-4, -1, 0\} = \text{any} \)
- \( \alpha\{0\} = \text{zero} \)

\((\mathcal{P}(\text{Int}), \subseteq) \langle \alpha, \gamma \rangle (\text{Sign}, \sqsubseteq)\) is a Galois connection: \( \gamma \) interprets the properties in \( \text{Sign} \), and \( \alpha \) maps each concrete set to the property that best describes the set [CousotCousot77].
The Galois connection is a “completion” of an abstraction relation

\[
\begin{align*}
\rho_{Sign} & \subseteq \text{Int} \times \text{Sign} \\
-n \rho_{Sign} & \text{neg} \\
0 \rho_{Sign} & \text{zero} \\
+m \rho_{Sign} & \text{any}, \text{ for all } m \in \text{Int}
\end{align*}
\]

Example: \(+3\) has property \text{pos}, because \(+3 \rho_{Sign} \text{ pos}\).

Intuition: for all \(a \in \text{Sign}, \gamma(a) = \{i \in \text{Int} \mid i \rho_{Sign} a\}\).

\(\rho \subseteq C \times A\) generates a Galois connection between \(C\) and \(A\) iff \(\rho\) possesses U-GLB-L-LUB-closure [Shmuely74,Schmidt04].
A Galois connection defines an internal logic that one uses to compute a static analysis

For \((PC, \subseteq)\langle \alpha, \gamma \rangle (A, \sqsubseteq)\), for all \(S \in PC\), and \(a \in A\), define

\[ S \vDash a \text{ iff } S \subseteq \gamma(a) \text{ (iff } \alpha(S) \sqsubseteq a) \text{ (iff } S \bar{\rho} a) \]

**Example:** For \(Sign\), \(\{2, 5\} \vDash pos\). The G.C. defines this internal logic:

\[ \phi ::= a \mid \phi_1 \sqcap \phi_2 \]

because \(\gamma\) preserves \(\sqcap\) as \(\cap\) (that is, \(\gamma(a_1 \sqcap a_2) = \gamma(a_1) \cap \gamma(a_2)\)):

\[ S \vDash a_1 \sqcap a_2 \text{ iff } S \vDash a_1 \text{ and } S \vDash a_2. \]

**Example:** In \(Sign\), \(\{2, 5\} \vDash pos \sqcap \text{any}\).

More importantly, for all \(a \in A\), \(a \sqsubseteq \phi\) implies \(\gamma(a) \vDash \phi\).

Static analysis crucially depends on this (cf. the earlier example).

But \(Sign\)’s logic excludes disjunction, e.g., \(\{0\} \vDash \text{any} = \text{neg} \sqcup pos\), yet \(\{0\} \not\vDash \text{neg}\) and \(\{0\} \not\vDash pos\) — \(\gamma\) does not preserve \(\sqcup\) as \(\cup\)!
Abstract transformers compute on properties

For \( f : \text{PC} \rightarrow \text{PC} \), \( f^\# : \mathbb{A} \rightarrow \mathbb{A} \) is \textit{sound} iff

\[
\alpha \circ f \sqsubseteq f^\# \circ \alpha \quad \text{iff} \quad f \circ \gamma \sqsubseteq \gamma \circ f^\#
\]

\( \alpha \) and \( \gamma \) act as \textit{semi-homomorphisms}; \( f^\# \) is a \textit{postcondition transformer}.

**Example:** For \( \text{succ} : \mathcal{P} (\text{Int}) \rightarrow \mathcal{P} (\text{Int}) \), \( \text{succ}\{0\} = \{1\} \),
\( \text{succ}^\#(\text{zero}) = \text{pos} \). \textit{This is how a static analysis computes.}

**Properties:** \( f(S) \models f^\#(\alpha(S))\) and \( f(\gamma(a)) \models f^\#(a) \).

For example, \( \{0\} \models \text{zero} \) and \( \text{succ}\{0\} \models \text{succ}^\#(\text{zero}) = \text{pos} \).

\( f^\#_{\text{best}} = \alpha \circ f \circ \gamma \) is the best — strongest postcondition — transformer in \( \mathbb{A}'s \) internal logic.
(Functional) completeness: from semi-homomorphism to homomorphism

For \( f : PC \rightarrow PC \), \( f^\# : A \rightarrow A \):

**Forwards(\( \gamma \))-completeness**

[GiacobazziQuintarelli01] :

\[
\gamma \circ f = f \circ f^\#
\]

\[
\begin{array}{c}
\gamma(a) \\
\gamma \\
a \\
\end{array} \xrightarrow{f} \bullet \\
\gamma \xrightarrow{f^\#} \gamma \xrightarrow{f^\#(a)}
\]

\( \gamma \) is a homomorphism from \( A \) to \( PC \) — it preserves \( f^\# \) as \( f \).

**Theorem:** \( S \models f^\#(a) \) iff \( S \subseteq f(\gamma(a)) \).

That is, \( f^\# \) is a logical operator in \( A \)'s internal logic (like \( \square \) is).

**Backwards(\( \alpha \))-completeness**

[Cousots79,GiacobazziJACM00] :

\[
\alpha \circ f = f^\# \circ \alpha
\]

\[
\begin{array}{c}
S \\
\alpha \\
f(S) \\
\end{array} \xrightarrow{f} f(S) \\
\alpha \xrightarrow{\alpha} \bullet \\
\alpha \xrightarrow{f^\#} \alpha \xrightarrow{\alpha}
\]

\( \alpha \) is a homomorphism from \( PC \) to \( A \) — it preserves \( f \) as \( f^\# \).

**Theorem:** \( f^\#(\alpha(S)) \subseteq a \) iff \( f(S) \models a \).

That is, we can decide properties of \( f \) in \( A \).
Often, one wants more than the internal logic

\[ \mathcal{L} \ni \phi ::= a \mid \phi_1 \land \phi_2 \mid \phi_1 \lor \phi_2 \mid [f]\phi, \text{ where } a \in A \]

The interpretation, \([ \cdot ] : \mathcal{L} \to \mathcal{P}(C)\), is defined as

\[
\begin{align*}
[a] &= \gamma(a) \\
[\phi_1 \land \phi_2] &= [\phi_1] \cap [\phi_2] \\
[\phi_1 \lor \phi_2] &= [\phi_1] \cup [\phi_2] \\
[[f]\phi] &= \widehat{pre}_f[\phi]
\end{align*}
\]

where \(\widehat{pre}_f(S) = \{ c \in C \mid f(c) \subseteq S \}\), and \(f : C \to \mathcal{P}(C)\) is a state-transition function.

Define \(S \models \phi\) iff \(S \subseteq [\phi]\). (In the internal logic, \(S \models a\) iff \(S \subseteq \gamma(a)\).)

\(\phi_1 \lor \phi_2\) and \([f]\phi\) might not fall in \(A\)’s internal logic. (E.g., for \(Sign\), there is no \(\bigcup : Sign \times Sign \to Sign\) such that \(\gamma(\text{neg} \bigcup \text{pos}) = [\phi_1 \lor \phi_2]\). And, most \(f^\#_{\text{best}}\) are not \(\gamma\)-complete for \(f\).)

What justifies these extra operators? Can we employ them within a sound static analysis?
Help comes from the logical relations for the types, \( \tau ::= b \mid \tau_1 \to \tau_2 \mid L(\tau) \mid U(\tau) \)

For \( \rho_\tau \subseteq C_\tau \times A_\tau \):

- \( \rho_b \) is given, for base type \( b \) (e.g., \( \rho_{\text{Sign}} \subseteq \text{Int} \times \text{Sign} \))

- \( f \rho_{\tau_1 \to \tau_2} f^\# \) iff for all \( c \in C_{\tau_1} \) and \( a \in A_{\tau_1} \), \( c \rho_{\tau_1} a \) implies \( f(c) \rho_{\tau_2} f^\#(a) \)

- \( S \rho_{L(\tau)} T \) iff for all \( c \in S \in C_{L(\tau)} \), exists \( a \in T \in A_{L(\tau)} \) s.t. \( c \rho_{\tau} a \)

- \( S \rho_{U(\tau)} T \) iff for all \( a \in T \in A_{U(\tau)} \), exists \( c \in S \in C_{U(\tau)} \) s.t. \( c \rho_{\tau} a \)

where

- \( D_b \) is given (e.g., \( \text{Int} \) and \( \text{Sign} \))

- \( D_{\tau_1 \to \tau_2} = D_{\tau_1} \to D_{\tau_2} \), the poset of monotone functions from \( D_{\tau_1} \) to \( D_{\tau_2} \).

- \( D_{L(\tau)} = \mathcal{P}_L(D_\tau) \), a lower powerset of \( D_\tau \), a collection of downclosed subsets of \( D \) that includes all \( \downarrow d \) for all \( d \in D \), partially ordered by \( \subseteq \), and closed under \( \cap \).

- \( D_{U(\tau)} = \mathcal{P}_U(D_\tau) \), an upper powerset of \( D_\tau \), a collection of upclosed subsets of \( D \) that includes all \( \uparrow d \) for all \( d \in D \), partially ordered by \( \supseteq \), and closed under \( \cup \).
**Lower-powerset approximation defines universal, disjunctive properties:** e.g., \( \{ \text{neg, zero, none} \} \) asserts \( \forall (\text{neg} \lor \text{zero}) \) — all data are nonpositive:

**Upper-powerset approximation defines conjunctive, existential properties:** e.g., \( \{ \text{neg, pos, any} \} \) asserts \( \exists \text{neg} \land \exists \text{pos} \) — there exists a negative and a positive datum:
Consequences of the “powerset lift”

♦ When $\phi_i$ are found in $A$’s internal logic, then $\forall (\bigvee_i \phi_i)$ is found in $\mathcal{P}_\downarrow(A)$’s internal logic.

♦ When $\phi_i$ are found in $A$’s internal logic, then $\bigwedge_i (\exists \phi_i)$ is found in $\mathcal{P}_\uparrow(A)$’s internal logic.

More importantly, we use down-closed and up-closed subsets of $A$ to define an external logic where judgements take form,

$$a \in \llbracket \phi \rrbracket^A \subseteq A, \text{ rather than } a \sqsubseteq \phi \in A.$$  

The sets let us define sound underapproximation, where $a \in \llbracket \phi \rrbracket^A$ implies $\gamma(a) \subseteq \llbracket \phi \rrbracket$.

The external logic falls even further outside of $A$’s internal logic because of problems with $f^\# : A_1 \rightarrow A_2$, which we “split” into pre- and post-image, which are rarely $\gamma$-complete....
A programming logic based on logical relations

**Types:** \( \tau ::= b \mid L(\tau) \mid U(\tau) \mid \tau_1 \rightarrow \tau_2 \)

**Assertions:** \( \phi ::= a \mid \bigvee_{i<k} \phi_i \mid \bigwedge_{i<k} \phi_i \mid f(\phi) \mid f; \phi \)

**Judgement typing:**

\[
\begin{align*}
\frac{a : b \quad \phi_i : \tau, \text{ for all } i < k}{\bigvee_{i<k} \phi_i : L(\tau)} & \quad \frac{\phi_i : \tau, \text{ for all } i < k}{\bigwedge_{i<k} \phi_i : U(\tau)} \\
f : \tau_1 \rightarrow \tau_2 \quad \phi : \tau_1 & \quad f : \tau_1 \rightarrow \tau_2 \quad \phi : \tau_2 \\
\frac{f(\phi) : \tau_2}{f; \phi : \tau_1}
\end{align*}
\]

**Concrete judgements:** have form, \( c \models_\tau \phi \), where \( c \in C_\tau \) and \( \phi : \tau \)

- \( c \models_b a \) is given by \( \rho_b \subseteq C_b \times A_b \), e.g., \( n \models_{Sign} a \) if \( n \rho_{Sign} a \)
- \( S \models_{L(\tau)} \bigvee_{i<k} \phi_i \), if for all \( c \in S \), there exists \( j < k \) such that \( c \models_\tau \phi_j \)
- \( S \models_{U(\tau)} \bigwedge_{i<k} \phi_i \), if for all \( i < k \), there exists \( c \in S \) such that \( c \models_\tau \phi_i \)
- \( c \models_{\tau_1} f; \phi \), if \( f(c) \models_{\tau_2} \phi \), for \( f \in C_{\tau_1} \rightarrow C_{\tau_2} \) (this defines \( c \in \text{pre}_f(\phi) \))
- \( c \models_{\tau_2} f(\phi) \), if there exists \( c' \in C_{\tau_1} \) such that \( c' \models_{\tau_1} \phi \) and \( f(c') = c \), for \( f \in C_{\tau_1} \rightarrow C_{\tau_2} \) (this defines \( c \in \text{post}_f(\phi) \))
The corresponding external logic for abstract domains

**Abstract judgements** have form, $a \models^A_\tau \phi$,

where $a \in A_\tau$ and $\phi : \tau$. (Read $a \models^A_\tau \phi$ as $a \in [\phi]^A_\tau$.)

$a \models^A_b a'$, if $a \sqsubseteq_b a'$, for $a, a' \in A_b$ (e.g., $pos \sqsubseteq Sign any$)

$T \models^A_{L(\tau)} \bigvee_{i < k} \phi_i$, if for all $a \in T$, there exists $j < k$ such that $a \models^A_\tau \phi_j$

$T \models^A_{U(\tau)} \bigwedge_{i < k} \phi_i$, if for all $i < k$, there exists $a \in T$ such that $a \models^A_\tau \phi_i$

$a \models^A_{\tau_1} f; \phi$, if $f^#(a) \models^A_{\tau_2} \phi$, where $f \rho_{\tau_1 \rightarrow L(\tau_2)} f^#$

(this underapproximates $\widetilde{pre}_f(\phi)$)

$a \models^A_{\tau_2} f(\phi)$, if there exists $a' \in A_{\tau_1}$ such that $a' \models^A_{\tau_1} \phi$

and $a' \in f^b(a)$, where $f^{-1} \rho_{\tau_1 \rightarrow U(\tau_2)} f^b$

(this underapproximates $pre_{f^{-1}}(\phi) = post_f(\phi)$)
Consequences

1. **Soundness**: $S \rho_\tau a$ and $a \models^A_\tau \phi$ imply $S \models_{\tau} \phi$.

2. **Completeness I**: When a logical operator, $f^\#$, is $\gamma$-complete for $f$, then the judgement form, $\cdot \models^A_{\tau} f^\#(\phi)$, is complete (falls in the internal logic) for $\cdot \models_{\tau} f(\phi)$.

3. **Completeness II**: When $f^\#$ is $\alpha$-complete for $f$, then $\cdot \models^A_{\tau} f^\#; \phi$ is complete for $\cdot \models_{\tau} f; \phi$

4. We can formally justify branching-time logics (e.g., Hennessy-Milner logic) as sound, best approximating, and complete for static analysis ([abstract model checking](Dams97)).


4. R. Giacobazzi and E. Quintarelli. Incompleteness, counterexamples, and refinements in abstract model checking. SAS’01, LNCS 2126.

