Static Program Analysis

Foundations of Abstract Interpretation

Sebastian Hack, Christian Hammer, Jan Reineke

Advanced Lecture, Winter 2014/15

Recap: Abstract Interpretation

- Semantics-based approach to program analysis
- Framework to develop provably correct and terminating analyses

Ingredients:

- Concrete semantics: Formalizes meaning of a program ✓
- Abstract semantics (✓)
- Both semantics defined as fixpoints of monotone functions over some domain ✓
- Relation between the two semantics establishing correctness (✓)

Abstract Semantics

Similar to concrete semantics:

- A complete lattice ($L^{\#}, \leq$) as the domain for abstract elements
- A monotone function $F^{\#}$ corresponding to the concrete function F
- Then the abstract semantics is the least fixed point of $F^{\#}$, lfp $F^{\#}$

If $F^{\#}$ "correctly approximates" F,

then lfp $F^{\#}$ "correctly approximates" lfp F.

Fixpoint Transfer Theorem

Let $(L, \leq)$ and $(L^{\#}, \leq^{\#})$ be two complete lattices, $\gamma : L^{\#} \to L$ a monotone function, and $F : L \to L$ and $F^{\#} : L^{\#} \to L^{\#}$ two monotone functions, with

$$\boxed{\forall l^{\#} \in L^{\#} : \gamma(F^{\#}(l^{\#})) \geq F(\gamma(l^{\#})).}$$

Local Correctness

Then:

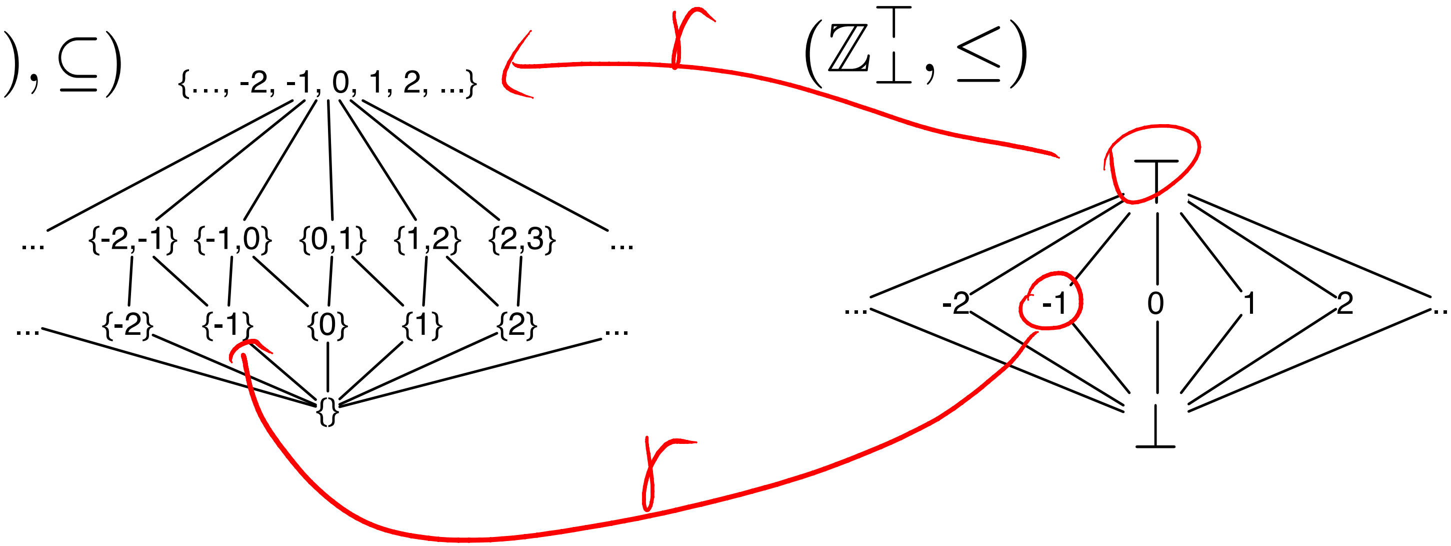
$$\boxed{\mathit{lfp}\ F \leq \gamma(\mathit{lfp}\ F^{\#}).}$$

Global Correctness

$$\begin{tikzcd}
l^{\#} \arrow[r, "F^{\#}"] \arrow[d, "\gamma"'] & F^{\#}(l^{\#}) \arrow[d, "\gamma"] \\
\gamma(l^{\#}) \arrow[r, "F"] & F(\gamma(l^{\#}))
\end{tikzcd}$$

An Example Abstract Domain for Values of Variables

$(\mathcal{P}(\mathbb{Z}), \subseteq)$ $(\mathbb{Z}_{\perp}^{\top}, \leq)$



How to relate the two?

➔ *Concretization function, specifying "meaning" of abstract values.*

$$\gamma : \mathbb{Z}_{\perp}^{\top} \to \mathcal{P}(\mathbb{Z})$$

➔ *Abstraction function: determines best representation concrete values.*

$$\alpha : \mathcal{P}(\mathbb{Z}) \to \mathbb{Z}_{\perp}^{\top}$$

Relation between the Abstract and Concrete Domains

$$\gamma(\top) := \mathbb{Z}$$
$$\gamma(\bot) := \emptyset$$
$$\gamma(x) := \{x\}$$

$$\alpha(A) := \begin{cases} \top & : |A| \geq 2 \\ x & : A = \{x\} \\ \bot & : A = \emptyset \end{cases}$$

1. *Are these functions monotone?*
2. *Should they be?*
3. *What is the meaning of the partial order in the abstract domain?*
4. *What if we first abstract and the concretize?*

$\gamma(\alpha(x)) \geq x$

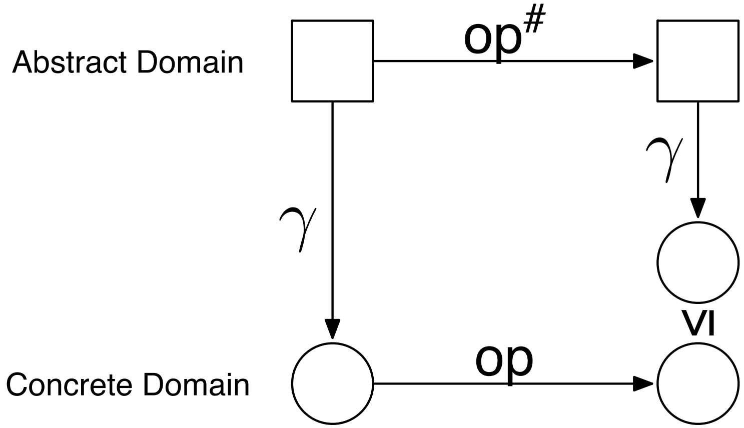
How to Compute in the Abstract Domain
Example: Multiplication on Flat Lattice

Denotes abstract version of operator

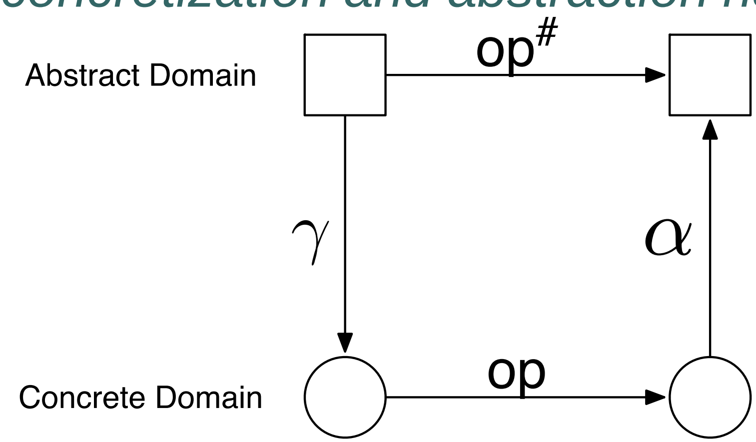
$*^{\#}$	$\top$	a	0	$\bot$
$\top$				
b				
0				
$\bot$				

How to Compute in the Abstract Domain: Correctness Conditions

Correctness Condition:



Correct by construction
(if concretization and abstraction have certain properties):



How to Compute in the Abstract Domain
Example: Multiplication on Flat Lattice

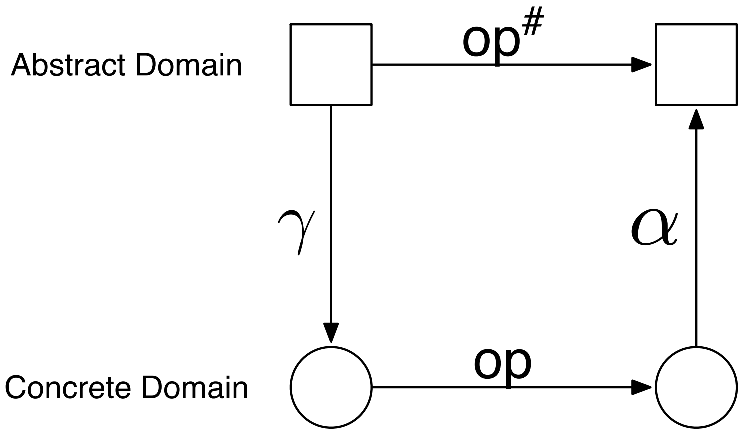
$$
\begin{tikzcd}
a \quad b \arrow[r, "*^{\#}"] \arrow[d, "\gamma"'] & a*b \\
\{a\} \quad \{b\} \arrow[r, "*"] & \{a * b\} \arrow[u, "\alpha"']
\end{tikzcd}
$$

How to Compute in the Abstract Domain
Example: Multiplication on Flat Lattice

$$
\begin{tikzcd}
\top \arrow[d, "\gamma"'] & 0 \arrow[d, "\gamma"'] \arrow[rr, "*^{\#}"] & & 0 \\
\mathbb{Z} & \{0\} \arrow[rr, "*"] & & \{0\} \arrow[u, "\alpha"]
\end{tikzcd}
$$

How to Compute in the Abstract Domain: Correct by Construction

Correct by construction
(if concretization and abstraction have certain properties):



"Certain properties": Notion of Galois connection:

Let $(L, \leq)$ *and* $(M, \sqsubseteq)$ *be partially ordered sets and* $\alpha \in L \to M$, $\gamma \in M \to L$. *We call* $(L, \leq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (M, \sqsubseteq)$ *a* Galois connection *if* α *and* γ *are monotone functions and*

$$\begin{array}{rcl} l & \leq & \gamma(\alpha(l)) \\ \alpha(\gamma(m)) & \sqsubseteq & m \end{array}$$

for all $l \in L$ *and* $m \in M$.

Galois connections

Notion of Galois connections:

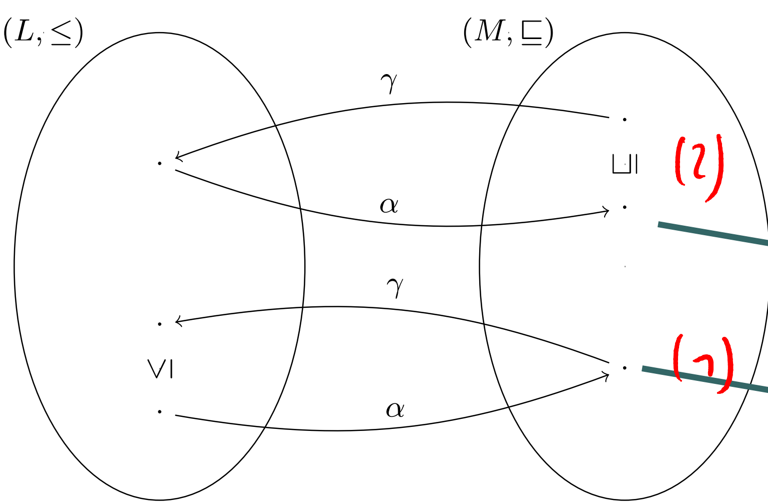
Let $(L, \leq)$ *and* $(M, \sqsubseteq)$ *be partially ordered sets and* $\alpha \in L \to M$, $\gamma \in M \to L$. *We call* $(L, \leq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (M, \sqsubseteq)$ *a* Galois connection *if* α *and* γ *are monotone functions and*

$$\begin{array}{rcl} l & \leq & \gamma(\alpha(l)) \qquad (1) \\ \alpha(\gamma(m)) & \sqsubseteq & m \qquad (2) \end{array}$$

for all $l \in L$ *and* $m \in M$.

Why monotone?

Graphically:



For precision.

For soundness.

Galois connections: Example

$$(\mathcal{P}(\mathbb{Z}), \subseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (\mathbb{Z}_\perp^\top, \leq)$$

with:

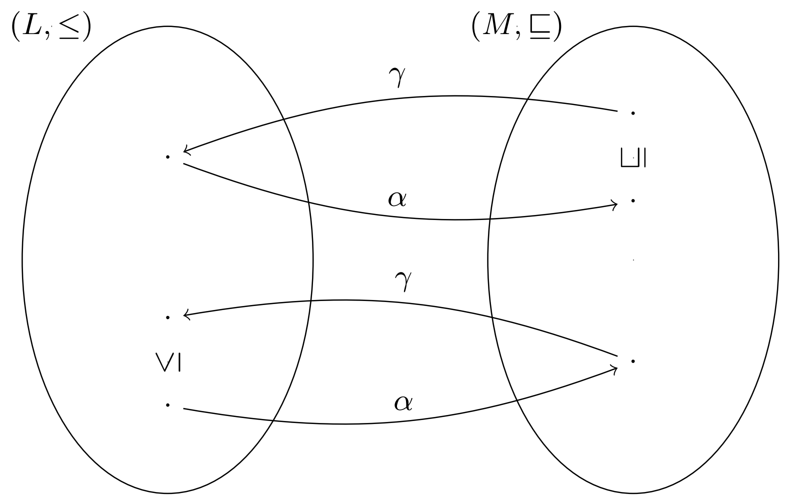
$$\alpha : \mathcal{P}(\mathbb{Z}) \to \mathbb{Z}_\perp^\top \qquad \gamma : \mathbb{Z}_\perp^\top \to \mathcal{P}(\mathbb{Z})$$

$$\alpha(A) := \begin{cases} \top & : |A| \geq 2 \\ x & : A = \{x\} \\ \perp & : A = \emptyset \end{cases} \qquad \begin{aligned} \gamma(\top) &:= \mathbb{Z} \\ \gamma(\perp) &:= \emptyset \\ \gamma(x) &:= \{x\} \end{aligned}$$

GALOIS INSERTION $\quad \alpha(\gamma(x)) = x \quad \forall x$

Galois connections: Properties

Graphically:



Properties:

1) *Can be used to systematically construct correct (and in fact the ~~most precise~~) abstract operations:* $op^{\#} = \alpha \circ op \circ \gamma$

2) *a) Abstraction function induces concretization function*

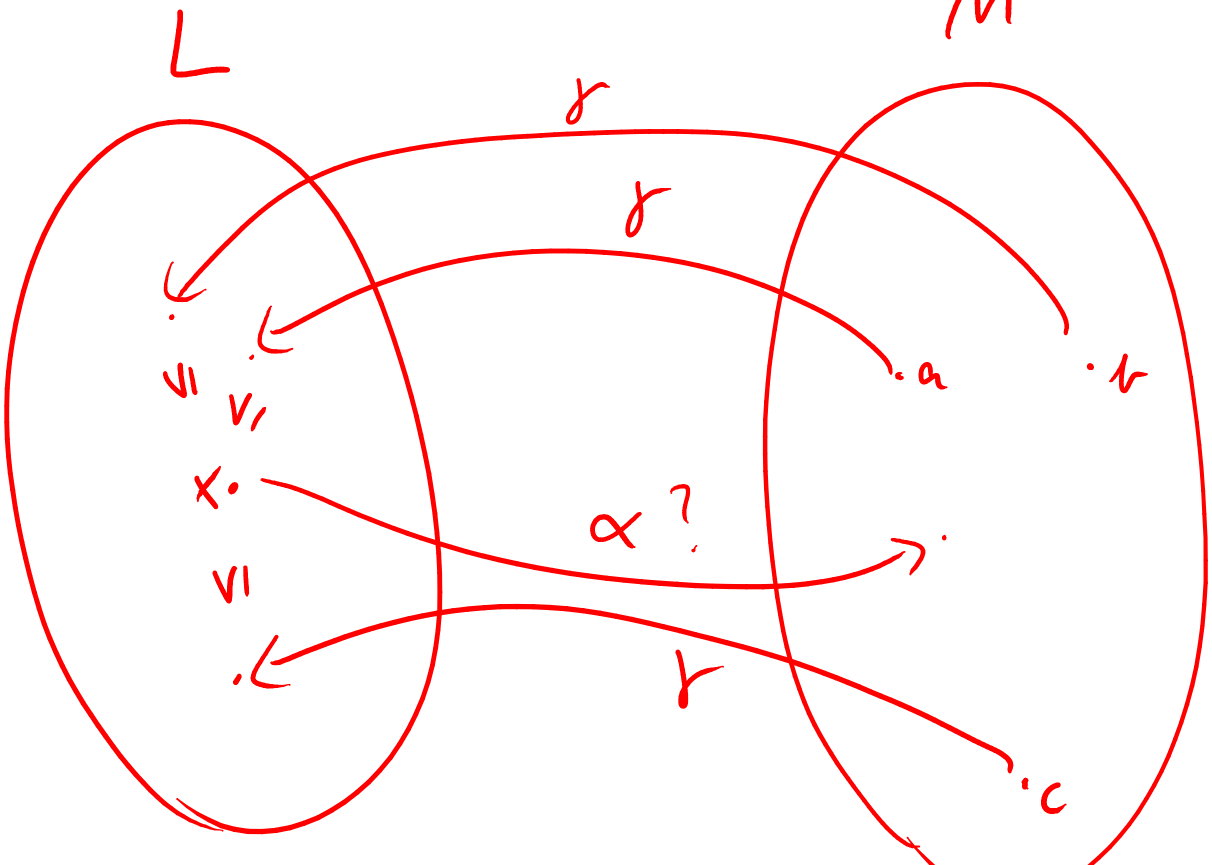
b) Concretization function induces abstraction function

Why?

How?

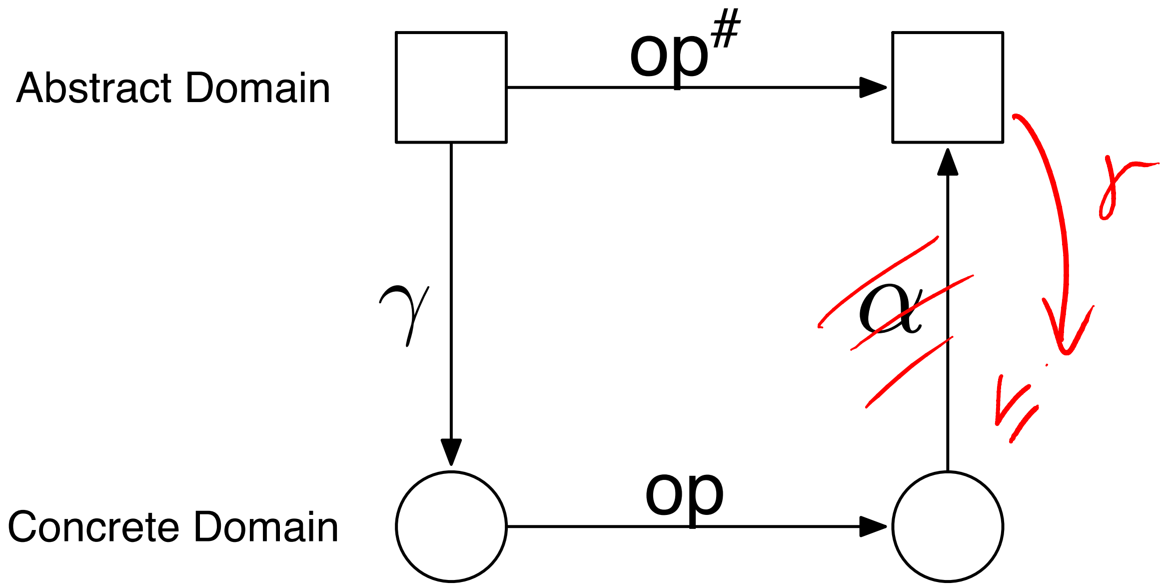
How do abstraction and concretization induce each other?

$$\gamma(a) := \bigsqcup_M \{ c \mid \alpha(c) \leq a \}$$



$$\alpha(x) := \bigsqcap \{ a \mid x \leq \gamma(a) \}$$

Why is $\alpha \circ op \circ \gamma$ a correct abstract operation?



Why is $\alpha \circ op \circ \gamma$ the **best** correct abstract transformer?

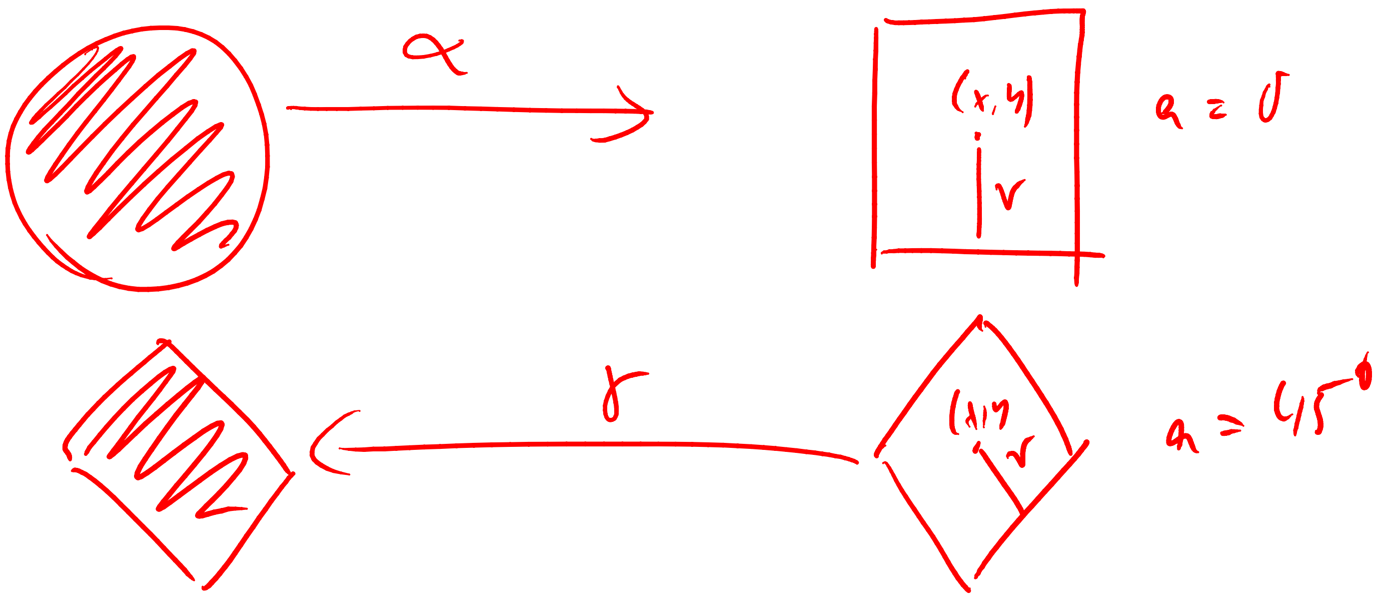
Could there not be multiple incomparable transformers?

Think of an abstraction that does not admit a Galois connection!

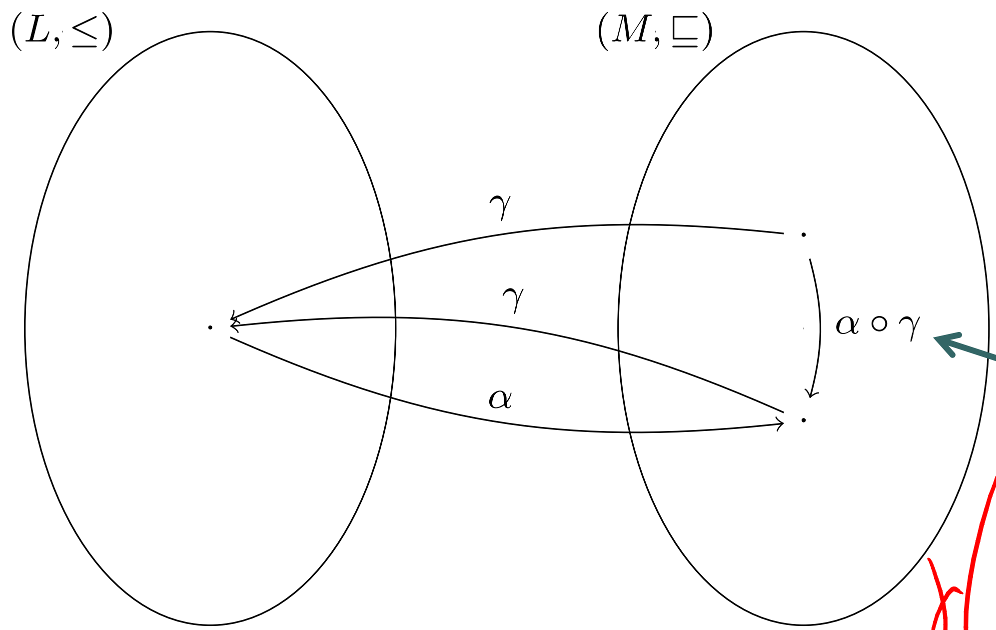
$(\mathcal{P}(\mathbb{Q} \times \mathbb{Q}), \subseteq)$

(x, y, r, a)

$\mathbb{Q} \times \mathbb{Q} \times \mathbb{Q} \times \mathbb{R}$



Semantic Reduction and Galois Insertions



Semantic Reduction

$V \to SIGN$

$\{+, -, 0\}$

$SIGN =$

$\gamma\left(\left\{\begin{array}{l} x \mapsto \emptyset, \\ y \mapsto \{+\} \end{array}\right\}\right) = \emptyset$

- "Improves" abstract value without affecting meaning.
- A *Galois Connection* is a *Galois Insertion* if $\alpha \circ \gamma = id$.
- Where might this occur?

$\alpha\left(\gamma\left(\left\{\begin{array}{l} x \mapsto \emptyset, \\ y \mapsto \{+\} \end{array}\right\}\right)\right) = \begin{array}{l} x \mapsto \emptyset \\ y \mapsto \emptyset \end{array}$

Abstracting Sets of Concrete States
Recap: Concrete States

Concrete states are not just sets of values...

Concrete states consist of variables and memory:

$$s = (\rho, \mu) \in States$$

$\rho : Vars \to int$ ← Values of Variables

$\mu : \mathbb{N} \to int$ ← Contents of Memory

$$\cancel{States = (Vars \to int) \times (\mathbb{N} \to int)}$$

$$States = (Vars \to \mathbb{Z}) \times (\mathbb{N} \to \mathbb{Z})$$

Abstracting Sets of Concrete States
Recap: Concrete States

Reachability semantics is defined on sets of states:

$$[\![statement]\!] \subseteq States \times States$$

$$[\![statement]\!] : \mathcal{P}(States) \to \mathcal{P}(States)$$

$$[\![statement]\!](S) := \{s' \mid \exists s \in S : (s, s') \in [\![statement]\!]\}$$

$$\mathcal{P}(States) = \mathcal{P}((Vars \to \mathbb{Z}) \times (\mathbb{N} \to \mathbb{Z}))$$

Relation between Concrete Domain and Abstract Domain

Concrete domain!

$$\mathcal{P}(States) = \mathcal{P}((Vars \to \mathbb{Z}) \times (\mathbb{N} \to \mathbb{Z}))$$

$$\{\{x \mapsto 3, y \mapsto 3\}, \{x \mapsto 4, y \mapsto 2\}\}$$

Abstract domain?

$$\widehat{States} = Vars \to \mathbb{Z}_{\perp}^{\top}$$

Relation between the two?

→ For ease of understanding, introduce Intermediate domain:

$$\widehat{PowerSetStates} = Vars \to \mathcal{P}(\mathbb{Z})$$

Relation between Concrete Domain and Intermediate Domain

Concrete domain:

$$\mathcal{P}(States) = \mathcal{P}((Vars \to \mathbb{Z}) \times (\mathbb{N} \to \mathbb{Z}))$$

Intermediate domain:

$$\widehat{PowerSetStates} = Vars \to \mathcal{P}(\mathbb{Z})$$

Abstraction:

$$\alpha_{C,I} : \mathcal{P}(States) \to \widehat{PowerSetStates}$$

$$\alpha_{C,I}(C) := \lambda x \in Vars.\{v(x) \in \mathbb{Z} \mid (v,m) \in C\}$$

Concretization:

$$\gamma_{I,C} : \widehat{PowerSetStates} \to \mathcal{P}(States)$$

$$\gamma_{I,C}(\widehat{c}) := \{(v,m) \in States \mid \forall x \in Vars : v(x) \in \widehat{c}(x)\}$$

Relation between Intermediate Domain and Abstract Domain

Intermediate domain:

$$\widehat{PowerSetStates} = Vars \rightarrow \mathcal{P}(\mathbb{Z})$$

Abstract domain:

$$\widehat{States} = Vars \rightarrow \mathbb{Z}_{\bot}^{\top}$$

Abstraction:

$$\alpha_{I,A} : \widehat{PowerSetStates} \rightarrow \widehat{States}$$

$$\alpha(\widehat{c}) := \lambda x \in Vars.\alpha(c(x))$$

Concretization:

$$\gamma_{A,I} : \widehat{States} \rightarrow \widehat{PowerSetStates}$$

$$\gamma(\widehat{a}) := \lambda x \in Vars.\gamma(\widehat{a}(x))$$

Abstraction and Concretization functions from before!

Could plug in other abstractions for sets of values…

Relation between Concrete Domain and Abstract Domain

Concrete domain:

$$\mathcal{P}(States) = \mathcal{P}((Vars \to \mathbb{Z}) \times (\mathbb{N} \to \mathbb{Z}))$$

Abstract domain:

$$\widehat{States} = Vars \to \mathbb{Z}_{\perp}^{\top}$$

Abstraction:

$$\alpha_{C,A} : \mathcal{P}(States) \to \widehat{States}$$

$$\alpha_{C,A} := \alpha_{I,A} \circ \alpha_{C,I}$$

Concretization:

$$\gamma_{A,C} : \widehat{States} \to \mathcal{P}(States)$$

$$\gamma_{A,C} := \gamma_{I,C} \circ \gamma_{A,I}$$

Galois connections can be composed to obtain new Galois connections.

$$C \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} I \underset{\gamma'}{\overset{\alpha'}{\rightleftarrows}} A$$

Meaning of Statements in the Abstract Domain

$$[\![R = e]\!]^\#(\widehat{a}) := \widehat{a}[R \mapsto [\![e]\!]^\#(\widehat{a})]$$

$$[\![R = M[e]]\!]^\#(\widehat{a}) := \widehat{a}[R \mapsto \top] \quad \checkmark$$

$$[\![M[e_1] = e_2]\!]^\#(\widehat{a}) := \widehat{a} \quad \checkmark$$

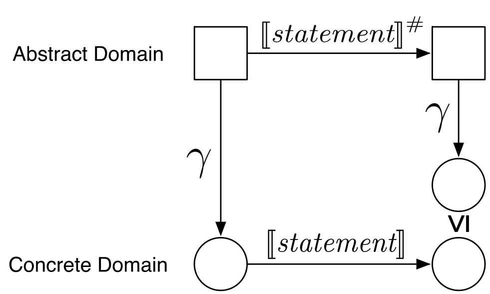
$$[\![Pos(e)]\!]^\#(\widehat{a}) := \widehat{a}$$

$$[\![Neg(e)]\!]^\#(\widehat{a}) := \widehat{a}$$

Can this be done better?

Again:

For Correctness:



For the best possible precision:

Meaning of Expressions

Evaluation of expressions is as expected:

$$[\![x]\!]^{\#}(\widehat{a}) := \widehat{a}(x) \qquad \qquad \textit{if } x \in \mathit{Vars}$$

$$[\![e_1 \ \ op \ e_2]\!]^{\#}(\widehat{a}) := [\![e_1]\!]^{\#}(\widehat{a}) \ op^{\#} \ [\![e_2]\!]^{\#}(\widehat{a})$$

As we have seen earlier!

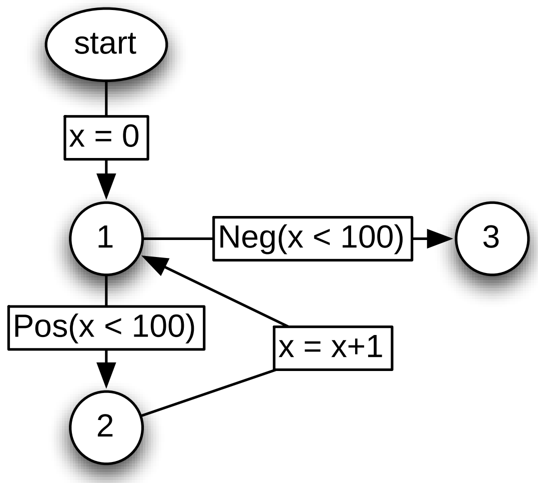
Putting it all together: The Abstract Reachability Semantics

Abstract Reachability Semantics captured as least fixed point of:

$$\widehat{Reach} : V \to \widehat{States}$$

$$\widehat{Reach}(start) = \top$$

$$\forall v' \in V \setminus \{start\} : \widehat{Reach}(v') = \bigsqcup_{v \in V, (v,v') \in E} [\![labeling(v, v')]\!]^{\#}(\widehat{Reach}(v))$$



$$\widehat{Reach}(1) = [\![labeling(start, 1)]\!]^{\#}(\widehat{Reach}(start)) \sqcup [\![labeling(2, 1)]\!](\widehat{Reach}(2))$$

$$\widehat{Reach}(2) = [\![labeling(1, 2)]\!]^{\#}(\widehat{Reach}(1))$$

$$\widehat{Reach}(3) = [\![labeling(1, 3)]\!]^{\#}(\widehat{Reach}(1))$$

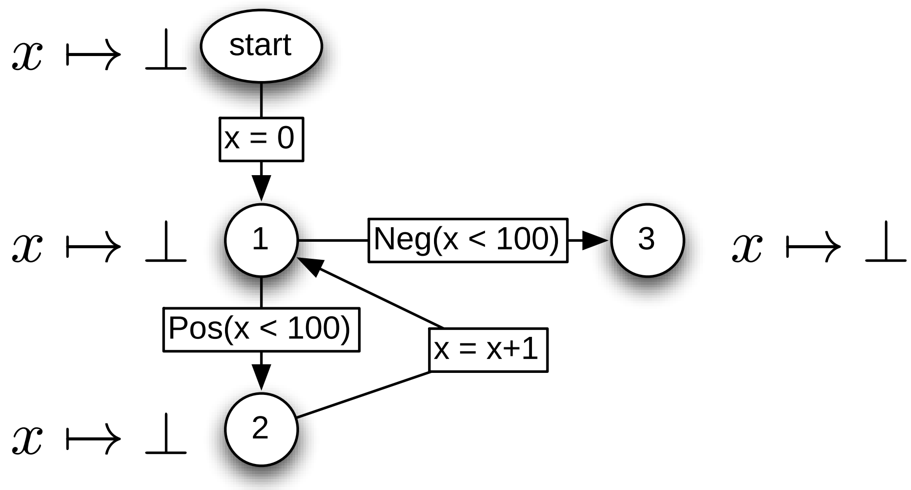
$$\Downarrow$$

$$\widehat{Reach}(1) = [\![x = 0]\!]^{\#}(\widehat{Reach}(start)) \sqcup [\![x = x + 1]\!]^{\#}(\widehat{Reach}(2))$$

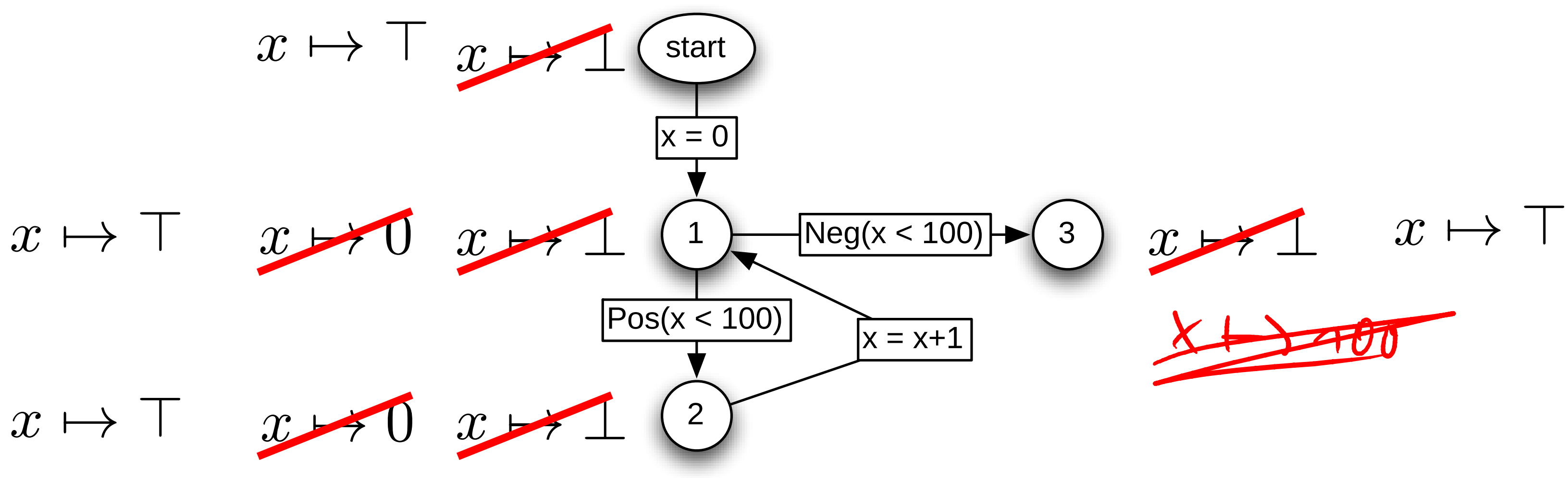
$$\widehat{Reach}(2) = [\![Pos(x < 100)]\!]^{\#}(\widehat{Reach}(1))$$

$$\widehat{Reach}(3) = [\![Neg(x < 100)]\!]^{\#}(\widehat{Reach}(1))$$

Example: Kleene Iteration to Compute Abstract Reachability Semantics

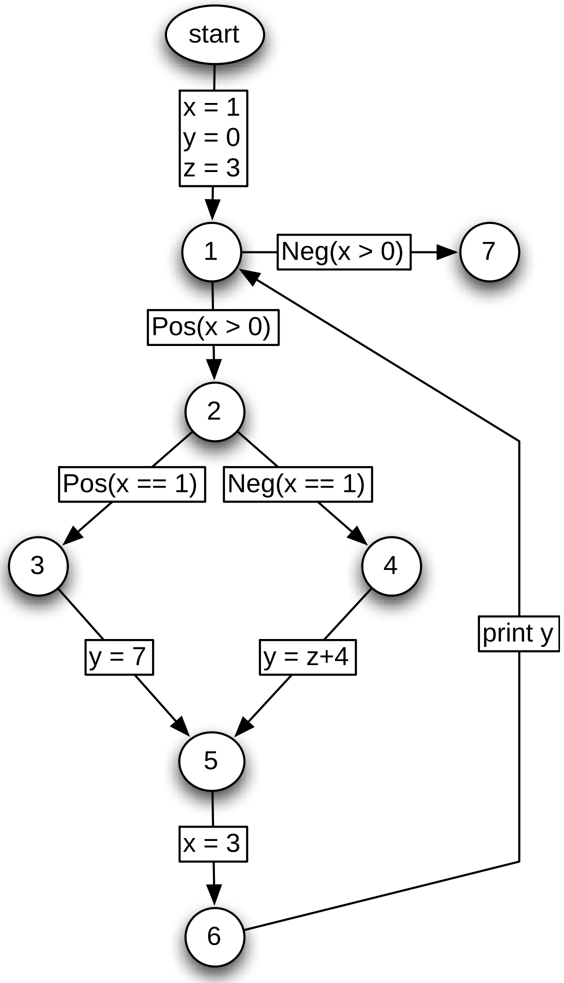


Example: Kleene Iteration to Compute Abstract Reachability Semantics



Example II: Kleene Iteration to Compute Abstract Reachability Semantics

```
y = 0;
x = 1;
z = 3;
while (x > 0) {
    if (x == 1) {
        y = 7;
    }
    else {
        y = z+4;
    }
    x = 3;
    print y;
}
```



Next: Other Numerical Abstractions

- Signs
- Parity ∈ EVEN OR ODD ?
- Intervals
- Octagons / POLYHEDRA
- Congruence x MOD 2

3