CIS 670: Program Analysis

Title: Abstract Interpretation.

Guest Lecturer: Sriram Sankaranarayanan.

NEC Labs America,

Princeton, NJ.

sriram@nec-labs.com

Date: Oct 3rd, 2007.

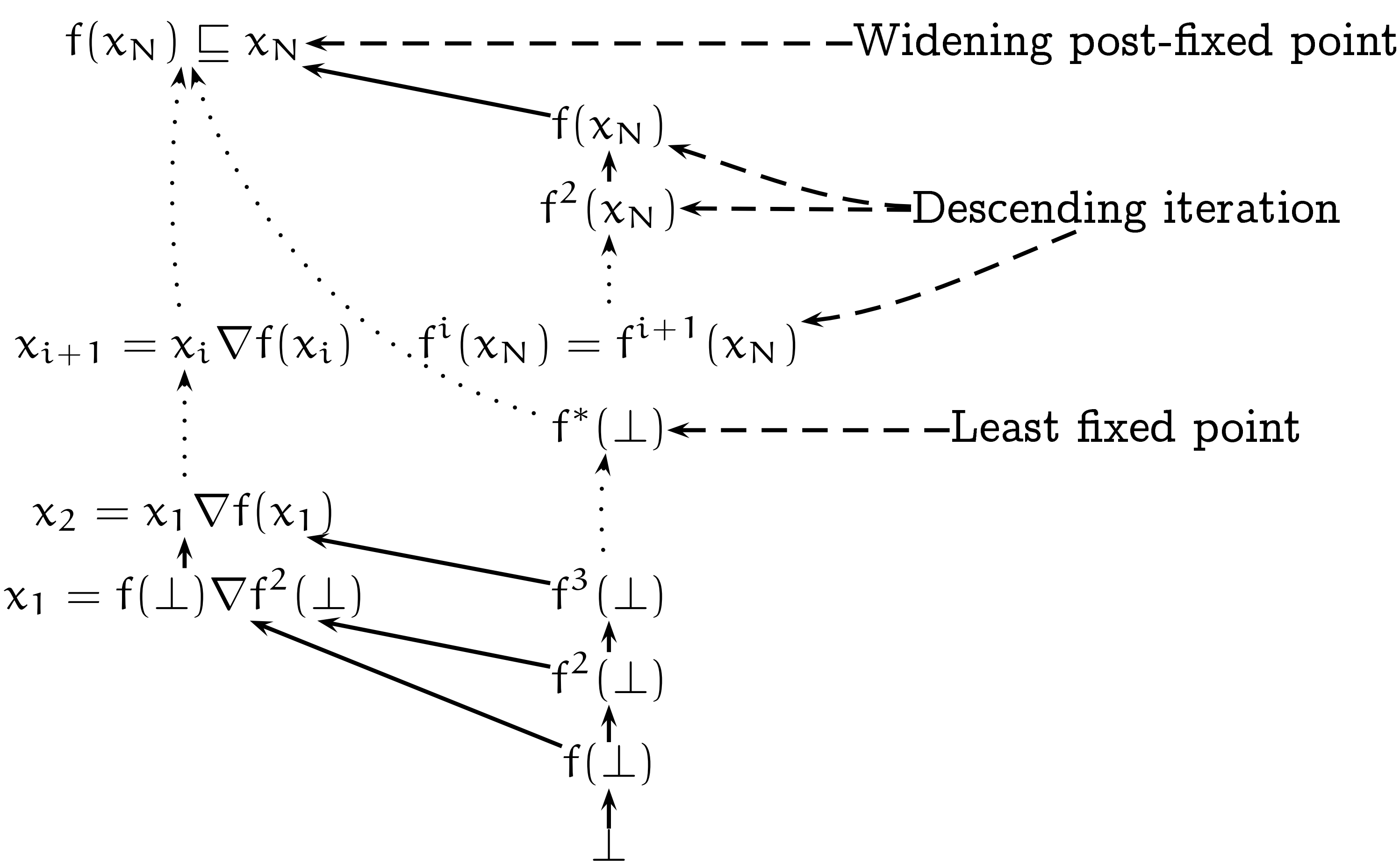
The story so far...

- Signs and Interval analyses: Lattice Inequalities.
- Iteration strategy for solving lattice inequalities.

$$x_0 = f(\bot),\ x_1 = f(x_0),\ \cdots$$

- The iteration converges if the lattice is finite.
- If the lattice is not finite, then iteration may diverge.
- We used widening to force convergence.
- Widening reaches a postfixed point

Ascending/Descending Iterations



Descending Iteration: Convergence

Descending Chain Condition: Dual to Ascending Chain condition.

Descending iteration need not necessarily converge in finitely many steps.

(1) Stop the iteration after some fixed number of steps.
This is not a good idea (for large programs).

(2) Use a "narrowing" operator to force convergence.

Narrowing

Let $b \sqsubset a$, then $a \triangle b$ is intermediate to a, b.

$$b \sqsubseteq a \triangle b \sqsubseteq a\,.$$

Let $a_1 \sqsupset a_2 \sqsupset a_3 \sqsupset \ldots$ be an infinite decreasing iteration.

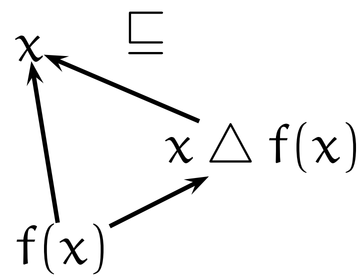
Narrowed iteration: Define sequence $b_1, b_2, \ldots,$:

$$b_1 = a_1, \;\; b_{i+1} = b_i \triangle (a_{i+1})\,.$$

(1) $b_1 \sqsupseteq b_2 \sqsupseteq \cdots \sqsupseteq b_N = b_{N+1}$ for $N > 0$.

(2) $\min_{\sqsubseteq}\{a_1, a_2, \ldots,\} \sqsubseteq b_N$.

Illustration:



Property:

$$\text{If } f(x) \sqsubseteq x \text{ then } f(x \bigtriangleup f(x)) \sqsubseteq x \bigtriangleup f(x)\,.$$

Therefore, result of narrowing is still part of the decreasing iteration.

Interval Narrowing

Let $[c, d] \sqsubseteq [a, b]$. Then $[a, b] \bigtriangleup [c, d] = [\ell, u]$.

$$\ell = \begin{cases} c & a = -\infty \\ a & \text{otherwise} \end{cases}$$

$$u = \begin{cases} d & b = \infty \\ b & \text{otherwise} \end{cases}$$

Special case: $x \bigtriangleup \bot = \bot$.

Interval Narrowing: Examples

$$\begin{aligned}
[1,1] \bigtriangleup \bot &= \bot \\
[-1,\infty) \bigtriangleup [1,10] &= [-1,10] \\
[-1,\infty) \bigtriangleup [5,\infty) &= [-1,\infty) \\
[-\infty,\infty] \bigtriangleup [0,10] &= [0,10]
\end{aligned}$$

Updated picture with Widening/Narrowing sequence

Delayed Widening

In order to improve precision:

- First apply $k > 0$ regular iterations,

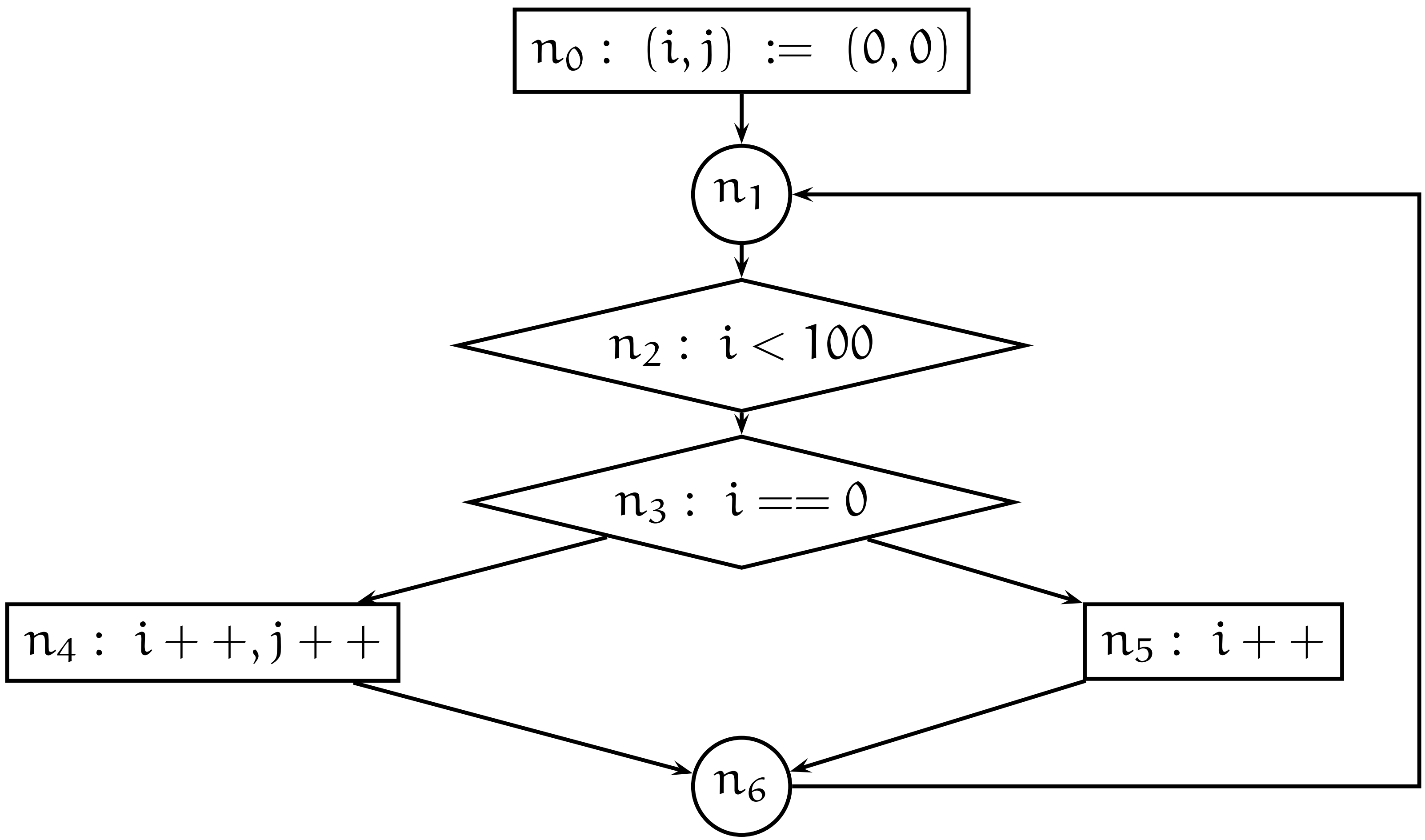
$$x^0 = \bot, \; x^{i+1} = f(x^i), \; \text{if } i < k \, .$$

- Then apply widening iteration until post fixed point.

$$x^{i+1} = x^i \nabla f(x^i) \, .$$

- Similarly narrowing iteration can be delayed.

Example: Delayed Widening



With no delay in widening, we compute the fixed point at n_1:

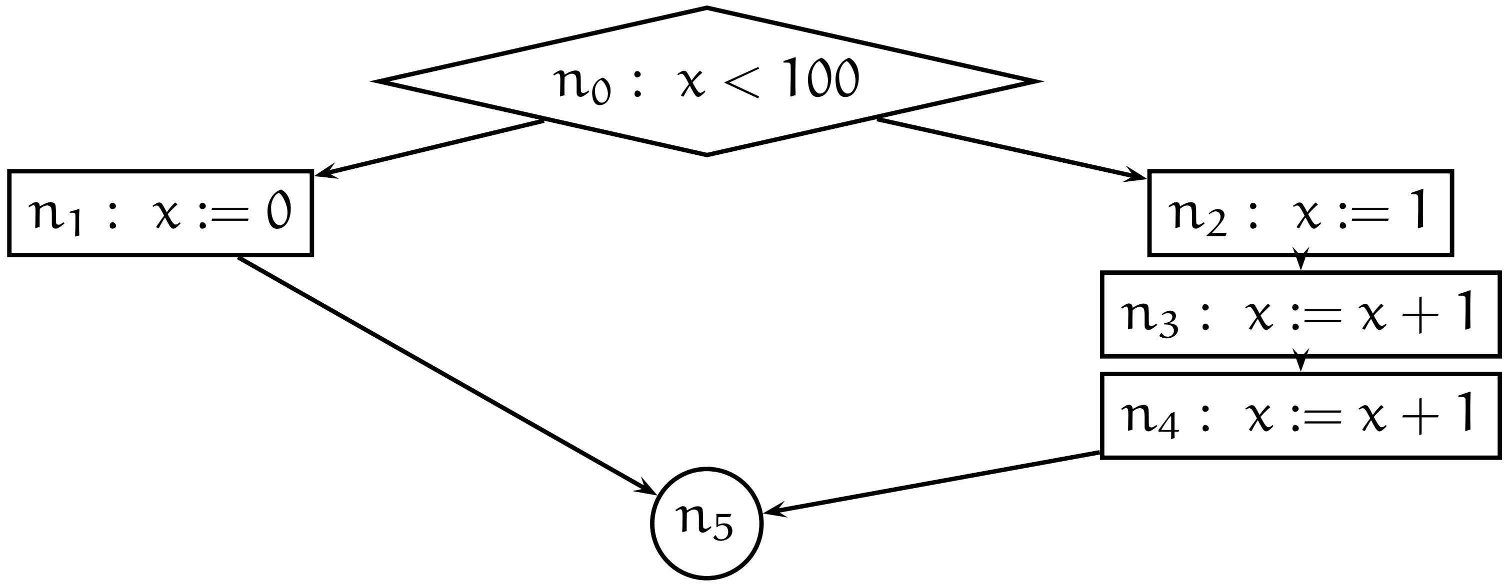
$$i \in [0, 100] \text{ and } j \in [0, \infty) .$$

With delay in widening (~ 5) step delay, we can compute:

$$i \in [0, 100] \text{ and } j \in [0, 1] .$$

Where to widen?

Our current approach says widen everywhere.



Question: With delayless widening, what is the solution computed at n_5?

Widening strategy

- Widening needs to be applied when there are loops in the code.
- Widening needs to be applied only at the loop heads:

$$x_j^{i+1} = \begin{cases} f(x_j^i) & \text{if } n_j \text{ not a loop head} \\ x_j^i \nabla f(x_j^i) & \text{if } n_j \text{ is the head of a loop} \end{cases}$$

- Similarly, we need to narrow only at the heads of loops.

Widening Upto Operator

- Current widening goes from finite to infinity in one step:

$$[0,0]\nabla[0,1] = [0,\infty),\ [0,1]\nabla[-1,1] = (-\infty,1]\,.$$

- Upto set: A set of integer points. Eg.,
  $U = \{-1, 0, 1, 100, 200, 1000\}$.

- Widening upto operator ∇_U: choose the smallest bound from the upto set to replace (if no bound exists, use $\pm\infty$).

- Eg., $[-1,5]\nabla_U[-1,6] = [-1,100]$, $[1,10]\nabla_U[0,10] = [-1,10]$, $\cdots$.

The Big Picture

- Signs Analysis: Compute a sign for every variable.
- Interval Analysis: Compute an interval for every variable.
- Are these analyses sound? What does soundness mean?

Collecting Semantics

a.k.a "Concrete Interpretation".

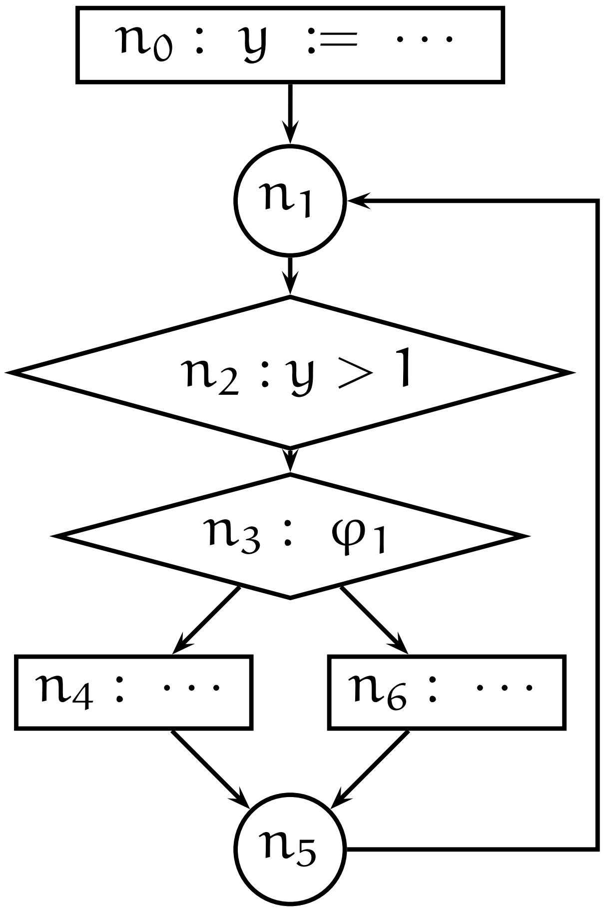
State: A program state is an assignment of integer values to variables.

$$s: \langle x_1 : v_1,\ x_2 : v_2,\ \ldots,\ x_n : v_n \rangle \ .$$

Let $\Sigma : Z \times Z \times \cdots \times Z$ be the set of all program states.

Reachable states: Let $\mathsf{Reach}(n) \subseteq \Sigma$ be the set of all states reaching a location n.

Concrete Interpretation



$$\mathsf{post}(n_0, \mathsf{Reach}(n_0)) \subseteq \mathsf{Reach}(n_1)$$

$$\mathsf{Reach}(n_5) \subseteq \mathsf{Reach}(n_1)$$

$$\mathsf{Reach}(n_1) \subseteq \mathsf{Reach}(n_2)$$

$$\mathsf{Reach}(n_2) \cap [\![y > 1]\!] \subseteq \mathsf{Reach}(n_3)$$

$$\mathsf{Reach}(n_3) \cap [\![\varphi_1]\!] \subseteq \mathsf{Reach}(n_4)$$

$$\mathsf{post}(n_4, \mathsf{Reach}(n_4)) \subseteq \mathsf{Reach}(n_5)$$

$$\mathsf{post}(n_6, \mathsf{Reach}(n_6)) \subseteq \mathsf{Reach}(n_5)$$

Reachable States

- The concrete lattice is $\mathcal{C} : 2^\Sigma$ ordered by $\subseteq$.
- Reachable states can be expressed as a fix point of a monotonic function over sets of states.

$$\mathsf{Reach}(\cdot) : \{\mathsf{F}(\emptyset) \cup \mathsf{F}^2(\emptyset) \cup \cdots \cup \mathsf{F}^n(\emptyset)\}\,.$$

- This is however, a purely theoretical exercise.
  - The lattice of state sets 2^Σ has infinite height.
  - Arbitrary infinite sets cannot be represented inside a computer.

Galois Connections

Galois Connection

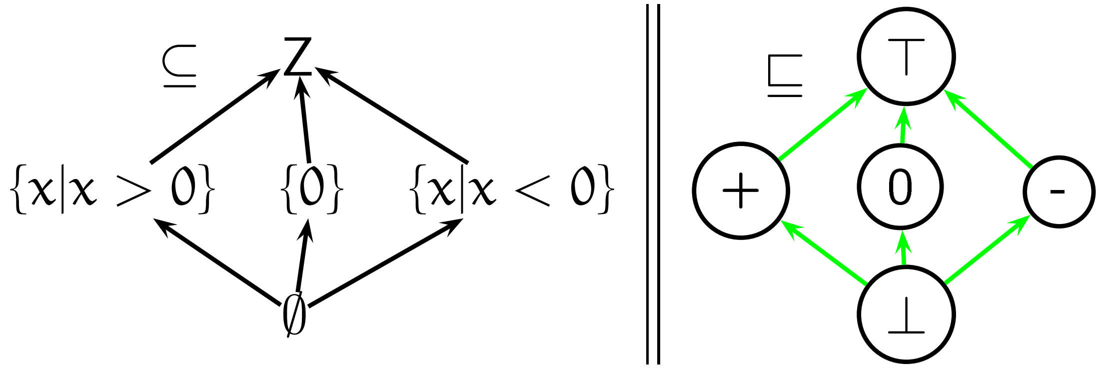
Consider two lattices $\langle C, \subseteq \rangle$ and $\langle A, \sqsubseteq \rangle$.

A <u>Galois Connection</u> between C and A is a pair of functions $\alpha : C \mapsto A$ and $\gamma : A \mapsto C$, such that

$$\text{for all } S \in C \text{ and } a \in A, \alpha(S) \sqsubseteq a \text{ iff } S \subseteq \gamma(a)\,.$$

α is called the "Abstraction Map" and γ is called the "Concretization Map".

Example #1: Signs Lattice



$$\alpha(I) = \begin{cases} \text{“}\bot\text{''}, \text{ if } I = \emptyset \\ \text{“}+\text{''}, \text{ if } I \subseteq \mathsf{Pos} \\ \text{“}-\text{''}, \text{ if } I \subseteq \mathsf{Neg} \\ \text{“}0\text{''}, \text{ if } I \equiv \{0\} \\ \text{“}\top\text{''}, \text{ o.w.} \end{cases}$$

$$\gamma(c) = [\![c]\!] .$$

Example# 2: Interval Lattice

Let $\mathrm{C} : 2^{\mathsf{Z}}$ and A : Intervals.

$$\alpha(\mathsf{X}) = [\mathsf{min}(\mathsf{X}), \mathsf{max}(\mathsf{X})$$

$$\gamma([\ell, u]) = [\![\ell, u]\!] = \{z \mid \ell \leq z \leq u\}\,.$$

Verify the Galois connection.

$$(\forall \mathrm{I} \subseteq \mathsf{Z}, [\ell, u] \in \mathsf{Intervals})\ \alpha(\mathrm{I}) \sqsubseteq [\ell, u] \text{ iff } \mathrm{I} \subseteq \gamma([\ell, u])\,.$$

Galois Connection: Intuition

$$\alpha: \text{ Sets of States} \mapsto \text{Abstraction (signs/intervals/...)}\,.$$

and

$$\gamma: \text{ Abstraction } \mapsto \text{ Sets of states it represents}\,.$$

Question: What does a Galois connection mean?

If a abstracts a set S iff the concretization of a overapproximates S.

Galois Connection: "Best" abstraction & concretization

Property # 0: Derive α given γ (and vice versa).

Idea:

"Best" abstraction of S should be the smallest abstract element that contains S.

$$\alpha_b(S) = \min\{a \mid S \subseteq \gamma(a)\}.$$

Similarly, "best" concretization given α is

$$\gamma_b(a) = \max\{S \mid \alpha(S) \sqsubseteq a\}.$$

Let us try to apply this to the two domains we have seen.

Galois Connection: Closure

Property # 1: $(\forall S \in C)\ S \subseteq \gamma(\alpha(S))$

Proof:

$$\alpha(S) \sqsubseteq \alpha(S). \text{Therefore, } S \subseteq \gamma(\alpha(S))\,.$$

Property # 2: $(\forall a \in A)\ \alpha(\gamma(a)) \subseteq a$

Proof:

$$\gamma(a) \subseteq \gamma(a). \text{Therefore, } \alpha(\gamma(a)) \sqsubseteq a\,.$$

Galois Connection: Monotonicity

Property # 3: α and γ are monotonic. I.e.,

$$\text{If } S_1 \subseteq S_2 \text{ then } \alpha(S_1) \sqsubseteq \alpha(S_2)\,.$$

Similarly,

$$\text{If } a_1 \sqsubseteq a_2 \text{ then } \gamma(a_1) \subseteq \gamma(a_2)\,.$$

Proof: Let $S_1 \subseteq S_2$. We know from Property #1 that $S_2 \subseteq \gamma(\alpha(S_2))$. Therefore, $S_1 \subseteq \gamma(\alpha(S_2))$. Applying Galois connection definition, $\alpha(S_1) \subseteq \alpha(S_2)$.

Similarly, we can prove the other part too.

Join Preservation

Property # 4: For all $S_1, S_2 \in C$,

$$\alpha(S_1 \cup S_2) = \alpha(S_1) \sqcup \alpha(S_2)\,.$$

Proof: We rely on a sub-fact about lattices.

> **Fact:** If for $a, b \in L$, for all $c \in L$,
> $a \sqsubseteq c \;\leftrightarrow\; b \sqsubseteq c$ then $a = b$.

$$
\begin{aligned}
\alpha(S_1 \cup S_2) \sqsubseteq c \quad & \text{iff} \quad S_1 \cup S_2 \subseteq \gamma(c) \\
& \text{iff} \quad S_1 \subseteq \gamma(c),\; S_2 \subseteq \gamma(c) \\
& \text{iff} \quad \alpha(S_1) \sqsubseteq c,\; \alpha(S_2) \sqsubseteq c \\
& \text{iff} \quad \alpha(S_1) \sqcup \alpha(S_2) \sqsubseteq c
\end{aligned}
$$

Now applying **fact**, we get

$$\alpha(S_1 \cup S_2) = \alpha(S_1) \sqcup \alpha(S_2)\,.$$

Meet Preservation

For all $S_1, S_2 \in C$,

$$\alpha(S_1 \cap S_2) = \alpha(S_1) \sqcap \alpha(S_2)\,.$$

Proof: Use dual fact.

Monotone Function Theorem

Let $f : C \mapsto C$ and $g : A \mapsto A$ be monotone functions on C, A respectively.

g is a <u>sound abstraction</u> of f iff

$$\forall\, S \in C,\;\; \alpha(f(S)) \sqsubseteq g(\alpha(S))\,.$$

Claim: $\alpha(\mathsf{LFP}_C(f)) \sqsubseteq \mathsf{LFP}_A(g)$.

1. $\alpha(\emptyset) = \bot$
2. $\forall\, n \geq 0,\; \forall S \in C,\;\; \alpha(f^n(S)) \sqsubseteq g^n(\alpha(S))$
3. $\alpha(\mathsf{LFP}(f)) \sqsubseteq \mathsf{LFP}(g)$.

Proving Soundness of Abstract Interpretation

Background

- We have a "concrete domain" $\mathcal{C} : 2^{\Sigma}$ and abstract domain $\langle L, \sqsubseteq \rangle$.
- Fixed point inside lattice $\mathcal{C}$: $\mathsf{Reach}(n)$.
- Dataflow analysis inside lattice L : $\mathsf{fp}(n)$ (eg., $\mathsf{sign}(n, x)$, $\mathsf{Rng}(n, y)$).
- **Goal:** Relate concrete fixed point $\mathsf{Reach}(n)$ with abstract fixed point $\mathsf{fp}_L(n)$.
- Let $\langle \alpha, \gamma \rangle$ be a galois connection between $\mathcal{C}$ and L.

Soundness: Basic Operations

We will establish $\alpha \circ f \sqsubseteq g \circ \alpha$.

- For any sets S_1, S_2,

  $$\alpha(S_1 \cup S_2) \sqsubseteq \alpha(S_1) \sqcup \alpha(S_2) .$$

  This is the join preservation result.

- For sets S_1, S_2,

  $$\alpha(S_1 \cap S_2) \sqsubseteq \alpha(S_1) \sqcap \alpha(S_2) .$$

  The meet preservation result.

- For any set S_1,

$$\alpha(\mathsf{post}_{\mathcal{C}}(n, S)) \sqsubseteq \mathsf{post}_{\mathrm{L}}(n, \alpha(S)) \, .$$

  This is a requirement.

- We can now lift the result to dataflow inequalities.

Soundness: Dataflow Inequalities

For a given program P,

Let $F(X) \subseteq X$ be the flow inequalities in the concrete domain.

Let $g(x) \sqsubseteq x$ be the flow inequalities in the abstract domain.

Obs. 1: F and g are structurally identical.

For example,

$$F : \mathsf{post}(n_0, X_0) \cup (X_1 \cap [\![I]\!]) \cup \mathsf{post}(n_1, X_1) \cup X_2 \,.$$

and

$$g : \mathsf{post}_L(n_0, x_0) \sqcup (x_1 \sqcap \alpha(I)) \sqcup \mathsf{post}(n_1, x_1) \sqcup x_2 \,.$$

Reason: The generation of dataflow inequalities is "syntax-directed".

Obs. 2: $\alpha(F(X)) \sqsubseteq g(\alpha(x))$.

Proof: Build this up from proof for basic operations.

Soundness

Let L be a dataflow lattice such that

1. There exists a Galois connection between L and concrete domain $\mathcal{C}$.
2. Post condition on L is sound, w.r.t post condition on $\mathcal{C}$,

$$\alpha(\mathsf{post}(n, S)) \sqsubseteq \mathsf{post}_{L}(n, \alpha(S))\,.$$

given program we get the dataflow inequalities:
$F(X) \subseteq X$ on $\mathcal{C}$ and $g(x) \sqsubseteq x$ on L,
then, the least fixed point of g on L abstracts the LFP of F on $\mathcal{C}$.

$$\mathsf{LFP}_{C}(F) \sqsubseteq \mathsf{LFP}_{L}(g)\,.$$