Abstract Interpretation, Re-Reloaded and Numerical and Structural Abstractions

Jan Midtgaard

Winter School, Day 4

`http://janmidtgaard.dk/aiws15/`

Saint Petersburg, Russia, 2015

With figures courtesy of David A. Schmidt, Patrick and Radhia Cousot

Yesterday

A first in-depth look at abstract interpretation based on Cousot-Cousot:JLP92.

- □ Foundations: Fixed points, Galois connections, . . .
- □ The Galois approach and friends: closure operators, Moore families, . . .
- □ From collecting semantics to analysis (soundness, optimality, completeness)

Analysing Plotkin's three counter machine

Today: Two parts

1. More approximation methods (Cousot-Cousot:JLP92)
   - Relational and attribute independent analysis
   - Inducing, abstracting, approximating fixed points
   - Widening, narrowing
   - Forwards/backwards analysis
2. A catalogue of abstractions
   - Toolbox abstractions
   - Structural abstractions: sums, pairs/tuples, . . .
   - Numerical abstractions: constants, intervals, polyhedra
   - Concretization-based abstract interpretation, briefly

   A retrospective on the 3 counter machine analysis

Relational vs. independent attribute analysis

Relational vs. independent attribute analysis

Definition. *We say an analysis is* attribute independent*, if attributes are analysed independently of each other.*

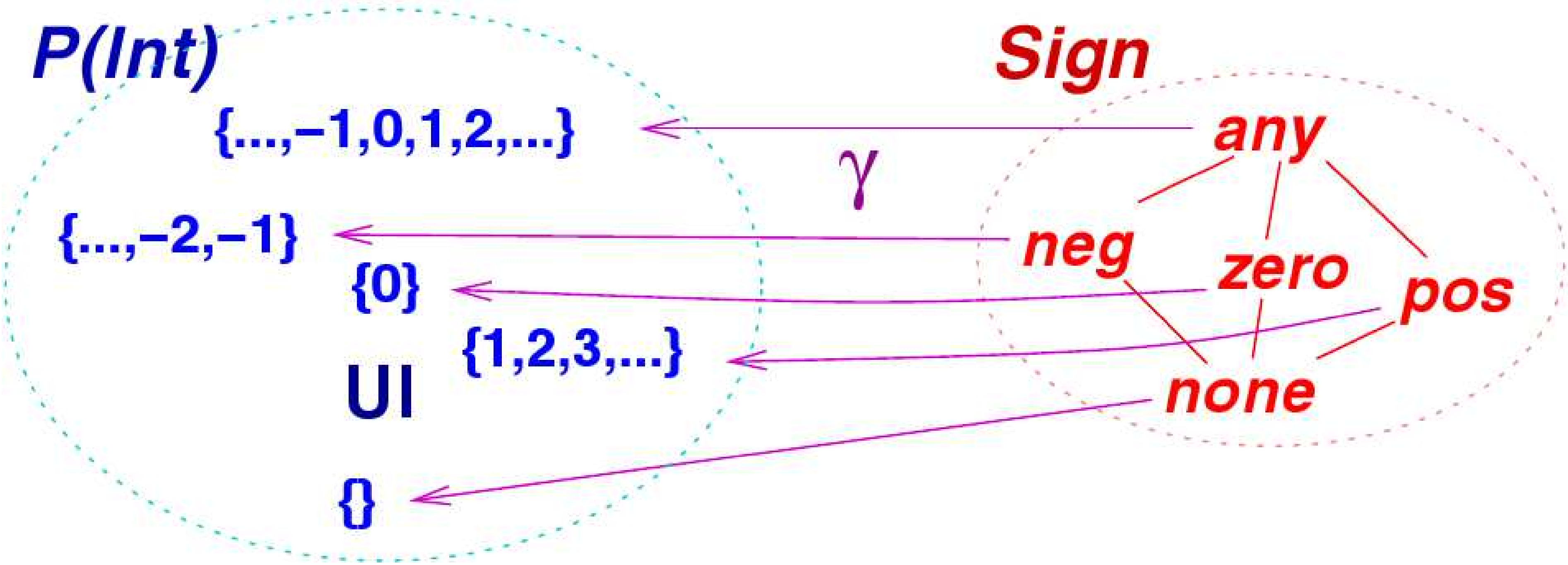
For example, the Parity analysis of `x`, `y`, and `z` we have developed, analyses the possible values of each variable in isolation.

Definition. *We say an analysis is* relational*, if it can determine relations between attributes.*

For example, imagine an analysis that can determine `x` is odd if and only if `y` is even.

The loss of information in joins

Consider the following Sign domain (a variant of the two Sign domains we studied yesterday):



Observe how γ doesn't preserve $\sqcup$.

Disjunctive/down-set completion

Given a Galois connection we can improve it, by considering sets of elements:

Proposition. *Given a Galois connection* $\langle \wp(C); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A; \leq \rangle$ *between complete lattices:* $\langle \wp(C); \subseteq, \emptyset, C, \cup, \cap \rangle$ *and* $\langle A; \leq, \bot, \top, \vee, \wedge \rangle$ *we can replace* A *by* $\wp_{\downarrow}(A) = \{\downarrow S \mid S \subseteq A\}$ *to form another Galois connection:*

$$\langle \wp(C); \subseteq \rangle \underset{\alpha^{\wp}}{\overset{\gamma^{\wp}}{\leftrightarrows}} \langle \wp_{\downarrow}(A); \subseteq \rangle$$

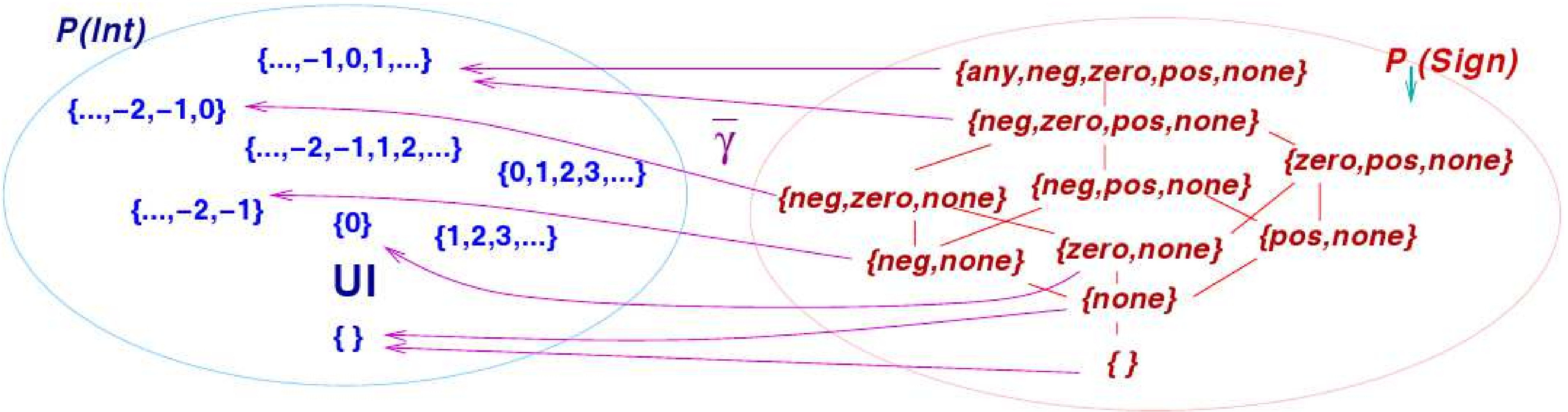
where $\downarrow S = \{a \in A \mid \exists a' \in S : a \leq a'\}$

$$\alpha^{\wp}(cs) = \cap\{as \mid cs \subseteq \gamma^{\wp}(as)\} = \downarrow \{\alpha(\{c\}) \mid c \in cs\}$$

$$\gamma^{\wp}(as) = \gamma(as) = \cup_{a \in as} \gamma(a)$$

Disjunctive completion to the rescue

The completion of the earlier Sign domain:



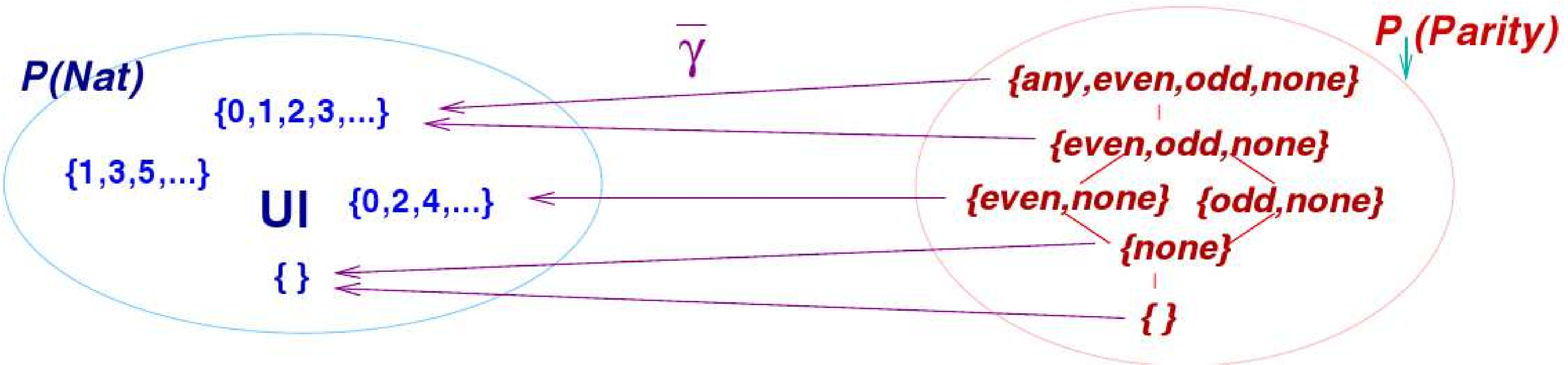
This domain is more expressive, however exponentially larger than the starting point.

In particular, it now preserves $\sqcup$ (from Sign)

Q: What do we obtain by domain reduction?

Not always a free lunch. . .

Disjunctive completion may not provide an improvement:



Q: What do we obtain by domain reduction?

From attribute independent to relational analysis

Observation: Given n independent analyses, the disjunctive completion of their reduced product provides a *relational analysis*.

Example: Consider the disjunctive completion of the reduced product for Parity, for two variables $\mathtt{x}$ and $\mathtt{y}$.

The resulting domain can express "x *is odd if and only if* y *is even*":

$$\begin{aligned} &\downarrow \{\langle odd,\ even\rangle, \langle even,\ odd\rangle\} \\ &\qquad = \{\langle odd,\ even\rangle, \langle even,\ odd\rangle, \ldots\} \end{aligned}$$

Inducing, abstracting, and approximating fixed points

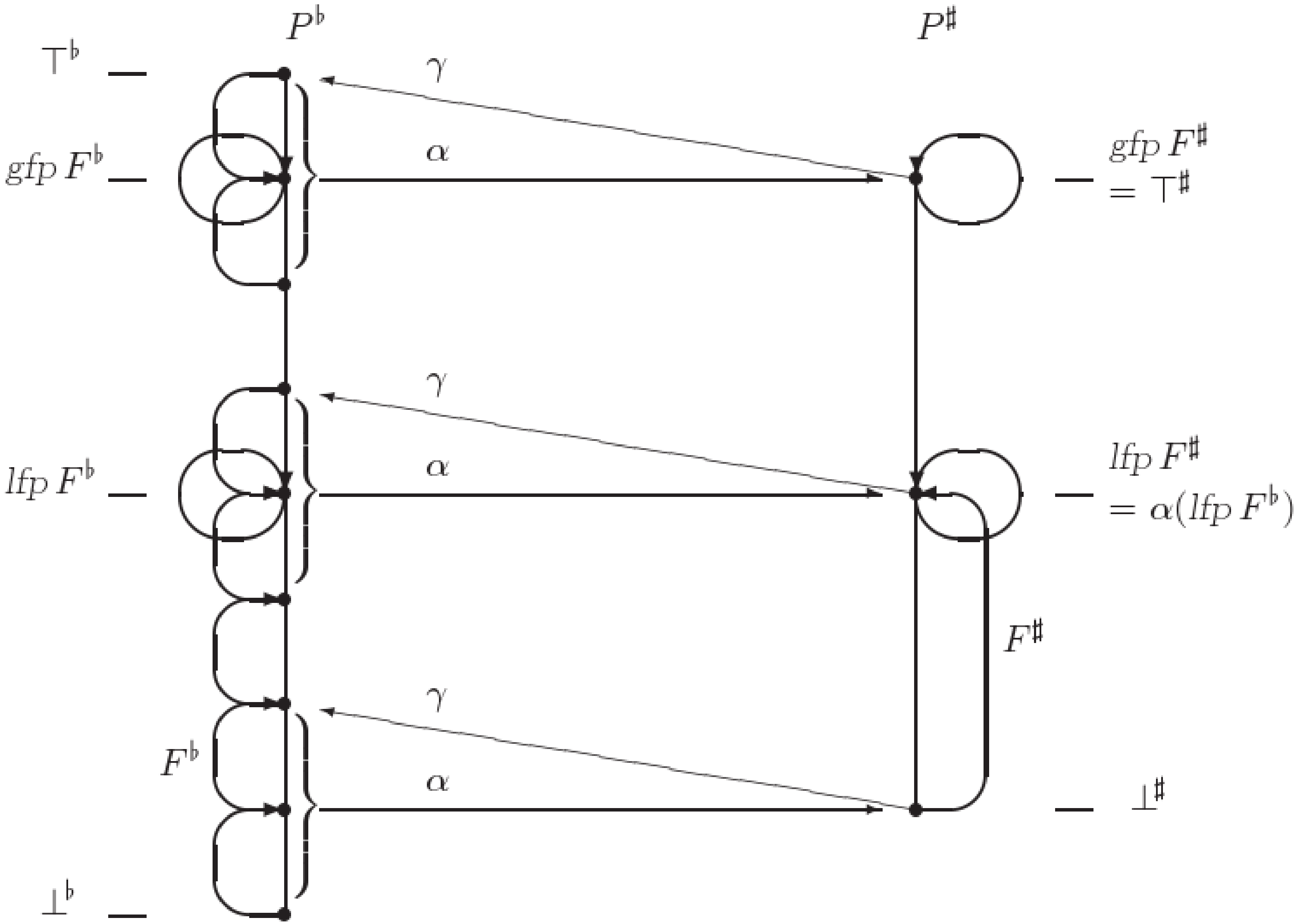
Fixed point inducing using Galois connections

Proposition. *If* $\langle C; \sqsubseteq \rangle \xleftrightarrow[\alpha]{\gamma} \langle A; \leq \rangle$ *is a Galois connection between posets* $\langle C; \sqsubseteq, \sqcup \rangle$ *and* $\langle A; \leq, \vee \rangle$, $F : C \to C$ *is such that* $\operatorname{lfp} F = \bigsqcup_{n \geq 0} F^n(\bot)$, $\alpha(\bot_c) = \bot_a$, $F^\# : A \to A$ *is such that* $\alpha \circ F = F^\# \circ \alpha$ *then* $\alpha(\operatorname{lfp} F) = \bigvee_{n \geq 0} {F^\#}^n(\bot_a)$ *and* $\bigvee_{n \geq 0} {F^\#}^n(\bot_a)$ *is a fixed point of* $F^\#$. *If* $F^\# : A \to A$ *is monotone, it is furthermore the* least *fixed point (*$\geq \bot_a$*).*

Note: this proposition concerns a *complete approximation*.

Also note: this proposition relaxes the preconditions of the stronger fixed point theorem (from yesterday).

Illustration of induced fixed point



Fixed point abstraction and approx. using Galois conn.

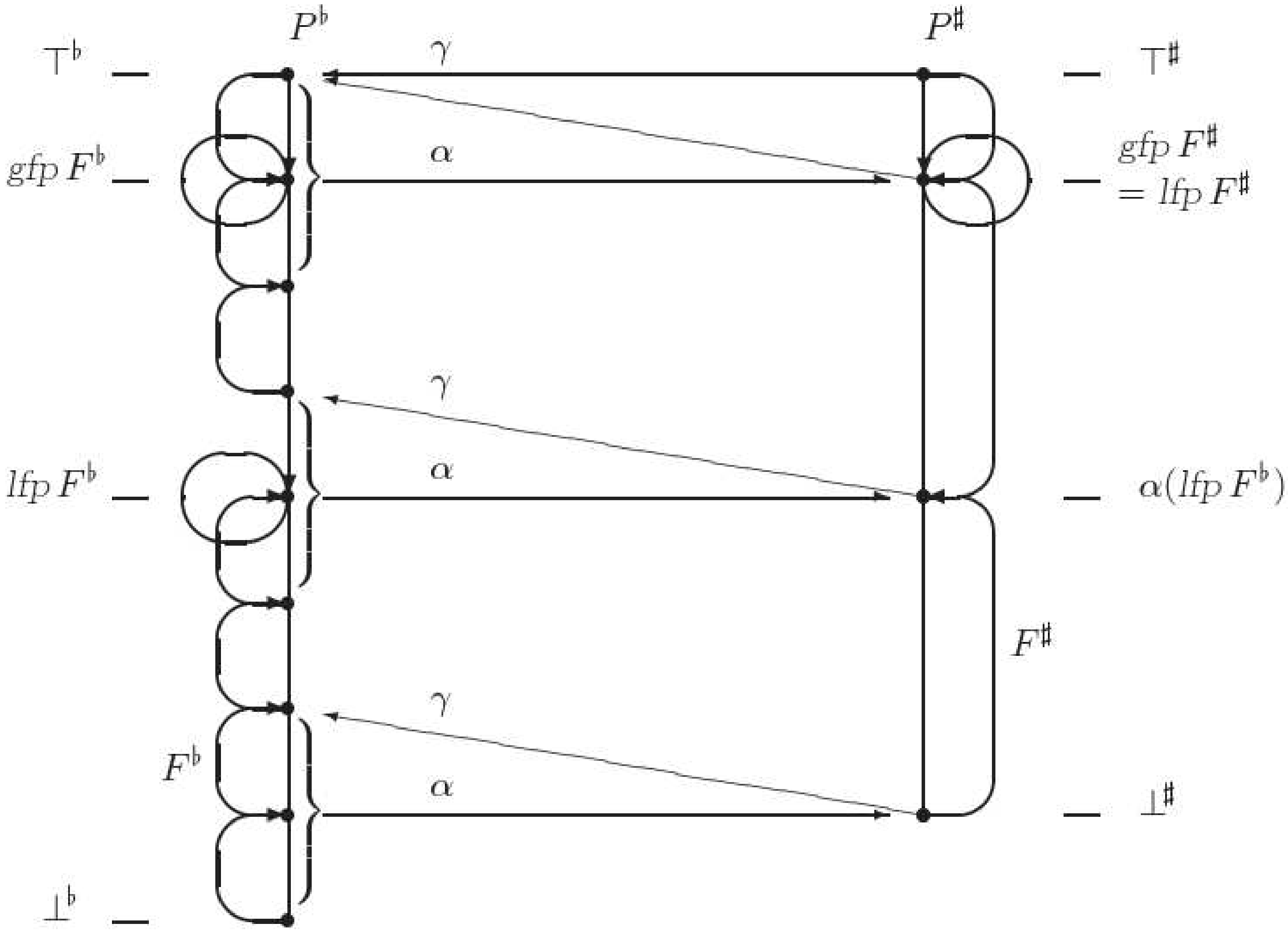
Proposition. *If* $\langle C; \sqsubseteq, \bot_c, \top_c, \sqcup, \sqcap \rangle$ *and* $\langle A; \leq, \bot_a, \top_a, \vee, \wedge \rangle$, *are complete lattices,* $\langle C; \sqsubseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A; \leq \rangle$ *and* $F : C \rightarrow C$ *is monotone, then* $\alpha(\mathrm{lfp}\, F) \leq \mathrm{lfp}(\alpha \circ F \circ \gamma)$

Note: this proposition concerns an *optimal approximation* (akin to what you did on Tuesday).

Proposition. *If* $\langle A; \leq, \bot_a, \top_a, \vee, \wedge \rangle$ *is a complete lattice,* $F^{\#}, F^{\#'} : A \rightarrow A$ *are monotone functions and* $F^{\#} \dot{\leq} F^{\#'}$*, then* $\mathrm{lfp}\, F^{\#} \leq \mathrm{lfp}\, F^{\#'}$*.*

Read: any monotone, upward judgement of the $\alpha \circ F \circ \gamma$ composition is fine.

Illustration of approximated fixed point



Widening/narrowing reloaded

Narrowing motivation

We are after a (finite) approximation sequence $\check{X}^0 \geq \check{X}^1 \geq \check{X}^2 \geq \cdots \geq \check{X}^n \geq \operatorname{lfp} F^\#$ of the least fixed point (from above).

We could start from, e.g., $\check{X}^0 = \top$.

For the inductive step, not much is available: the previous iterate $\check{X}^k$ and the function $F^\#$. Assuming $\operatorname{lfp} F^\# \leq \check{X}^k$ and $F^\#$ is monotone, we want to ensure $\operatorname{lfp} F^\# \leq \check{X}^{k+1}$.

The narrowing operator $\vartriangle$ simply combines the available information:

$$\check{X}^{k+1} = \check{X}^k \vartriangle F^\#(\check{X}^k)$$

Narrowing definition

Definition. *A narrowing operator* $\triangle$ *satisfies the following:*

- *For all* $x, y : (x \triangle y) \leq x$ (ensure decr. seq.)
- *For all* $x, y, z : x \leq y \ \wedge \ x \leq z \implies x \leq (y \triangle z)$
  (keep above)
- *For any decreasing chain* X_i *the alternative chain defined as* $\check{X}^0 = X_0$ *and* $\check{X}^{k+1} = \check{X}^k \triangle X_{k+1}$ *stabilizes after a finite number of steps.*
  (terminate)

Example: interval narrowing

Consider the domain of intervals: $\langle \wp(\mathbb{Z}); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \mathit{Interval}; \sqsubseteq \rangle$ defined as follows:

$$\mathit{Interval} = \{[l; u] \mid l \in \mathbb{Z} \cup \{-\infty\} \ \wedge \ u \in \mathbb{Z} \cup \{+\infty\} \ \wedge \ l \leq u\} \cup \{\bot\}$$

$$[a; b] \sqsubseteq [c; d] \iff c \leq a \ \wedge \ b \leq d$$

$$\alpha(\emptyset) = \bot$$

$$\alpha(X) = [\min X, \max X] \qquad \min \mathbb{Z} = -\infty \qquad \max \mathbb{Z} = +\infty$$

Strictly decreasing interval chains can be infinite:

$$[0; +\infty] \sqsupset [1; +\infty] \sqsupset [2; +\infty] \sqsupset \ldots$$

Hence we need a narrowing operator:

$$\bot \vartriangle I = \bot \qquad I \vartriangle \bot = \bot$$

$$[a; b] \vartriangle [c; d] = [\text{if } a = -\infty \text{ then } c \text{ else } a; \ \text{if } b = +\infty \text{ then } d \text{ else } b]$$

Downward iteration with narrowing

Proposition. *If $F^{\#} : A \to A$ is a monotone function, and $\triangle : A \times A \to A$ is a narrowing operator and $F^{\#}(a) = a \leq a'$ then $\check{X}^0 = a', \ldots, \check{X}^{k+1} = \check{X}^k \triangle F^{\#}(\check{X}^k)$ converges with limit $\check{X}^n$, $n \in \mathbb{N}$ such that $a \leq \check{X}^n \leq a'$.*

Intuition: this decreasing chain is finite and may take us closer to $F^{\#}$'s fixed point from above.

Note: In a complete lattice, if all strictly decreasing chains are finite, we can use $\triangle = \sqcap$.

Widening motivation

We aim for a better initial approximation than $\top$.

We are after a (finite) approximation sequence $\hat{X}^0 \leq \hat{X}^1 \leq \hat{X}^2 \leq \cdots \leq \hat{X}^n \geq \operatorname{lfp} F^\#$ of the least fixed point (starting below, ending above).

We could, e.g., try to iterate *above* a standard fixed point iteration: $X^0 = \bot, X^{k+1} = F^\#(X^k)$ towards $\operatorname{lfp} F^\#$.

Hence start from $\hat{X}^0 = \bot$

and use the widening operator $\triangledown$ to combine the available information:

$$\hat{X}^{k+1} = \hat{X}^k \triangledown F^\#(\hat{X}^k)$$

Widening definition

Definition. *A widening operator satisfies the following:*

- *For all* $x, y : x \leq (x \triangledown y) \ \wedge \ y \leq (x \triangledown y)$

  (keep above)

- *For any increasing chain* $X_0 \sqsubseteq X_1 \sqsubseteq X_2 \sqsubseteq \ldots$ *the alternative chain defined as* $\hat{X}^0 = X_0$ *and* $\hat{X}^{k+1} = \hat{X}^k \triangledown X_{k+1}$ *stabilizes after a finite number of steps.*

Example: interval widening

Consider again the domain of intervals:

$$\langle \wp(\mathbb{Z}); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle Interval; \sqsubseteq \rangle$$

For intervals strictly increasing chains can be infinite:

$$[0;0] \sqsubset [0;1] \sqsubset [0;2] \sqsubset \ldots$$

Hence we need a widening operator:

$$\bot \triangledown I = I \qquad\qquad I \triangledown \bot = I$$

$$[a;b] \triangledown [c;d] = [\text{if } c < a \text{ then } -\infty \text{ else } a;\ \text{if } d > b \text{ then } +\infty \text{ else } b]$$

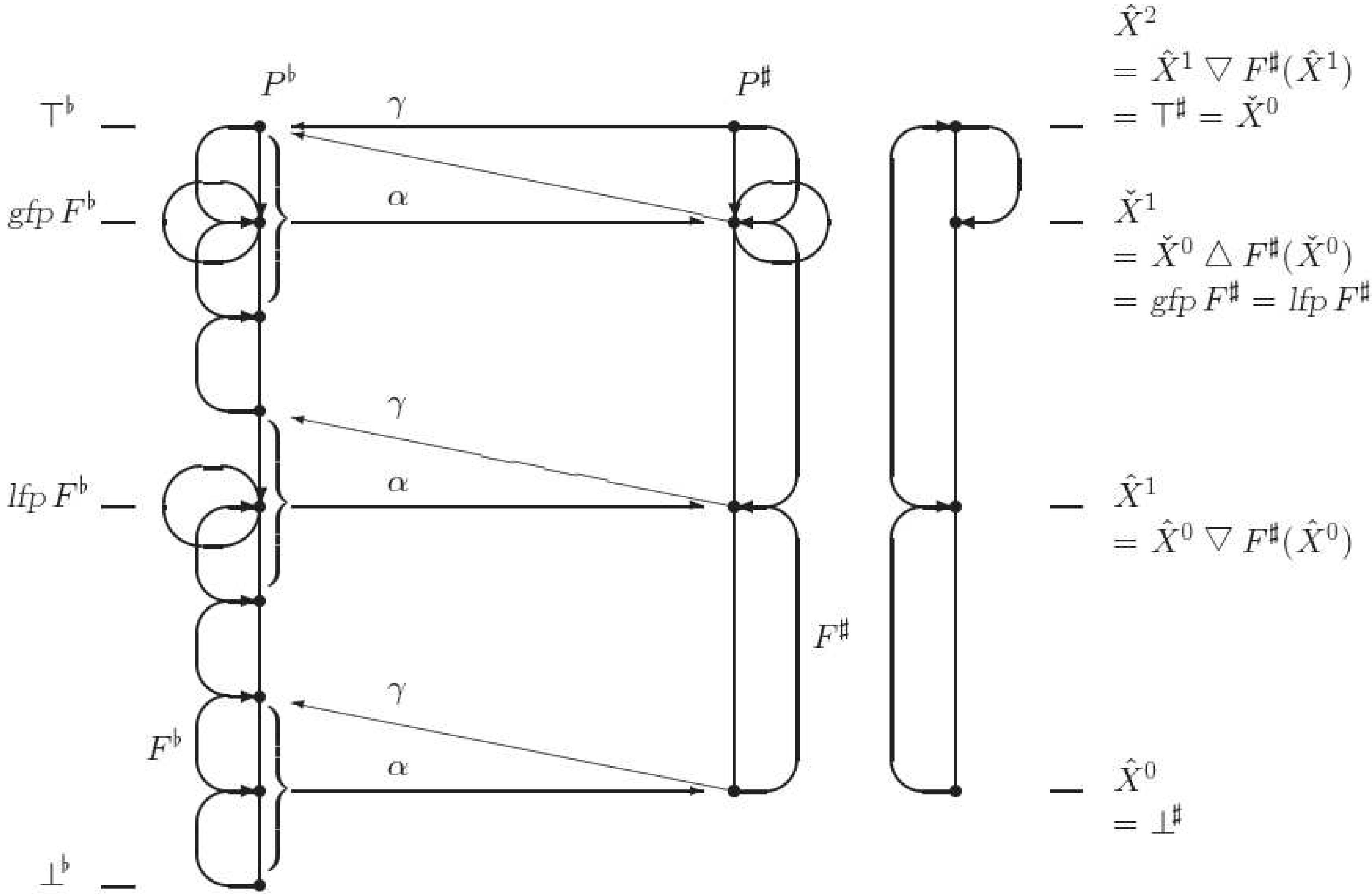
Upward iteration with widening

Proposition. *If* $F^\# : A \to A$ *is a monotone function, and* $\nabla : A \times A \to A$ *is a widening operator then* $\hat{X}^0 = \bot, \ldots, \hat{X}^{k+1} = \hat{X}^k \,\nabla\, F^\#(\hat{X}^k)$ *converges with limit* $\hat{X}^n$*,* $n \in \mathbb{N}$ *such that* $\mathrm{lfp}\, F^\# \leq \hat{X}^n$*.*

Note: In a complete lattice, if all strictly increasing chains are finite, we can use $\nabla = \sqcup$.

We don't actually need to widen in such a situation.

Combining widening/narrowing iteration



Forwards/backwards analysis

Transition systems with final states

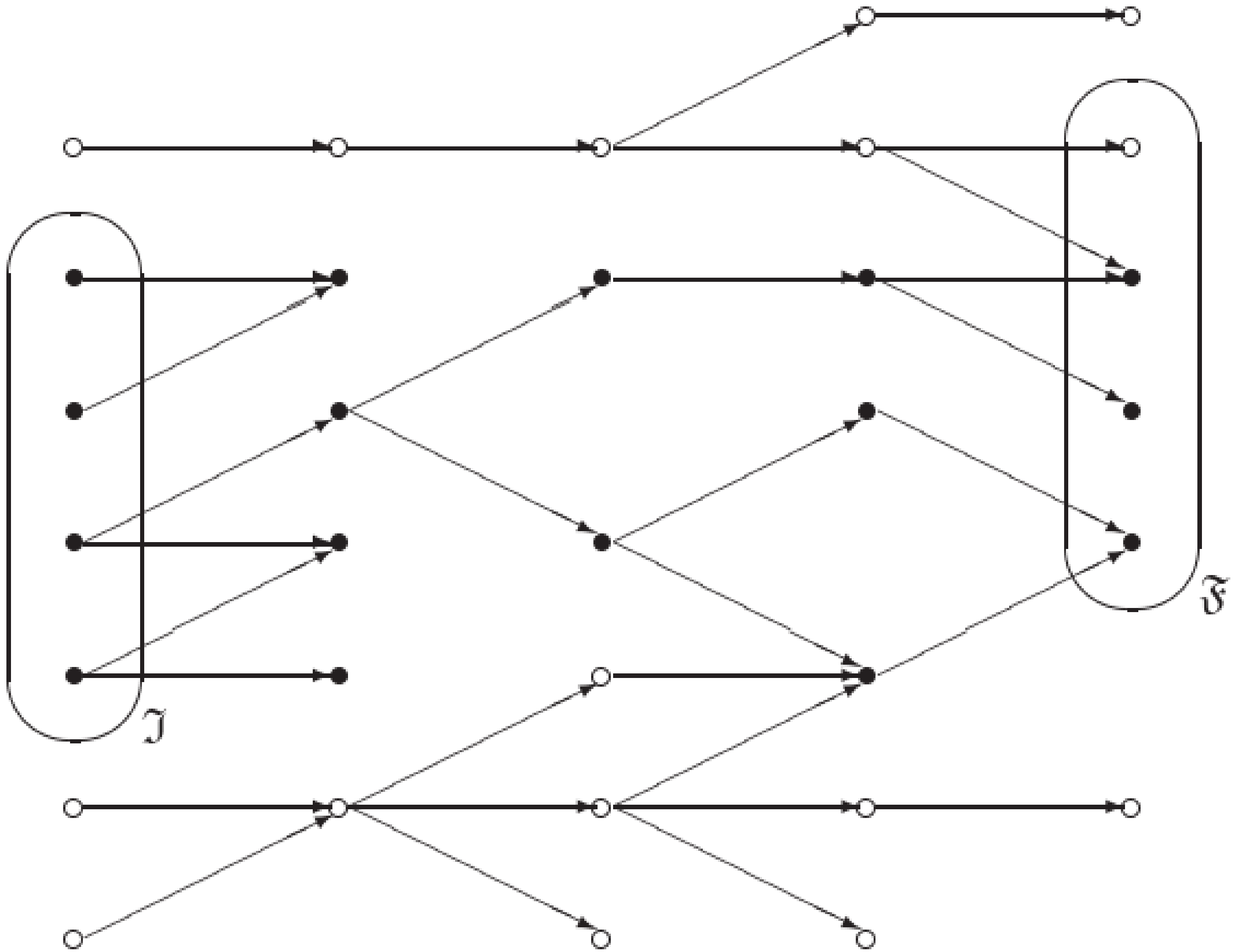
All though the transition system definition from day 1 included final states we haven't used them much:

Definition. *A transition system is a quadruple* $\langle S, S_i, S_f, \rightarrow \rangle$*, where*

- S *is a set of states*
- $S_i \subseteq S$ *is a set of initial states*
- $S_f \subseteq S$ *is a set of final states (*$\forall s \in S_f, s' \in S : s \not\rightarrow s'$*)*
- $\rightarrow \subseteq S \times S$ *is a transition relation relating a state to its (possible) successors*

Forwards collecting semantics (1/2)

Descendants of initial states (aka reachable states):



Forwards collecting semantics (2/2)

The forwards (top-down) collecting semantics can be expressed as a fixed point:

$$\begin{aligned}\operatorname{lfp} F \quad \text{where} \quad F(X) &= S_i \cup \{s \mid \exists s' \in X : s' \to s\} \\ &= S_i \cup post[\to](X)\end{aligned}$$

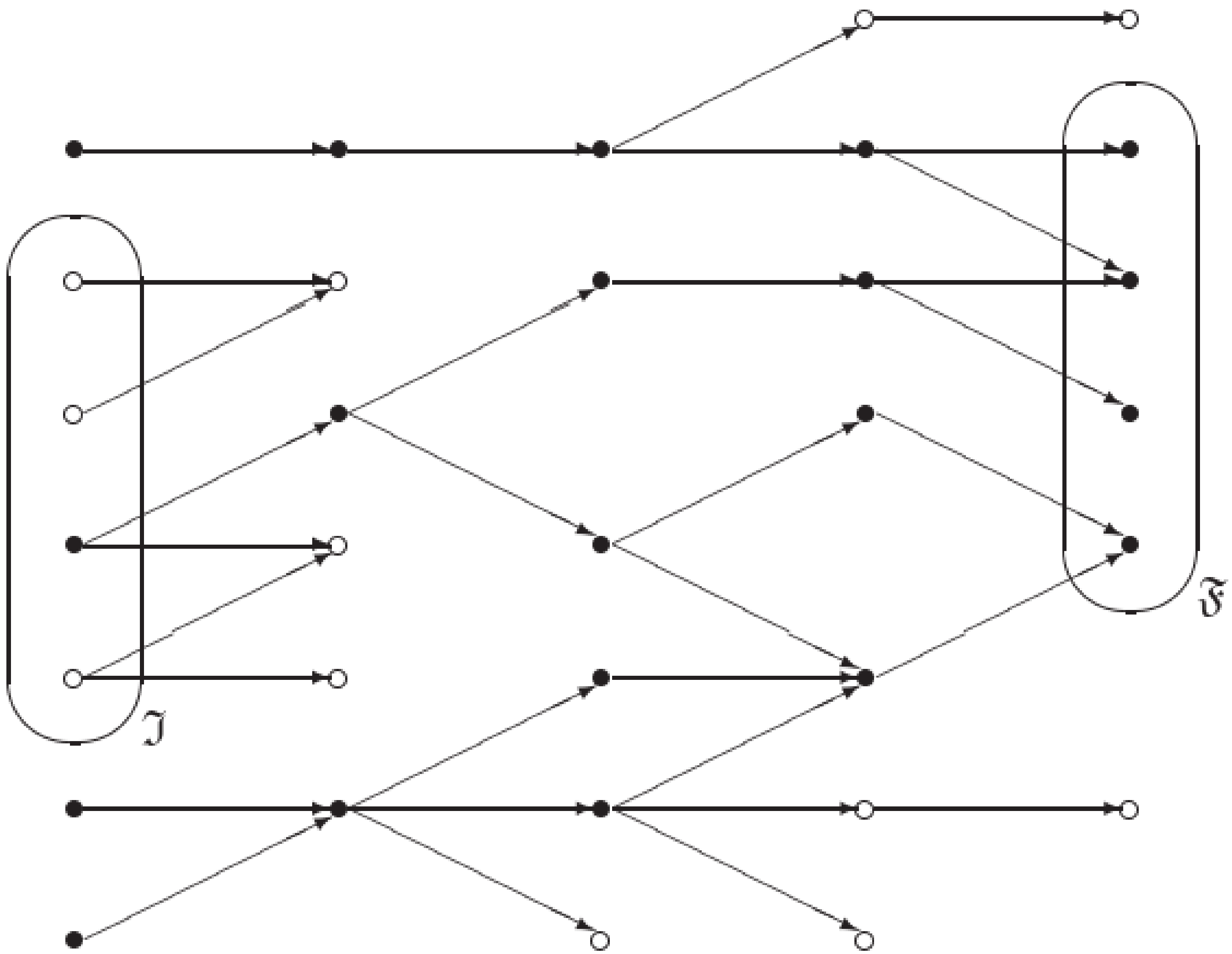
with

$$post[r](X) = \{s \mid \exists s' \in X : \langle s',\, s\rangle \in r\}$$

as we have already seen.

Backwards collecting semantics (1/2)

Ascendants of final states:



Backwards collecting semantics (2/2)

The backwards (bottom-up) collecting semantics can also be expressed as a fixed point:

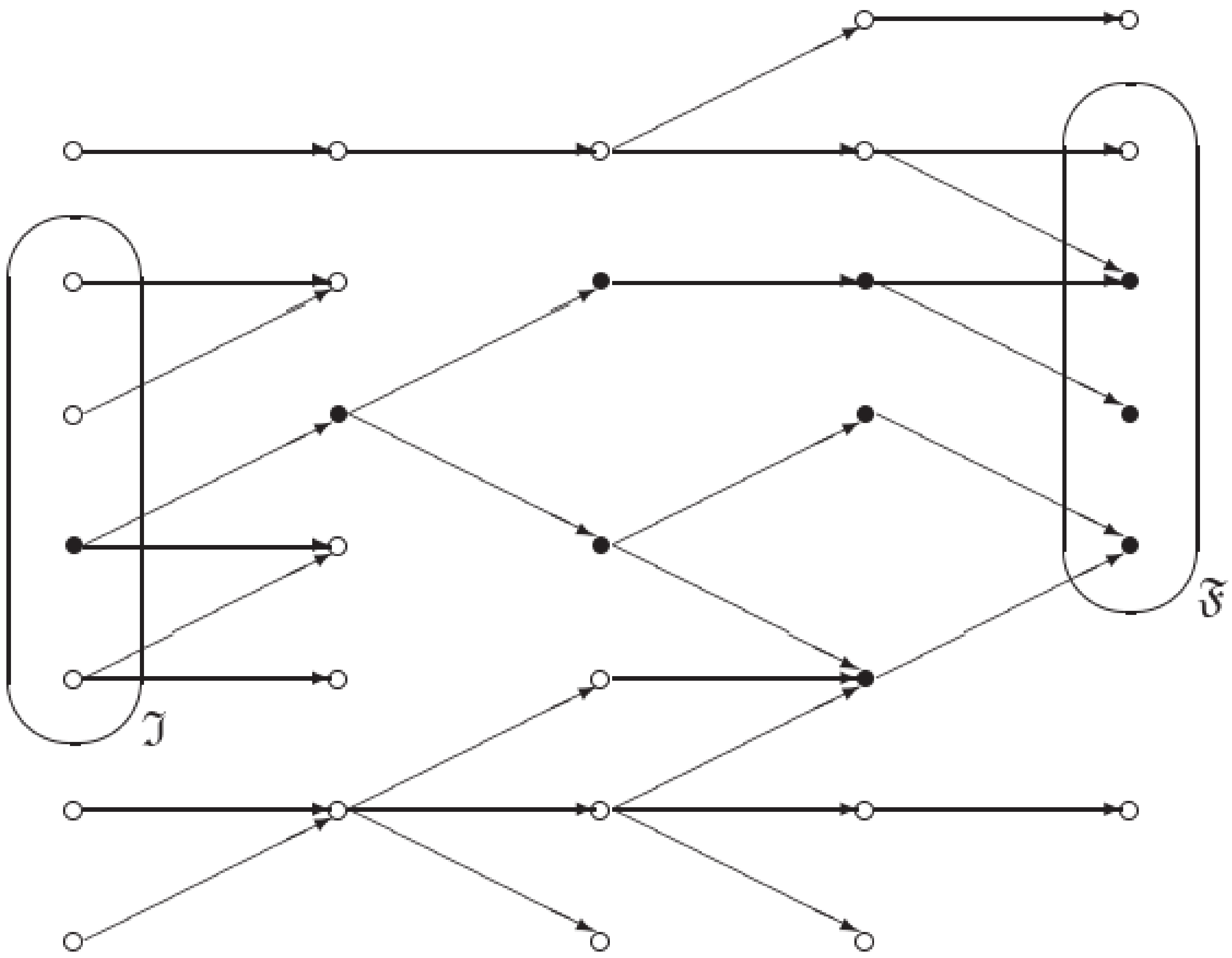
$$\begin{aligned}\operatorname{lfp} B \;\; \text{where} \;\; B(X) &= S_f \cup \{s \mid \exists s' \in X : s \to s'\} \\ &= S_f \cup pre[\to](X)\end{aligned}$$

with

$$pre[r](X) = \{s \mid \exists s' \in X : \langle s,\, s' \rangle \in r\}$$

Forwards/backwards collecting semantics (1/2)

Descendants of initial states which are also ascendants of final states:



Forwards/backwards collecting semantics (2/2)

This set of states can be expressed as the intersection of the two fixed points just defined:

$$\operatorname{lfp} F \cap \operatorname{lfp} B$$

The above is not computable in general, but the intuition is:

1. “run program forwards”,
2. “run program backwards”,
3. intersect.

Forwards/backwards collecting semantics in other words

We can express forwards/backward collecting semantics in several ways:

Proposition. *Given a transition system* $\langle S,\, S_i,\, S_f,\, \rightarrow\rangle$ *with* $X \subseteq S$*, we have*

1. $pre[\rightarrow](X) \cap \operatorname{lfp} F \subseteq pre[\rightarrow](X \cap \operatorname{lfp} F)$
2. $post[\rightarrow](X) \cap \operatorname{lfp} B \subseteq post[\rightarrow](X \cap \operatorname{lfp} B)$

$$\operatorname{lfp} F \cap \operatorname{lfp} B$$

3. $= \operatorname{lfp}(\lambda X.\, \operatorname{lfp} F \cap B(X))$
4. $= \operatorname{lfp}(\lambda X.\, \operatorname{lfp} B \cap F(X))$
5. $= \operatorname{lfp}(\lambda X.\, \operatorname{lfp} F \cap \operatorname{lfp} B \cap B(X))$
6. $= \operatorname{lfp}(\lambda X.\, \operatorname{lfp} F \cap \operatorname{lfp} B \cap F(X))$

Forwards/backwards analysis

Once we move to an abstract domain, a sequence akin to the alternative characterizations is more precise:

Proposition. *If* $\langle C; \sqsubseteq, \bot_c, \top_c, \sqcup, \sqcap\rangle$ *and* $\langle A; \leq, \bot_a, \top_a, \vee, \wedge\rangle$*, are complete lattices,* $\langle C; \sqsubseteq\rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A; \leq\rangle$*,* $F, B : C \to C$ *are monotone functions satisfying (5) and (6),* $F^\#, B^\# : A \to A$ *are monotone functions, such that* $\alpha \circ F \circ \gamma \leq F^\#$ *and* $\alpha \circ B \circ \gamma \leq B^\#$*, then the sequence*

- $\dot{X}^0 = \operatorname{lfp} F^\#$ *(or* $\operatorname{lfp} B^\#$*)*
- $\dot{X}^{2n+1} = \operatorname{lfp}(\lambda X.\, \dot{X}^{2n} \wedge B^\#(X))$
- $\dot{X}^{2n+2} = \operatorname{lfp}(\lambda X.\, \dot{X}^{2n+1} \wedge F^\#(X))$

satisfies for all $k \in \mathbb{N} : \alpha(\operatorname{lfp} F \cap \operatorname{lfp} B) \leq \dot{X}^{k+1} \leq \dot{X}^k$

Hence, we have an ascending sequence.

Forwards/backwards analysis over infinite domains

We may also need to *narrow* in order to ensure termination of the downward iteration (if descending chains can be infinite):

$$\dot{X}^0 > \dot{X}^1 > \dot{X}^2 > \dots$$

Similarly, we may need to *widen* (and *narrow*) to ensure termination of the fixed point computation in each iterate.

- $\dot{X}^0 = \mathrm{lfp} \dots$
- $\dot{X}^{2n+1} = \mathrm{lfp}(\dots)$
- $\dot{X}^{2n+2} = \mathrm{lfp}(\dots)$

Part II:
Toolbox abstractions

Warm up: Collapsing abstractions

The collapsing abstraction into a two element lattice:

$$\begin{aligned}\alpha(\emptyset) &= \bot\\ \alpha(S) &= \top \quad \text{if} \quad S \neq \emptyset\\ \gamma(\bot) &= \emptyset\\ \gamma(\top) &= S\end{aligned}$$

$$\langle \wp(S); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \{\top, \bot\}; \sqsubseteq \rangle$$

is slightly better than the completely collapsing abstraction:

$$\begin{aligned}\alpha(S) &= \bot\\ \gamma(\bot) &= S\end{aligned}$$

$$\langle \wp(S); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \{\bot\}; \sqsubseteq \rangle$$

Subset abstraction

Given a set C and a strict subset $A \subset C$ hereof, the restriction to the subset induces a Galois connection:

$$\langle \wp(C); \subseteq \rangle \underset{\alpha_{\subset}}{\overset{\gamma_{\subset}}{\leftrightarrows}} \langle \wp(A); \subseteq \rangle$$

$$\alpha_{\subset}(X) = X \cap A$$

$$\gamma_{\subset}(Y) = Y \cup (C \setminus A)$$

For example, in a *control-flow analysis* of untyped functional programs one can choose to focus on functional values (closures) and not model numbers:

$$\langle \wp(Clo + Num); \subseteq \rangle \underset{\alpha_{\subset}}{\overset{\gamma_{\subset}}{\leftrightarrows}} \langle \wp(Clo); \subseteq \rangle$$

(Note: by a sum $A + B$ we mean the disjoint union)

Elementwise abstraction

Let an elementwise operator @ $: C \to A$ be given.
Define

$$\alpha(P) = \{@(p) \mid p \in P\}$$
$$\gamma(Q) = \{p \mid @(p) \in Q\}$$

Then

$$\langle \wp(C); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \wp(A); \subseteq \rangle$$

In particular, if @ is onto, we have

$$\langle \wp(C); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftarrows\!\!\twoheadrightarrow}} \langle \wp(A); \subseteq \rangle$$

For example, Parity is isomorphic to an elementwise abstraction.
Q: what would A and @ be in this case?

Structural abstractions

Structural?

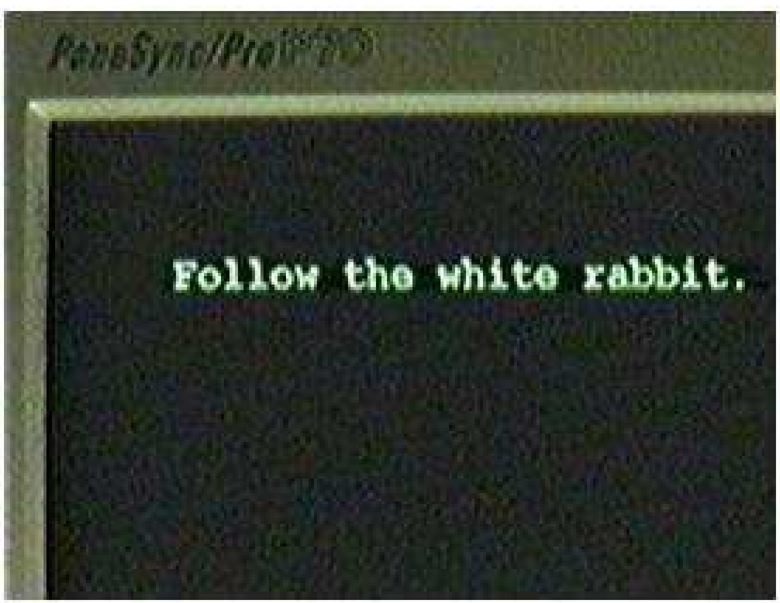
How is $State^{\#}$ constructed? It is possible to invent $State^{\#}$, and then the pair of adjoined functions. Another approach consists in inducing $State^{\#}$ from the structure of $State$.

—Alain Deutsch, POPL'90

Structural?

How is $State^{\#}$ constructed? It is possible to invent $State^{\#}$, and then the pair of adjoined functions. Another approach consists in inducing $State^{\#}$ from the structure of $State$.

—Alain Deutsch, POPL'90



Abstracting sums as a product

We can abstract sums by first utilizing a simple isomorphism:

$$\langle \wp(A + B); \subseteq \rangle \xleftarrow[\alpha]{\gamma} \langle \wp(A) \times \wp(B); \subseteq_\times \rangle$$

where

$$\alpha(S) = (\{a \mid a \in S \cap A\}, \{b \mid b \in S \cap B\})$$

This isomorphism will typically enable further approximation.

For example, the values of a mini-Scheme language could be such a disjoint sum: closure or number

Componentwise abstraction

We can abstract a Cartesian product (e.g., the outcome of the previous isomorphism) componentwise:

$$\frac{\langle \wp(C_i); \subseteq \rangle \underset{\alpha_i}{\overset{\gamma_i}{\leftrightarrows}} \langle A_i; \sqsubseteq_i \rangle \qquad i \in \{1, \ldots, n\}}{\langle \wp(C_1) \times \ldots \times \wp(C_n); \subseteq_\times \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A_1 \times \ldots \times A_n; \sqsubseteq_\times \rangle}$$

with

$$\alpha(\langle X_1, \ldots, X_n \rangle) = \langle \alpha_1(X_1), \ldots, \alpha_n(X_n) \rangle$$
$$\gamma(\langle x_1, \ldots, x_n \rangle) = \langle \gamma_1(x_1), \ldots, \gamma_n(x_n) \rangle$$

and writing $\sqsubseteq_\times$ for componentwise inclusion.

For example, we used the “triple version” ($n = 3$) for abstracting the 3 Counter Machine memory.

Abstracting pairs, coarsely

We can approximate a set-of-pairs by an abstract pair:

$$\frac{\langle \wp(C_1); \subseteq \rangle \underset{\alpha_1}{\overset{\gamma_1}{\leftrightarrows}} \langle A_1; \leq_1 \rangle \qquad \langle \wp(C_2); \subseteq \rangle \underset{\alpha_2}{\overset{\gamma_2}{\leftrightarrows}} \langle A_2; \leq_2 \rangle}{\langle \wp(C_1 \times C_2); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A_1 \times A_2; \leq_\times \rangle}$$

where

$$\alpha(S) = \langle \alpha_1(\{a \mid (a, b) \in S\}),\ \alpha_2(\{b \mid (a, b) \in S\}) \rangle$$

For example, our transitive abstraction of the three memory registers of the 3CM boils down to this approach.

Abstracting pairs, better

Utilizing the well-known isomorphism

$$\langle \wp(C_1 \times C_2); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle C_1 \to \wp(C_2); \dot{\subseteq} \rangle$$

we can approximate the set-of-pairs as a function between abstract domains:

$$\frac{\langle \wp(C_1); \subseteq \rangle \underset{\alpha_1}{\overset{\gamma_1}{\leftrightarrows}} \langle A_1; \leq_1 \rangle \qquad \langle \wp(C_2); \subseteq \rangle \underset{\alpha_2}{\overset{\gamma_2}{\leftrightarrows}} \langle A_2; \leq_2 \rangle}{\langle \wp(C_1 \times C_2); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A_1 \to A_2; \dot{\leq}_2 \rangle}$$

where

$$\alpha(S) = \dot{\bigsqcup}\{[\alpha_1(\{a\}) \mapsto \alpha_2(\{b\})] \mid \langle a,\, b \rangle \in S\}$$

Abstracting pairs, relationally

Finally we can go all-in and approximate the set-of-pairs as an abstract set-of-pairs:

$$\frac{\langle \wp(C_1); \subseteq \rangle \xleftarrow[\alpha_1]{\gamma_1} \langle A_1; \leq_1 \rangle \qquad \langle \wp(C_2); \subseteq \rangle \xleftarrow[\alpha_2]{\gamma_2} \langle A_2; \leq_2 \rangle}{\langle \wp(C_1 \times C_2); \subseteq \rangle \xleftarrow[\alpha]{\gamma} \langle \wp(A_1 \times A_2)/_{\equiv}; \subseteq \rangle}$$

where $\quad \alpha(S) = \{\langle \alpha_1(\{a\}),\, \alpha_2(\{b\})\rangle \mid \langle a,\, b\rangle \in S\}$

Note: this requires a domain reduction, equating all elements with the same meaning, e.g., in $\wp(Par \times Par)$, $\{\langle \top,\, even\rangle\} \equiv \{\langle odd,\, even\rangle, \langle even,\, even\rangle\}$.

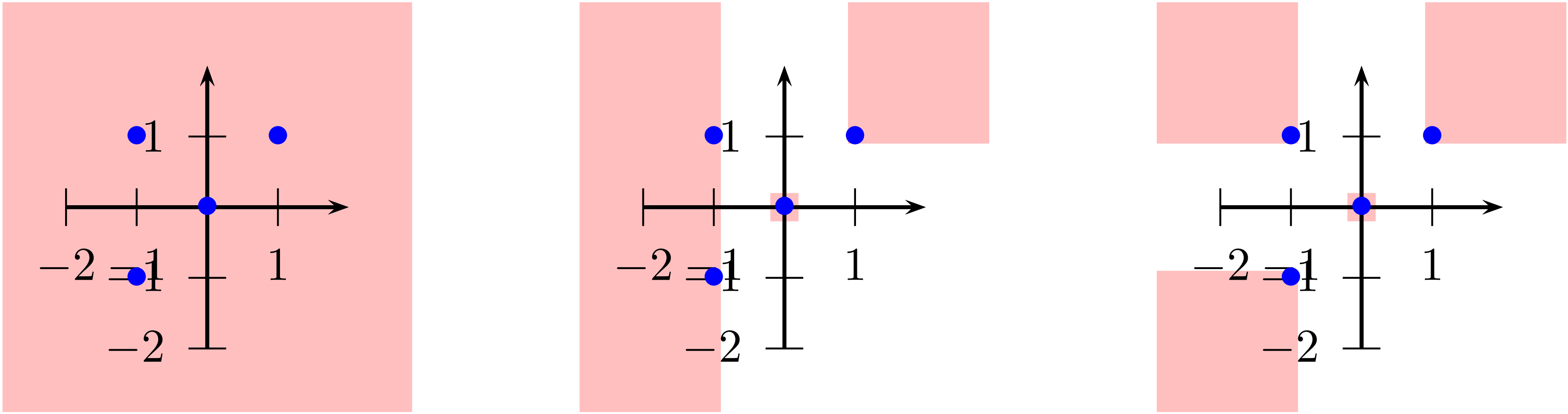
Perhaps a fun project abstracting the 3CM in this manner?

Comparing the three pair abstractions

Suppose we abstract the signs of the following set

$$S = \{\langle -1,\, -1\rangle, \langle 0,\, 0\rangle, \langle 1,\, 1\rangle, \langle -1,\, 1\rangle\}$$

coarsely: better: relationally:



$$\alpha(S) = \langle \top,\, \top\rangle$$

$$\alpha(S) = [neg \mapsto \top,\; 0 \mapsto 0,\; pos \mapsto pos]$$

$$\alpha(S) = \{\langle neg,\, neg\rangle, \langle 0,\, 0\rangle, \langle pos,\, pos\rangle, \langle neg,\, pos\rangle\}$$

A projecting abstraction

$$\frac{}{\langle \wp(C_1 \times \cdots \times C_n); \subseteq \rangle \xleftrightarrows[\alpha]{\gamma} \langle \wp(C_i); \subseteq \rangle}$$

where

$$\begin{aligned} \alpha(T) &= \{\pi_i t \mid t \in T\} \\ \gamma(E) &= \{t \mid \pi_i t \in E\} \end{aligned}$$

(which can be seen as an ’elementwise abstraction’)

The 3CM analysis uses a product of three such projections.

Abstracting monotone functions

Similar to the 'better abstraction' of pairs, we can approximate monotone functions by monotone abstract functions:

$$\frac{\langle C_1; \subseteq_1 \rangle \underset{\alpha_1}{\overset{\gamma_1}{\leftrightarrows}} \langle A_1; \leq_1 \rangle \qquad \langle C_2; \subseteq_2 \rangle \underset{\alpha_2}{\overset{\gamma_2}{\leftrightarrows}} \langle A_2; \leq_2 \rangle}{\langle C_1 \xrightarrow{m} C_2; \dot{\subseteq}_2 \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A_1 \xrightarrow{m} A_2; \dot{\leq}_2 \rangle}$$

where $X \xrightarrow{m} Y$ are the monotone functions from X to Y and

$$\alpha(f) = \alpha_2 \circ f \circ \gamma_1$$
$$\gamma(g) = \gamma_2 \circ g \circ \alpha_1$$

For $C_1 = C_2, A_1 = A_2$ this reduces to an *optimal approximation of* f and how we approximate fixed points

Abstracting sequences

We can abstract a set of sequences (rather crudely) by collapsing their elements:

$$\frac{\langle \wp(C); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A; \leq \rangle}{\langle \wp(C^*); \subseteq \rangle \underset{\alpha^*}{\overset{\gamma^*}{\leftrightarrows}} \langle A; \leq \rangle}$$

where

$$\alpha^*(S) = \alpha(\{x \mid x \in s \ \wedge \ s \in S\})$$

Numerical abstractions

Numerical abstractions

We've already come across a few numerical abstract domains: parity, signs, intervals, . . .

All of these were attribute independent (or non-relational): they don't express relations between (the values of) variables.

Let's recap what we have seen and supplement with some new ones, both non-relational and relational.

What is a numerical abstract domain?

A computer-representable property, with

- top and bottom: $\top$, $\bot$
- join, meet, and ordering operators: $\sqcup$, $\sqcap$, and $\sqsubseteq$
- widening and narrowing operators (optional, for domains with infinite strictly incr./decr. chains)
- some primitive operations: $+, -, *, /$
- other basic operations: test, assignment
- with matching backwards operations (optional, for (forwards/) backwards analysis)
- a γ-function mapping elements to their meaning (mathematical, not necessarily computable)

The parity domain

$$Par = \{\top, odd, even, \bot\}$$

$$\langle \wp(\mathbb{N}_0); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle Par; \sqsubseteq \rangle$$

$$\begin{tikzcd}[column sep=small] & \top \arrow[dl, dash] \arrow[dr, dash] & \\ odd \arrow[dr, dash] & & even \arrow[dl, dash] \\ & \bot & \end{tikzcd}$$

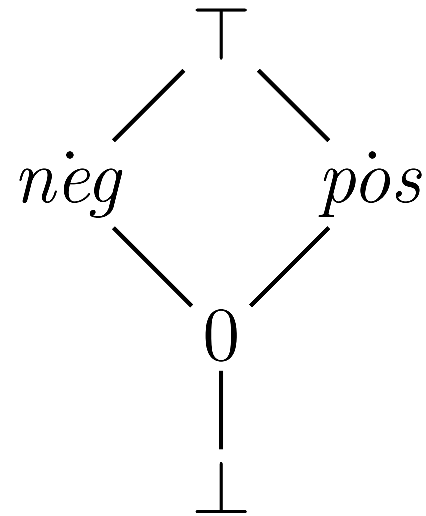
where

$$\begin{aligned} \gamma(\bot) &= \emptyset \\ \gamma(odd) &= \{n \in \mathbb{N}_0 \mid n \mod 2 = 1\} \\ \gamma(even) &= \{n \in \mathbb{N}_0 \mid n \mod 2 = 0\} \\ \gamma(\top) &= \mathbb{N}_0 \end{aligned}$$

A simple sign domain

$$Sign = \{\top, p\dot{o}s, n\dot{e}g, 0, \bot\}$$

$$\langle \wp(\mathbb{Z}); \subseteq \rangle \mathrel{\underset{\alpha}{\overset{\gamma}{\leftrightarrows}}} \langle Sign; \sqsubseteq \rangle$$



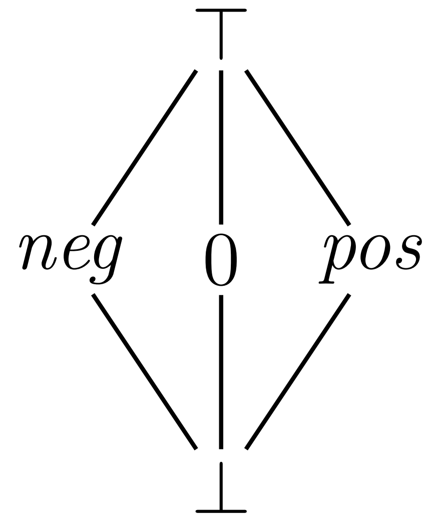
where

$$\begin{aligned}
\gamma(\bot) &= \emptyset \\
\gamma(0) &= \{0\} \\
\gamma(p\dot{o}s) &= \{n \in \mathbb{Z} \mid n \geq 0\} \\
\gamma(n\dot{e}g) &= \{n \in \mathbb{Z} \mid n \leq 0\} \\
\gamma(\top) &= \mathbb{Z}
\end{aligned}$$

Another simple sign domain

$$Sign = \{\top, pos, neg, 0, \bot\}$$

$$\langle \wp(\mathbb{Z}); \subseteq \rangle \mathrel{\substack{\xleftarrow{\quad\gamma\quad} \\ \underset{\alpha}{\twoheadrightarrow}}} \langle Sign; \sqsubseteq \rangle$$

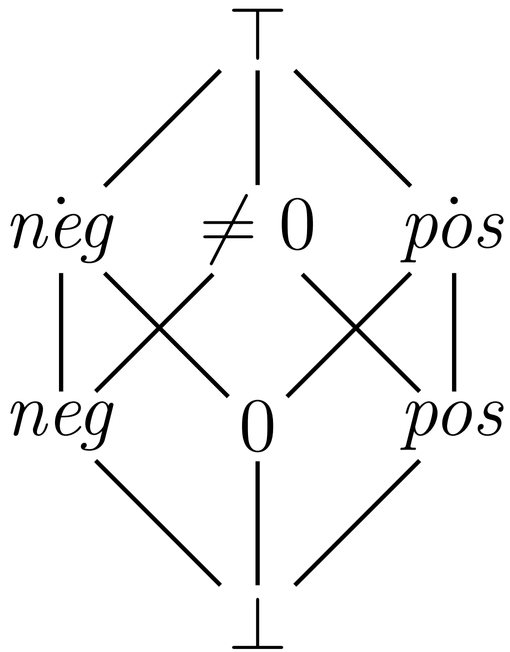


where

$$\begin{aligned}
\gamma(\bot) &= \emptyset \\
\gamma(0) &= \{0\} \\
\gamma(pos) &= \{n \in \mathbb{Z} \mid n > 0\} \\
\gamma(neg) &= \{n \in \mathbb{Z} \mid n < 0\} \\
\gamma(\top) &= \mathbb{Z}
\end{aligned}$$

The improved sign domain

$$Sign = \{\top, \neq 0, p\dot{o}s, n\dot{e}g, pos, neg, 0, \bot\}$$



$$\langle \wp(\mathbb{Z}); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle Sign; \sqsubseteq \rangle$$

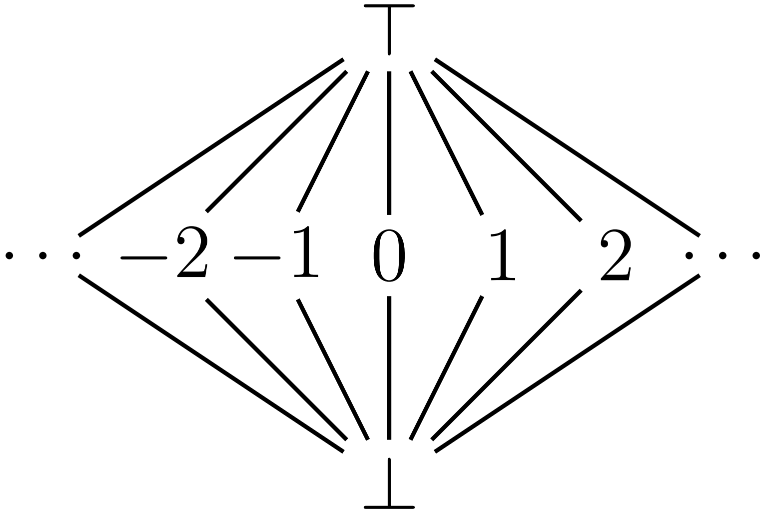
where

$$\begin{aligned}
\gamma(\bot) &= \emptyset \\
\gamma(0) &= \{0\} \\
\gamma(pos) &= \{n \in \mathbb{Z} \mid n > 0\} \\
\gamma(neg) &= \{n \in \mathbb{Z} \mid n < 0\} \\
\gamma(p\dot{o}s) &= \{n \in \mathbb{Z} \mid n \geq 0\} \\
\gamma(n\dot{e}g) &= \{n \in \mathbb{Z} \mid n \leq 0\} \\
\gamma(\neq 0) &= \{n \in \mathbb{Z} \mid n \neq 0\} \\
\gamma(\top) &= \mathbb{Z}
\end{aligned}$$

The constant propagation domain (Kildall:73)

$$Const = \mathbb{Z} \cup \{\top, \bot\}$$

$$\langle \wp(\mathbb{Z}); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle Const; \sqsubseteq \rangle$$



where

$$\gamma(\top) = \mathbb{Z} \qquad \alpha(\{n_1, n_2, \ldots\}) = \top$$
$$\gamma(n) = \{n\} \qquad \alpha(\{n\}) = n$$
$$\gamma(\bot) = \emptyset \qquad \alpha(\emptyset) = \bot$$

This domain extends naturally to string, characters, ...

Simple congruences (Granger'89)

$$Cong = \{\bot\} \cup \{a + b\mathbb{Z} \mid a, b \in \mathbb{Z} : \\ (b = 0) \;\vee\; (0 \leq a < b)\}$$

$$\langle \wp(\mathbb{Z}); \subseteq \rangle \xrightleftharpoons[\alpha]{\gamma} \langle Cong; \sqsubseteq \rangle$$

$$\gamma(\bot) = \emptyset$$
$$\gamma(a + b\mathbb{Z}) = \{a + bz \mid z \in \mathbb{Z}\}$$

Ordering:

$$\bot \sqsubseteq (a + b\mathbb{Z})$$
$$(a + b\mathbb{Z}) \sqsubseteq (a' + b'\mathbb{Z}) \iff (b' \mid \gcd(|a - a'|, b))$$

```
             0+1Z
           /  |     \
0+2Z     1+2Z     1+3Z   ...
 / \          \   /
...  ...     1+6Z

     ...  ...  ...

...  0+0Z 1+0Z 2+0Z  ...
   \    \   |   /    /
           bot
```

$$x \equiv a \mod b$$

Simple congruences, continued

Join:

$$\perp \sqcup (a + b\mathbb{Z}) = a + b\mathbb{Z}$$

$$(a + b\mathbb{Z}) \sqcup \perp = a + b\mathbb{Z}$$

$$(a + b\mathbb{Z}) \sqcup (a' + b'\mathbb{Z}) = (\min(a, a') + \gcd(|a - a'|, b, b')\mathbb{Z})$$

Note: there are no infinite, strictly increasing chains.
However there are infinite, strictly decreasing chains:

$$0 + 1\mathbb{Z} \sqsupset 1 + 2\mathbb{Z} \sqsupset 1 + 6\mathbb{Z} \sqsupset 1 + 12\mathbb{Z} \sqsupset \ldots$$

hence we may need a narrowing…

Simple congruences, continued

Join:
$$\bot \sqcup (a + b\mathbb{Z}) = a + b\mathbb{Z}$$
$$(a + b\mathbb{Z}) \sqcup \bot = a + b\mathbb{Z}$$
$$(a + b\mathbb{Z}) \sqcup (a' + b'\mathbb{Z}) = (\min(a, a') + \gcd(|a - a'|, b, b')\mathbb{Z})$$

Note: there are no infinite, strictly increasing chains.
However there are infinite, strictly decreasing chains:

$$0 + 1\mathbb{Z} \sqsupset 1 + 2\mathbb{Z} \sqsupset 1 + 6\mathbb{Z} \sqsupset 1 + 12\mathbb{Z} \sqsupset \dots$$

hence we may need a narrowing…

Q: what do the elements $0 + 2\mathbb{Z}$ and $1 + 2\mathbb{Z}$ represent together with $\bot$ and $0 + 1\mathbb{Z}$?

Simple congruences, continued

Join:
$$\perp \sqcup (a + b\mathbb{Z}) = a + b\mathbb{Z}$$
$$(a + b\mathbb{Z}) \sqcup \perp = a + b\mathbb{Z}$$
$$(a + b\mathbb{Z}) \sqcup (a' + b'\mathbb{Z}) = (\min(a, a') + \gcd(|a - a'|, b, b')\mathbb{Z})$$

Note: there are no infinite, strictly increasing chains.
However there are infinite, strictly decreasing chains:

$$0 + 1\mathbb{Z} \sqsupset 1 + 2\mathbb{Z} \sqsupset 1 + 6\mathbb{Z} \sqsupset 1 + 12\mathbb{Z} \sqsupset \dots$$

hence we may need a narrowing...

Q: what do the elements $0 + 2\mathbb{Z}$ and $1 + 2\mathbb{Z}$ represent together with $\perp$ and $0 + 1\mathbb{Z}$?

Q: what about $\dots, 0 + 0\mathbb{Z}, 1 + 0\mathbb{Z}, 2 + 0\mathbb{Z}, 3 + 0\mathbb{Z}, \dots$ together with $\perp$ and $0 + 1\mathbb{Z}$?

Simple congruence operations

The arithmetic operators over congruences, e.g., addition:

$$
\begin{aligned}
(a + b\mathbb{Z}) + \perp &= \perp \\
\perp + (a + b\mathbb{Z}) &= \perp \\
(a + b\mathbb{Z}) + (c + d\mathbb{Z}) &= ((a + c) \mod \gcd(b, d)) + \gcd(b, d)\mathbb{Z}
\end{aligned}
$$

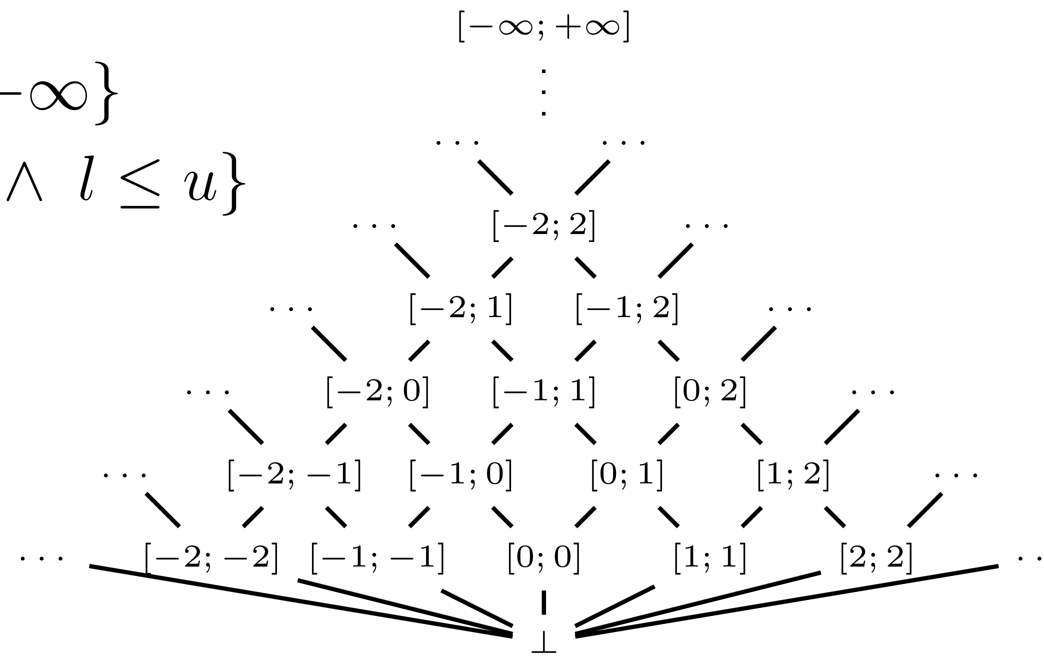
and multiplication:

$$
\begin{aligned}
(a + b\mathbb{Z}) * \perp &= \perp \\
\perp * (a + b\mathbb{Z}) &= \perp \\
(a + b\mathbb{Z}) * (c + d\mathbb{Z}) &= (ac \mod \gcd(ad, bc, bd)) \\
&\qquad + \gcd(ad, bc, bd)\mathbb{Z}
\end{aligned}
$$

Intervals (Moore'66, Cousot-Cousot'76)

$$Interval = \{\bot\} \cup \{[l;u] \mid l \in \mathbb{Z} \cup \{-\infty\} \wedge u \in \mathbb{Z} \cup \{+\infty\} \wedge l \leq u\}$$

$$\langle \wp(\mathbb{Z}); \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle Interval; \sqsubseteq \rangle$$



$$\gamma(\bot) = \emptyset \qquad \alpha(\emptyset) = \bot$$

$$\gamma([a;b]) = \{n \in \mathbb{Z} \mid a \leq n \leq b\} \qquad \alpha(S) = [\min S; \max S]$$

Note: intervals over $\mathbb{R}$ also work, however over $\mathbb{Q}$ the resulting domain is not complete.

Intervals, continued (2/3)

Least upper bounds:

$$
\begin{aligned}
X \sqcup \bot &= X \\
\bot \sqcup Y &= Y \\
[a; b] \sqcup [c; d] &= [\min(a, c); \max(b, d)]
\end{aligned}
$$

Greatest lower bounds:

$$
\begin{aligned}
X \sqcap \bot &= \bot \\
\bot \sqcap Y &= \bot \\
[a; b] \sqcap [c; d] &= \begin{cases} [\max(a, c); \min(b, d)] & \text{if} \quad \max(a, c) \leq \min(b, d) \\ \bot & \text{otherwise} \end{cases}
\end{aligned}
$$

Intervals, continued (3/3)

Interval addition:

$$\begin{aligned} \bot + X &= \bot \\ X + \bot &= \bot \\ [a;b] + [c;d] &= [a+c; b+d] \end{aligned}$$

Widening and narrowing:

$$\bot \mathbin{\triangledown} I = I \qquad\qquad I \mathbin{\triangledown} \bot = I$$

$$[a;b] \mathbin{\triangledown} [c;d] = \left[\begin{cases} -\infty & c < a \\ a & c \geq a \end{cases} ; \begin{cases} +\infty & d > b \\ b & d \leq b \end{cases} \right]$$

$$\bot \mathbin{\vartriangle} I = \bot \qquad\qquad I \mathbin{\vartriangle} \bot = \bot$$

$$[a,b] \mathbin{\vartriangle} [c,d] = \left[\begin{cases} c & a = -\infty \\ a & \text{otherwise} \end{cases} ; \begin{cases} d & b = +\infty \\ b & \text{otherwise} \end{cases} \right]$$

Interval widening example

Widening with $\perp$ yields identity:

$$\perp \nabla [1; 100] = [1; 100]$$

Increasing upper bounds expand to $+\infty$:

$$[1; 100] \nabla [1; 101] = [1; +\infty]$$

Decreasing lower bounds expand to $-\infty$:

$$[1; +\infty] \nabla [0; 102] = [-\infty; +\infty]$$

Convex Polyhedra

Convex Polyhedra (1/2)

We can use inequalities to describe the relationship between numerical variables of a program, e.g.:

$$\mathtt{y} \geq 1 \quad \wedge \quad \mathtt{x} + \mathtt{y} \geq 3 \quad \wedge \quad -\mathtt{x} + \mathtt{y} \leq 1$$

for two variables `x` and `y`.

The inequalities represent a *convex polyhedron*.

These form the abstract values of the *polyhedra* domain, which is a *relational* abstract domain.

```
y ^     /
  |    /
3 |   /
  |  /
2 |  \
  |   \
1 |    ---------------
  |
0 +---------------------->
  0 1 2 3 4 5 6 ...    x
```

Representation (implementation)

Convex polyhedra are represented using *double description* (with variables $X = \{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$):

- a *system of inequalities* (A, B) where A is an $m \times n$ matrix, B is an m vector, and
  $\gamma(A, B) = \{X \mid AX \geq B\}$
- a *system of generators* (V, R) of vertices and rays where $V = \{V_1, \ldots, V_k\}$, $R = \{R_1, \ldots, R_l\}$, and
  $$\gamma(V, R) = \{\Sigma_{i=1}^{k} \lambda_i V_i + \Sigma_{i=1}^{l} \mu_i R_i \mid \lambda_i \geq 0 \ \wedge \ \mu_i \geq 0 \ \wedge \ \Sigma_{i=1}^{k} \lambda_i = 1\}$$

An domain implementation will typically translate back and forth between the two, trying to minimize the number of conversions.

Representation example

For example, we can represent

$$\mathtt{y} \geq 1 \quad \wedge \quad \mathtt{x} + \mathtt{y} \geq 3 \quad \wedge \quad -\mathtt{x} + \mathtt{y} \leq 1$$

as a system of inequalities: $AX \geq B$

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \geq \begin{bmatrix} 1 \\ 3 \\ -1 \end{bmatrix}$$

as a system of generators:

$$V = \{V_0 : (2, 1), V_1 : (1, 2)\}$$
$$R = \{R_0 : (1, 0), R_1 : (1, 1)\}$$

```
y ^       /
  |    _
3 |    /| R1
  |V1/
2 |   \
  |    \   R0
1 |     x-> - - - - - -
  |    V0
0 +---------------------->
  0 1 2 3 4 5 6 ...     x
```

Convex Polyhedra (2/2)

Operations, some of which are easier on one representation, rather than the other:

- $\sqcup$ returns a *convex hull*, which is an over-approximation of the union of two polyhedra.

  Easily expressed as a union of the corresponding generators.

- $\sqcap$ returns the polyhedron representing the intersection of two polyhedra.

  Easily expressed as the conjunction of the two constraint systems.

But there is a catch…

The polyhedra lattice is not complete: there exists strictly infinite chains for which the limit is not in the domain. Example: a disk.

Hence for some sets, e.g., a disk, there is no best abstraction.

As a consequence the abstraction to polyhedra is not a Galois connection.

A possible relaxation is to consider only concretization functions…

Concretization-based abstract interpretation

Proposition. *Assume* $\langle C; \sqsubseteq, \sqcup \rangle$ *is a poset,* $F : C \to C$ *is a continuous function,* $\bot_c \in C$ *such that* $\bot_c \sqsubseteq F(\bot_c)$*, and* $\bigsqcup_{n \in \mathbb{N}} F^n(\bot_c)$ *exists.*

Assume A *is a set,* $\gamma : A \to C$ *is a function,* $\leq$ *is a preorder, defined as:* $a \leq a' \iff \gamma(a) \sqsubseteq \gamma(a')$*,* $\bot_a \in A$ *such that* $\bot_c \sqsubseteq \gamma(\bot_a)$*,* $F^\sharp : A \to A$ *is a monotone function such that* $F \circ \gamma \sqsubseteq \gamma \circ F^\sharp$ *and* $\triangledown$ *is a widening operator.*

Then the upward iteration sequence with widening is ultimately stationary with limit a*, such that* $\mathrm{lfp}\, F \sqsubseteq \gamma(a)$ *and* $F^\sharp(a) \leq a$*.*

Alternative frameworks

If we relax the Galois connection requirement there are other options.

Cousot-Cousot *“Abstract interpretation frameworks”* (JLC92) contains a range of alternative frameworks, like the previous concretization-based one.

As an alternative, Miné suggests a framework based on *partial Galois connections*, in which α is a partial function.

Want more abstractions?

There are many more numerical abstractions, see, e.g., Miné's thesis or this link:

`http://bugseng.com/products/ppl/abstractions`

The *Two Variables per Inequality* (TVPI) domain is a restricted form of polyhedra, only expressing relations between two variables: $a_{ij}\mathtt{x}_i + b_{ij}\mathtt{x}_j \leq c_{ij}$

Miné's *Octagon* domain is another restricted form of polyhedra, also expressing relations between two variables: $\pm\mathtt{x}_i \pm \mathtt{x}_j \leq c_{ij}$

Q: what do we get by restricting to one variable per inequality?

Numerical domains, botanically

ATTRIBUTE INDEPENDENT DOMAINS (NON-RELATIONAL):

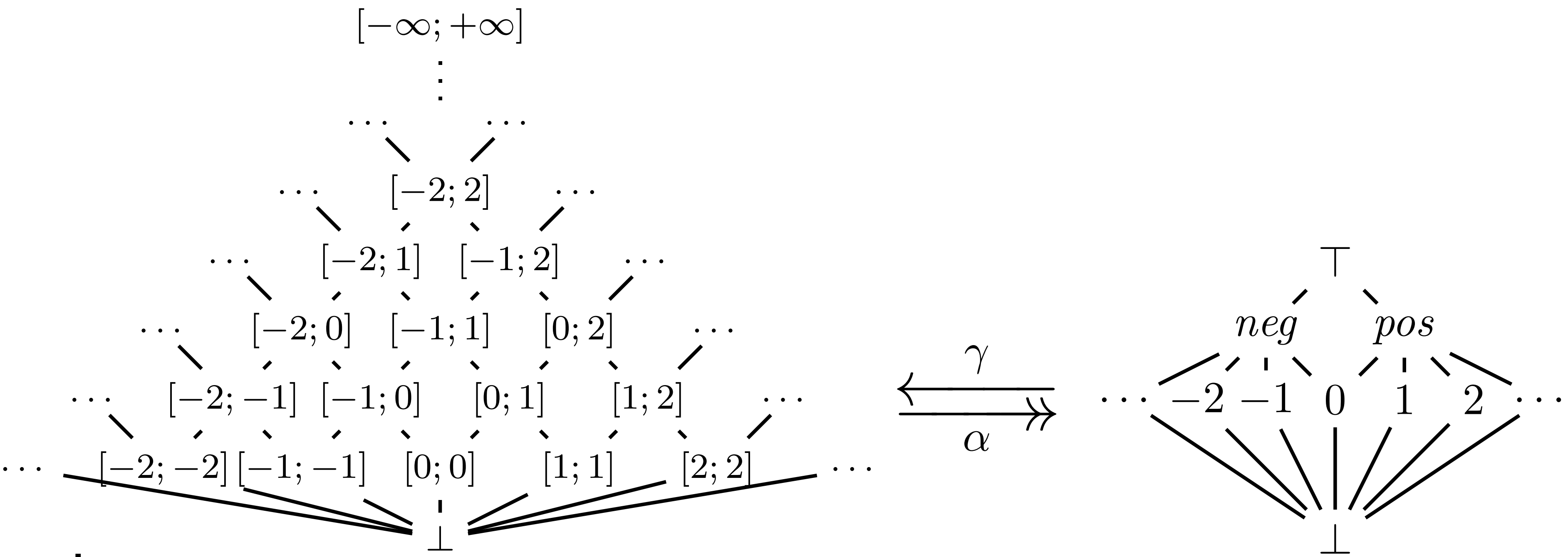
Parity, Sign, Constants, Simple Congruences, Intervals, . . .

RELATIONAL DOMAINS:

Polyhedra, Octagons, TVPI, . . .

A few connections between numerical abstractions

From intervals to a constant/sign combination

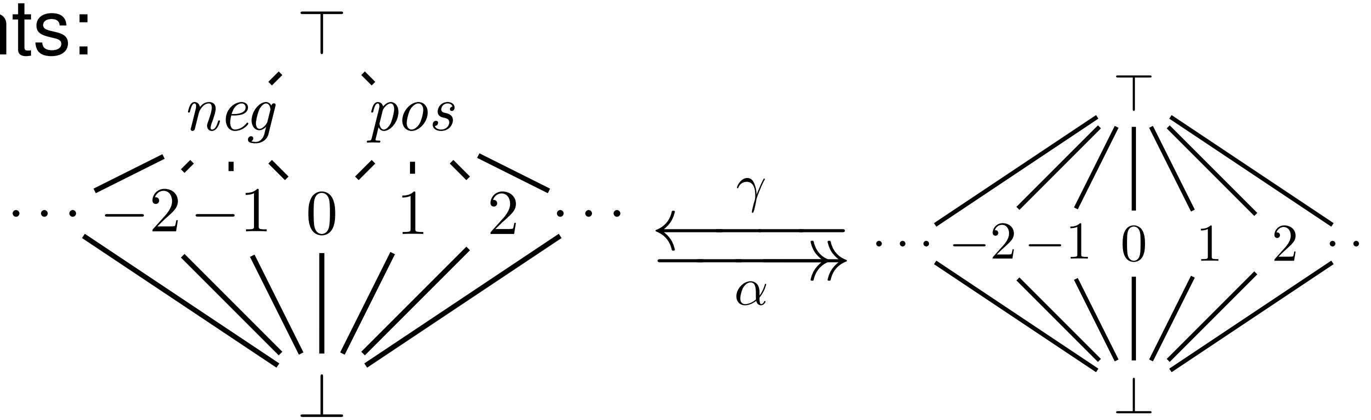


where

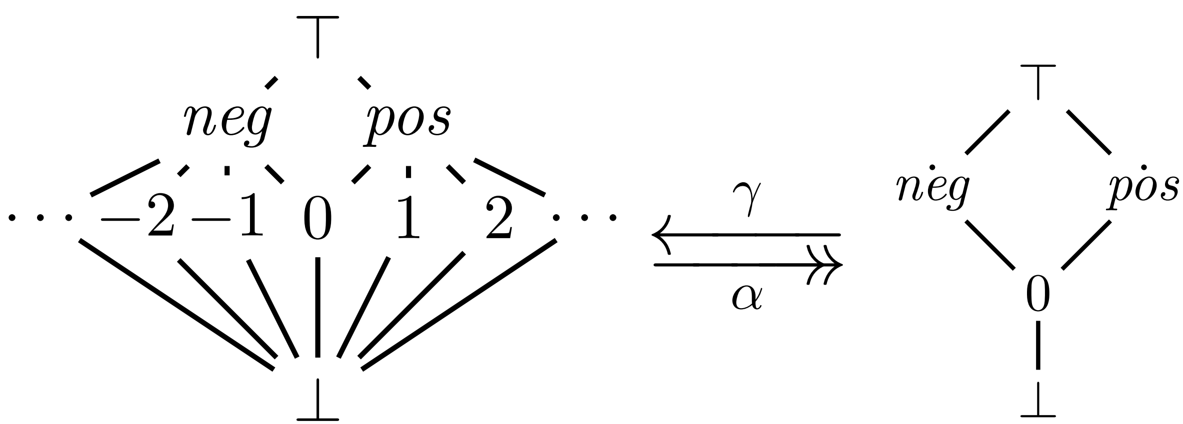
$$
\begin{aligned}
\alpha(\perp) &= \perp \\
\alpha([a; a]) &= a \\
\alpha([a; b]) &= \begin{cases} pos & \text{if } a \geq 0, a \neq b \\ neg & \text{if } b \leq 0, a \neq b \\ \top & \text{otherwise} \end{cases}
\end{aligned}
$$

From the constant/sign combination to . . .

This domain can (naturally) be abstracted to both constants:



and signs:



From the constant/sign combination to . . .

Both can be abstracted into a simple two-point domain:

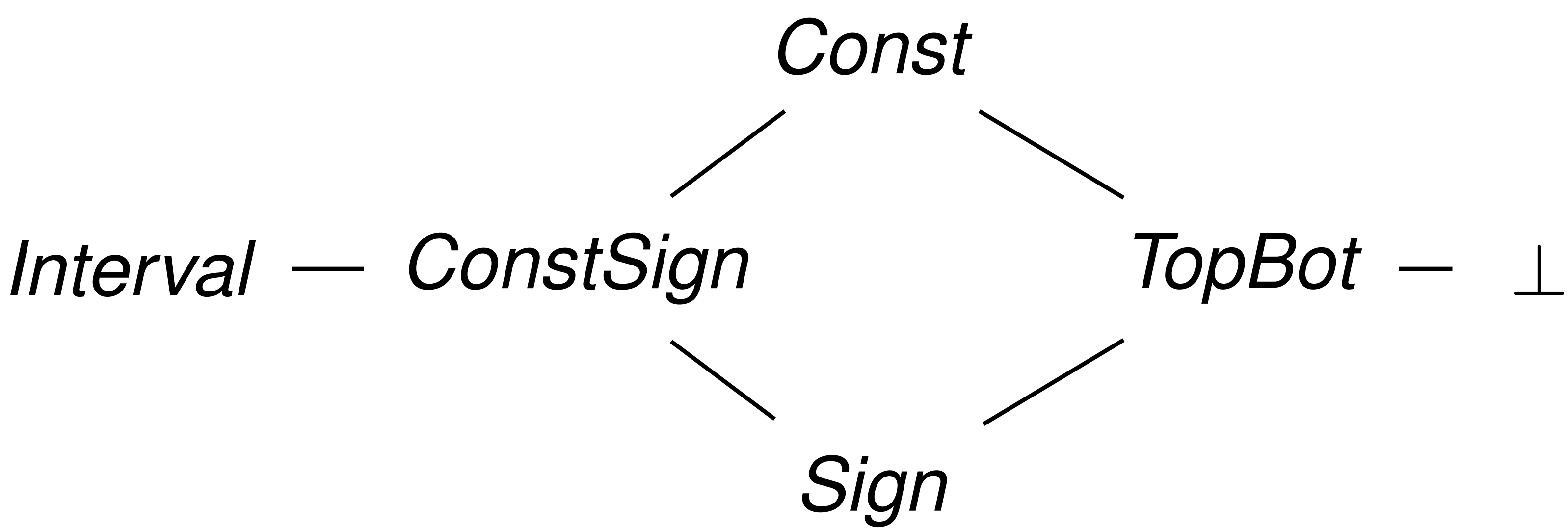
$$\begin{tikzcd}[column sep=small]
 & \top \arrow[dl, dash] \arrow[dr, dash] & & & \\
\dot{neg} \arrow[dr, dash] & & \dot{pos} \arrow[dl, dash] & \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} & \top \arrow[dd, dash] \\
 & 0 \arrow[d, dash] & & & \\
 & \bot & & & \bot
\end{tikzcd}$$

and all the way down to a one-point lattice:

$$\begin{tikzcd}
\top \arrow[d, dash] & & \\
\bot & \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} & \bot
\end{tikzcd}$$

Connection summary

To summarize:



A nice lattice of lattices! $\smile$

The 3 counter machine analysis, revisited

The 3 counter machine analysis, revisited

We arrived at an abstract transition function $\mathtt{F\#}$, but the analysis is the least fixed point lfp of $\mathtt{F\#}$.

Q: Which fixed point theorem(s) from this morning (slide 12+14) are we relying on?

Design choices of the analysis

The resulting analysis associates an abstract memory to each program point:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} PC \to (Parity \times Parity \times Parity)$$

Design choices of the analysis

The resulting analysis associates an abstract memory to each program point:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} PC \to (Parity \times Parity \times Parity)$$

Alternatively we could have abstracted the components separately as follows:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \wp(PC) \times (Parity \times Parity \times Parity)$$

Design choices of the analysis

The resulting analysis associates an abstract memory to each program point:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} PC \to (Parity \times Parity \times Parity)$$

Alternatively we could have abstracted the components separately as follows:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \wp(PC) \times (Parity \times Parity \times Parity)$$

Q: how would you characterize the first analysis?

Design choices of the analysis

The resulting analysis associates an abstract memory to each program point:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} PC \to (Parity \times Parity \times Parity)$$

Alternatively we could have abstracted the components separately as follows:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \wp(PC) \times (Parity \times Parity \times Parity)$$

Q: how would you characterize the first analysis?

Q: how would you characterize the second?

Design choices of the analysis

The resulting analysis associates an abstract memory to each program point:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} PC \rightarrow (Parity \times Parity \times Parity)$$

Alternatively we could have abstracted the components separately as follows:

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \wp(PC) \times (Parity \times Parity \times Parity)$$

Q: how would you characterize the first analysis?

Q: how would you characterize the second?

Q: how would you characterize a first projection?

$$\wp(PC \times \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \wp(PC)$$

Alternative 3 counter machine analyses?

Q: What changes if we want to switch to a different numerical abstraction (intervals, congruences, . . .)?

or rather,

Q: which assumptions about Parity did we rely on?

Alternative 3 counter machine analyses?

Q: What changes if we want to switch to a different numerical abstraction (intervals, congruences, . . .)?

or rather,

Q: which assumptions about Parity did we rely on?

$$\dfrac{\dfrac{\dfrac{}{\wp(\mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \leftrightarrows \wp(\mathbb{N}_0) \times \wp(\mathbb{N}_0) \times \wp(\mathbb{N}_0)} \qquad \dfrac{\dfrac{}{\wp(\mathbb{N}_0) \mathrel{\overset{\longleftarrow}{\twoheadrightarrow}} Par} \quad \dfrac{}{\wp(\mathbb{N}_0) \mathrel{\overset{\longleftarrow}{\twoheadrightarrow}} Par} \quad \dfrac{}{\wp(\mathbb{N}_0) \mathrel{\overset{\longleftarrow}{\twoheadrightarrow}} Par}}{\wp(\mathbb{N}_0) \times \wp(\mathbb{N}_0) \times \wp(\mathbb{N}_0) \leftrightarrows Par \times Par \times Par}}{\wp(\mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \leftrightarrows Par \times Par \times Par}}{PC \to \wp(\mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0) \leftrightarrows PC \to Par \times Par \times Par}$$

Summary

Summary

1. More approximation methods (Cousot-Cousot:JLP92):
   - □ Relational and attribute independent analysis
   - □ Inducing, abstracting, approximating fixed points
   - □ Widening, narrowing
   - □ Forwards/backwards analysis
2. A catalogue of abstractions
   - □ Toolbox abstractions
   - □ Structural abstractions: sums, pairs/tuples, . . .
   - □ Numerical abstractions: constants, intervals, congruences, polyhedra, . . .
   - □ Concretization-based abstract interpretation, briefly

   A retrospective on the 3 counter machine analysis