Theory of Static Program Analysis

Mooly Sagiv

Textbook: Principles of Program Analysis

Chapter 4, Appendix A

CC79, CC92

Content

- Mathematical Background
- Chaotic Iterations
- Examples
- Soundness, Precision and more examples next week

Mathematical Background

- Declaratively define
  - The result of the analysis
  - The exact solution
  - Allow comparison

Posets

- A partial ordering is a binary relation
  $\sqsubseteq : L \times L \to \{false, true\}$
  - For all $l \in L : l \sqsubseteq l$ (Reflexive)
  - For all $l_1, l_2, l_3 \in L : l_1 \sqsubseteq l_2, l_2 \sqsubseteq l_3 \Rightarrow l_1 \sqsubseteq l_3$ (Transitive)
  - For all $l_1, l_2 \in L : l_1 \sqsubseteq l_2, l_2 \sqsubseteq l_1 \Rightarrow l_1 = l_2$ (Anti-Symmetric)
- Denoted by $(L, \sqsubseteq)$
- In program analysis
  - $l_1 \sqsubseteq l_2 \Leftrightarrow l_1$ is more precise than $l_2 \Leftrightarrow$ l_1 represents fewer concrete states than l_2
- Examples
  - Total orders $(N, \leq)$
  - Powersets $(P(S), \subseteq)$
  - Powersets $(P(S), \supseteq)$
  - Even/Odd
  - Constant propagation

Posets

- More notations
  - $l_1 \sqsupseteq l_2 \Leftrightarrow l_2 \sqsubseteq l_1$
  - $l_1 \sqsubset l_2 \Leftrightarrow l_1 \sqsubseteq l_2 \wedge l_1 \neq l_2$
  - $l_1 \sqsupset l_2 \Leftrightarrow l_2 \sqsubset l_1$

Upper and Lower Bounds

- Consider a poset $(L, \sqsubseteq)$
- A subset $L' \subseteq L$ has a lower bound $l \in L$ if for all $l' \in L'$: $l \sqsubseteq l'$
- A subset $L' \subseteq L$ has an upper bound $u \in L$ if for all $l' \in L'$: $l' \sqsubseteq u$
- A greatest lower bound of a subset $L' \subseteq L$ is a lower bound $l_0 \in L$ such that $l \sqsubseteq l_0$ for any lower bound l of L'
- A lowest upper bound of a subset $L' \subseteq L$ is an upper bound $u_0 \in L$ such that $u_0 \sqsubseteq u$ for any upper bound u of L'
- For every subset $L' \subseteq L$:
  - The greatest lower bound of L' is unique if at all exists
    - » $\sqcap L'$ (meet) $a \sqcap b$
  - The lowest upper bound of L' is unique if at all exists
    - » $\sqcup L'$ (join) $a \sqcup b$

Complete Lattices

- A poset $(L, \sqsubseteq)$ is a complete lattice if every subset has least and upper bounds
- $L = (L, \sqsubseteq) = (L, \sqsubseteq, \sqcup, \sqcap, \bot, \top)$
  - $\bot = \bigsqcup \emptyset = \bigsqcap L$
  - $\top = \bigsqcup L = \bigsqcap \emptyset$
- Examples
  - Total orders $(N, \leq)$
  - Powersets $(P(S), \subseteq)$
  - Powersets $(P(S), \supseteq)$
  - Constant propagation

Complete Lattices

- Lemma For every poset $(L, \sqsubseteq)$ the following conditions are equivalent
  - L is a complete lattice
  - Every subset of L has a least upper bound
  - Every subset of L has a greatest lower bound

Cartesian Products

- A complete lattice $(L_1, \sqsubseteq_1) = (L_1, \sqsubseteq, \sqcup_1, \sqcap_1, \perp_1, \top_1)$
- A complete lattice $(L_2, \sqsubseteq_2) = (, \sqsubseteq, \sqcup_2, \sqcap_2, \perp_2, \top_2)$
- Define a Poset $L = (L_1 \times L_2 , \sqsubseteq)$ where
  - $(x_1, x_2) \sqsubseteq (y_1, y_2)$ if
    - » $x_1 \sqsubseteq y_1$ and
    - » $x_2 \sqsubseteq y_2$
- L is a complete lattice

Finite Maps

- A complete lattice $(L_1, \sqsubseteq_1) = (L_1, \sqsubseteq, \sqcup_1, \sqcap_1, \perp_1, \top_1)$
- A finite set V
- Define a Poset $L = (V \rightarrow L_1, \sqsubseteq)$ where
  - $e_1 \sqsubseteq e_2$ if for all $v \in V$
    - » $e_1 v \sqsubseteq e_2 v$
- L is a complete lattice

Chains

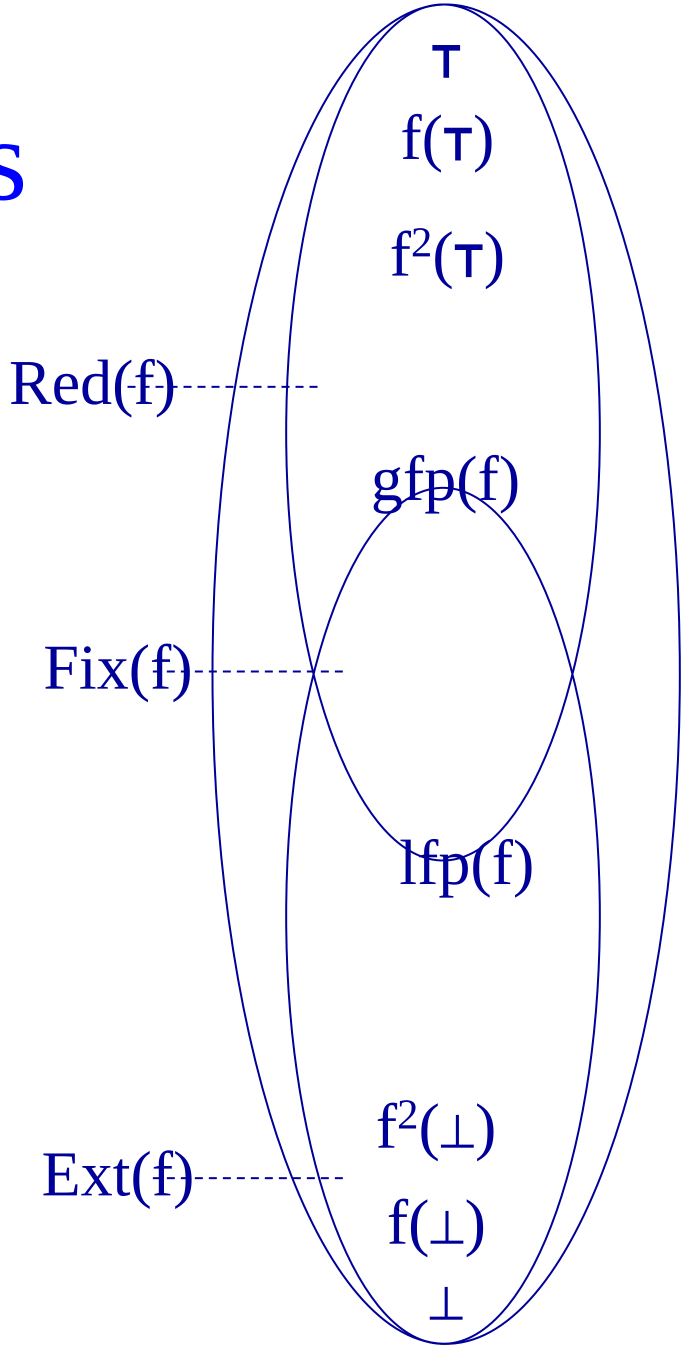
- A subset $Y \subseteq L$ in a poset $(L, \sqsubseteq)$ is a chain if every two elements in Y are ordered
  - For all $l_1, l_2 \in Y$: $l_1 \sqsubseteq l_2$ or $l_2 \sqsubseteq l_1$
- An ascending chain is a sequence of values
  - $l_1 \sqsubseteq l_2 \sqsubseteq l_3 \sqsubseteq \ldots$
- A strictly ascending chain is a sequence of values
  - $l_1 \sqsubset l_2 \sqsubset l_3 \sqsubset \ldots$
- A descending chain is a sequence of values
  - $l_1 \sqsupseteq l_2 \sqsupseteq l_3 \sqsupseteq \ldots$
- A strictly descending chain is a sequence of values
  - $l_1 \sqsupset l_2 \sqsupset l_3 \sqsupset \ldots$
- L has a finite height if every chain in L is finite
- Lemma A poset $(L, \sqsubseteq)$ has finite height if and only if every strictly decreasing and strictly increasing chains are finite

Monotone Functions

- A poset $(L, \sqsubseteq)$
- A function $f: L \rightarrow L$ is monotone if for every $l_1, l_2 \in L$:
  - $l_1 \sqsubseteq l_2 \Rightarrow f(l_1) \sqsubseteq f(l_2)$

Fixed Points

- A monotone function f: $L \rightarrow L$ where $(L, \sqsubseteq, \sqcup, \sqcap, \bot, \top)$ is a complete lattice
- $Fix(f) = \{ l : l \in L, f(l) = l \}$
- $Red(f) = \{ l : l \in L, f(l) \sqsubseteq l \}$
- $Ext(f) = \{ l : l \in L, l \sqsubseteq f(l) \}$
  - $l_1 \sqsubseteq l_2 \Rightarrow f(l_1) \sqsubseteq f(l_2)$
- Tarski's Theorem 1955: if f is monotone then:
  - $lfp(f) = \sqcap Fix(f) = \sqcap Red(f) \in Fix(f)$
  - $gfp(f) = \sqcup Fix(f) = \sqcup Ext(f) \in Fix(f)$



Special Case Finite Height

- A monotone function $f: L \to L$ where $(L, \sqsubseteq, \sqcup, \sqcap, \bot, \top)$ is a complete lattice
- L does not include infinite ascending chains

```
x := ⊥
while changes do
  x := f(x)
```

Chaotic Iterations

- A lattice $L = (L, \sqsubseteq, \sqcup, \sqcap, \perp, \top)$ with finite strictly increasing chains
- $L^n = L \times L \times \ldots \times L$
- A monotone function $\underline{f}: L^n \rightarrow L^n$
- Compute $lfp(\underline{f})$
- The simultaneous least fixed of the system $\{x[i] = \underline{f}_i(x) : 1 \leq i \leq n \}$

$\underline{x} := (\perp, \perp, \ldots, \perp)$

while ($\underline{f}(x) \neq \underline{x}$) do

$\underline{x} := f(\underline{x})$

```
for i :=1  to n do
    x[i] = ⊥
WL = {1, 2, …, n}
while (WL ≠  Ø )  do
   select and remove an element i ∈ WL
   new := f_i(x)
   if (new ≠ x[i]) then
       x[i] := new;
       Add all the indexes that directly depends on i to WL
```

Specialized Chaotic Iterations System of Equations

$S =$

$$\left\{ \begin{array}{l} df_{entry}[s] = \iota \\ df_{entry}[v] = \sqcup \{ f(u, v)\ (df_{entry}[u]) \mid (u, v) \in E \} \end{array} \right\}$$

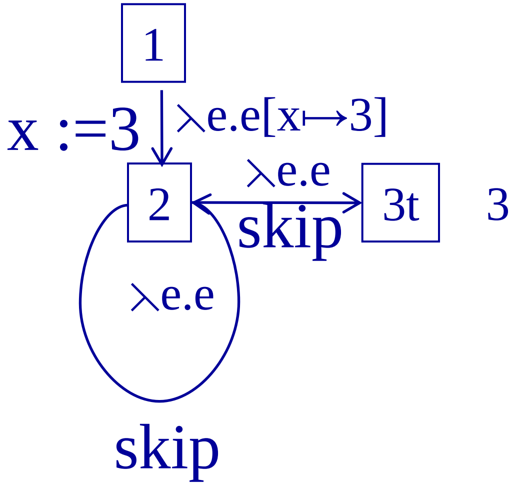
$F_S : L^n \rightarrow L^n$

$$F_S\ (X)[s] = \iota$$

$$F_S(X)[v] = \sqcup \{ f(u, v)(X[u]) \mid (u, v) \in E \}$$

$$lfp(S) = lfp(F_S)$$

Example Constant Propagation



$$DF(1) = [x \mapsto 0]$$

$$DF(2) = DF(1)[x \mapsto 3] \sqcup DF(2)$$

$$DF(3) = DF(2)$$

DF[1]	**DF[2]**	**DF[3]**
$[x \mapsto 0]$	$[x \mapsto 3]$	$[x \mapsto 3]$
$[x \mapsto 0]$	$[x \mapsto ?]$	$[x \mapsto ?]$
$[x \mapsto 7]$	$[x \mapsto 9]$	$[x \mapsto 7]$
$[x \mapsto ?]$	$[x \mapsto 3]$	$[x \mapsto 3]$

Specialized Chaotic Iterations

Chaotic(G(V, E): Graph, s: Node, L: Lattice, ι: L, f: E $\rightarrow$(L $\rightarrow$L)){

for each v in V to n do $df_{entry}[v] := \perp$

df[s] = ι

WL = {s}

while (WL $\neq$ Ø) do

select and remove an element u $\in$ WL

for each v, such that. (u, v) $\in$E do

temp = $f(e)(df_{entry}[u])$

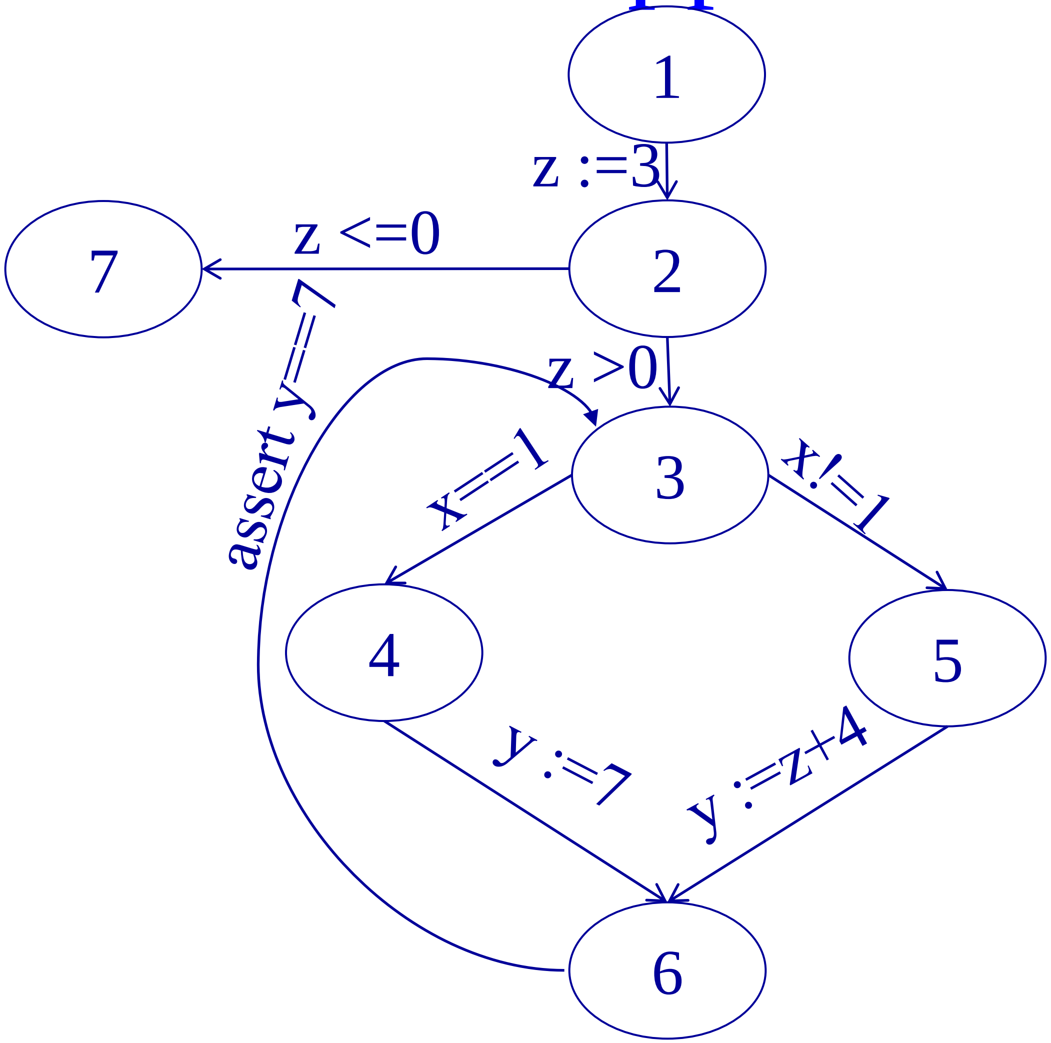
new := $df_{entry}(v) \sqcup$ temp

if (new $\neq df_{entry}[v]$) then

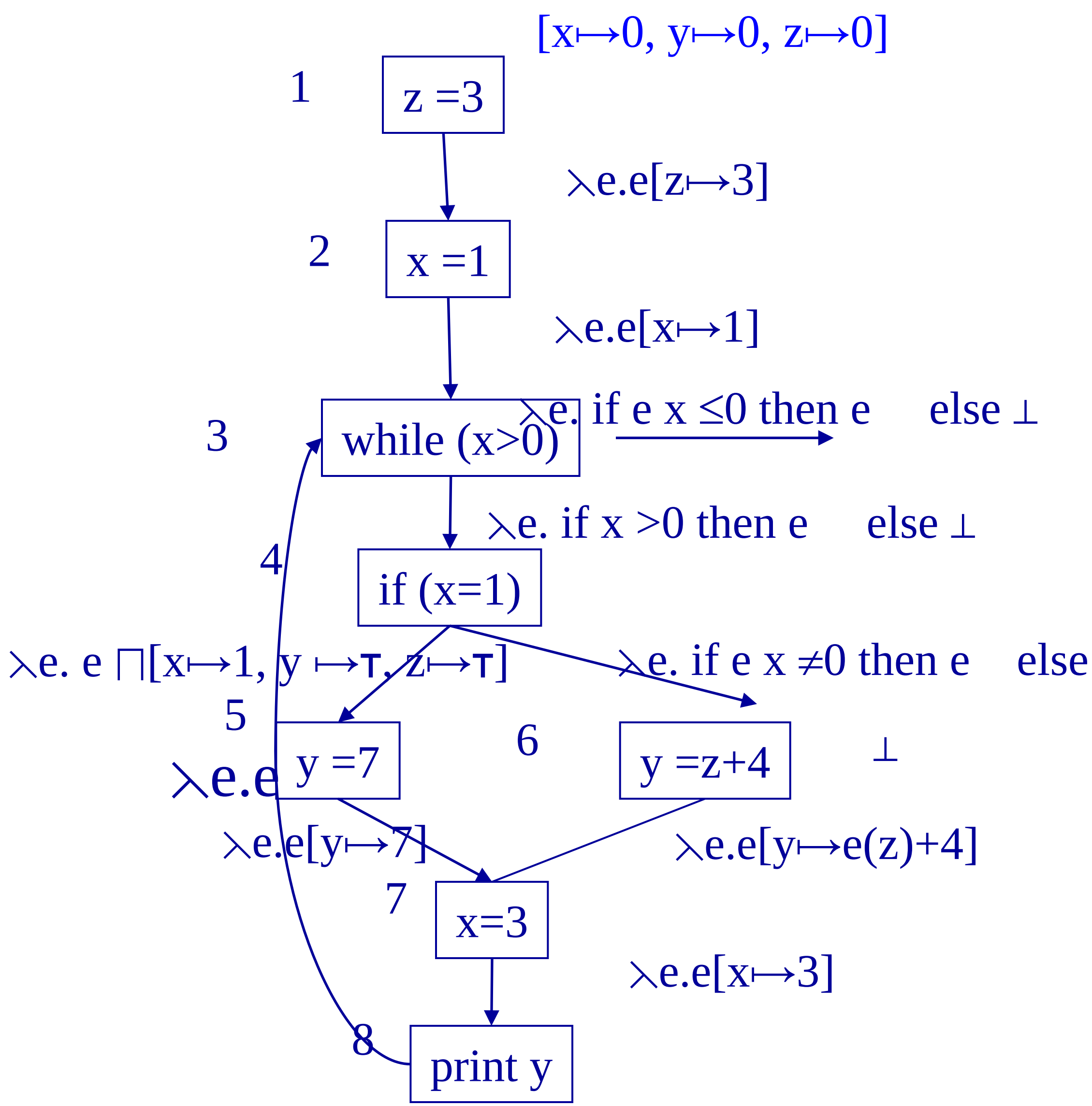
$df_{entry}[v]$:= new;

WL := WL $\cup$\{v\}

Iterative Approximation



N	Value	WL
1	[x⟼?, y⟼?, z⟼?]	{2, 3, 4, 5, 6, 7}
2	[x⟼?, y⟼?, z⟼3]	{ 3, 4, 5, 6, 7}
3	[x⟼?, y⟼?, z⟼3]	{ 4, 5, 6, 7}
4	[x⟼1, y⟼7, z⟼3]	{5, 6, 7}
5	[x⟼?, y⟼7, z⟼3]	{6, 7}
6	[x⟼?, y⟼7, z⟼3]	{7}



WL	$df_{entry}[v]$
{1}	
{2}	df[2]:=[x↦0, y↦0, z↦3]
{3}	df[3]:=[x↦1, y↦0, z↦3]
{4}	df[4]:=[x↦1, y↦0, z↦3]
{5}	df[5]:=[x↦1, y↦0, z↦3]
{7}	df[7]:=[x↦1, y↦7, z↦3]
{8}	df[8]:=[x↦3, y↦7, z↦3]
{3}	df[3]:=[x↦т, y↦т, z↦3]
{4}	df[4]:=[x↦т, y↦т, z↦3]
{5,6}	df[5]:=[x↦1, y↦т, z↦3]
{6,7}	df[6]:=[x↦т, y↦т, z↦3]
{7}	df[7]:=[x↦т, y↦7, z↦3]

Complexity of Chaotic Iterations

- Parameters:
  - n the number of CFG nodes
  - k is the maximum outdegree of edges
  - A lattice of height h
  - c is the maximum cost of
    - applying $f_{(e)}$
    - $\sqcup$
    - L comparisons
- Complexity
  O(n * h * c * k)

Soundness

- Every detected constant is indeed such
- Every error will be detected
- The least fixed points represents all occurring runtime states

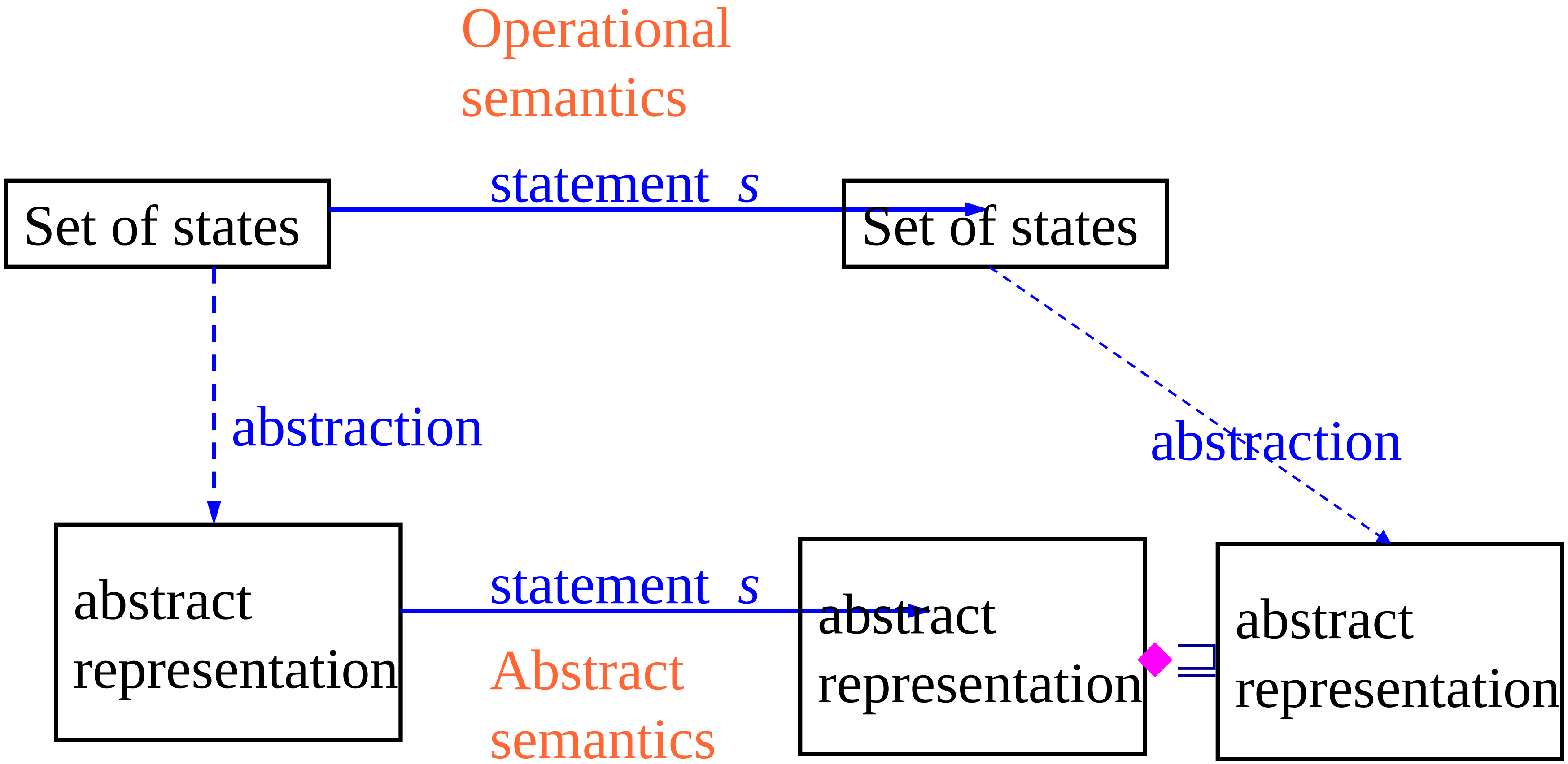
Completeness

- Every constant is indeed detected as such
- Every detected error is real
- Every state represented by the least fixed is reachable for some input

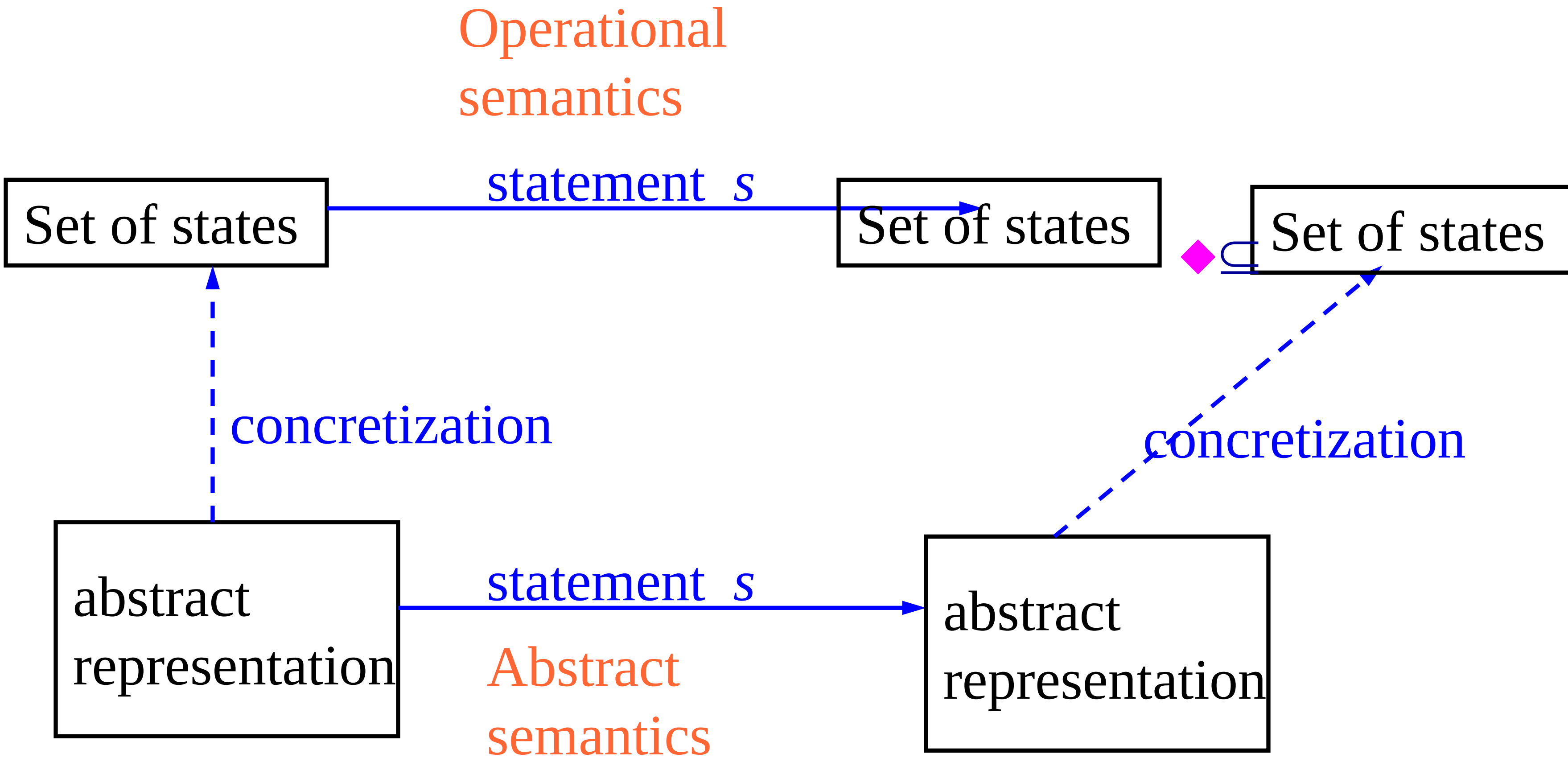
The Abstract Interpretation Technique (Cousot & Cousot)

- The foundation of program analysis
- Defines the meaning of the information computed by static tools
- A mathematical framework
- Allows proving that an analysis is sound in a local way
- Identify design bugs
- Understand where precision is lost
- New analysis from old
- Not limited to certain programming style

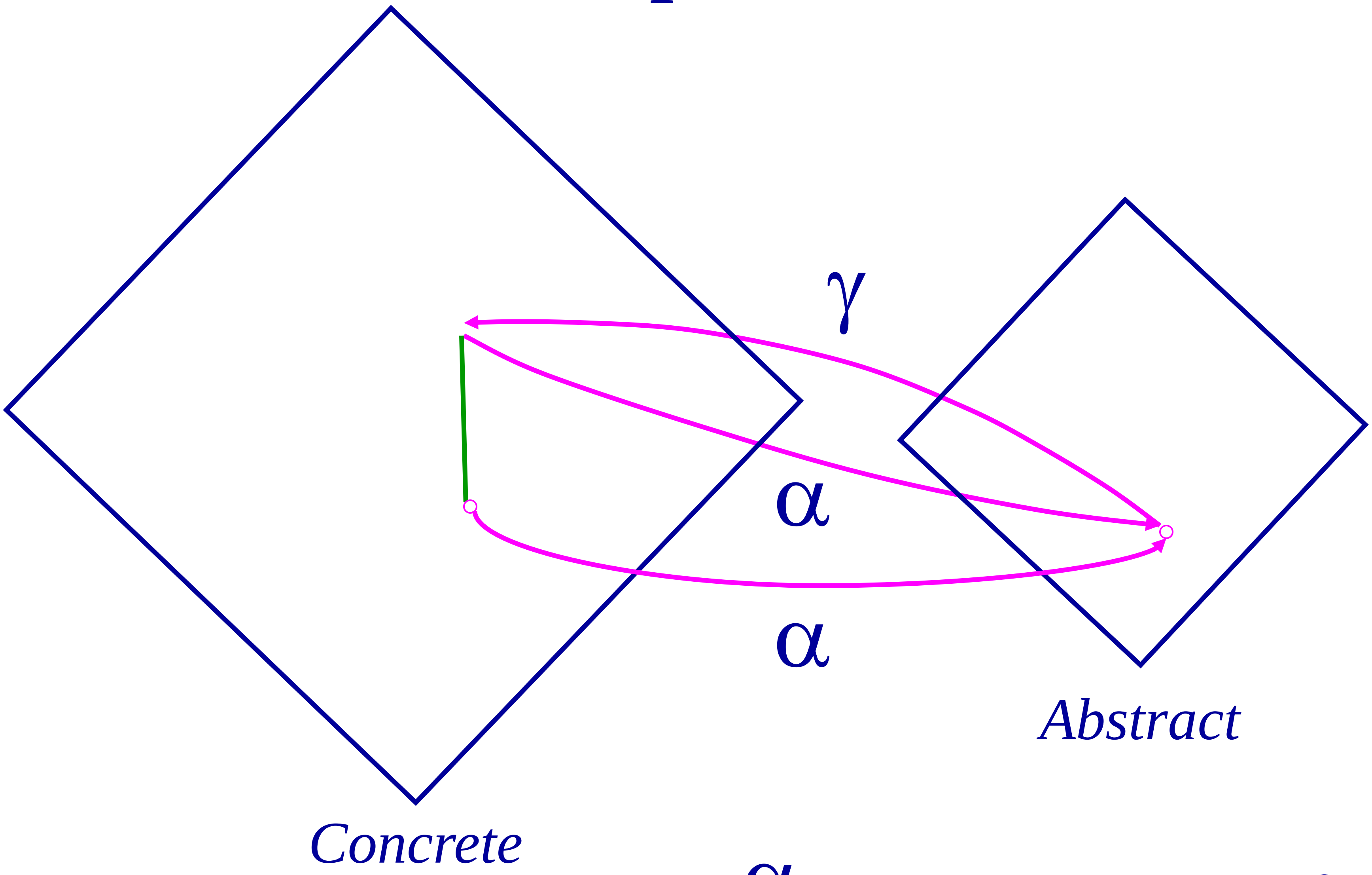
Abstract (Conservative) interpretation



Abstract (Conservative) interpretation



Abstract Interpretation



Sets of stores $\xrightarrow{\alpha}$ *Descriptors of sets of stores*

Galois Connections

- Lattices C and A and functions α: C $\rightarrow$A and γ: A $\rightarrow$C
- The pair of functions (α, γ) form Galois connection if
  - α and γ are monotone
  - $\forall$ a $\in$ A
    - » $\alpha(\gamma(a)) \sqsubseteq a$
  - $\forall$ c $\in$ C
    - » $c \sqsubseteq \gamma(\alpha(C))$
- Alternatively if:

  $\forall$ c $\in$ C

  $\forall$ a $\in$ A

  $$\alpha(c) \sqsubseteq a \text{ iff } c \sqsubseteq \gamma(a)$$

- α and γ uniquely determine each other

The Abstraction Function (CP)

- Map collecting states into constants
- The abstraction of an individual state

  $\beta_{CP}:[Var_* \rightarrow Z] \rightarrow [Var_* \rightarrow Z \cup \{\bot, \top\}]$

  $\beta_{CP}(\sigma) = \sigma$
- The abstraction of set of states

  $\alpha_{CP}:P([Var_* \rightarrow Z]) \rightarrow [Var_* \rightarrow Z \cup \{\bot, \top\}]$

  $\alpha_{CP}(CS) = \sqcup \{ \beta_{CP}(\sigma) \mid \sigma \in CS\} = \sqcup \{\sigma \mid \sigma \in CS\}$
- Soundness

  $\alpha_{CP}(Reach(v)) \sqsubseteq df(v)$
- Completeness

The Concretization Function

- Map constants into collecting states
- The formal meaning of constants
- The concretization

  γ_{CP}: $[Var_* \rightarrow Z \cup \{\perp, \top\}] \rightarrow P([Var_* \rightarrow Z])$

  $\gamma_{CP}(df) = \{\sigma | \beta_{CP}(\sigma) \sqsubseteq df\} = \{\sigma \mid \sigma \sqsubseteq df\}$
- Soundness

  $Reach(v) \subseteq \gamma_{CP}(df(v))$
- Completeness

Galois Connection Constant Propagation

- α_{CP} is monotone
- γ_{CP} is monotone
- $\forall\ df \in [Var_* \rightarrow Z \cup \{\bot, \top\}]$
  - $\alpha_{CP}(\gamma_{CP}(df)) \sqsubseteq df$
- $\forall\ c \in P([Var_* \rightarrow Z])$
  - $c_{CP} \sqsubseteq \gamma_{CP}(\alpha_{CP}(C))$

Upper Closures

- Define abstractions on sets of concrete states
- $\uparrow$: $P(\Sigma) \rightarrow P(\Sigma)$ such that
  - $\uparrow$ is monotone, i.e., $X \subseteq Y \rightarrow \uparrow X \subseteq \uparrow Y$
  - $\uparrow$ is extensive, i.e., $\uparrow X \supseteq X$
  - $\uparrow$ is closure, i.e., $\uparrow(\uparrow X) = \uparrow X$
- Every Galois connection defines an upper closure

Proof of Soundness

- Define an “appropriate” operational semantics
- Define “collecting” structural operational semantics
- Establish a Galois connection between collecting states and abstract states
- (Local correctness) Show that the abstract interpretation of every atomic statement is sound w.r.t. the collecting semantics
- (Global correctness) Conclude that the analysis is sound

Collecting Semantics

- The input state is not known at compile-time
- "Collect" all the states for all possible inputs to the program
- No lost of precision

A Simple Example Program

$\{[x \mapsto 0, y \mapsto 0, z \mapsto 0]\}$

z = 3 $\{[x \mapsto 0, y \mapsto 0, z \mapsto 3]\}$

x = 1 $\{[x \mapsto 1, y \mapsto 0, z \mapsto 3]\}$

while (x > 0) ($\{[x \mapsto 1, y \mapsto 0, z \mapsto 3], [x \mapsto 3, y \mapsto 0, z \mapsto 3],\}$

if (x = 1) then y = 7 $\{[x \mapsto 1, y \mapsto 7, z \mapsto 3], [x \mapsto 3, y \mapsto 7, z \mapsto 3]\}$

else y = z + 4

$\{[x \mapsto 1, y \mapsto 7, z \mapsto 3], [x \mapsto 3, y \mapsto 7, z \mapsto 3]\}$

x = 3

$\{[x \mapsto 3, y \mapsto 7, z \mapsto 3]\}$

print y

$\{[x \mapsto 3, y \mapsto 7, z \mapsto 3]\}$

)

Another Example

```
x= 0
while (true) do
   x = x +1
```

An “Iterative” Definition

- Generate a system of monotone equations
- The least solution is well-defined
- The least solution is the collecting interpretation
- But may not be computable

Equations Generated for Collecting Interpretation

- Equations for elementary statements
  - [skip]

    $CS_{exit}(1) = CS_{entry}(l)$
  - [b]

    $CS_{exit}(1) = \{\sigma: \sigma \in CS_{entry}(l), [\![b]\!]\sigma=tt\}$
  - [x := a]

    $CS_{exit}(1) = \{(s[x \mapsto \mathbf{A}[\![a]\!]s]) \mid s \in CS_{entry}(l)\}$
- Equations for control flow constructs

  $CS_{entry}(l) = \cup\, CS_{exit}(l')$ l' immediately precedes l in the control flow graph
- An equation for the entry

  $CS_{entry}(1) = \{\sigma \mid \sigma \in Var_{*} \rightarrow Z\}$

Specialized Chaotic Iterations System of Equations (Collecting Semantics)

S =

$CS_{entry}[s] = \{\sigma_0\}$

$CS_{entry}[v] = \cup\{f(e)(CS_{entry}[u]) \mid (u, v) \in E\}$

where $f(e) = \lambda X. \{[\![st(e)]\!]\, \sigma \mid \sigma \in X\}$ for atomic statements

$f(e) = \lambda X.\{\sigma \mid [\![b(e)]\!]\, \sigma = tt\}$

$F_S: L^n \rightarrow L^n$

$F_s(X)[v] = \cup\{f(e)[u] \mid (u, v) \in E\}$

$lfp(S) = lfp(F_S)$

The Least Solution

- 2n sets of equations
  $CS_{entry}(1), \ldots, CS_{entry}(n), CS_{exit}(1), \ldots, CS_{exit}(n)$
- Can be written in vectorial form

$$\overrightarrow{CS} = F_{cs}(\overrightarrow{CS})$$

- The least solution $lfp(F_{cs})$ is well-defined
- Every component is minimal
- Since F_{cs} is monotone such a solution always exists
- $CS_{entry}(v) = \{s | \exists s_0 | \langle P, s_0 \rangle \Rightarrow^* (S', s)),$
  $init(S')=v\}$
- Simplify the soundness criteria

Specialized Chaotic Iterations System of Equations (Collecting Semantics)

S =

$$\left\{ \begin{array}{l} CS_{entry}[s] = \{\sigma_0\} \\ CS_{entry}[v] = \cup\{f(e)(CS_{entry}[u]) \mid (u, v) \in E\} \\ \text{where } f(e) = \lambda X.\ \{[\![st(e)]\!]\ \sigma \mid \sigma \in X\} \text{ for atomic statements} \end{array} \right\}$$

$$f(e) = \lambda X.\{\sigma \mid [\![b(e)]\!]\ \sigma = tt\}$$

$$F_S : L^n \to L^n$$

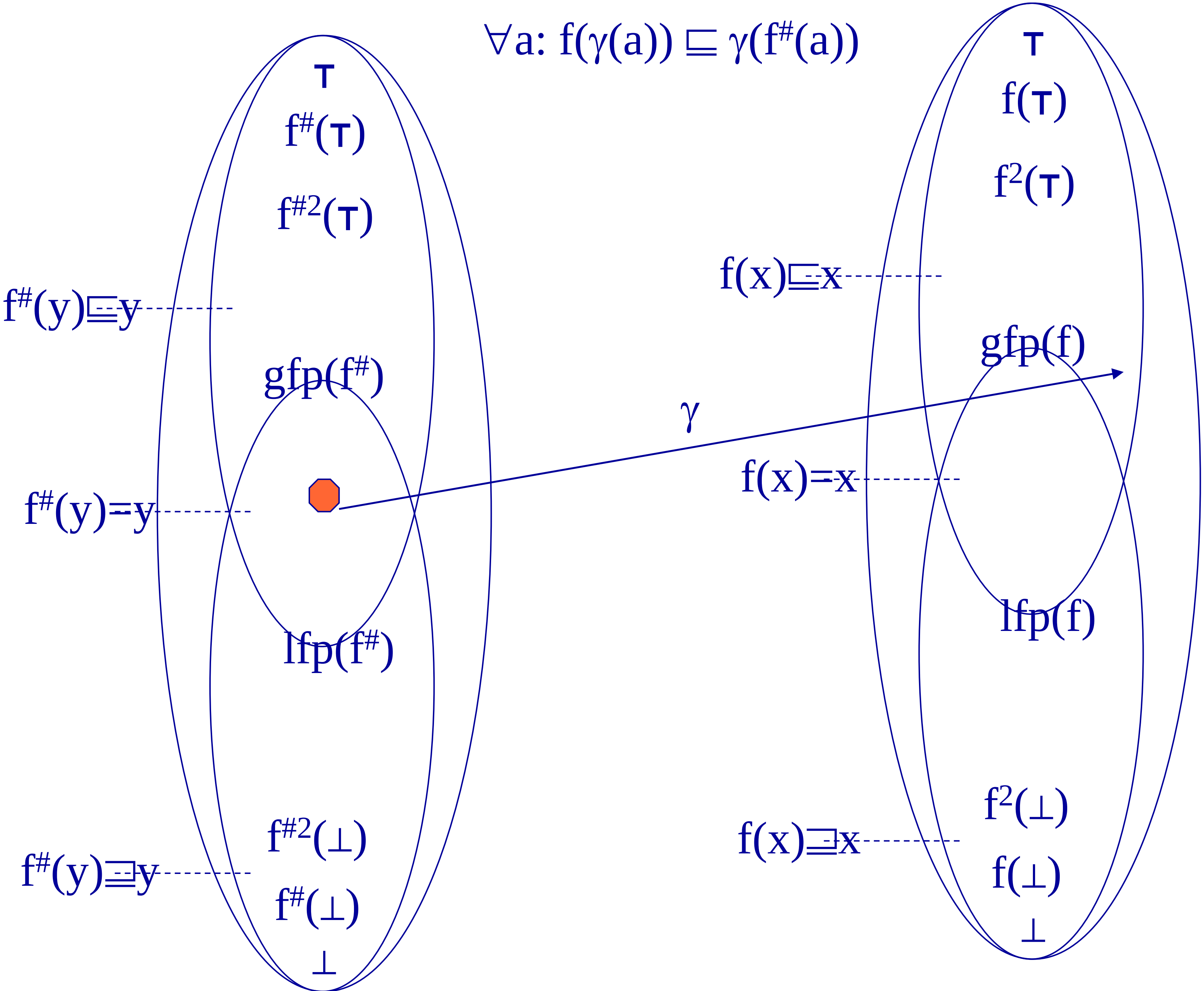
$$F_s(X)[v] = \cup\{f(e)[u] \mid (u, v) \in E\}$$

$$lfp(S) = lfp(F_S)$$

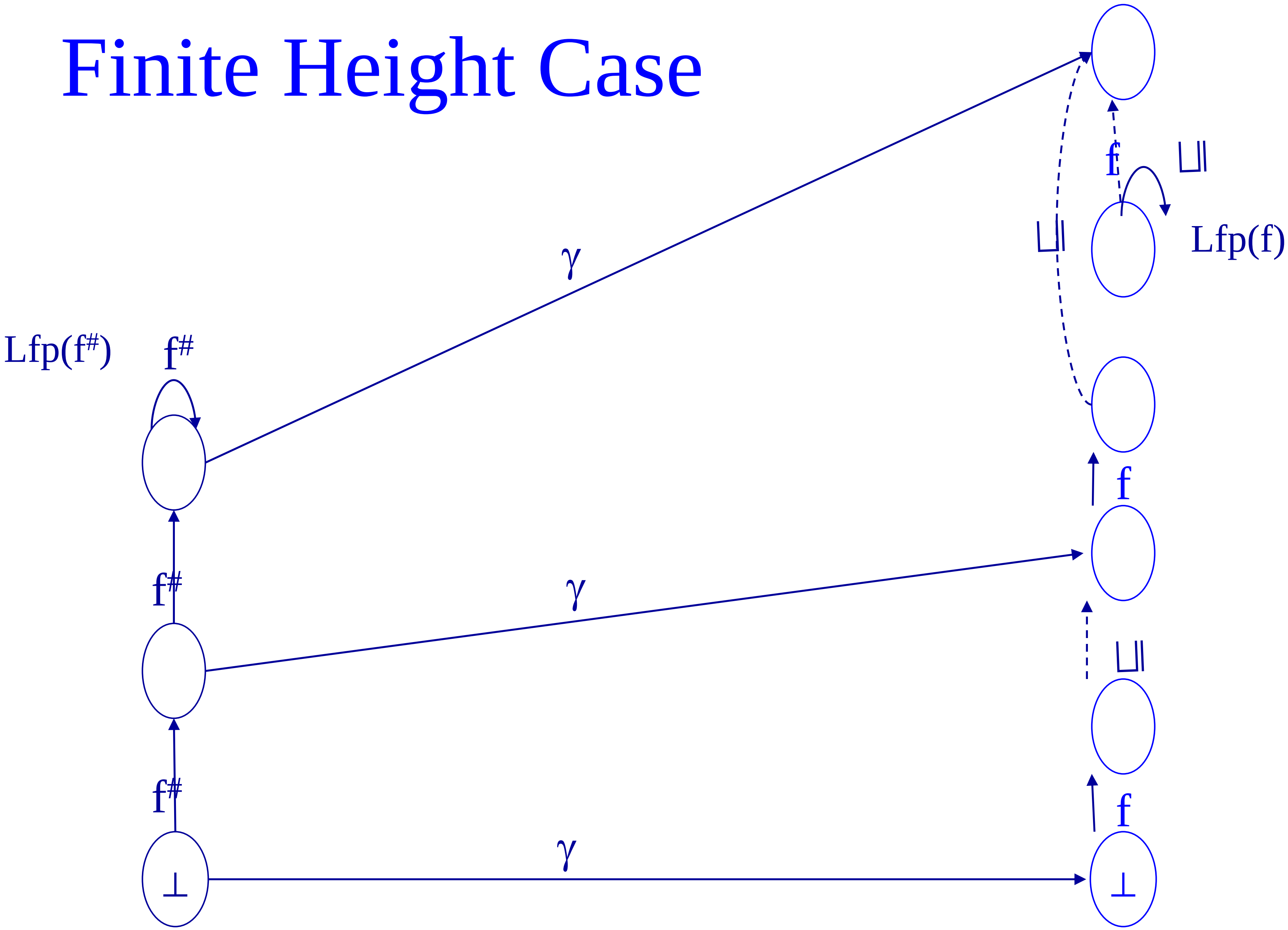
The Least Solution

- 2n sets of equations
  $CS_{entry}(1), \ldots, CS_{entry}(n), CS_{exit}(1), \ldots, CS_{exit}(n)$
- Can be written in vectorial form
  $$\overrightarrow{CS} = F_{cs}(\overrightarrow{CS})$$
- The least solution $lfp(F_{cs})$ is well-defined
- Every component is minimal
- Since F_{cs} is monotone such a solution always exists
- $CS_{entry}(v) = \{s|\exists s_0| <P, s_0> \Rightarrow^* (S', s)),$
  $\mathit{init}(S')=v\}$
- Simplify the soundness criteria

$\forall a: f(\gamma(a)) \sqsubseteq \gamma(f^{\#}(a))$



Finite Height Case



Soundness Theorem(2)

1. Let (α, γ) form Galois connection from C to A
2. $f: C \to C$ be a monotone function
3. $f^{\#}: A \to A$ be a monotone function
4. $\forall c \in C: \alpha(f(c)) \sqsubseteq f^{\#}(\alpha(c))$

$$\alpha(lfp(f)) \sqsubseteq lfp(f^{\#})$$

$$lfp(f) \sqsubseteq \gamma(lfp(f^{\#}))$$

Soundness Theorem(3)

1. Let (α, γ) form Galois connection from C to A
2. $f: C \rightarrow C$ be a monotone function
3. $f^{\#} : A \rightarrow A$ be a monotone function
4. $\forall a \in A: \alpha(f(\gamma(a))) \sqsubseteq f^{\#}(a)$

$$\alpha(\mathrm{lfp}(f)) \sqsubseteq \mathrm{lfp}(f^{\#})$$

$$\mathrm{lfp}(f) \sqsubseteq \gamma(\mathrm{lfp}(f^{\#}))$$

Proof of Soundness (Summary)

- Define an “appropriate” structural operational semantics
- Define “collecting” structural operational semantics
- Establish a Galois connection between collecting states and reaching definitions
- (Local correctness) Show that the abstract interpretation of every atomic statement is sound w.r.t. the collecting semantics
- (Global correctness) Conclude that the analysis is sound

Completeness

$$\alpha(\mathrm{lfp}(f)) = \mathrm{lfp}(f^{\#})$$

$$\mathrm{lfp}(f) = \gamma(\mathrm{lfp}(f^{\#}))$$

Constant Propagation

- β: [Var $\rightarrow$ Z] $\rightarrow$ [Var $\rightarrow$ Z$\cup\{\top, \bot\}$]
  - $\beta(\sigma) = (\sigma)$
- α: P([Var $\rightarrow$ Z]) $\rightarrow$ [Var $\rightarrow$ Z$\cup\{\top, \bot\}$]
  - $\alpha(X) = \sqcup \{\beta(\sigma) \mid \sigma \in X\} = \sqcup \{\sigma \mid \sigma \in X\}$
- γ:[Var $\rightarrow$ Z $\cup\{\top, \bot\}$] $\rightarrow$ P([Var $\rightarrow$ Z])
  - $\gamma(\sigma^{\#}) = \{\sigma \mid \beta(\sigma) \sqsubseteq \sigma^{\#} \} = \{\sigma \mid \sigma \sqsubseteq \sigma^{\#} \}$
- Local Soundness
  - $[\![st]\!]^{\#}(\sigma^{\#}) \sqsupseteq \alpha(\{ [\![st]\!]\, \sigma \mid \sigma \in \gamma(\sigma^{\#}) = \sqcup \{ [\![st]\!]\, \sigma \mid \sigma \sqsubseteq \sigma^{\#} \}$
- Optimality (Induced)
  - $[\![st]\!]^{\#}(\sigma^{\#}) = \alpha(\{ [\![st]\!]\, \sigma \mid \sigma \in \gamma\, (\sigma^{\#})\} = \sqcup \{ [\![st]\!]\, \sigma \mid \sigma \sqsubseteq \sigma^{\#} \}$
- Soundness
- Completeness

Summary

- Abstract interpretation Connects Abstract and Concrete Semantics
- Galois Connection
- Local Correctness
- Global Correctness

Conclusions

- Chaotic iterations is a powerful technique
- Easy to implement
- Rather precise
- But expensive
  - More efficient methods exist for structured programs