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
  \[ \sqsubseteq : L \times L \rightarrow \{\text{false, true}\} \]
  - 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
  - \( l_1 \sqsubseteq l_2 \Leftrightarrow l_1 \text{ is more precise than } l_2 \Leftrightarrow l_1 \text{ represents fewer concrete states than } l_2 \)
- Examples
  - Total orders \((N, \leq)\)
  - Powersets \((P(S), \subseteq)\)
    - Powersets \((P(S), \supseteq)\)
  - Even/Odd
  - Constant propagation
Posets

- More notations
  - $l_1 \equiv l_2 \iff l_2 \sqsubseteq l_1$
  - $l_1 \subset l_2 \iff l_1 \sqsubseteq l_2 \land l_1 \neq l_2$
  - $l_1 \supset l_2 \iff l_2 \sqsubset l_1$
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.
- $L = (L, \sqsubseteq) = (L, \sqsubseteq, \sqcup, \sqcap, \bot, \top)$
  - $\bot = \sqcup \emptyset = \sqcap L$
  - $\top = \sqcup L = \sqcap \emptyset$

- Examples
  - Total orders $(\mathbb{N}, \leq)$
  - Powersets $(\mathcal{P}(S), \subseteq)$
  - Powersets $(\mathcal{P}(S), \supseteq)$
  - 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
  \((L_1, \sqsubseteq_1) = (L_1, \sqsubseteq, \sqcup_1, \sqcap_1, \bot_1, \top_1)\)

◆ A complete lattice
  \((L_2, \sqsubseteq_2) = (L_2, \sqsubseteq, \sqcup_2, \sqcap_2, \bot_2, \top_2)\)

◆ 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
  \((L_1, \subseteq_1) = (L_1, \subseteq, \cup_1, \cap_1, \bot_1, T_1)\)

- A finite set \(V\)

- Define a Poset \(L = (V \rightarrow L_1, \subseteq)\) where
  - \(e_1 \subseteq e_2\) if for all \(v \in V\)
    - \(e_1v \subseteq e_2v\)

- \(L\) 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
  - $l_1 \subset l_2 \subset l_3 \subset \ldots$

- A descending chain is a sequence of values
  - $l_1 \supset l_2 \supset l_3 \supset \ldots$

- A strictly descending chain is a sequence of values
  - $l_1 \supset l_2 \supset l_3 \supset \ldots$

- $L$ 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 \((L, \subseteq)\)

- A function \(f: L \rightarrow L\) is monotone if for every \(l_1, l_2 \in L:\)
  - \(l_1 \subseteq l_2 \Rightarrow f(l_1) \subseteq f(l_2)\)
Fixed Points

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

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

\[
x := \bot
\]

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 \( f: L^n \rightarrow L^n \)
◆ Compute \( \text{lfp}(f) \)
◆ The simultaneous least fixed of the system \( \{x[i] = f_i(x) : 1 \leq i \leq n \} \)

\[
\begin{align*}
\text{for } i := 1 \text{ to } n \text{ do} \\
\quad x[i] = \perp
\end{align*}
\]

\[
\begin{align*}
WL &= \{1, 2, \ldots, n\} \\
x &:= (\perp, \perp, \ldots, \perp)
\end{align*}
\]

while (\( WL \neq \emptyset \)) do

\[
\begin{align*}
\text{select and remove an element } i \in WL \\
\text{new} &:= f_i(x) \\
\text{if } (\text{new} \neq x[i]) \text{ then} \\
\quad x[i] &:= \text{new};
\end{align*}
\]

Add all the indexes that directly depends on \( i \) to \( WL \)
Specialized Chaotic Iterations
System of Equations

\[
S = \begin{cases} 
\text{df}_{\text{entry}}[s] = \tau \\
\text{df}_{\text{entry}}[v] = \bigsqcup \{ f(u, v) \left( \text{df}_{\text{entry}}[u] \right) \mid (u, v) \in E \} 
\end{cases}
\]

\[
F_S : L^n \rightarrow L^n
\]

\[
F_S(X)[s] = \tau
\]

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

\[
lfp(S) = lfp(F_S)
\]
Example Constant Propagation

\[ \text{DF}(1) = [x \mapsto 0] \]
\[ \text{DF}(2) = \text{DF}(1)[x \mapsto 3] \sqcup \text{DF}(2) \]
\[ \text{DF}(3) = \text{DF}(2) \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[x \mapsto 0]</td>
<td>[x \mapsto 3]</td>
<td>[x \mapsto 3]</td>
</tr>
<tr>
<td>[x \mapsto 0]</td>
<td>[x \mapsto ?]</td>
<td>[x \mapsto ?]</td>
</tr>
<tr>
<td>[x \mapsto 7]</td>
<td>[x \mapsto 9]</td>
<td>[x \mapsto 7]</td>
</tr>
<tr>
<td>[x \mapsto ?]</td>
<td>[x \mapsto 3]</td>
<td>[x \mapsto 3]</td>
</tr>
</tbody>
</table>
Specialized Chaotic Iterations

**Chaotic(G(V, E):** Graph, s: Node, L: Lattice, \( \iota: L \rightarrow (L \rightarrow L) \))

for each \( v \) in \( V \) to \( n \) do \( df_{entry}[v] := \bot \)

\( df[s] = \iota \)

\( WL = \{s\} \)

while (\( WL \neq \emptyset \)) do

select and remove an element \( u \in WL \)

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

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

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

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

\( df_{entry}[v] := new; \)

\( WL := WL \cup \{v\} \)
Iterative Approximation

N | Value | WL
---|---|---
1 | \([x\mapsto?, y\mapsto?, z\mapsto?]\) | \{2, 3, 4, 5, 6, 7\}
2 | \([x\mapsto?, y\mapsto?, z\mapsto]\) | \{3, 4, 5, 6, 7\}
3 | \([x\mapsto?, y\mapsto?, z\mapsto]\) | \{4, 5, 6, 7\}
4 | \([x\mapsto1, y\mapsto7, z\mapsto]\) | \{5, 6, 7\}
5 | \([x\mapsto?, y\mapsto7, z\mapsto]\) | \{6, 7\}
6 | \([x\mapsto?, y\mapsto7, z\mapsto]\) | \{7\}
z = 3
x = 1
while (x > 0)
    if (x = 1)
        y = 7
        y = z + 4

print y
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 \( f(e) \)
    » \( \square \)
    » L comparisons

◆ Complexity
  \( O(n \times h \times c \times k) \)
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

Operational semantics

Set of states

abstraction

abstract representation

statement $s$

Abstract semantics

Set of states

abstraction

abstract representation

statement $s$
Abstract (Conservative) interpretation

Set of states

abstract representation

concretization

Statement $s$

Operational semantics

Set of states

abstract representation

Concretization

Abstract semantics

Set of states

Set of states
Abstract Interpretation

Concrete
Sets of stores

Abstract
Descriptors of sets of stores

\(\alpha\)

\(\gamma\)
Galois Connections

- Lattices $C$ and $A$ and functions $\alpha: C \rightarrow A$ and $\gamma: A \rightarrow C$

- The pair of functions $(\alpha, \gamma)$ form Galois connection if
  - $\alpha$ and $\gamma$ are monotone
  - $\forall a \in A$  
    $\quad \Rightarrow \quad \alpha(\gamma(a)) \subseteq a$
  - $\forall c \in C$  
    $\quad \Rightarrow \quad c \subseteq \gamma(\alpha(C))$

- Alternatively if:  
  $\forall c \in C$  
  $\forall a \in A$  
  $\quad \alpha(c) \subseteq a \quad$ iff $c \subseteq \gamma(a)$

- $\alpha$ and $\gamma$ uniquely determine each other
The Abstraction Function (CP)

- Map collecting states into constants
- The abstraction of an individual state
  \[ \beta_{\text{CP}}: [\text{Var}_* \rightarrow \mathbb{Z}] \rightarrow [\text{Var}_* \rightarrow \mathbb{Z} \cup \{ \bot, \top \}] \]
  \[ \beta_{\text{CP}}(\sigma) = \sigma \]
- The abstraction of set of states
  \[ \alpha_{\text{CP}}: \mathcal{P}([\text{Var}_* \rightarrow \mathbb{Z}]) \rightarrow [\text{Var}_* \rightarrow \mathbb{Z} \cup \{ \bot, \top \}] \]
  \[ \alpha_{\text{CP}}(\text{CS}) = \bigcup \{ \beta_{\text{CP}}(\sigma) \mid \sigma \in \text{CS} \} = \bigcup \{ \sigma \mid \sigma \in \text{CS} \} \]
- Soundness
  \[ \alpha_{\text{CP}}(\text{Reach}(v)) \subseteq \text{df}(v) \]
- Completeness
The Concretization Function

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

\[ \gamma_{\text{CP}}: [\text{Var}_* \rightarrow \mathbb{Z} \cup \{\bot, \top\}] \rightarrow \mathcal{P}([\text{Var}_* \rightarrow \mathbb{Z}]) \]

\[ \gamma_{\text{CP}}(\text{df}) = \{\sigma | \beta_{\text{CP}}(\sigma) \subseteq \text{df}\} = \{\sigma | \sigma \subseteq \text{df}\} \]

- Soundness
  \[ \text{Reach}(v) \subseteq \gamma_{\text{CP}}(\text{df}(v)) \]

- Completeness
Galois Connection Constant

Propagation

● $\alpha_{CP}$ is monotone
● $\gamma_{CP}$ is monotone
● $\forall \ df \in [\text{Var}_* \rightarrow \mathbb{Z} \cup \{\bot, \top\}]$
  - $\alpha_{CP}(\gamma_{CP}(df)) \subseteq df$
● $\forall \ c \in P([\text{Var}_* \rightarrow \mathbb{Z}])$
  - $c_{CP} \subseteq \gamma_{CP}(\alpha_{CP}(C))$
Upper Closures

- Define abstractions on sets of concrete states
- $\uparrow: \mathcal{P}(\Sigma) \rightarrow \mathcal{P}(\Sigma)$ 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→0, y→0, z→0]}

\[ z = 3 \quad \{[x→0, y→0, z→3]\} \]

\[ x = 1 \quad \{[x→1, y→0, z→3]\} \]

while (x > 0) ( {[x→1, y→0, z→3], [x→3, y→0, z→3]},

if (x = 1) then \( y = 7 \)

\{[x→1, y→7, z→3], [x→3, y→7, z→3]\}

else \( y = z + 4 \)

\{[x→1, y→7, z→3], [x→3, y→7, z→3]\}

\[ x = 3 \]

{[x→1, y→7, z→3], [x→3, y→7, z→3]}

print \( y \)

{[x→3, y→7, z→3]}

) \{[x→3, y→7, z→3]\}
Another Example

\[
\begin{align*}
x &= 0 \\
\text{while (true) do} \\
\quad x &= x + 1
\end{align*}
\]
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]`
    \[ CS_{exit}(1) = \{ \sigma: \sigma \in CS_{entry}(l), \semantics{b}\sigma=tt \} \]
  - `[x := a]`
    \[ CS_{exit}(1) = \{ (s[x \mapsto A[a]]s) | s \in CS_{entry}(l) \} \]

◆ Equations for control flow constructs
  \[ CS_{entry}(l) = \bigcup CS_{exit}(l') \]
  \( l' \) immediately precedes \( l \)
  in the control flow graph

◆ An equation for the entry
  \[ CS_{entry}(1) = \{ \sigma | \sigma \in \text{Var}_* \rightarrow \mathbb{Z} \} \]
Specialized Chaotic Iterations
System of Equations
(Collecting Semantics)

\[ S = \]

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

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

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

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

\[ F_S : L^n \rightarrow L^n \]

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

\[ \text{lfp}(S) = \text{lfp}(F_S) \]
The Least Solution

- 2n sets of equations
  \[ CS_{\text{entry}}(1), \ldots, CS_{\text{entry}}(n), CS_{\text{exit}}(1), \ldots, CS_{\text{exit}}(n) \]
- Can be written in vectorial form
  \[ \vec{CS} = F_{cs}(\vec{CS}) \]
- The least solution lfp\( (F_{cs}) \) is well-defined
- Every component is minimal
- Since \( F_{cs} \) is monotone such a solution always exists
- \( CS_{\text{entry}}(v) = \{ s \mid \exists s_0 \mid \langle P, s_0 \rangle \Rightarrow^* (S', s), \text{init}(S') = v \} \)
- Simplify the soundness criteria
Specialized Chaotic Iterations
System of Equations
(Collecting Semantics)

\[ S = \]

\[ \begin{cases}
CS_{\text{entry}}[s] = \{ \sigma_0 \} \\
CS_{\text{entry}}[v] = \bigcup\{ f(e)(CS_{\text{entry}}[u]) \mid (u, v) \in E \}
\end{cases} \]

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

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

\[ F_S : L^n \rightarrow L^n \]

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

\[ \text{lfp}(S) = \text{lfp}(F_S) \]
The Least Solution

- 2n sets of equations
  \[ \text{CS}_{\text{entry}}(1), \ldots, \text{CS}_{\text{entry}}(n), \text{CS}_{\text{exit}}(1), \ldots, \text{CS}_{\text{exit}}(n) \]
- Can be written in vectorial form
  \[ \overrightarrow{\text{CS}} = F_{cs}(\overrightarrow{\text{CS}}) \]
- The least solution \( \text{lfp}(F_{cs}) \) is well-defined
- Every component is minimal
- Since \( F_{cs} \) is monotone such a solution always exists
- \( \text{CS}_{\text{entry}}(v) = \{ s | \exists s_0 \quad <P, s_0 \quad \Rightarrow^* (S', s)) \}, \quad \text{init}(S')=v \}
- Simplify the soundness criteria
∀a: f(γ(a)) ⊆ γ(f#(a))
Finite Height Case

\[ Lfp(f^\#) \]

\[ f^\# \]

\[ f^\# \]

\[ \bot \]

\[ \gamma \]

\[ \gamma \]

\[ f \]

\[ Lfp(f) \]
Soundness Theorem(2)

1. Let \((\alpha, \gamma)\) 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. \(\forall c \in C: \alpha(f(c)) \sqsubseteq f^\#(\alpha(c))\)

\[\alpha(\text{lfp}(f)) \sqsubseteq \text{lfp}(f^\#)\]

\[\text{lfp}(f) \sqsubseteq \gamma(\text{lfp}(f^\#))\]
Soundness Theorem (3)

1. Let \((\alpha, \gamma)\) 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. \(\forall a \in A: \alpha(f(\gamma(a))) \sqsubseteq f^#(a)\)

\[\alpha(\text{lfp}(f)) \sqsubseteq \text{lfp}(f^#)\]

\[\text{lfp}(f) \sqsubseteq \gamma(\text{lfp}(f^#))\]
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

\[ \alpha(\text{lfp}(f)) = \text{lfp}(f^#) \]

\[ \text{lfp}(f) = \gamma(\text{lfp}(f^#)) \]
Constant Propagation

- $\beta: [\text{Var} \rightarrow \mathbb{Z}] \rightarrow [\text{Var} \rightarrow \mathbb{Z} \cup \{\top, \bot\}]$
  - $\beta(\sigma) = (\sigma)$

- $\alpha: \mathcal{P}([\text{Var} \rightarrow \mathbb{Z}]) \rightarrow [\text{Var} \rightarrow \mathbb{Z} \cup \{\top, \bot\}]$
  - $\alpha(X) = \bigcup \{\beta(\sigma) | \sigma \in X\} = \bigcup \{\sigma | \sigma \in X\}$

- $\gamma: [\text{Var} \rightarrow \mathbb{Z} \cup \{\top, \bot\}] \rightarrow \mathcal{P}([\text{Var} \rightarrow \mathbb{Z}])$
  - $\gamma(\sigma^\#) = \{\sigma | \beta(\sigma) \subseteq \sigma^\#\} = \{\sigma | \sigma \subseteq \sigma^\#\}$

- **Local Soundness**
  - $\llbracket \text{st} \rrbracket^\#(\sigma^\#) \supseteq \alpha(\{ \llbracket \text{st} \rrbracket \sigma | \sigma \in \gamma(\sigma^\#)\}) = \bigcup \{ \llbracket \text{st} \rrbracket \sigma | \sigma \subseteq \sigma^\#\}$

- **Optimality (Induced)**
  - $\llbracket \text{st} \rrbracket^\#(\sigma^\#) = \alpha(\{ \llbracket \text{st} \rrbracket \sigma | \sigma \in \gamma(\sigma^\#)\}) = \bigcup \{ \llbracket \text{st} \rrbracket \sigma | \sigma \subseteq \sigma^\#\}$

- **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