AAA616: Program Analysis

Lecture 4 — Abstract Interpretation Framework

Hakjoo Oh 2016 Fall

Abstract Interpretation Framework

A powerful framework for designing correct static analysis

- "framework": correct static analysis comes out, reusable
- "powerful": all static analyses are understood in this framework
- "simple": prescription is simple
- "eye-opening": any static analysis is an abstract interpretation

 MATHAGE DERIVATION IN A DISTURB NOTION PORT PROFESSIONAL OF PERSONAL ACCOMPANIES ON APPRICIPATION OF EXECUTE INTERFER TOWARD OF MEMORY ACCOUNTS.
 MATHAGE ACCOUNTS AND ACCOUNTS ACCOUNTS.

1. Dependention

• else a second seco

A thread

pactics 5 described the space and solutionial summities of a subol insertion 1 despects over and Tanakine (T.). This support has been able to avoid or its built a super-solution mention in the sequence of a program, in the first beauter of her the start of the start of the solution of her the start of the solution of the solution of her the start of the solution of a solution.

- · duals is believe as C.7.7.7. Monthly
- re this and any supported by INE-SOLAL under

.

CC77

appear of proving of treatme

th will use finite fineheats as a tengong inb praine represention of program.

c spear of a fielder

has manufacer and productions under 1 Network, manufactor 1 Network (Network) (Network

Fraction, we note |B| the condimiting of a set B, for |B| = 1 or that B = 10 or considers use B to result A.

¹ Source and M.S. Strength, "Association the set forer, "An encourt and "Astra" pertition the set forer, and our account, Chrystophic - \$1 and norme one ofference research francescoling and reverse the research of the approximate and on the same reserving. For the same reserving of the ofference of the same approximation of the ofference of the same of the same reserving of the the same of the same of the same reserving of the the same of the same

10

CC79

In Institute 6 we study and ever while adopt on its used is near of againstand searching or spinbacket spincould be adopt number of the last future fragment to data type of the last future fragment to spin future spinerary spin-term graders that we equal to the last spin future that we equal to

conserver setti activiti nel consection della especializza della es

and the second secon

anna marpine un un un cond discontigi (n 1018 Film medicia Filmadori, 2006 discontigi (n 1018 Film medicia Filmadori, 2006 distorte et un ninamente un shartij colozitote di conteste et une rukament Liardori. En Stellion Tues consister glatikal program analysis

even in a second sec

AAA616 2016 Fall, Lecture 4

Step 1: Define Concrete Semantics

The concrete semantics describes the real executions of the program. Described by semantic domain and function.

- A semantic domain **D**, which is a CPO:
 - D is a partially ordered set with a least element \perp .
 - Any increasing chain $d_0 \sqsubseteq d_1 \sqsubseteq \ldots$ in D has a least upper bound $\bigsqcup_{n \ge 0} d_n$ in D.
- A semantic function F:D o D, which is continuous: for all chains $d_0\sqsubseteq d_1\sqsubseteq\ldots$,

$$F(\bigsqcup_{n\geq 0}d_i)=\bigsqcup_{n\geq 0}F(d_n).$$

Then, the concrete semantics (or collecting semantics) is defined as the least fixed point of semantic function $F: D \rightarrow D$:

fix
$$F = \bigsqcup_{i \in N} F^i(\bot)$$
.

Example: Concrete Semantics

- Program representation:
 - P is represented by control flow graph $(\mathbb{C},
 ightarrow, c_0)$
 - Each program point c is associated with a command $\operatorname{cmd}(c)$

$$cmd \hspace{.1in}
ightarrow \hspace{.1in} skip \mid x := e \ e \hspace{.1in}
ightarrow \hspace{.1in} n \mid x \mid e + e \mid e - e$$

- Semantics of commands:
- Concrete memory states: $\mathbb{M} = \mathbf{Var} \to \mathbb{Z}$
- Concrete semantics:

$$\begin{bmatrix} c \end{bmatrix} : \mathbb{M} \to \mathbb{M} \\ \begin{bmatrix} skip \end{bmatrix}(m) &= m \\ \llbracket x := e \rrbracket(m) &= m[x \mapsto \llbracket e \rrbracket(s)] \\ \llbracket e \rrbracket : \mathbb{M} \to \mathbb{Z} \\ \llbracket n \rrbracket(m) &= n \\ \llbracket x \rrbracket(m) &= m(x) \\ \llbracket e_1 + e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) + \llbracket e_2 \rrbracket(m) \\ \llbracket e_1 - e_2 \rrbracket(m) &= \llbracket e_1 \rrbracket(m) + \llbracket e_2 \rrbracket(m) \\ \end{bmatrix}$$

Example: Concrete Semantics

• Program states: State = $\mathbb{C} \times \mathbb{M}$

• A trace $\sigma \in \mathbf{State}^+$ is a (partial) execution sequence of the program:

 $\sigma_0 \in I \land orall k. \sigma_k \leadsto \sigma_{k+1}$

where $I \subseteq$ **State** is the initial program states

$$I=\{(c_0,m_0)\mid m_0\in\mathbb{M}\}$$

and $(\sim) \subseteq$ **State** × **State** is the relation for the one-step execution:

$$(c_i,s_i) \rightsquigarrow (c_j,s_j) \iff c_i \rightarrow c_j \land s_j = \llbracket \mathsf{cmd}(c_j)
rbracket (s_i)$$

Example: Concrete Semantics

The collecting semantics of program P is defined as the set of all finite traces of the program:

$$\llbracket P
rbracket = \{ \sigma \in \mathsf{State}^+ \mid \sigma_0 \in I \land \forall k. \sigma_k \leadsto \sigma_{k+1} \}$$

The semantic domain:

$$D = \wp(\mathsf{State}^+)$$

The semantic function:

$$egin{array}{rcl} F & : & \wp({\sf State}^+) o \wp({\sf State}^+) \ F(\Sigma) & = & I \cup \{ \sigma \cdot (c,m) \mid \sigma \in \Sigma \ \land \ \sigma \dashv \leadsto (c,m) \} \end{array}$$

Lemma

 $\llbracket P \rrbracket = fix F.$

Step 2: Define Abstract Semantics

Define the abstract semantics of the input program.

- Define an abstract semantic domain CPO \hat{D} .
 - Intuition: \hat{D} is an abstraction of D
- Define an abstract semantic function $\hat{F}:\hat{D}
 ightarrow\hat{D}.$
 - Intuition: \hat{F} is an abstraction of F.
 - \hat{F} must be monotone:

$$orall \hat{x}, \hat{y} \in \hat{D}. \ \hat{x} \sqsubseteq \hat{y} \implies \hat{F}(\hat{x}) \sqsubseteq \hat{F}(\hat{y})$$

(or extensive: $orall x \in \hat{D}. \ x \sqsubseteq \hat{F}(x))$

Then, static analysis is to compute an upper bound of:

$$igsqcup_{i\in\mathbb{N}}\hat{F}^i(ot)$$

How can we ensure that the result soundly approximate the concrete semantics?

Requirement 1: Galois Connection

D and \hat{D} must be related with Galois-connection:

$$D \stackrel{\gamma}{\underset{\alpha}{\longleftarrow}} \hat{D}$$

That is, we have

- abstraction function: $lpha\in D o \hat{D}$
 - \blacktriangleright represents elements in D as elements of \hat{D}
- concretization function: $\gamma\in\hat{D} o D$
 - ullet gives the meaning of elements of \hat{D} in terms of D
- $\forall x \in D, \hat{x} \in \hat{D}. \ \alpha(x) \sqsubseteq \hat{x} \iff x \sqsubseteq \gamma(\hat{x})$

- lpha and γ respect the orderings of D and \hat{D}

Galois-Connection



Example: Sign Abstraction

~

Sign abstraction:

where

$$\wp(\mathbb{Z}) \xrightarrow{\gamma} \{\bot, +, 0, -\top\}$$

$$\alpha(Z) = \begin{cases} \bot & Z = \emptyset \\ + & \forall z \in Z. \ z > \\ 0 & Z = \{0\} \\ - & \forall z \in Z. \ z < \\ \top & \text{otherwise} \end{cases}$$

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

$$\gamma(\top) = \mathbb{Z}$$

$$\gamma(+) = \{z \in \mathbb{Z} \mid z > 0\}$$

$$\gamma(0) = \{0\}$$

$$\gamma(-) = \{z \in \mathbb{Z} \mid z < 0\}$$

0

0

Example: Interval Abstraction

$$\begin{split} \wp(\mathbb{Z}) & \xleftarrow{\gamma}{\alpha} \{\bot\} \cup \{[a,b] \mid a \in \mathbb{Z} \cup \{-\infty\}, b \in \mathbb{Z} \cup \{+\infty\}\} \\ & \gamma(\bot) = \emptyset \\ & \gamma([a,b]) = \{z \in \mathbb{Z} \mid a \le z \le b\} \\ & \gamma([a,+\infty]) = \{z \in \mathbb{Z} \mid z \ge a\} \\ & \gamma([-\infty,b]) = \{z \in \mathbb{Z} \mid z \le b\} \\ & \gamma([-\infty,+\infty]) = \mathbb{Z} \end{split}$$

Requirement 2: \hat{F} and F

• \hat{F} and F must satisfy

$$lpha \circ F \sqsubseteq \hat{F} \circ lpha$$
 (i.e., $F \circ \gamma \sqsubseteq \gamma \circ \hat{F}$)

• or, alternatively,

$$lpha(x) \sqsubseteq \hat{x} \implies lpha(F(x)) \sqsubseteq \hat{F}(\hat{x})$$

Soundness Guarantee

Theorem (Fixpoint Transfer)

Let D and \hat{D} be related by Galois-connection $D \xleftarrow{\gamma}{\alpha} \hat{D}$. Let $F : D \to D$ be a continuous function and $\hat{F} : \hat{D} \to \hat{D}$ be a monotone function such that $\alpha \circ F \sqsubseteq \hat{F} \circ \alpha$. Then,

$$lpha(\mathit{fix} F) \sqsubseteq \bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\perp}).$$

Theorem (Fixpoint Transfer2)

Let D and \hat{D} be related by Galois-connection $D \xleftarrow{\gamma}{\alpha} \hat{D}$. Let $F: D \to D$ be a continuous function and $\hat{F}: \hat{D} \to \hat{D}$ be a monotone function such that $\alpha(x) \sqsubseteq \hat{x} \implies \alpha(F(x)) \sqsubseteq \hat{F}(\hat{x})$. Then,

$$lpha(\mathit{fix}F) \sqsubseteq \bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\perp}).$$

A Property of Galois-Connection

The functional composition of two Galois-connections is also Galois-connection:

Lemma
If
$$D_1 \xleftarrow{\gamma_1}{\alpha_1} D_2$$
 and $D_2 \xleftarrow{\gamma_2}{\alpha_2} D_3$, then
 $D_1 \xleftarrow{\gamma_1 \circ \gamma_2}{\alpha_2 \circ \alpha_1} D_3.$

Proof.

Exercise

Example: Partitioning Abstraction

Galois-connection: $\wp(\mathsf{State}^+) \xleftarrow{\gamma_1}{\alpha_1} \mathbb{C} \to \wp(\mathbb{M})$

$$lpha_1(\Sigma) \;\;=\;\; \lambda c. \{m \in \mathbb{M} \;|\; \exists \sigma \in \Sigma \;\wedge\; \exists i. \sigma_i = (c,m) \}$$

Semantic function:

$$\hat{F}_1: (\mathbb{C} o \wp(\mathbb{M})) o (\mathbb{C} o \wp(\mathbb{M}))$$
 $\hat{F}_1(X) = lpha_1(I) \sqcup \lambda c \in \mathbb{C}. \ f_c(igcup_{c' o c} X(c'))$

where $f_c: \wp(\mathbb{M}) \to \wp(\mathbb{M})$ is a transfer function at program point c:

$$f_c(M) \hspace{.1in} = \hspace{.1in} \{m' \mid m \in M \hspace{.1in} \land \hspace{.1in} m' = \llbracket \mathsf{cmd}(c) \rrbracket(m) \}$$

Lemma (Soundness of Partitioning Abstraction) $lpha_1(fixF) \sqsubseteq igsqcup_{i\in\mathbb{N}} \hat{F}_1^i(\bot).$

Example: Memory State Abstraction

Galois-connection:

$$\begin{split} \mathbb{C} &\to \wp(\mathbb{M}) \xleftarrow{\gamma_2}{\alpha_2} \mathbb{C} \to \hat{\mathbb{M}} \\ \alpha_2(f) &= \lambda c. \ \alpha_m(f(c)) \\ \gamma_1(\hat{f}) &= \lambda c. \ \gamma_m(\hat{f}(c)) \end{split}$$

where we assume

$$\wp(\mathbb{M}) \xrightarrow{\gamma_m} \hat{\mathbb{M}}$$

Semantic function $\hat{F}: (\mathbb{C} \to \hat{\mathbb{M}}) \to (\mathbb{C} \to \hat{\mathbb{M}})$:

$$\hat{F}(X) = (lpha_2 \circ lpha_1)(I) \sqcup \lambda c \in \mathbb{C}. \ \hat{f}_c(\bigsqcup_{c' o c} X(c'))$$

where abstract transfer function $\hat{f}_c:\hat{\mathbb{M}} o\hat{\mathbb{M}}$ is given such that

$$\alpha_m \circ f_c \sqsubseteq \hat{f}_c \circ \alpha_m \tag{1}$$

Theorem (Soundness) $\alpha(fixF) \sqsubseteq \bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\bot)$ where $\alpha = \alpha_2 \circ \alpha_1$.

Example: Sign Analysis

Memory state abstraction:

$$\wp(\mathbb{M}) \xleftarrow{\gamma_m}{\alpha_m} \hat{\mathbb{M}} \ lpha_m(M) = \lambda x \in \mathsf{Var.} \ lpha_s(\{m(x) \mid m \in M\})$$

....

where α_s is the sign abstraction:

$$\wp(\mathbb{Z}) \xleftarrow{\gamma_s}{\alpha_s} \hat{\mathbb{Z}}$$

The transfer function $\hat{f}_c: \hat{\mathbb{M}} \to \hat{\mathbb{M}}$:

$$egin{array}{rll} \hat{f}_{c}(\hat{m}) &=& \hat{m} & c = skip \ \hat{f}_{c}(\hat{m}) &=& \hat{m}[x\mapsto\hat{\mathcal{V}}(e)(\hat{m})] & c = x := e \ \hat{\mathcal{V}}(n)(\hat{m}) &=& lpha_{s}(\{n\}) \ \hat{\mathcal{V}}(x)(\hat{m}) &=& \hat{m}(x) \ \hat{\mathcal{V}}(e_{1}+e_{2}) &=& \hat{\mathcal{V}}(e_{1})(\hat{m}) + \hat{\mathcal{V}}(e_{2})(\hat{m}) \ \hat{\mathcal{V}}(e_{1}-e_{2}) &=& \hat{\mathcal{V}}(e_{1})(\hat{m}) - \hat{\mathcal{V}}(e_{2})(\hat{m}) \end{array}$$

Lemma

 $\alpha_m \circ f_c \sqsubseteq \hat{f}_c \circ \alpha_m$

Hakjoo Oh

Example: Interval Analysis

Memory state abstraction:

$$lpha_m(M) \;\;=\;\; \lambda x \in {\sf Var.}\; lpha_n(\{m(x) \mid m \in M\})$$

where α_n is the interval abstraction:

$$\wp(\mathbb{Z}) \xleftarrow{\gamma_n}{\alpha_n} \hat{\mathbb{Z}}$$

 $\hat{\mathbb{Z}} = \{\bot\} \cup \{[l,u] \mid l,u \in \mathbb{Z} \cup \{-\infty,+\infty\} \land l \leq u\}$ The transfer function $\hat{f}_c : \hat{\mathbb{M}} \to \hat{\mathbb{M}}$:

$$egin{array}{rll} \hat{f}_{c}(\hat{m}) &=& \hat{m} & c = skip \ \hat{f}_{c}(\hat{m}) &=& \hat{m}[x\mapsto\hat{\mathcal{V}}(e)(\hat{m})] & c = x := e \ \hat{\mathcal{V}}(n)(\hat{m}) &=& lpha_{s}(\{n\}) \ \hat{\mathcal{V}}(x)(\hat{m}) &=& \hat{m}(x) \ \hat{\mathcal{V}}(e_{1}+e_{2}) &=& \hat{\mathcal{V}}(e_{1})(\hat{m}) + \hat{\mathcal{V}}(e_{2})(\hat{m}) \ \hat{\mathcal{V}}(e_{1}-e_{2}) &=& \hat{\mathcal{V}}(e_{1})(\hat{m}) - \hat{\mathcal{V}}(e_{2})(\hat{m}) \end{array}$$

Lemma

 $\alpha_m \circ f_c \sqsubset \hat{f}_c \circ \alpha_m$

Hakjoo Oh

October 11, 2016 18 / 29

Computing an upper bound of $\bigsqcup_{i\in\mathbb{N}}\hat{F}^i(\hat{\perp})$

• If the abstract domain \hat{D} has finite height (i.e., all chains are finite), we can directly calculate

$$\bigsqcup_{i\in\mathbb{N}}\hat{F}^{i}(\hat{\perp}).$$

$$igsqcup_{i\in\mathbb{N}}\hat{F}^i(\hat{ot})\sqsubseteq \lim_{i\in\mathbb{N}}\hat{X}_i$$

Finite Chain \hat{X}_i

Define finite chain \hat{X}_i by an widening operator $\nabla: \hat{D} \times \hat{D} \to \hat{D}$:

$$\begin{array}{rcl} \hat{X}_0 &=& \bot \\ \hat{X}_i &=& \hat{X}_{i-1} & \text{ if } \hat{F}(\hat{X}_{i-1}) \sqsubseteq \hat{X}_{i-1} & (2) \\ &=& \hat{X}_{i-1} \bigtriangledown \hat{F}(\hat{X}_{i-1}) & \text{ otherwise} \end{array}$$

Conditions on ∇ :

• $\forall a,b\in \hat{D}.\;(a\sqsubseteq a\bigtriangledown b)\;\wedge\;(b\sqsubseteq a\bigtriangledown b)$

• For all increasing chains $(x_i)_i$, the increasing chain $(y_i)_i$ defined as

$$y_i = \left\{egin{array}{cc} x_0 & ext{if } i=0 \ y_{i-1} \bigtriangledown x_i & ext{if } i>0 \end{array}
ight.$$

eventually stabilizes (i.e., the chain is finite).

Then, the limit of the chain is safe analysis result.

Theorem (Widening's Safety)

Let \hat{D} be a CPO, $\hat{F} : \hat{D} \to \hat{D}$ a monotone function, $\bigtriangledown : \hat{D} \times \hat{D} \to \hat{D}$ a widening operator. Then, chain $(\hat{X}_i)_i$ defined as (2) eventually stabilizes and

$$igsqcup_{\in\mathbb{N}}\hat{F}^i(\hat{ot})\sqsubseteq \lim_{i\in\mathbb{N}}\hat{X}_i.$$

Narrowing

- We can refine the widening result $\lim_{i\in\mathbb{N}}\hat{X}_i$ by a narrowing operator $\triangle:\hat{D}\times\hat{D}\to\hat{D}$.
- Compute chain $(\hat{Y}_i)_i$

$$\hat{Y}_{i} = \begin{cases} \lim_{i \in \mathbb{N}} \hat{X}_{i} & \text{if } i = 0\\ \hat{Y}_{i-1} \bigtriangleup \hat{F}(\hat{Y}_{i-1}) & \text{if } i > 0 \end{cases}$$
(3)

- Conditions on riangle
 - $\blacktriangleright \ \forall a,b \in \hat{D}. \ a \sqsubseteq b \implies a \sqsubseteq a \bigtriangleup b \sqsubseteq b$
 - \blacktriangleright For all decreasing chain $(x_i)_i$, the decreasing chain $(y_i)_i$ defined as

$$y_i = \left\{egin{array}{cc} x_i & ext{if } i=0 \ y_{i-1} igtriangleq x_i & ext{if } i>0 \end{array}
ight.$$

eventually stabilizes.

Theorem (Narrowing's Safety)

Let \hat{D} be a CPO, $\hat{F} : \hat{D} \to \hat{D}$ a monotone function, $\triangle : \hat{D} \times \hat{D} \to \hat{D}$ a narrowing operator. Then, chain $(\hat{Y}_i)_i$ defined as (3) eventually stabilizes and

$$igsqcup_{i\in\mathbb{N}}\hat{F}^i(\hat{ot})\sqsubseteq \lim_{i\in\mathbb{N}}\hat{Y}_i.$$

i = 0;
while (i<10)
 i++;</pre>

• Abstract equation:

$$egin{array}{rcl} X_1&=&[0,0]\ X_2&=&(X_1\sqcup X_3]\sqcap [-\infty,9]\ X_3&=&X_2+[1,1]\ X_4&=&(X_1\sqcup X_3)\sqcap [10,+\infty] \end{array}$$

• Abstract domain $\hat{D} =$ Interval × Interval × Interval × Interval • Semantic function $\hat{F}:\hat{D}\to\hat{D}$ such that

$$(X_1, X_2, X_3, X_4) = \hat{F}(X_1, X_2, X_3, X_4)$$

$$egin{array}{rcl} X_1&=&[0,0]\ X_2&=&(X_1\sqcup X_3]\sqcap [-\infty,9]\ X_3&=&X_2\stackrel{.}{+} [1,1]\ X_4&=&(X_1\sqcup X_3)\sqcap [10,+\infty] \end{array}$$

 $igsqcup_{i\in\mathbb{N}}\hat{F}^{i}(\hat{\perp})$:

	0	1	2	3	4	5	6	
X_1	Ĵ	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
X_2	Î	Ĵ	[0, 0]	[0, 0]	[0,1]	[0,1]	[0, 2]	[0,9]
X_3	Î	Ĵ	Î	[1,1]	[1,1]	[1, 2]	[1, 2]	[1, 10]
X_4	Î	Ĵ	Î	Î	Î	Î	Î	[10, 10]

A simple widening operator for the Interval domain:

$$egin{array}{rll} [a,b] &\bigtriangledown &\perp &= [a,b] \ &\perp &\bigtriangledown & [c,d] &= [c,d] \ [a,b] &\bigtriangledown & [c,d] &= [(c < a? - \infty:a), (b < d? + \infty:b)] \end{array}$$

A simple narrowing operator:

$$egin{array}{rcl} [a,b] & \bigtriangleup & \bot & = \bot \ & \bot & \bigtriangleup & [c,d] & = \bot \ [a,b] & \bigtriangleup & [c,d] & = [(a=-\infty?c:a), (b=+\infty?d:b)] \end{array}$$

Widening iteration:

	0	1	2	3	4	5	6	7
X_1	Î	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0,0]	[0, 0]	[0, 0]
X_2	Î	Î Î	[0,0]	[0, 0]	$[0,+\infty]$	$[0, +\infty]$	$[0,+\infty]$	$[0,+\infty]$
X_3	ÎÎ	Î Î	Î Î	[1, 1]	[1, 1]	$[1, +\infty]$	$[1,+\infty]$	$[1,+\infty]$
X_4	Î	Î	Î	Î	Ĵ	Î	$[10,+\infty]$	$[10,+\infty]$

Narrowing iteration:

	0	1	2	3	4
X_1	[0, 0]	[0, 0]	[0, 0]	[0, 0]	[0, 0]
X_2	$[0,+\infty]$	[0, 9]	[0, 9]	[0, 9]	[0, 9]
X_3	$[1, +\infty]$	$[1,+\infty]$	[1, 10]	[1, 10]	[1, 10]
X_4	$[10,+\infty]$	$[10,+\infty]$	$[10,+\infty]$	[10, 10]	[10, 10]

Worklist Algorithm

 $W \in Worklist = \wp(\mathbb{C})$ $T \in \mathbb{C} \to \hat{\mathbb{S}}$ $\hat{f}_c \in \hat{\mathbb{S}} \to \hat{\mathbb{S}}$ $W := \mathbb{C}$ $T := \lambda c$. repeat c := choose(W) $W := W - \{c\}$ $\hat{s}_{in} := | |_{c' \to c} \hat{f}_{c'}(T(c'))$ if $\hat{s}_{in} \not \sqsubseteq \hat{X}(c)$ if c is a head of a flow cycle $\hat{s}_{in} := T(c) \bigtriangledown \hat{s}_{in}$ $\hat{X}(c) := \hat{s}_{in}$ $W := W \cup \{c' \mid c \to c'\}$ until $W = \emptyset$

Example

