Static Analysis Design Framework: Abstract Interpration

Kwangkeun Yi

Seoul National University, Korea
http://ropas.snu.ac.kr/~kwang

2/26/2012 – 3/2/2012
17th Estonian Winter School in Computer Science, Palmse, Estonia

Abstract Interpretation

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

Why Abstraction?

- without abstraction, can't capture all possible executions
- without abstraction, can't terminate

Abstraction is not omission

- reality: $\{2, 4, 6, 8, \cdots\}$
- "even number" (abstraction) vs "multiple of 4" (omission)

Abstract Interpretation Framework

real execution	$[\![C]\!] = \mathit{fix} F \in D$
abstract execution	$[\![\hat{C}]\!] = \lim_{i\in\mathbb{N}} \hat{F}^i(\bot_{\hat{D}}) \ \in \hat{D}$
correctness	$[\![C]\!] \approx [\![\hat{C}]\!]$
implementation	computation of $[\![\hat{C}]\!]$

The framework requires:

- a relation between D and $\hat{D}$
- a relation between $F \in D \to D$ and $\hat{F} \in \hat{D} \to \hat{D}$

The framework guarantees:

- correctness: $[\![C]\!] \approx [\![\hat{C}]\!]$
- implmentation: computation of $[\![\hat{C}]\!]$
- freedom: any such $\hat{F}$ and $\hat{D}$ are fine

Static Analysis Design: step 1

Define the input program's real executions(concrete semantics)

- Define semantic domain CPO D.
- Define the real executions as the least fixed point $\mathit{fix} F$ of continuous function $F \in D \to D$

$$\mathit{fix} F = \bigsqcup_{i \in \mathbb{N}} F^i(\perp_D)$$

$\boxed{\text{Plan: define an abstraction that captures } \mathit{fix} F}$

Static Analysis Design: step 2

Define the input program's abstract semantics

- Define abstract domain CPO $\hat{D}$.
  - Establish a Galois connection between D and $\hat{D}$
- Define an abstract semantic function $\hat{F} \in \hat{D} \to \hat{D}$
  - $\hat{F}$ must be monotonic or extensive

Plan: define an abstraction that captures $\mathit{fix} F$ by using $\hat{F}$

Requirement 1: about $\hat{D}$ in relation with D

$\hat{D}$ must be Galois-connected with D

$$D \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \hat{D}.$$

- Galois connection:

$$\forall x \in D, \hat{x} \in \hat{D} : \alpha(x) \sqsubseteq \hat{x} \iff x \sqsubseteq \gamma(\hat{x}).$$

- Galois connection captures our intention:
  - bigger elements in $\hat{D}$ means more.
  - α abstracts .
  - γ concretizes.

Plan: static analysis is computing an upper bound of $\bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\bot})$

Requirement 2: about $\hat{F}$

- $\hat{F}$ must be monotonic:

$$\forall x, y \in \hat{D} : x \sqsubseteq y \Rightarrow \hat{F}(x) \sqsubseteq \hat{F}(y)$$

or extensive:

$$\forall x \in \hat{D} : x \sqsubseteq \hat{F}(x).$$

Plan: static analysis is computing an upper bound of $\bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\bot})$

Requirement 3: $\hat{F}$ in relation with F

- Concrete semantic ftn F and its abstract version $\hat{F}$ must satisfy

$$\alpha \circ F \sqsubseteq \hat{F} \circ \alpha, \quad \text{i.e.,} \quad F \circ \gamma \sqsubseteq \gamma \circ \hat{F}$$

or,

- Concrete semantic ftn F and its abstract version $\hat{F}$ must satisfy

$$\alpha(f) \sqsubseteq \hat{f} \;\Rightarrow\; \alpha(F\ f) \sqsubseteq \hat{F}\ \hat{f}$$

Plan: static analysis is computing an upper bound of $\bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\bot})$

Then: a Correct Static Analysis

static analysis = computing an upper bound of $\bigsqcup_{i\in\mathbb{N}} \hat{F}^i(\hat{\bot})$.

- Such an upper bound $\hat{A}$ is correct:

$$\alpha(\mathit{fix}\,F) \sqsubseteq \hat{A}, \quad \text{that is,}$$
$$\mathit{fix}\,F \sqsubseteq \gamma\hat{A}$$

  Theorem[fixpoint-transfer]

- Analysis result $\hat{A}$ subsumes the real executions $\mathit{fix}\,F$

How to Compute an Upper Bound of $\bigsqcup_{i\in\mathbb{N}} \hat{F}^i(\hat{\bot})$

- If abstract semantic domain $\hat{D}$'s height is finite then, we can directly compute
$$\bigsqcup_{i\in\mathbb{N}} \hat{F}^i(\hat{\bot}).$$
The computation always terminates.
- Otherwise, we compute a finite chain $\{\hat{X}_i\}_i$ such that
$$\bigsqcup_{i\in\mathbb{N}} (\hat{F}^i(\hat{\bot})) \sqsubseteq \lim_{i\in\mathbb{N}} (\hat{X}_i).$$

Searching for Finite Chain $\{\hat{X}_i\}_i$

Finite chain $\{\hat{X}_i\}_i$ such that

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

- If $\hat{F}$ is monotonic, a chain by an widening operator $\triangledown$:

$$\begin{array}{rcl} \hat{X}_0 & = & \hat{\bot} \\ \hat{X}_{i+1} & = & \begin{cases} \hat{X}_i & \text{if } \hat{F}(\hat{X}_i) \sqsubseteq \hat{X}_i \\ \hat{X}_i \triangledown \hat{F}(\hat{X}_i) & \text{o.w.} \end{cases} \end{array}$$

Conditions on Widening $\triangledown$

Conditions

- $\forall a, b \in \hat{D} : (a \sqsubseteq a \triangledown b) \ \wedge \ (b \sqsubseteq a \triangledown b)$
- $\forall$increasing chain$\{a_i\}_i$: chain$x_0 = a_0, x_{i+1} = x_i \triangledown a_{i+1}$is finite

Then

- $\{\hat{X}_i\}_i$ is a finite chain.
- Its limit($\hat{X}$) such that $\hat{F}(\hat{X}) \sqsubseteq \hat{X}$ is correct:

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

Theorem[widen's safety]

Refining the Widening Result

If $\hat{F}$ is monotonic,

- We can refine the widened result $\hat{\mathcal{A}} \stackrel{\text{let}}{=} \lim_{i \in \mathbb{N}}(\hat{X}_i)$ by a narrowing operator $\triangle$.
- Compute chain $\{\hat{Y}_i\}_i$

$$\begin{array}{rcl} \hat{Y}_0 & = & \hat{\mathcal{A}} \\ \hat{Y}_{i+1} & = & \hat{Y}_i \triangle \hat{F}(\hat{Y}_i) \end{array}$$

Conditions

- $\forall a, b \in \hat{D} : a \sqsupseteq b \Rightarrow a \sqsupseteq (a \triangle b) \sqsupseteq b$
- $\forall$decreasing chain$\{a_i\}_i$: chain$y_0 = a_0, y_{i+1} = y_i \triangle a_{i+1}$is finite

Then

- $\{\hat{Y}_i\}_i$ is a finite chain.
- Its limit $\lim_{i \in \mathbb{N}}(\hat{Y}_i)$ is still correct:

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

Theorem[narrow's safety]

Fixpoint Transfer Theorem

Theorem (fixpoint transfer)

Let CPOs D *and* $\hat{D}$ *are Galois-connected. Function* $F : D \to D$ *is continuous.* $\hat{F} : \hat{D} \to \hat{D}$ *is either monotonic or extensive. Either* $\alpha \circ F \sqsubseteq \hat{F} \circ \alpha$ *or* $\alpha\ f \sqsubseteq \hat{f}$ *implies* $\alpha(F\ f) \sqsubseteq \hat{F}\ \hat{f}$*. Then,*

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

Why Above Prescription Is Correct? (2/2)

Widening/Narrowing Theorems

Theorem (widen's safety)

Let $\hat{F} : \hat{D} \to \hat{D}$ *be monotonic over CPO* $\hat{D}$. *Let widening operator* $\bigtriangledown : \hat{D} \times \hat{D} \to \hat{D}$ *satisfies the widending conditions. Then the widened chain* $\{\hat{X}_i\}_i$ *is finite and its limit satisfies* $\lim_{i \in \mathbb{N}} \hat{X}_i \sqsupseteq \bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\bot})$.

Theorem (narrow's safety)

Let $\hat{F} : \hat{D} \to \hat{D}$ *be monotonic over CPO* $\hat{D}$. *Let narrowing operator* $\bigtriangleup : \hat{D} \times \hat{D} \to \hat{D}$ *satisfies the narrowng conditions. If* $\hat{F}(\hat{\mathcal{A}}) \sqsubseteq \hat{\mathcal{A}}$ *then the narrowed chain* $\{\hat{Y}_i\}_i$ *is finite and its limit satisfies* $\lim_{i \in \mathbb{N}} \hat{Y}_i \sqsupseteq \bigsqcup_{i \in \mathbb{N}} \hat{F}^i(\hat{\bot})$.

Abstract Interpretation Example

(or, a Special Abstract Interpretation Framework)

Semantics as Trace

Program C's semantics $[\![C]\!]$ is the set of all execution traces

$$\begin{array}{rcccl} [\![C]\!] & \in & 2^{\mathit{Trace}} & & \\ \tau, \tau_0\tau_1\cdots\tau_n & \in & \mathit{Trace} & = & \mathit{State}^* \\ & & \mathit{State} & = & \mathit{Command} \times \mathit{Memory} \times \cdots \end{array}$$

Side:

Trace	$=$	State^ω	v.s.	State^*
		liveness analysis		safety analysis
		prop. after infinite traces		prop. within finite traces

$$2^{Trace} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \hat{Trace}$$

$\xrightarrow{\alpha_0}$ Trace of set of states: sequence of set of states appearing at a given time along at least one of the traces

$$\alpha_0(X) = \lambda i.\{\tau_i \mid \tau \in X, 0 \leq i < |\tau|\} \quad \in \hat{Trace} = \mathbb{N} \xrightarrow{\text{fin}} 2^{State}$$

$\xrightarrow{\alpha_1 \circ \alpha_0}$ Set of reachable states (global invariant): set of states appearing at least once along a trace

$$\alpha_1(Y) = \bigcup\{Y(i) \mid i \in \text{Dom } Y\} \quad \in \hat{Trace} = 2^{State}$$

$\xrightarrow{\alpha_2 \circ \alpha_1 \circ \alpha_0}$ Partitioned set of reachable states (local invariant): e.g., project along each control point $\in \Delta$ (a finite set)

$$\alpha_2(Z) = \lambda c.\{s_i \mid \langle c_i, s_i\rangle \in Z, c_i = c \in \Delta\} \quad \in \hat{Trace} = \Delta \to 2^{State}$$

$\xrightarrow{\alpha_3 \circ \alpha_2 \circ \alpha_1 \circ \alpha_0}$ Abstracting the partitioned set of reachable states

$$\alpha_3(\Phi) = \lambda c.\alpha(\Phi\ c) \quad \in \hat{Trace} = \Delta \to \hat{State}$$

where

$$2^{State} \leftrightarrows \hat{State}$$

Trace Abstract Interpretation's Correctness Condition

$$\mathit{fix}(F \stackrel{\text{let}}{=} \lambda T.T_0 \cup \mathit{Next}\ T) \quad \text{and} \quad \mathit{fix}(\hat{F} \stackrel{\text{let}}{=} \lambda \hat{T}.\alpha(T_0) \sqcup \hat{\mathit{Next}}\ \hat{T})$$

where

$$F \in 2^{\mathit{Trace}} \to 2^{\mathit{Trace}} \quad \text{and} \quad \hat{F} \in \hat{\mathit{Trace}} \to \hat{\mathit{Trace}}.$$

To show is $\alpha(\mathit{fix} F) \sqsubseteq \mathit{fix} \hat{F}$, i.e., $\alpha \circ F \sqsubseteq \hat{F} \circ \alpha$.
A sufficient condition, if Trace and $\hat{\mathit{Trace}}$ are $\sqcup$-closed, is:

$$\alpha \circ \mathit{Next} \sqsubseteq \hat{\mathit{Next}} \circ \alpha.$$

(easy to see, by Galois-connection.)

A Sufficient Condition for $\alpha \circ Next \sqsubseteq \hat{Next} \circ \alpha$ (1/4)

Focus on:

$$2^{State} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (\Delta \to \hat{State})$$

that is,

- program's all executions = the collection of all the machine states occuring during the executions

$$[\![C]\!] \in 2^{State}$$

- program's abstract semantics = partition and abstract the collection:

$$[\![\hat{C}]\!] \in \Delta \to \hat{State}$$

  - Δ: a finite set of partinitiong indices
  - e.g.) Δ = the set of program points

Notation $\wp f$

For $f \in A \to B$,

- $\wp f \in 2^A \to 2^B$ is $(\wp f)X = \{fx \mid x \in X\}$.
- Abusely, $\wp f \in (\Delta \to A) \to 2^B$ is $(\wp f)X = \{fx \mid x \in range X\}$.

A Sufficient Condition for $\alpha \circ Next \sqsubseteq \hat{Next} \circ \alpha$ (2/4)

The Galois-connection

$$2^{State} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (\Delta \to \hat{State})$$

is

$$\alpha = (\wp\alpha_1) \circ \pi.$$

- α_1 abstracts sets of states into abstract states:

$$2^{State} \underset{\alpha_1}{\overset{\gamma_1}{\leftrightarrows}} \hat{State}.$$

- π and $\hat{\pi}$ are partition functions:

$$\begin{array}{lcl} \pi & \in & 2^{State} \to 2^{2^{State}} \\ \hat{\pi} & \in & 2^{\hat{State}} \to (\Delta \to 2^{\hat{State}}) \end{array}$$

Define

$$
\begin{array}{lll}
Next & = \wp next & \in 2^{State} \to 2^{State} \\
\hat{Next} & = (\wp \sqcup) \circ \hat{\pi} \circ \cup \circ (\wp \hat{next}) & \in (\Delta \to \hat{State}) \to (\Delta \to \hat{State})
\end{array}
$$

where

- concrete transition $next$:

  $$next \in State \to State$$

  (transitions terminal state into itself)
- abstract transition $\hat{next}$:

  $$\hat{next} \in \hat{State} \to 2^{\hat{State}}$$

  (may transition one abstract state into multiple abstract states)

A Sufficient Condition for $\alpha \circ Next \sqsubseteq \hat{Next} \circ \alpha$ (4/4)

Theorem (Correctness)

Let $Next$ and $\hat{Next}$ be:

$$\begin{array}{lll} Next & = \wp next & \in 2^{State} \to 2^{State} \\ \hat{Next} & = (\wp \sqcup) \circ \hat{\pi} \circ \cup \circ (\wp \hat{next}) & \in (\Delta \to \hat{State}) \to (\Delta \to \hat{State}) \end{array}$$

If the below two conditions hold then $\alpha \circ Next \sqsubseteq \hat{Next} \circ \alpha$.

1. *Condition on abstract partitioning($\hat{\pi}$):*

$$(\wp \alpha_1) \circ \pi \circ \cup \circ (\wp \gamma) \ \sqsubseteq \ (\wp \sqcup) \circ \hat{\pi} \tag{1}$$

2. *Condition on abstract transition($\hat{next}$):*

$$next\ x \in (\cup \circ (\wp \gamma) \circ \hat{next} \circ \alpha_1)\ \{x\} \tag{2}$$

Notation

- $\uparrow \in X \to 2^X$ is $\uparrow x = \{x\}$.
- For $f \in A \to B$, $\wp f \in 2^A \to 2^B$ is $(\wp f)X = \{fx \mid x \in X\}$.
- Abusely, $\wp f \in (\Delta \to A) \to 2^B$ is $(\wp f)X = \{fx \mid x \in rangeX\}$.
- For $f \in A \to 2^B$, $\wp_{\cup} f = \cup \circ \wp f$.

Facts

- $\wp_{\cup}(f \circ g) = (\wp_{\cup} f) \circ (\wp g)$.
- $\wp_{\cup}(\wp_{\cup} f) \circ (\wp g) = (\wp_{\cup} f) \circ (\wp_{\cup} g)$.
- For $x \in A, X \in 2^A$, $fx \in gx$ implies $(\wp f)X \subseteq (\wp_{\cup} g)X$.

Proof. First, from condition (2) the following holds:

$$\wp next \sqsubseteq (\wp_\cup \gamma) \circ (\wp_\cup \hat{next}) \circ \alpha \qquad (3)$$

Because,

$$\begin{array}{lcll}
\wp next & \sqsubseteq & \wp_\cup((\wp_\cup \gamma) \circ \hat{next} \circ \alpha_1 \circ \uparrow) & (\text{cond. (2), } (fx \in gx \text{ then } (\wp f)X \subseteq (\wp_\cup g)X \\
& = & \wp_\cup(\wp_\cup \gamma) \circ \wp(\hat{next} \circ \alpha_1 \circ \uparrow) & (\wp_\cup(f \circ g) = (\wp_\cup f) \circ (\wp g)) \\
& = & (\wp_\cup \gamma) \circ (\wp_\cup \hat{next}) \circ (\wp \alpha_1) \circ (\wp \uparrow) & (\wp_\cup(\wp_\cup f) \circ (\wp g) = (\wp_\cup f) \circ (\wp_\cup g)) \\
& \sqsubseteq & (\wp_\cup \gamma) \circ (\wp_\cup \hat{next}) \circ (\wp \alpha_1) \circ \pi & (\gamma, \hat{next}, \alpha_1 \text{ are all monotonic}) \\
& = & (\wp_\cup \gamma) \circ (\wp_\cup \hat{next}) \circ \alpha. &
\end{array}$$

Therefore,

$$\begin{array}{lcll}
\alpha \circ Next & = & (\wp \alpha_1) \circ \pi \circ (\wp next) & \\
& \sqsubseteq & (\wp \alpha_1) \circ \pi \circ (\wp_\cup \gamma) \circ (\wp_\cup \hat{next}) \circ \alpha & (\text{cond. (3)}) \\
& \sqsubseteq & (\wp \sqcup) \circ \hat{\pi} \circ (\wp_\cup \hat{next}) \circ \alpha & (\text{cond. (1)}) \\
& = & \hat{Next} \circ \alpha. &
\end{array}$$

That is, from condition (1) and condition (2), $\alpha \circ Next \sqsubseteq \hat{Next} \circ \alpha$ holds. Hence by the Fixpoint Transfer Theorem,

$$\alpha(\mathit{fix}(\lambda T.T_0 \cup Next\ T)) \sqsubseteq \mathit{fix}(\lambda \hat{T}.\alpha(T_0) \sqcup \hat{Next}\ \hat{T}).$$

$\square$

Static analysis is to compute $[\![\hat{C}]\!]$, which is

$$\mathit{fix}(\hat{F} \stackrel{\text{let}}{=} \lambda \hat{T}.\alpha(T_0) \sqcup \hat{Next}\,\hat{T})$$

where

$$\begin{array}{rcl} \hat{F} & \in & \hat{Trace} \to \hat{Trace} \\ \hat{Trace} & = & \Delta \to \hat{State} \\ \hat{Next} & = & (\wp\sqcup) \circ \hat{\pi} \circ (\wp_{\cup}\hat{next}) \\ \hat{next} & \in & \hat{State} \to 2^{\hat{State}}. \end{array}$$

Computing $\mathit{fix}\hat{F}$ is to compute Y_i until no change:

$$Y_0 = \alpha(T_0), \quad Y_{n+1} = \alpha(T_0) \sqcup \hat{Next}(Y_n)$$

Hence,

$T, T' : \Delta \to \hat{State}$;
begin
  $T := T' := \alpha(T_0)$;
  repeat
    $T' := T$;
    $T := \alpha(T_0) \;\sqcup\; ((\wp\sqcup) \circ \hat{\pi})(\bigcup_{i \in \Delta} \hat{next}\,T[i])$;
  until $T \sqsubseteq T'$; (* no more increase *)
return T';
end

Figure: Naive algorithm

When widening($\triangledown$) and narrowing($\triangle$) are necessary, we compute the folloing two things in sequence:

$$Widen(\hat{F}) = \lim_{i\in\mathbb{N}} \begin{cases} \hat{Y}_0 = \alpha(T_0) \\ \hat{Y}_{i+1} = \begin{cases} \hat{Y}_i & \text{if } \hat{F}(\hat{Y}_i) \sqsubseteq \hat{Y}_i \\ \hat{Y}_i \triangledown \hat{F}(\hat{Y}_i) & \text{o.w.} \end{cases} \end{cases}$$

$$Narrow(\hat{m}) = \lim_{i\in\mathbb{N}} \begin{cases} \hat{Z}_0 = \hat{m} \\ \hat{Z}_{i+1} = \hat{Z}_i \triangle \hat{F}(\hat{Z}_i) \end{cases}$$

Hence,

$T, T', Y : \Delta \to \hat{State};$
begin
  $T := T' := \alpha(T_0);$
  repeat
    $T' := T;$
    $Y := \alpha(T_0) \sqcup ((\wp\sqcup) \circ \hat{\pi})(\bigcup_{i\in\Delta} \hat{next}\, T[i]);$
    $T :=$ **if** $Y \sqsubseteq T'$ **then** T' **else** $T' \triangledown Y;$
  until $T \sqsubseteq T'$; (* no more increase *)

  repeat
    $T := T'$
    $T' \triangle := \alpha(T_0) \sqcup ((\wp\sqcup) \circ \hat{\pi})(\bigcup_{i\in\Delta} \hat{next}\, T[i]);$
  until $T \sqsubseteq T'$; (* no more decrease *)
  return T;
end

Figure: Naive algorithm with widening and narrowing

Trace Abstract Interpretation's Algorithm (3/4)

Worklist method:

- wasteful at each iteration to compute

$$\bigcup_{i \in \Delta} \hat{next}\, T[i]$$

  for every index in Δ.
- enough to compute those affected from the previous iteration

$T, T' : \Delta \to \hat{State};$
$W : 2^{\Delta};$ (* worklist *)
begin
 $T := T' := \alpha(T_0);\quad W := \Delta;$
 repeat
  $T' := T;$
  $T := \alpha(T_0) \sqcup ((\wp \sqcup) \circ \hat{\pi})(\bigcup_{i \in W} \hat{next}\, T[i]);$
  $W := \{i \in \Delta \mid T[i] \not\sqsubseteq T'[i]\};$
 until $W = \{\}$; (* no more increase *)
return T';
end

Figure: Worklist algorithm

$T, T', Y : \Delta \to \hat{State};$
$W : 2^{\Delta};$ (* worklist *)
begin
$\quad T := T' := \alpha(T_0); \quad W := \Delta;$
$\quad$ **repeat**
$\quad\quad T' := T;$
$\quad\quad Y := \alpha(T_0) \sqcup ((\wp\sqcup) \circ \hat{\pi})(\bigcup_{i \in W} \hat{next}\ T[i]);$
$\quad\quad T :=$ **if** $Y \sqsubseteq T'$ **then** T' **else** $T' \triangledown Y;$
$\quad\quad W := \{i \in \Delta \mid T[i] \not\sqsubseteq T'[i]\};$
$\quad$ **until** $W = \{\};$ (* no more increase *)

$\quad W := \Delta;$
$\quad$ **repeat**
$\quad\quad T := T';$
$\quad\quad T' \ \triangle := \alpha(T_0) \sqcup ((\wp\sqcup) \circ \hat{\pi})(\bigcup_{i \in W} \hat{next}\ T[i]);$
$\quad\quad W := \{i \in \Delta \mid T[i] \not\sqsubseteq T'[i]\};$
$\quad$ **until** $W = \{\};$ (* no more decrease *)
$\quad$ **return** $T;$
end

Figure: Worklist algorithm with widening and narrowing