

# Semantics and Verification of Software

## Lecture 19: Dataflow Analysis

Thomas Noll

Lehrstuhl für Informatik 2  
RWTH Aachen University  
noll@cs.rwth-aachen.de

<http://www-i2.informatik.rwth-aachen.de/i2/svsw/>

Summer semester 2007

- 1 Repetition: The MOP Solution
- 2 Another Analysis: Constant Propagation
- 3 Undecidability of the MOP Solution
- 4 MOP vs. Fixpoint Solution

# The MOP Solution I

- Other **solution method** for dataflow systems
- MOP = **Meet Over all Paths**
- Analysis information for block  $B^l :=$   
**least upper bound over all paths leading to  $l$**

## Definition (Paths)

Let  $S = (Lab, E, F, (D, \sqsubseteq), \iota, \varphi)$  be a dataflow system. For every  $l \in Lab$ , the set of **paths up to  $l$**  is given by

$$Path(l) := \{[l_1, \dots, l_{k-1}] \mid k \geq 1, l_1 \in E, \\ (l_i, l_{i+1}) \in F \text{ for every } 1 \leq i \leq k, l_k = l\}.$$

For a path  $p = [l_1, \dots, l_{k-1}] \in Path(l)$ , we define the **transfer function**  $\varphi_p : D \rightarrow D$  by

$$\varphi_p := \varphi_{l_{k-1}} \circ \dots \circ \varphi_{l_1} \circ \text{id}_D$$

(so that  $\varphi_{[]} = \text{id}_D$ ).

## Definition (MOP solution)

Let  $S = (Lab, E, F, (D, \sqsubseteq), \iota, \varphi)$  be a dataflow system where  $Lab = \{l_1, \dots, l_n\}$ . The **MOP solution** for  $S$  is determined by

$$\text{mop}(S) := (\text{mop}(l_1), \dots, \text{mop}(l_n)) \in D^n$$

where, for every  $l \in Lab$ ,

$$\text{mop}(l) := \bigsqcup \{\varphi_p(\iota) \mid p \in Path(l)\}.$$

## Remark:

- $Path(l)$  is generally infinite

⇒ not clear how to compute  $\text{mop}(l)$

- In fact: MOP solution generally undecidable (later)

- 1 Repetition: The MOP Solution
- 2 Another Analysis: Constant Propagation
- 3 Undecidability of the MOP Solution
- 4 MOP vs. Fixpoint Solution

# Goal of Constant Propagation Analysis

## Constant Propagation Analysis

The goal of **Constant Propagation Analysis** is to determine, for each program point, whether a variable has a constant value whenever execution reaches that point.

Used for **Constant Folding**: replace reference to variable by constant value

## Example 19.1 (Constant Propagation Analysis)

```
[x := 1]1;  
[y := 1]2;  
[z := 1]3;  
while [z > 0]4 do  
  [w := x+y]5;  
  if [w = 2]6 then  
    [x := y+2]7
```

- $y = z = 1$  at labels 4–7
- $w, x$  not constant at labels 4–7
- possible optimizations:  $[w := x+1]<sup>5</sup>  
     $[x := 3]<sup>7</sup>$$

# Formalizing Constant Propagation Analysis I

The **dataflow system**  $S = (Lab, E, F, (D, \sqsubseteq), \iota, \varphi)$  is given by

- set of labels  $Lab := Lab_c$ ,
- extremal labels  $E := \{\text{init}(c)\}$  (forward problem),
- flow relation  $F := \text{flow}(c)$  (forward problem),
- complete lattice  $(D, \sqsubseteq)$  where
  - $D := \{\delta \mid \delta : Var_c \rightarrow \mathbb{Z} \cup \{\perp, \top\}\}$ 
    - $\delta(x) = z \in \mathbb{Z}$ :  $x$  has constant value  $z$
    - $\delta(x) = \perp$ :  $x$  undefined
    - $\delta(x) = \top$ :  $x$  overdefined (i.e., different possible values)
  - $\sqsubseteq \subseteq D \times D$  defined by pointwise extension of  $\perp \sqsubseteq z \sqsubseteq \top$  (for every  $z \in \mathbb{Z}$ )

## Example 19.2

$$\begin{aligned} Var_c &= \{w, x, y, z\}, \\ \delta_1 &= (\underbrace{\perp}_w, \underbrace{1}_x, \underbrace{2}_y, \underbrace{\top}_z), \quad \delta_2 = (\underbrace{3}_w, \underbrace{1}_x, \underbrace{4}_y, \underbrace{\top}_z) \\ \implies \delta_1 \sqcup \delta_2 &= (\underbrace{3}_w, \underbrace{1}_x, \underbrace{\top}_y, \underbrace{\top}_z) \end{aligned}$$

**Dataflow system**  $S = (Lab, E, F, (D, \sqsubseteq), \iota, \varphi)$  (continued):

- extremal value  $\iota := \delta_{\top} \in D$  where  $\delta_{\top}(x) := \top$  for every  $x \in Var_c$ ,
- transfer functions  $\{\varphi_l \mid l \in Lab\}$  defined by

$$\varphi_l(\delta) := \begin{cases} \delta & \text{if } B^l = \text{skip or } B^l \in BExp \\ \delta[x \mapsto \mathfrak{A}[[a]]\delta] & \text{if } B^l = (x := a) \end{cases}$$

where

$$\begin{aligned} \mathfrak{A}[[x]]\delta &:= \delta(x) \\ \mathfrak{A}[[z]]\delta &:= z \end{aligned} \quad \mathfrak{A}[[a_1 \text{ op } a_2]]\delta := \begin{cases} z_1 \text{ op } z_2 & \text{if } z_1, z_2 \in \mathbb{Z} \\ \perp & \text{if } z_1 = \perp \text{ or } z_2 = \perp \\ \top & \text{otherwise} \end{cases}$$

if  $z_1 := \mathfrak{A}[[a_1]]\delta$  and  $z_2 := \mathfrak{A}[[a_2]]\delta$



## Example 19.3

If  $\delta = (\underbrace{\perp}_w, \underbrace{1}_x, \underbrace{2}_y, \underbrace{\top}_z)$ , then

$$\varphi_l(\delta) = \begin{cases} (\underbrace{0}_w, \underbrace{1}_x, \underbrace{2}_y, \underbrace{\top}_z) & \text{if } B^l = (w := 0) \\ (\underbrace{3}_w, \underbrace{1}_x, \underbrace{2}_y, \underbrace{\top}_z) & \text{if } B^l = (w := y+1) \\ (\underbrace{\perp}_w, \underbrace{1}_x, \underbrace{2}_y, \underbrace{\top}_z) & \text{if } B^l = (w := w+x) \\ (\underbrace{\top}_w, \underbrace{1}_x, \underbrace{2}_y, \underbrace{\top}_z) & \text{if } B^l = (w := z+2) \end{cases}$$

## Example 19.4

Constant Propagation Analysis for

$c := [x := 1]^1;$	$\varphi_1((a, b, c, d)) = (a, 1, c, d)$
$[y := 1]^2;$	$\varphi_2((a, b, c, d)) = (a, b, 1, d)$
$[z := 1]^3;$	$\varphi_3((a, b, c, d)) = (a, b, c, 1)$
$\text{while } [z > 0]^4 \text{ do}$	$\varphi_4((a, b, c, d)) = (a, b, c, d)$
$[w := x+y]^5;$	$\varphi_5((a, b, c, d)) = (b + c, b, c, d)$
$\text{if } [w = 2]^6 \text{ then}$	$\varphi_6((a, b, c, d)) = (a, b, c, d)$
$[x := y+2]^7$	$\varphi_7((a, b, c, d)) = (a, c + 2, c, d)$

- ❶ Fixpoint solution (on the board)
- ❷ MOP solution (on the board)

- 1 Repetition: The MOP Solution
- 2 Another Analysis: Constant Propagation
- 3 Undecidability of the MOP Solution
- 4 MOP vs. Fixpoint Solution

# Undecidability of the MOP Solution

## Theorem 19.5 (Undecidability of MOP solution)

*The MOP solution for Constant Propagation is undecidable.*

### Proof.

Based on undecidability of **Modified Post Correspondence Problem**:

Let  $\Gamma$  be some alphabet,  $n \in \mathbb{N}$ , and  $u_1, \dots, u_n, v_1, \dots, v_n \in \Gamma^+$ .

Does there exist  $i_1, \dots, i_m \in \{1, \dots, n\}$  with  $m \geq 1$  and  $i_1 = 1$  such that  $u_{i_1}u_{i_2} \dots u_{i_m} = v_{i_1}v_{i_2} \dots v_{i_m}$ ?

(on the board)

