

Semantics and Verification of Software

Lecture 19: Dataflow Analysis VI (The MOP Solution)

Thomas Noll

Lehrstuhl für Informatik 2
(Software Modeling and Verification)

RWTH Aachen University

`noll@cs.rwth-aachen.de`

`http://www-i2.informatik.rwth-aachen.de/i2/svsw08/`

Winter semester 2008/09

Online Registration for Seminars and Practical Courses (Praktika) in Summer Term 2009

Who?

- Students of:
- Hauptstudium Informatik
 - Master Courses
 - Bachelor Informatik (~~Pro~~Seminar!)

Where?

www.graphics.rwth-aachen.de/apse

When?

05.01.2009 - 18.01.2009

Seminar: Applying Formal Verification Methods to Embedded Systems

- Joint weekly Seminar with Embedded Software Laboratory
- Theoretical and Practical CS
- Topics:
 - Static program analysis
 - Abstract interpretation
 - Software model checking (of assembly and source code)
 - Analysis of timed behavior
 - Resource awareness
- Requirements:
 - Vordiplom/Bachelor
 - In particular: Automata Theory and Formal Languages
 - Helpful: basic knowledge in
 - this course
 - (Formal Methods for) Embedded Systems
 - Model Checking Technology

- 1 The MOP Solution
- 2 Another Analysis: Constant Propagation
- 3 Undecidability of the MOP Solution

The MOP Solution I

- Other **solution method** for dataflow systems

The MOP Solution I

- Other **solution method** for dataflow systems
- MOP = **Meet Over all Paths**

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**

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 19.1 (Paths)

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

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

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 19.1 (Paths)

Let $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$ be a dataflow system. For every $l \in L$, 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 19.2 (MOP solution)

Let $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$ be a dataflow system where $L = \{l_1, \dots, l_n\}$. 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 $l \in L$,

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

Definition 19.2 (MOP solution)

Let $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$ be a dataflow system where $L = \{l_1, \dots, l_n\}$. 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 $l \in L$,

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

Remark:

- $\text{Path}(l)$ is generally infinite

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

Definition 19.2 (MOP solution)

Let $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$ be a dataflow system where $L = \{l_1, \dots, l_n\}$. 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 $l \in L$,

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

Remark:

- $\text{Path}(l)$ is generally infinite

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

- In fact: MOP solution generally undecidable (later)

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

```
c = [x := 2]1;  
    [y := 4]2;  
    [x := 1]3;  
    if [y > 0]4 then  
        [z := x]5  
    else  
        [z := y*y]6;  
    [x := z]7
```

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

```
c = [x := 2]1;  
    [y := 4]2;  
    [x := 1]3;  
    if [y > 0]4 then  
        [z := x]5  
    else  
        [z := y*y]6;  
        [x := z]7  
  
⇒ Path(1) = {[7, 5, 4, 3, 2],  
              [7, 6, 4, 3, 2]}
```

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned} c = & [x := 2]^1; & \implies \text{mop}(1) = \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\ & [y := 4]^2; \\ & [x := 1]^3; \\ & \text{if } [y > 0]^4 \text{ then} \\ & \quad [z := x]^5 \\ & \text{else} \\ & \quad [z := y*y]^6; \\ & \quad [x := z]^7 \\ \implies \text{Path}(1) = & \{ [7, 5, 4, 3, 2], \\ & \quad [7, 6, 4, 3, 2] \} \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned} c = & [x := 2]^1; & \implies \text{mop}(1) = & \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\ & [y := 4]^2; & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{x, y, z\})))) \sqcup \\ & [x := 1]^3; & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{x, y, z\})))) \\ & \text{if } [y > 0]^4 \text{ then} \\ & \quad [z := x]^5 \\ & \text{else} \\ & \quad [z := y*y]^6; \\ & \quad [x := z]^7 \\ \implies \text{Path}(1) = & \{ [7, 5, 4, 3, 2], \\ & \quad [7, 6, 4, 3, 2] \} \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned}
 c = & [x := 2]^1; & \implies \text{mop}(1) = & \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\
 & [y := 4]^2; & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{x, y, z\})))) \sqcup \\
 & [x := 1]^3; & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{x, y, z\})))) \\
 & \text{if } [y > 0]^4 \text{ then} & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\{y, z\}))) \sqcup \\
 & \quad [z := x]^5 & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\{y, z\})))) \\
 & \text{else} & & \\
 & \quad [z := y*y]^6; & & \\
 & [x := z]^7 & & \\
 \implies \text{Path}(1) = & \{ [7, 5, 4, 3, 2], \\
 & [7, 6, 4, 3, 2] \}
 \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned}
 c = & [x := 2]^1; & \implies \text{mop}(1) = & \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\
 & [y := 4]^2; & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{x, y, z\})))) \sqcup \\
 & [x := 1]^3; & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{x, y, z\})))) \\
 & \text{if } [y > 0]^4 \text{ then} & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\{y, z\}))) \sqcup \\
 & \quad [z := x]^5 & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\{y, z\})))) \\
 & \text{else} & & = \varphi_2(\varphi_3(\varphi_4(\{x, y\}))) \sqcup \\
 & \quad [z := y*y]^6; & & \varphi_2(\varphi_3(\varphi_4(\{y\}))) \\
 & [x := z]^7 \\
 \implies \text{Path}(1) = & \{ [7, 5, 4, 3, 2], \\
 & [7, 6, 4, 3, 2] \}
 \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned}
 c = & [x := 2]^1; & \implies \text{mop}(1) = & \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\
 & [y := 4]^2; & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{\mathbf{x}, \mathbf{y}, \mathbf{z}\})))) \sqcup \\
 & [x := 1]^3; & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{\mathbf{x}, \mathbf{y}, \mathbf{z}\})))) \\
 & \text{if } [y > 0]^4 \text{ then} & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\{\mathbf{y}, \mathbf{z}\}))) \sqcup \\
 & \quad [z := x]^5 & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\{\mathbf{y}, \mathbf{z}\})))) \\
 & \text{else} & & = \varphi_2(\varphi_3(\varphi_4(\{\mathbf{x}, \mathbf{y}\}))) \sqcup \\
 & \quad [z := y*y]^6; & & \varphi_2(\varphi_3(\varphi_4(\{\mathbf{y}\}))) \\
 & [x := z]^7 & & = \varphi_2(\varphi_3(\{\mathbf{x}, \mathbf{y}\})) \sqcup \varphi_2(\varphi_3(\{\mathbf{y}\})) \\
 \implies \text{Path}(1) = & \{ [7, 5, 4, 3, 2], \\
 & [7, 6, 4, 3, 2] \}
 \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned}
 c = & [x := 2]^1; & \implies \text{mop}(1) = & \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\
 & [y := 4]^2; & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{x, y, z\})))) \sqcup \\
 & [x := 1]^3; & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{x, y, z\})))) \\
 & \text{if } [y > 0]^4 \text{ then} & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\{y, z\}))) \sqcup \\
 & \quad [z := x]^5 & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\{y, z\})))) \\
 & \text{else} & & = \varphi_2(\varphi_3(\varphi_4(\{x, y\}))) \sqcup \\
 & \quad [z := y*y]^6; & & \varphi_2(\varphi_3(\varphi_4(\{y\}))) \\
 & \quad [x := z]^7 & & = \varphi_2(\varphi_3(\{x, y\})) \sqcup \varphi_2(\varphi_3(\{y\})) \\
 & & & = \varphi_2(\{y\}) \sqcup \varphi_2(\{y\}) \\
 \implies \text{Path}(1) = & \{[7, 5, 4, 3, 2], \\
 & \quad [7, 6, 4, 3, 2]\}
 \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{aligned}
 c = & [x := 2]^1; & \implies \text{mop}(1) = & \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\
 & [y := 4]^2; & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{x, y, z\})))) \sqcup \\
 & [x := 1]^3; & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{x, y, z\})))) \\
 & \text{if } [y > 0]^4 \text{ then} & & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\{y, z\}))) \sqcup \\
 & \quad [z := x]^5 & & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\{y, z\})))) \\
 & \text{else} & & = \varphi_2(\varphi_3(\varphi_4(\{x, y\}))) \sqcup \\
 & \quad [z := y*y]^6; & & \varphi_2(\varphi_3(\varphi_4(\{y\}))) \\
 & \quad [x := z]^7 & & = \varphi_2(\varphi_3(\{x, y\})) \sqcup \varphi_2(\varphi_3(\{y\})) \\
 & & & = \varphi_2(\{y\}) \sqcup \varphi_2(\{y\}) \\
 \implies \text{Path}(1) = & \{[7, 5, 4, 3, 2], & & = \emptyset \sqcup \emptyset \\
 & [7, 6, 4, 3, 2]\}
 \end{aligned}$$

Example 19.3 (Live Variables; cf. Examples 15.3 and 17.4)

$$\begin{array}{ll}
 c = [x := 2]^1; & \implies \text{mop}(1) = \varphi_{[7,5,4,3,2]}(\iota) \sqcup \varphi_{[7,6,4,3,2]}(\iota) \\
 [y := 4]^2; & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\varphi_7(\{x, y, z\})))) \sqcup \\
 [x := 1]^3; & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\varphi_7(\{x, y, z\})))) \\
 \text{if } [y > 0]^4 \text{ then} & = \varphi_2(\varphi_3(\varphi_4(\varphi_5(\{y, z\}))) \sqcup \\
 [z := x]^5 & \varphi_2(\varphi_3(\varphi_4(\varphi_6(\{y, z\})))) \\
 \text{else} & = \varphi_2(\varphi_3(\varphi_4(\{x, y\}))) \sqcup \\
 [z := y*y]^6; & \varphi_2(\varphi_3(\varphi_4(\{y\}))) \\
 [x := z]^7 & = \varphi_2(\varphi_3(\{x, y\})) \sqcup \varphi_2(\varphi_3(\{y\})) \\
 & = \varphi_2(\{y\}) \sqcup \varphi_2(\{y\}) \\
 \implies \text{Path}(1) = \{[7, 5, 4, 3, 2], & = \emptyset \sqcup \emptyset \\
 [7, 6, 4, 3, 2]\} & = \emptyset
 \end{array}$$

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

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

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 19.4 (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
```

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 19.4 (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

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 19.4 (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**

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 19.4 (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:
 $[w := x+1]⁵ [x := 3]⁷$

Formalizing Constant Propagation Analysis I

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

- set of labels $L := L_c$,
- extremal labels $E := \{\text{init}(c)\}$ (forward problem),
- flow relation $F := \text{flow}(c)$ (forward problem),
- complete lattice $(D, \sqsubseteq)$ where
 - $D := \{\delta \mid \delta : \text{Var}_c \rightarrow \mathbb{Z} \cup \{\perp, \top\}\}$
 - $\delta(x) = z \in \mathbb{Z}$: 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 $z \in \mathbb{Z}$)

Formalizing Constant Propagation Analysis I

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

- set of labels $L := L_c$,
- extremal labels $E := \{\text{init}(c)\}$ (forward problem),
- flow relation $F := \text{flow}(c)$ (forward problem),
- complete lattice $(D, \sqsubseteq)$ where
 - $D := \{\delta \mid \delta : \text{Var}_c \rightarrow \mathbb{Z} \cup \{\perp, \top\}\}$
 - $\delta(x) = z \in \mathbb{Z}$: 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 19.5

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

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

- extremal value $\iota := \delta_{\top} \in D$ where $\delta_{\top}(x) := \top$ for every $x \in \text{Var}_c$,
- transfer functions $\{\varphi_l \mid l \in L\}$ 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 19.6

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 19.7

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)