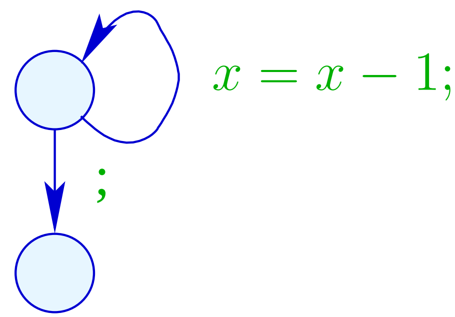
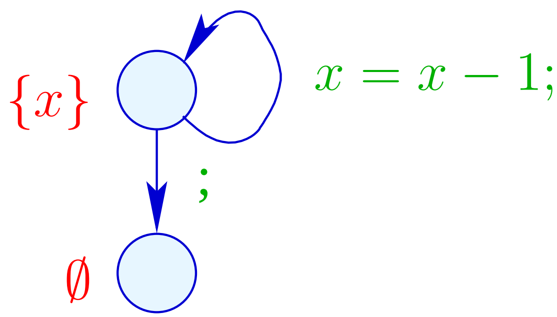
- True liveness detects more superfluous assignments than repeated liveness !!!



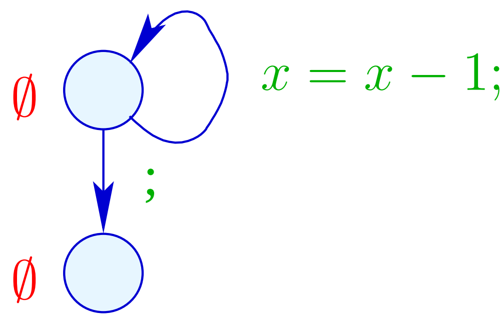
- True liveness detects more superfluous assignments than repeated liveness !!!

Liveness:



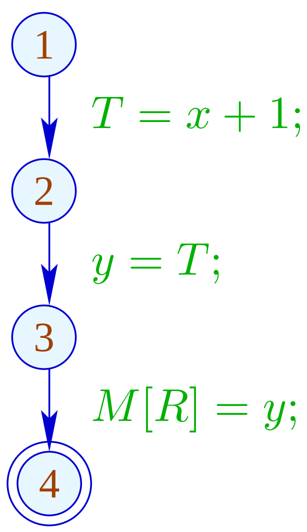
- True liveness detects more superfluous assignments than repeated liveness !!!

True Liveness:



1.3 Removing Superfluous Moves

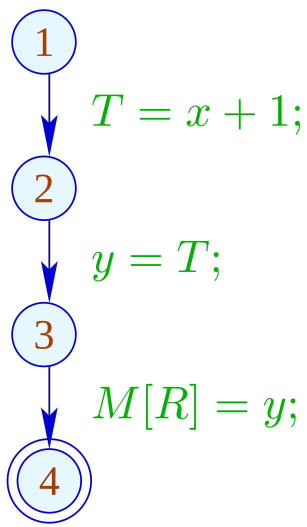
Example:



This variable-variable assignment is obviously useless :-(

1.3 Removing Superfluous Moves

Example:

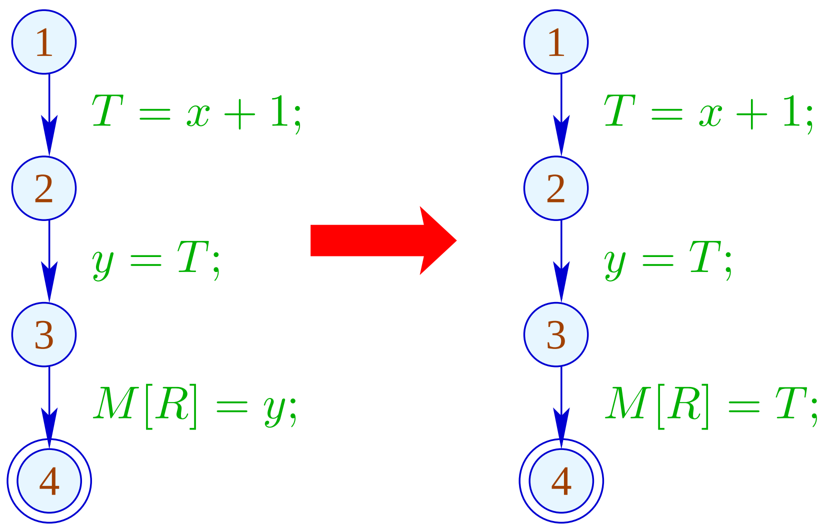


This variable-variable assignment is obviously useless :-(

Instead of y, we could also store T :-)

1.3 Removing Superfluous Moves

Example:

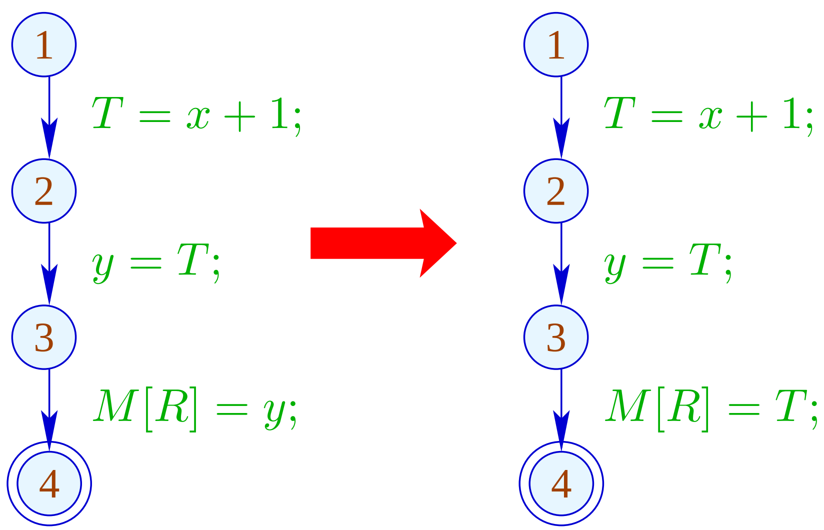


This variable-variable assignment is obviously useless :-(

Instead of y, we could also store T :-)

1.3 Removing Superfluous Moves

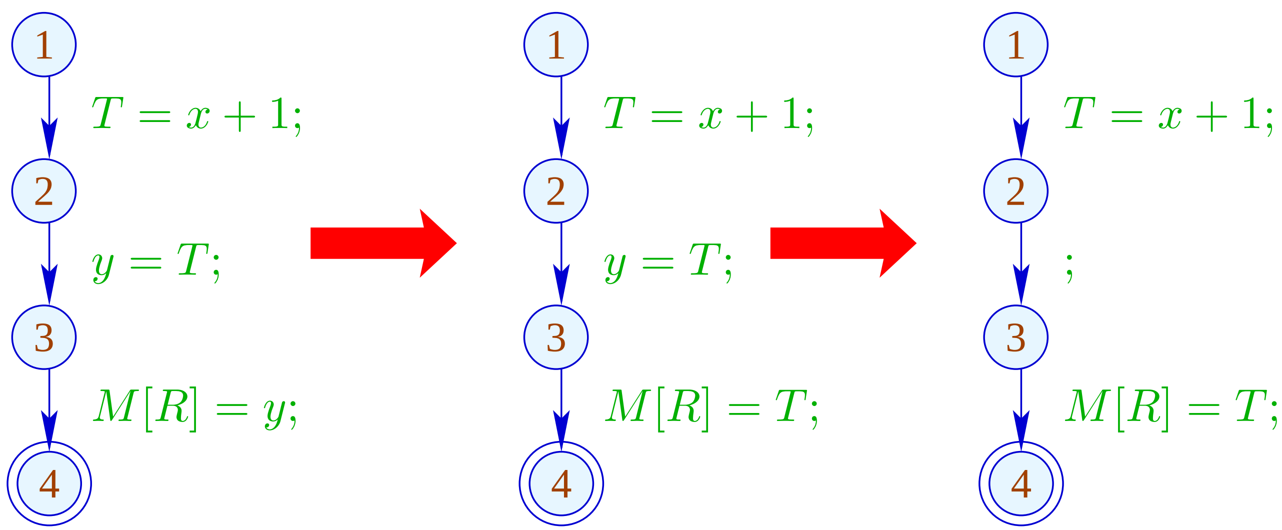
Example:



Advantage: Now, y has become dead :-))

1.3 Removing Superfluous Moves

Example:



Advantage: Now, y has become dead :-))

Idea:

For each expression, we record the variable which currently contains its value :-)

We use: $\mathbb{V} = \mathit{Expr} \to 2^{\mathit{Vars}}$ ...

Idea:

For each expression, we record the variable which currently contains its value :-)

We use: $\mathbb{V} = \mathit{Expr} \to 2^{\mathit{Vars}}$ and define:

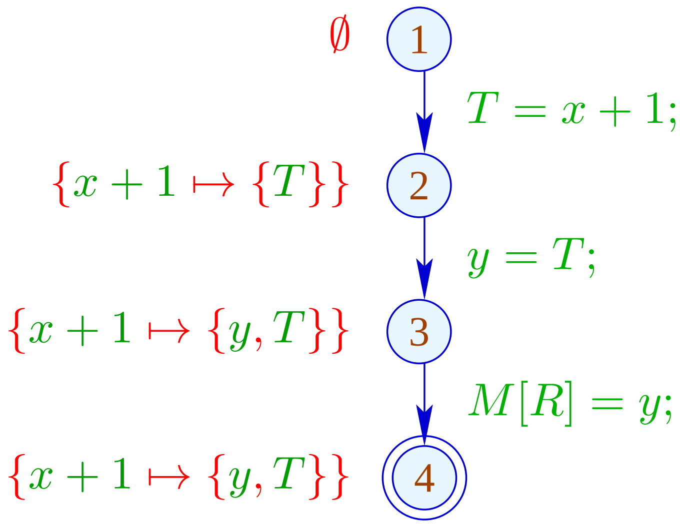
$$
\begin{array}{lllll}
[\![;]\!]^\sharp\, V & = & V & & \\
[\![\mathrm{Pos}(e)]\!]^\sharp\, V\, e' & = & [\![\mathrm{Neg}(e)]\!]^\sharp\, V\, e' & = & \begin{cases} \emptyset & \text{if } e' = e \\ V\, e' & \text{otherwise} \end{cases}
\end{array}
$$

$\ldots$

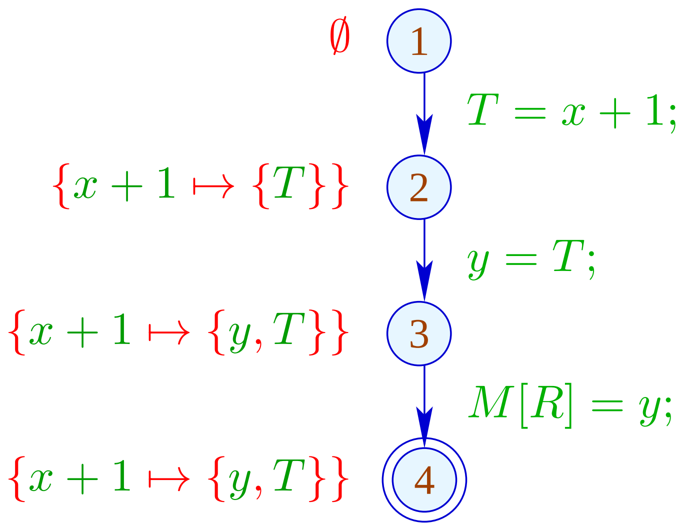
$$
\begin{array}{lll}
[\![x = c;]\!]^\sharp\, V\, e' & = & \begin{cases} (V\, c) \cup \{x\} & \text{if} \quad e' = c \\ (V\, e') \backslash \{x\} & \text{otherwise} \end{cases} \\
[\![x = y;]\!]^\sharp\, V\, e & = & \begin{cases} (V\, e) \cup \{x\} & \text{if} \quad y \in V\, e \\ (V\, e) \backslash \{x\} & \text{otherwise} \end{cases} \\
[\![x = e;]\!]^\sharp\, V\, e' & = & \begin{cases} \{x\} & \text{if} \quad e' = e \\ (V\, e') \backslash \{x\} & \text{otherwise} \end{cases} \\
[\![x = M[c];]\!]^\sharp\, V\, e' & = & (V\, e') \backslash \{x\} \\
[\![x = M[y];]\!]^\sharp\, V\, e' & = & (V\, e') \backslash \{x\} \\
[\![x = M[e];]\!]^\sharp\, V\, e' & = & \begin{cases} \emptyset & \text{if } e' = e \\ (V\, e') \backslash \{x\} & \text{otherwise} \end{cases}
\end{array}
$$

// analogously for the diverse stores

In the Example:



In the Example:



$\rightarrow$ We propagate information in forward direction :-)

At *start*, $V_0\, e = \emptyset$ for all e;

$\rightarrow$ $\sqsubseteq\ \subseteq\ \mathbb{V} \times \mathbb{V}$ is defined by:

$$V_1 \sqsubseteq V_2 \quad \text{iff} \quad V_1\, e \supseteq V_2\, e \quad \text{for all } e$$

Observation:

The new effects of edges are distributive:

To show this, we consider the functions:

(1) $f_1^x \, V \, e = (V \, e) \backslash \{x\}$

(2) $f_2^{e,a} \, V = V \oplus \{e \mapsto a\}$

(3) $f_3^{x,y} \, V \, e = (y \in V \, e) \, ? \, (V \, e \cup \{x\}) : ((V \, e) \backslash \{x\})$

Obviously, we have:

$$
\begin{array}{lll}
[\![x = e;]\!]^\sharp & = & f_2^{e,\{x\}} \circ f_1^x \\
[\![x = y;]\!]^\sharp & = & f_3^{x,y} \\
[\![x = M[e];]\!]^\sharp & = & f_2^{e,\emptyset} \circ f_1^x
\end{array}
$$

By closure under composition, the assertion follows :-))

(1) For $f\,V\,e = (V\,e)\backslash\{x\}$, we have:

$$
\begin{array}{lll}
f\,(V_1 \sqcup V_2)\,e & = & ((V_1 \sqcup V_2)\,e)\backslash\{x\} \\
& = & ((V_1\,e) \cap (V_2\,e))\backslash\{x\} \\
& = & ((V_1\,e)\backslash\{x\}) \cap ((V_2\,e)\backslash\{x\}) \\
& = & (f\,V_1\,e) \cap (f\,V_2\,e) \\
& = & (f\,V_1 \sqcup f\,V_2)\,e \qquad \text{:-)}
\end{array}
$$

(2) For $f\,V = V \oplus \{e \mapsto a\}$, we have:

$$
\begin{array}{lll}
f\,(V_1 \sqcup V_2)\,e' & = & ((V_1 \sqcup V_2) \oplus \{e \mapsto a\})\,e' \\
& = & (V_1 \sqcup V_2)\,e' \\
& = & (f\,V_1 \sqcup f\,V_2)\,e' \qquad \text{given that} \quad e \neq e'
\end{array}
$$

$$
\begin{array}{lll}
f\,(V_1 \sqcup V_2)\,e & = & ((V_1 \sqcup V_2) \oplus \{e \mapsto a\})\,e \\
& = & a \\
& = & ((V_1 \oplus \{e \mapsto a\})\,e) \cap ((V_2 \oplus \{e \mapsto a\})\,e) \\
& = & (f\,V_1 \sqcup f\,V_2)\,e \qquad \text{:-)}
\end{array}
$$

(3) For $f\,V\,e = (y \in V\;e)\;?\;(V\;e \cup \{x\}) : ((V\;e)\backslash\{x\})$, we have:

$$\begin{aligned}
f\,(V_1 \sqcup V_2)\,e &= (((V_1 \sqcup V_2)\,e)\backslash\{x\}) \cup (y \in (V_1 \sqcup V_2)\,e)\;?\;\{x\} : \emptyset \\
&= ((V_1\,e \cap V_2\,e)\backslash\{x\}) \cup (y \in (V_1\,e \cap V_2\,e))\;?\;\{x\} : \emptyset \\
&= ((V_1\,e \cap V_2\,e)\backslash\{x\}) \cup \\
&\quad ((y \in V_1\,e)\;?\;\{x\} : \emptyset) \cap ((y \in V_2\,e)\;?\;\{x\} : \emptyset) \\
&= (((V_1\,e)\backslash\{x\}) \cup (y \in V_1\,e)\;?\;\{x\} : \emptyset) \cap \\
&\quad (((V_2\,e)\backslash\{x\}) \cup (y \in V_2\,e)\;?\;\{x\} : \emptyset) \\
&= (f\,V_1 \sqcup f\,V_2)\,e \qquad \text{:-)}
\end{aligned}$$

We conclude:

→ Solving the constraint system returns the MOP solution :-)

→ Let $\mathcal{V}$ denote this solution.

If $x \in \mathcal{V}[u]\, e$, then x at u contains the value of e — which we have stored in T_e

$\Longrightarrow$

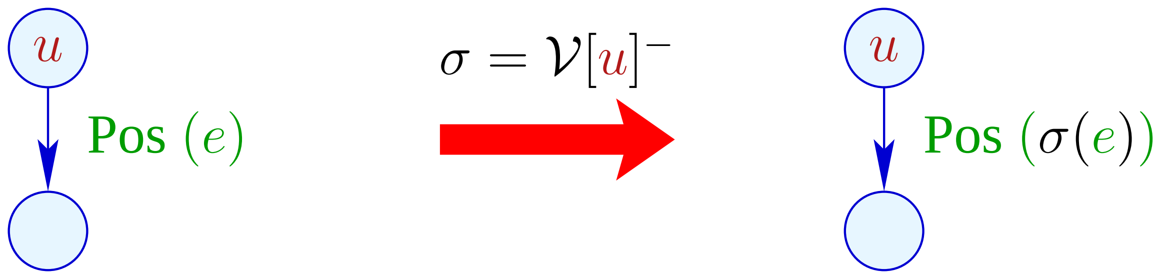
the access to x can be replaced by the access to T_e :-)

For $V \in \mathbb{V}$, let V^- denote the variable substitution with:

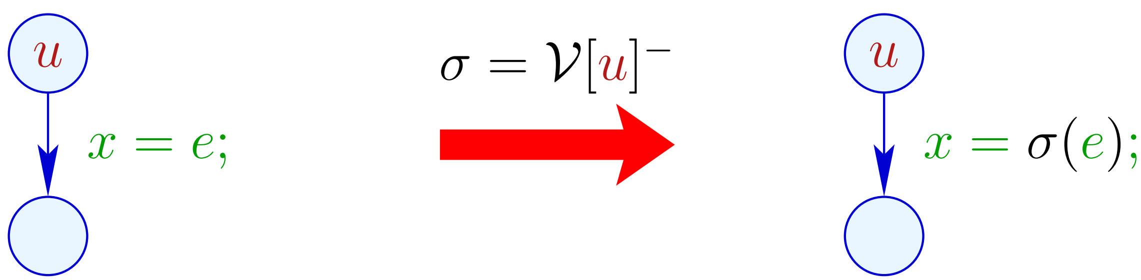
$$V^- x = \begin{cases} T_e & \text{if } x \in V\, e \\ x & \text{otherwise} \end{cases}$$

if $V\, e \cap V\, e' = \emptyset$ for $e \neq e'$. Otherwise: $V^- x = x$:-)

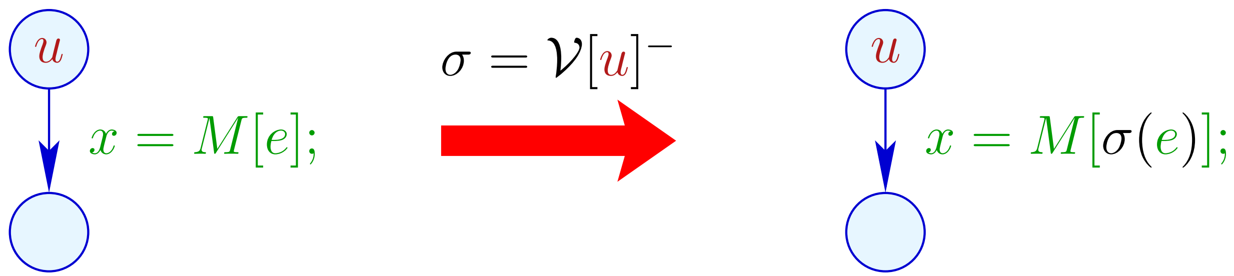
Transformation 3:

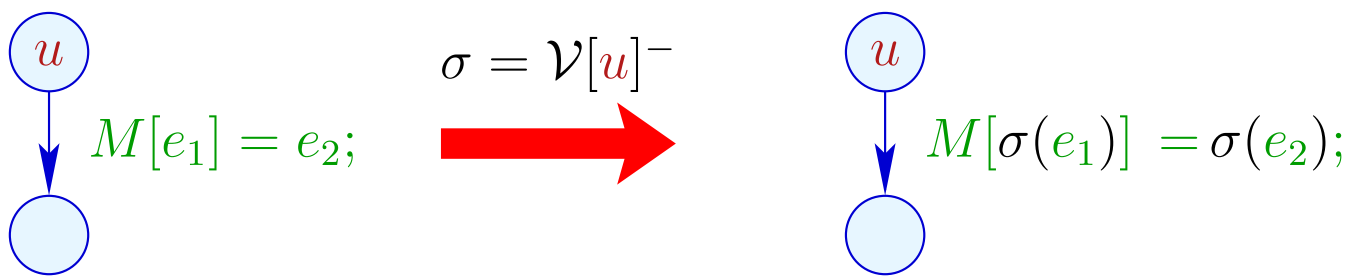


... analogously for edges with Neg (e)



Transformation 3 (cont.):





Procedure as a whole:

(1) Availability of expressions: T1

- \+ removes arithmetic operations
- – inserts superfluous moves

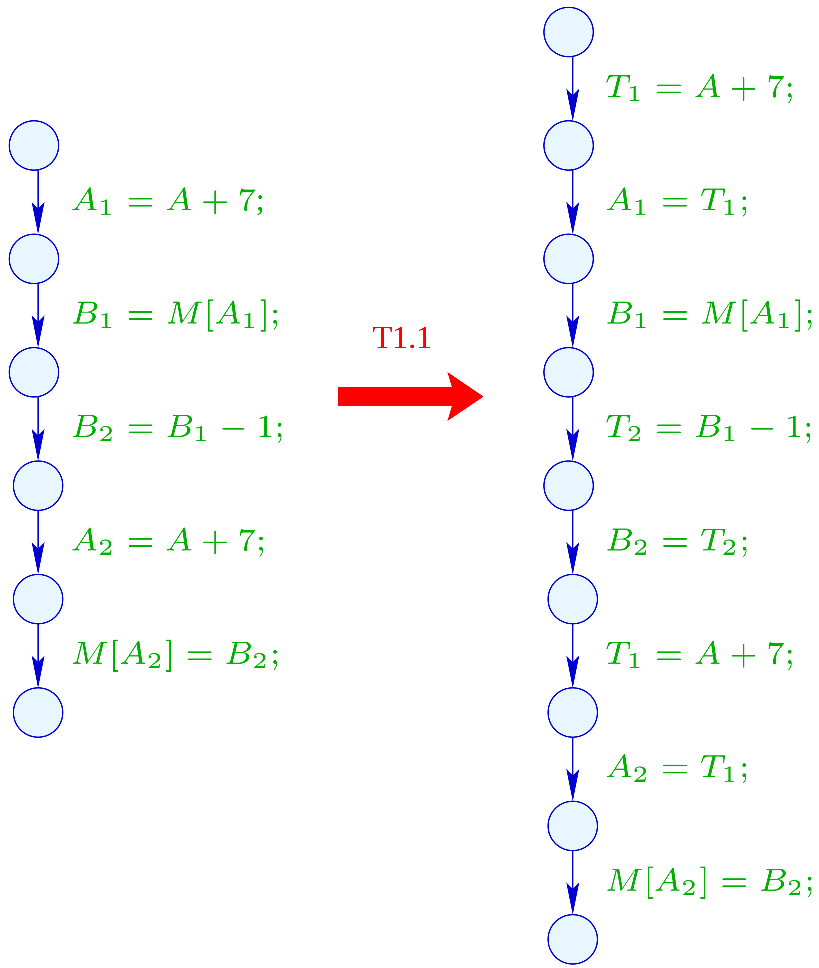
(2) Values of variables: T3

- \+ creates dead variables

(3) (true) liveness of variables: T2

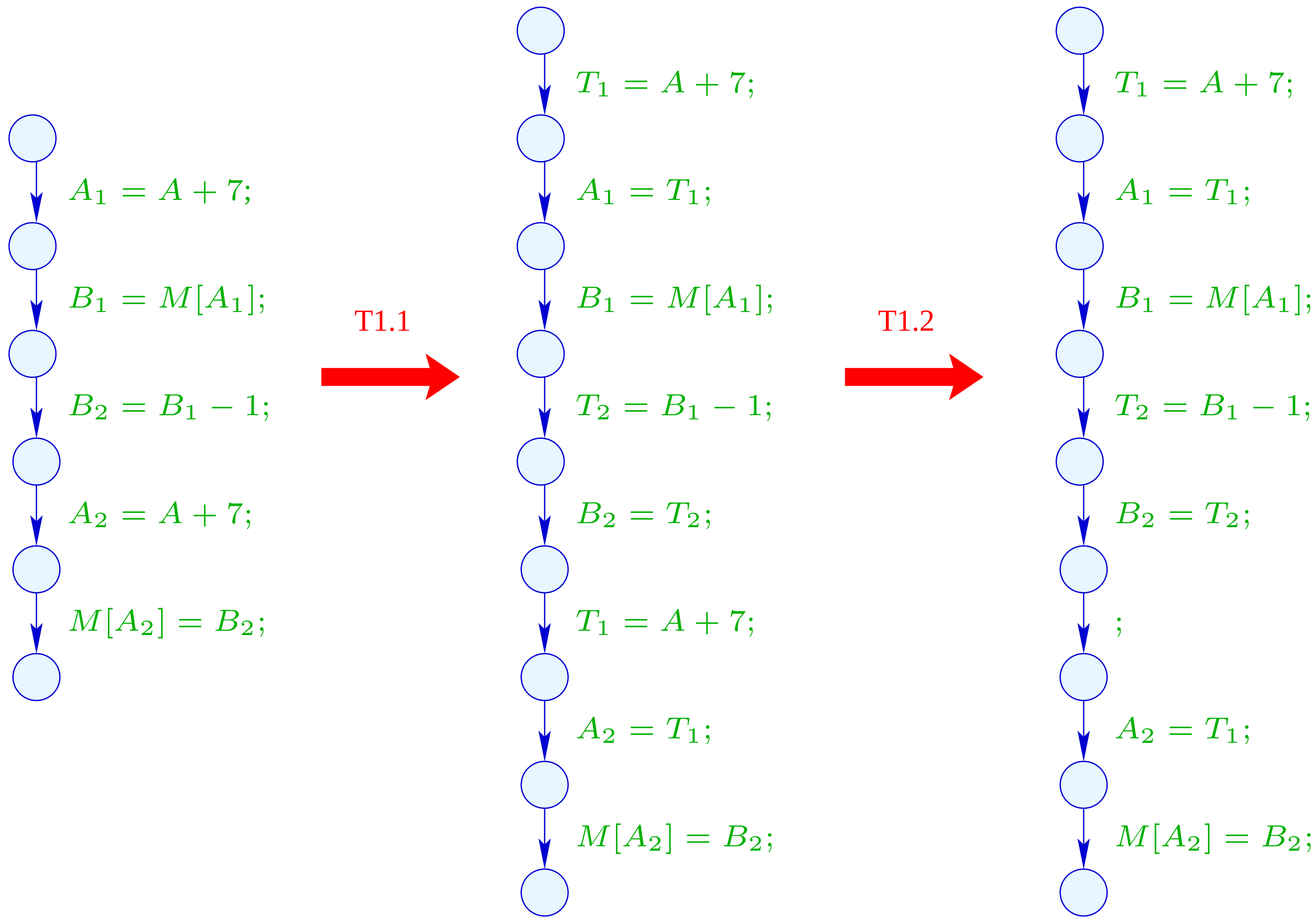
- \+ removes assignments to dead variables

Example: `a[7]--;`

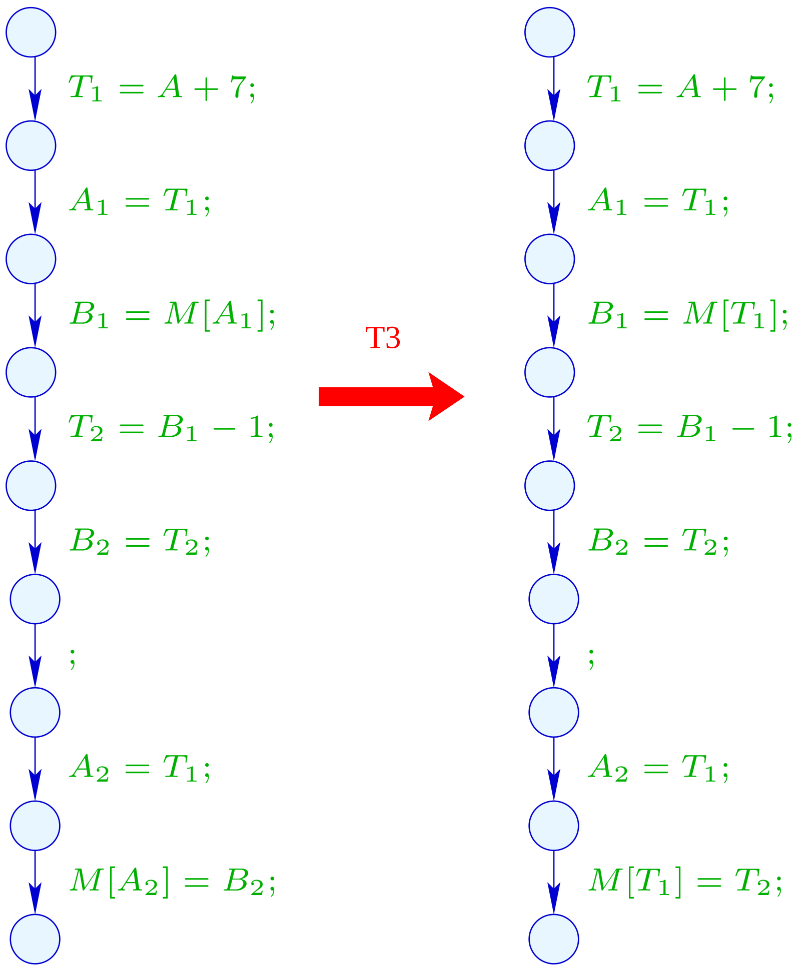


$A_1 = A + 7;$
$B_1 = M[A_1];$
$B_2 = B_1 - 1;$
$A_2 = A + 7;$
$M[A_2] = B_2;$
T1.1
$T_1 = A + 7;$
$A_1 = T_1;$
$B_1 = M[A_1];$
$T_2 = B_1 - 1;$
$B_2 = T_2;$
$T_1 = A + 7;$
$A_2 = T_1;$
$M[A_2] = B_2;$

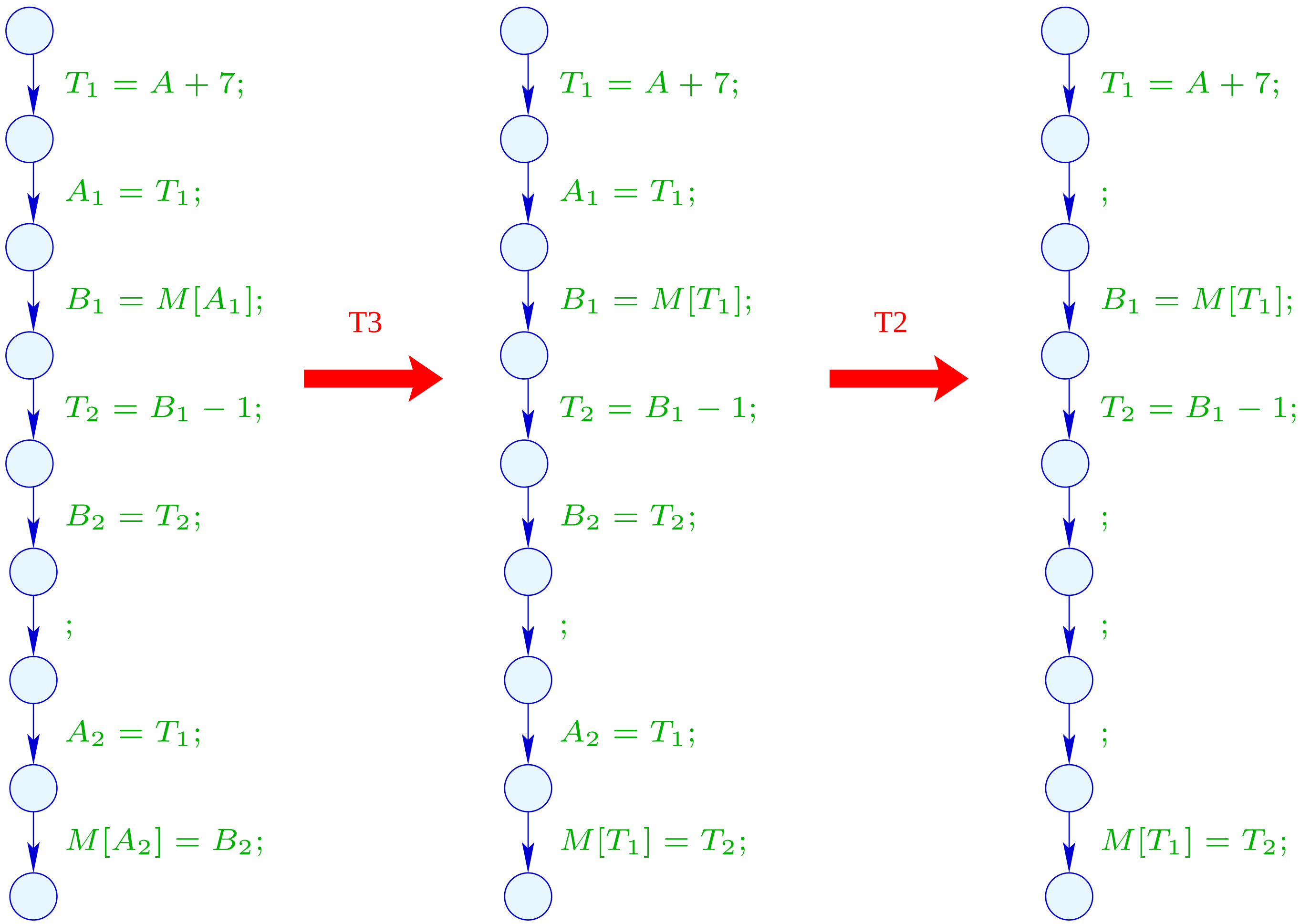
Example: `a[7]--;`


$A_1 = A + 7;$
$B_1 = M[A_1];$
$B_2 = B_1 - 1;$
$A_2 = A + 7;$
$M[A_2] = B_2;$
T1.1
$T_1 = A + 7;$
$A_1 = T_1;$
$B_1 = M[A_1];$
$T_2 = B_1 - 1;$
$B_2 = T_2;$
$T_1 = A + 7;$
$A_2 = T_1;$
$M[A_2] = B_2;$
T1.2
$T_1 = A + 7;$
$A_1 = T_1;$
$B_1 = M[A_1];$
$T_2 = B_1 - 1;$
$B_2 = T_2;$
;
$A_2 = T_1;$
$M[A_2] = B_2;$

Example (cont.): `a[7]--;`



Example (cont.): `a[7]--;`



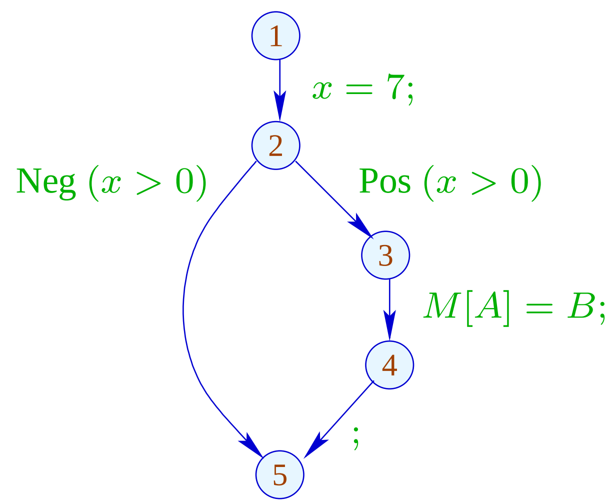
1.4 Constant Propagation

Idea:

Execute as much of the code at compile-time as possible!

Example:

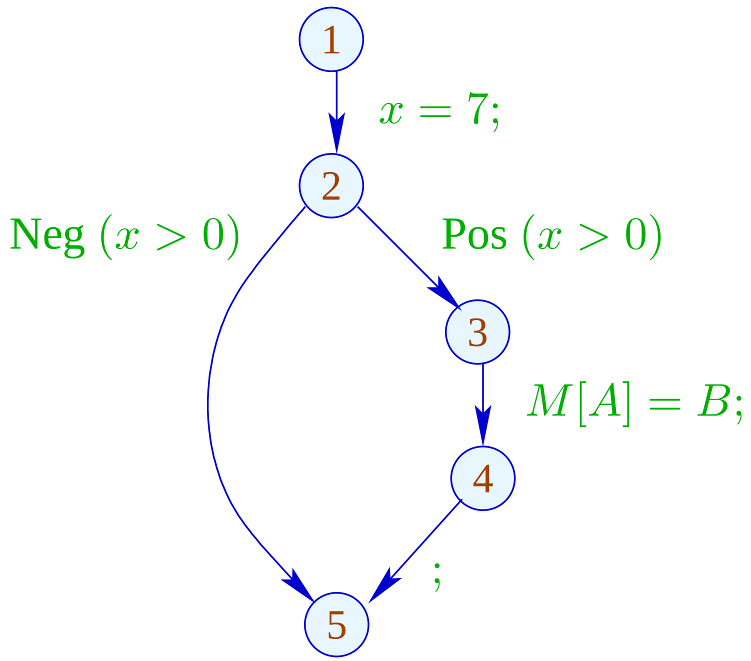
$$x = 7;$$
$$\mathsf{if}\ (x > 0)$$
$$\qquad M[A] = B;$$



Obviously, x has always the value 7 :-)

Thus, the memory access is always executed :-))

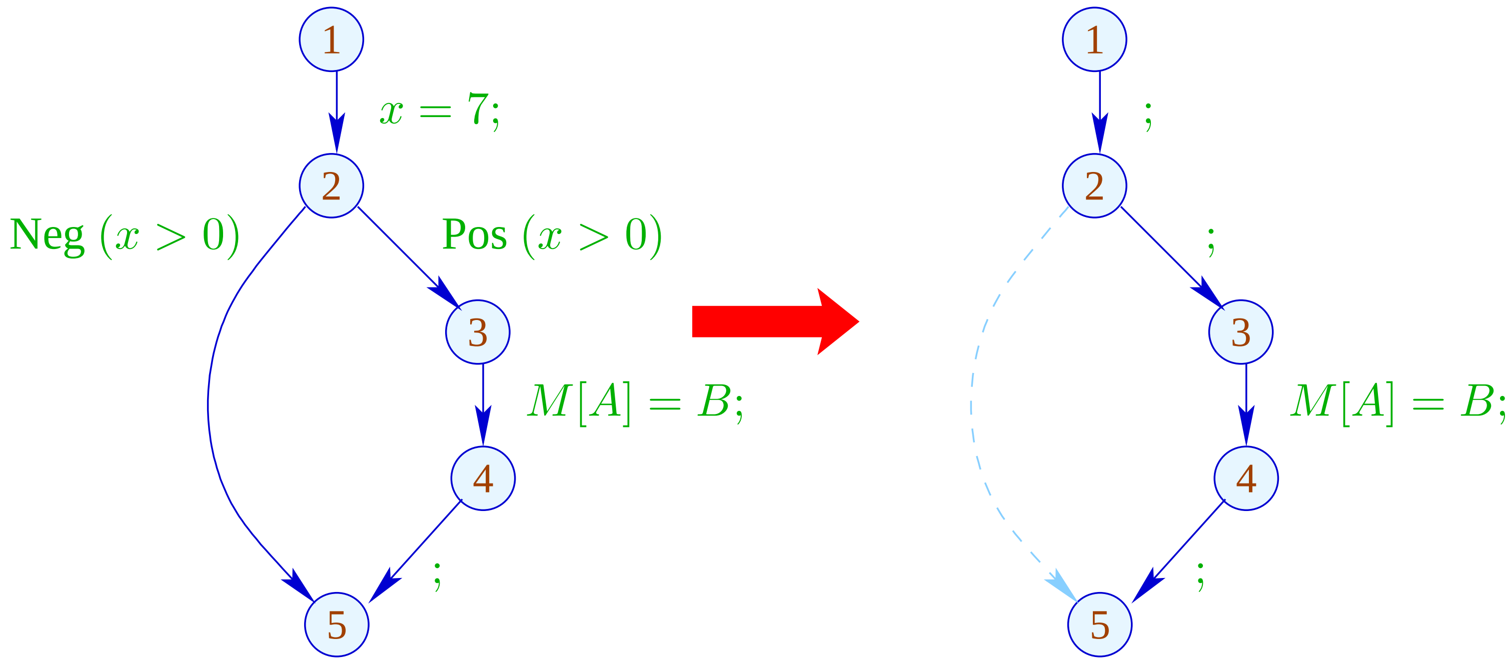
Goal:



Obviously, x has always the value 7 :-)

Thus, the memory access is always executed :-))

Goal:



Generalization: Partial Evaluation

Neil D. Jones, DIKU, Kopenhagen

Idea:

Design an analysis which for every u,

- determines the values which variables definitely have;
- tells whether u can be reached at all :-)

Idea:

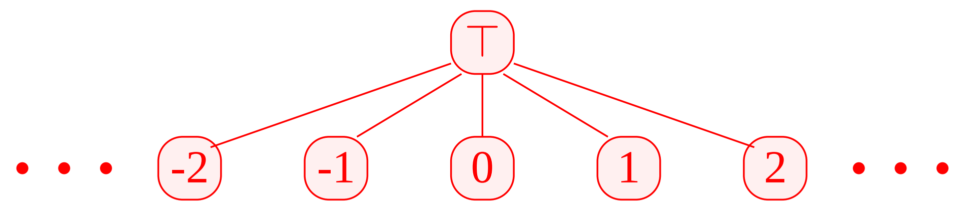
Design an analysis which for every u,

- determines the values which variables definitely have;
- tells whether u can be reached at all :-)

The complete lattice is constructed in two steps.

(1) The potential values of variables:

$$\mathbb{Z}^\top = \mathbb{Z} \cup \{\top\} \qquad \text{with} \quad x \sqsubseteq y \quad \text{iff } y = \top \text{ or } x = y$$



Caveat: $\mathbb{Z}^\top$ is not a complete lattice in itself :-(

(2) $\mathbb{D} = (\mathit{Vars} \to \mathbb{Z}^\top)_\bot = (\mathit{Vars} \to \mathbb{Z}^\top) \cup \{\bot\}$

// $\bot$ denotes: "not reachable" :-))

with $D_1 \sqsubseteq D_2$ iff $\bot = D_1$ or

$D_1\, x \sqsubseteq D_2\, x \quad (x \in \mathit{Vars})$

Remark: $\mathbb{D}$ is a complete lattice :-)

Caveat: $\mathbb{Z}^\top$ is not a complete lattice in itself :-(

(2) $\mathbb{D} = (\mathit{Vars} \to \mathbb{Z}^\top)_\perp = (\mathit{Vars} \to \mathbb{Z}^\top) \cup \{\perp\}$

// $\perp$ denotes: “not reachable” :-))

with $D_1 \sqsubseteq D_2$ iff $\perp = D_1$ or

$D_1\, x \sqsubseteq D_2\, x \quad (x \in \mathit{Vars})$

Remark: $\mathbb{D}$ is a complete lattice :-)

Consider $X \subseteq \mathbb{D}$. W.l.o.g., $\perp \notin X$.

Then $X \subseteq \mathit{Vars} \to \mathbb{Z}^\top$.

If $X = \emptyset$, then $\bigsqcup X = \perp \in \mathbb{D}$:-)

If $X \neq \emptyset$, then $\bigsqcup X = D$ with

$$D\, x = \bigsqcup \{ f\, x \mid f \in X \}$$
$$= \begin{cases} z & \text{if} \quad f\, x = z \quad (f \in X) \\ \top & \text{otherwise} \end{cases}$$

:-))

If $X \neq \emptyset$, then $\bigsqcup X = D$ with

$$D\,x \;=\; \bigsqcup\{f\,x \mid f \in X\}$$
$$= \begin{cases} z & \text{if} \quad f\,x = z \quad (f \in X) \\ \top & \text{otherwise} \end{cases}$$

:-))

For every edge $k = (\_, lab, \_)$, construct an effect function $[\![k]\!]^\sharp = [\![lab]\!]^\sharp \;:\; \mathbb{D} \to \mathbb{D}$ which simulates the concrete computation.

Obviously, $[\![lab]\!]^\sharp \perp = \perp$ for all lab :-)

Now let $\perp \neq D \in \mathit{Vars} \to \mathbb{Z}^\top$.

Idea:

- We use D to determine the values of expressions.

Idea:

- We use D to determine the values of expressions.
- For some sub-expressions, we obtain $\top$:-)

Idea:

- We use D to determine the values of expressions.
- For some sub-expressions, we obtain $\top$:-)

  $\Longrightarrow$

  We must replace the concrete operators $\Box$ by abstract operators $\Box^\sharp$ which can handle $\top$:

$$a \, \Box^\sharp \, b = \begin{cases} \top & \text{if} \quad a = \top \text{ or } b = \top \\ a \, \Box \, b & \text{otherwise} \end{cases}$$

Idea:

- We use D to determine the values of expressions.
- For some sub-expressions, we obtain $\top$:-)

  $\Longrightarrow$

  We must replace the concrete operators $\Box$ by abstract operators $\Box^\sharp$ which can handle $\top$:

$$a \,\Box^\sharp\, b = \begin{cases} \top & \text{if} \quad a = \top \text{ or } b = \top \\ a \,\Box\, b & \text{otherwise} \end{cases}$$

- The abstract operators allow to define an abstract evaluation of expressions:

$$[\![e]\!]^\sharp \;:\; (Vars \to \mathbb{Z}^\top) \to \mathbb{Z}^\top$$

Abstract evaluation of expressions is like the concrete evaluation — but with abstract values and operators. Here:

$$
\begin{array}{lll}
[\![c]\!]^\sharp\, D & = & c \\
[\![e_1 \,\square\, e_2]\!]^\sharp\, D & = & [\![e_1]\!]^\sharp\, D \,\square^\sharp\, [\![e_2]\!]^\sharp\, D
\end{array}
$$

... analogously for unary operators :-)

Abstract evaluation of expressions is like the concrete evaluation — but with abstract values and operators. Here:

$$\begin{array}{lll} [\![c]\!]^\sharp\, D & = & c \\ [\![e_1 \,\Box\, e_2]\!]^\sharp\, D & = & [\![e_1]\!]^\sharp\, D \,\Box^\sharp\, [\![e_2]\!]^\sharp\, D \end{array}$$

... analogously for unary operators :-)

Example: $D = \{x \mapsto 2, y \mapsto \top\}$

$$\begin{array}{lll} [\![x + 7]\!]^\sharp\, D & = & [\![x]\!]^\sharp\, D \;+^\sharp\; [\![7]\!]^\sharp\, D \\ & = & 2 \;+^\sharp\; 7 \\ & = & 9 \\ [\![x - y]\!]^\sharp\, D & = & 2 \;-^\sharp\; \top \\ & = & \top \end{array}$$

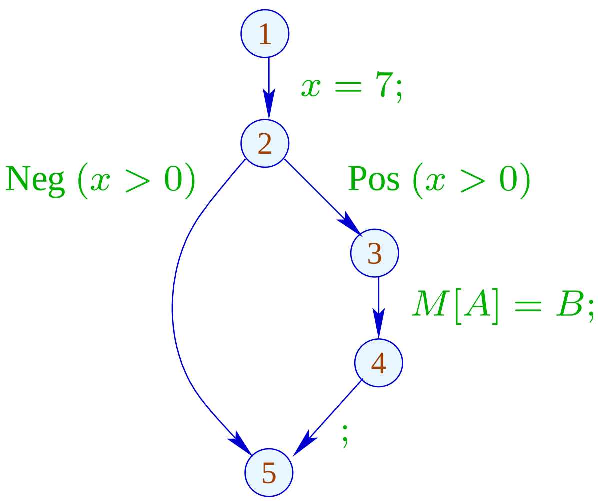
Thus, we obtain the following effects of edges $[\![lab]\!]^\sharp$:

$$
\begin{array}{lll}
[\![;]\!]^\sharp\, D & = & D \\
[\![\text{Pos}\,(e)]\!]^\sharp\, D & = & \begin{cases} \bot & \text{if} \quad 0 = [\![e]\!]^\sharp\, D \\ D & \text{otherwise} \end{cases} \\
[\![\text{Neg}\,(e)]\!]^\sharp\, D & = & \begin{cases} D & \text{if} \quad 0 \sqsubseteq [\![e]\!]^\sharp\, D \\ \bot & \text{otherwise} \end{cases} \\
[\![x = e;]\!]^\sharp\, D & = & D \oplus \{x \mapsto [\![e]\!]^\sharp\, D\} \\
[\![x = M[e];]\!]^\sharp\, D & = & D \oplus \{x \mapsto \top\} \\
[\![M[e_1] = e_2;]\!]^\sharp\, D & = & D
\end{array}
$$

... whenever $D \neq \bot$:-)

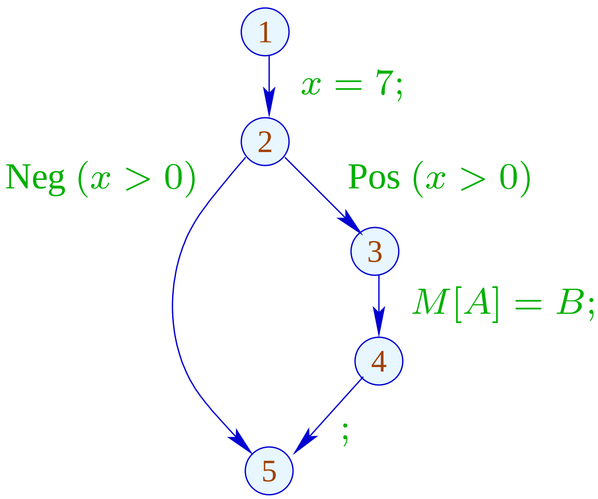
At *start*, we have $D_\top = \{x \mapsto \top \mid x \in \mathit{Vars}\}$.

Example:



At *start*, we have $D_\top = \{x \mapsto \top \mid x \in \mathit{Vars}\}$.

Example:



1	$\{x \mapsto \top\}$
2	$\{x \mapsto 7\}$
3	$\{x \mapsto 7\}$
4	$\{x \mapsto 7\}$
5	$\bot \sqcup \{x \mapsto 7\} = \{x \mapsto 7\}$

The abstract effects of edges $[\![k]\!]^\sharp$ are again composed to the effects of paths $\pi = k_1 \ldots k_r$ by:

$$[\![\pi]\!]^\sharp = [\![k_r]\!]^\sharp \circ \ldots \circ [\![k_1]\!]^\sharp \quad : \mathbb{D} \to \mathbb{D}$$

Idea for Correctness: Abstract Interpretation

Cousot, Cousot 1977

Patrick Cousot, ENS, Paris

The abstract effects of edges $[\![k]\!]^\sharp$ are again composed to the effects of paths $\pi = k_1 \ldots k_r$ by:

$$[\![\pi]\!]^\sharp = [\![k_r]\!]^\sharp \circ \ldots \circ [\![k_1]\!]^\sharp \qquad : \mathbb{D} \to \mathbb{D}$$

Idea for Correctness: Abstract Interpretation

Cousot, Cousot 1977

Establish a description relation Δ between theconcrete values and their descriptions with:

$$x \,\Delta\, a_1 \quad \wedge \quad a_1 \sqsubseteq a_2 \quad \Longrightarrow \quad x \,\Delta\, a_2$$

Concretization: $\gamma\, a = \{x \mid x \,\Delta\, a\}$

// returns the set of described values :-)

(1) Values: $\Delta \subseteq \mathbb{Z} \times \mathbb{Z}^\top$

$$z \, \Delta \, a \quad \text{iff} \quad z = a \vee a = \top$$

Concretization:

$$\gamma \, a = \begin{cases} \{a\} & \text{if} \quad a \sqsubset \top \\ \mathbb{Z} & \text{if} \quad a = \top \end{cases}$$

(1) Values: $\Delta \subseteq \mathbb{Z} \times \mathbb{Z}^\top$

$$z \,\Delta\, a \quad \text{iff} \quad z = a \vee a = \top$$

Concretization:

$$\gamma\, a = \begin{cases} \{a\} & \text{if} \quad a \sqsubset \top \\ \mathbb{Z} & \text{if} \quad a = \top \end{cases}$$

(2) Variable Assignments: $\Delta \subseteq (Vars \to \mathbb{Z}) \times (Vars \to \mathbb{Z}^\top)_\perp$

$$\rho \,\Delta\, D \quad \text{iff} \quad D \neq \perp \wedge \rho\, x \sqsubseteq D\, x \quad (x \in Vars)$$

Concretization:

$$\gamma\, D = \begin{cases} \emptyset & \text{if} \quad D = \perp \\ \{\rho \mid \forall\, x :\ (\rho\, x)\, \Delta\, (D\, x)\} & \text{otherwise} \end{cases}$$