Correctness:

Assume $y_i^{(d)}$ is the i-th component of $F^d \perp$.

Assume $x_i^{(d)}$ is the value of x_i after the d-th RR-iteration.

Correctness:

Assume $y_i^{(d)}$ is the i-th component of $F^d \underline{\perp}$.

Assume $x_i^{(d)}$ is the value of x_i after the i-th RR-iteration.

One proves:

(1) $y_i^{(d)} \sqsubseteq x_i^{(d)}$:-)

Correctness:

Assume $y_i^{(d)}$ is the i-th component of $F^d \underline{\perp}$.

Assume $x_i^{(d)}$ is the value of x_i after the i-th RR-iteration.

One proves:

(1) $y_i^{(d)} \sqsubseteq x_i^{(d)}$:-)

(2) $x_i^{(d)} \sqsubseteq z_i$ for every solution $(z_1, \ldots, z_n)$:-)

Correctness:

Assume $y_i^{(d)}$ is the i-th component of $F^d \underline{\bot}$.

Assume $x_i^{(d)}$ is the value of x_i after the i-th RR-iteration.

One proves:

(1) $y_i^{(d)} \sqsubseteq x_i^{(d)}$:-)

(2) $x_i^{(d)} \sqsubseteq z_i$ for every solution $(z_1, \ldots, z_n)$:-)

(3) If RR-iteration terminates after d rounds, then $(x_1^{(d)}, \ldots, x_n^{(d)})$ is a solution :-))

Caveat:

The efficiency of RR-iteration depends on the ordering of the unknowns !!!

Caveat:

The efficiency of RR-iteration depends on the ordering of the unknowns !!!

Good:

- $\rightarrow$ u before v, if $u \rightarrow^* v$;
- $\rightarrow$ entry condition before loop body :-)

Caveat:

The efficiency of RR-iteration depends on the ordering of the unknowns !!!

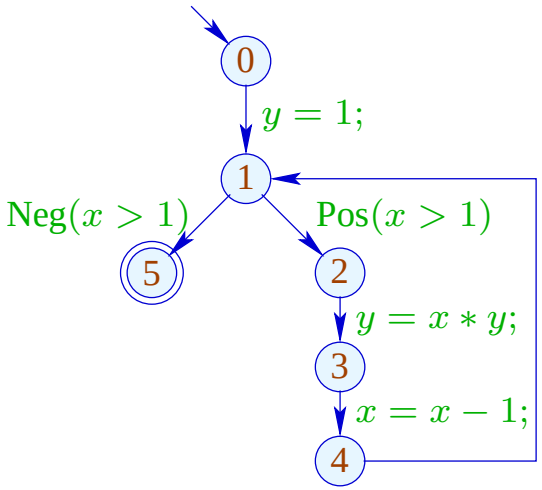
Good:

- $\rightarrow$ u before v, if $u \rightarrow^* v$;
- $\rightarrow$ entry condition before loop body :-)

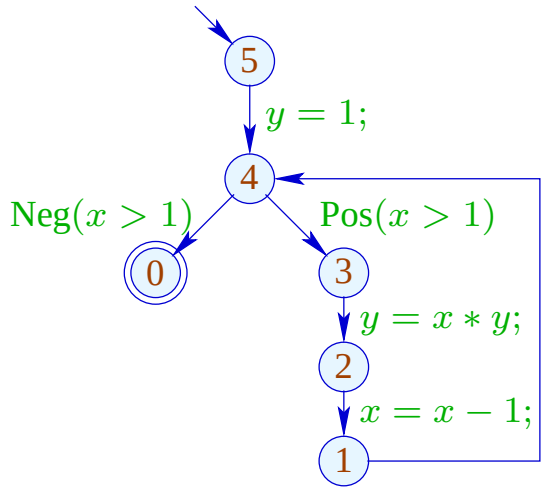
Bad:

e.g., post-order DFS of the CFG, starting at start :-)

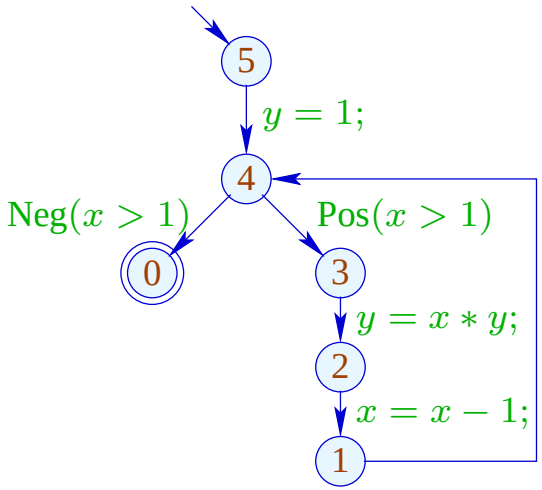
Good:


0
$y = 1;$
1
$\text{Neg}(x > 1)$
$\text{Pos}(x > 1)$
5
2
$y = x * y;$
3
$x = x - 1;$
4

Bad:

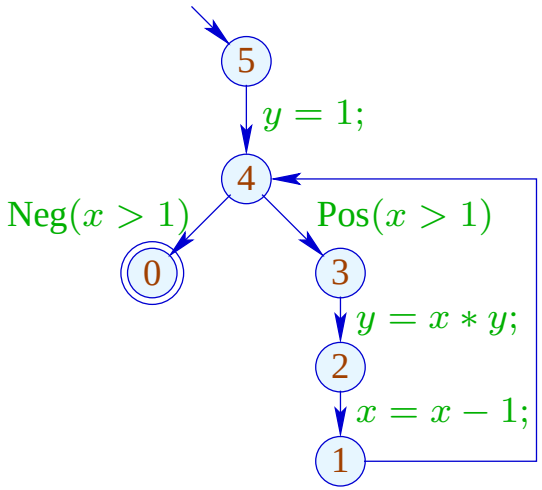

5
$y = 1;$
4
$\text{Neg}(x > 1)$
$\text{Pos}(x > 1)$
0
3
$y = x * y;$
2
$x = x - 1;$
1

Inefficient Round Robin Iteration:



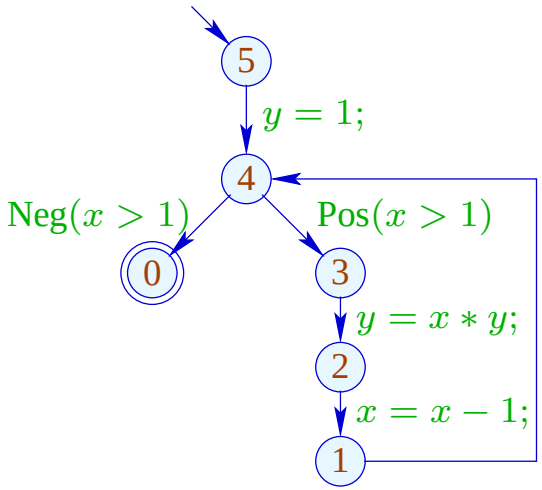
0
1
2
3
4
5

Inefficient Round Robin Iteration:



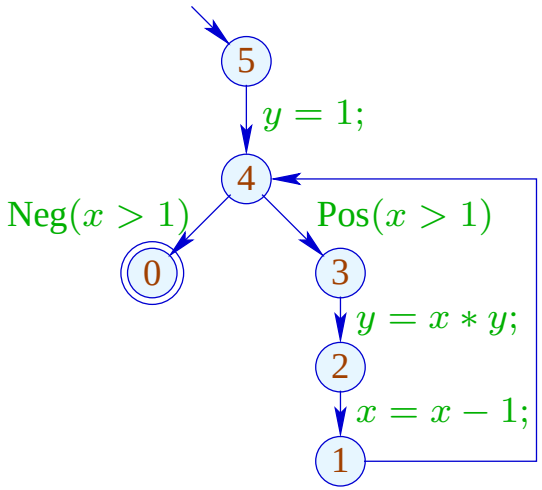
	1
0	*Expr*
1	$\{1\}$
2	$\{1, x-1, x>1\}$
3	*Expr*
4	$\{1\}$
5	$\emptyset$

Inefficient Round Robin Iteration:



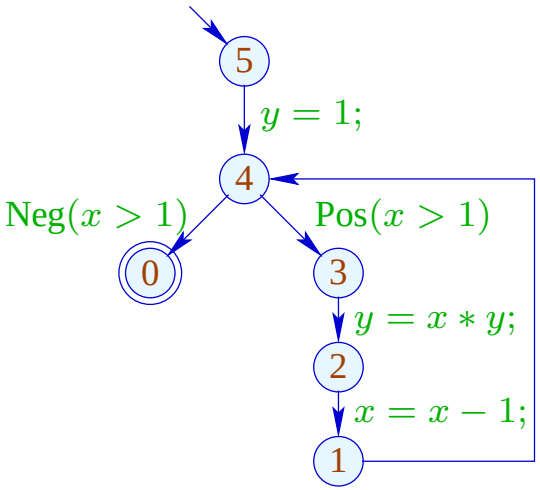
	1	2
0	*Expr*	$\{1, x > 1\}$
1	$\{1\}$	$\{1\}$
2	$\{1, x - 1, x > 1\}$	$\{1, x - 1, x > 1\}$
3	*Expr*	$\{1, x > 1\}$
4	$\{1\}$	$\{1\}$
5	$\emptyset$	$\emptyset$

Inefficient Round Robin Iteration:



	1	2	3
0	*Expr*	$\{1, x > 1\}$	$\{1, x > 1\}$
1	$\{1\}$	$\{1\}$	$\{1\}$
2	$\{1, x - 1, x > 1\}$	$\{1, x - 1, x > 1\}$	$\{1, x > 1\}$
3	*Expr*	$\{1, x > 1\}$	$\{1, x > 1\}$
4	$\{1\}$	$\{1\}$	$\{1\}$
5	$\emptyset$	$\emptyset$	$\emptyset$

Inefficient Round Robin Iteration:



	1	2	3	4
0	*Expr*	$\{1, x > 1\}$	$\{1, x > 1\}$	
1	$\{1\}$	$\{1\}$	$\{1\}$	
2	$\{1, x - 1, x > 1\}$	$\{1, x - 1, x > 1\}$	$\{1, x > 1\}$	dito
3	*Expr*	$\{1, x > 1\}$	$\{1, x > 1\}$	
4	$\{1\}$	$\{1\}$	$\{1\}$	
5	$\emptyset$	$\emptyset$	$\emptyset$	

$\Longrightarrow$ significantly less efficient :-)

... end of background on: Complete Lattices

... end of background on: Complete Lattices

Final Question:

Why is a (or the least) solution of the constraint system useful ???

... end of background on: Complete Lattices

Final Question:

Why is a (or the least) solution of the constraint system useful ???

For a complete lattice $\mathbb{D}$, consider systems:

$$\begin{array}{lcll} \mathcal{I}[start] & \sqsupseteq & d_0 & \\ \mathcal{I}[v] & \sqsupseteq & [\![k]\!]^\sharp\,(\mathcal{I}[u]) & k = (u, \_, v) \quad \text{edge} \end{array}$$

where $d_0 \in \mathbb{D}$ and all $[\![k]\!]^\sharp : \mathbb{D} \to \mathbb{D}$ are monotonic ...

... end of background on: Complete Lattices

Final Question:

Why is a (or the least) solution of the constraint system useful ???

For a complete lattice $\mathbb{D}$, consider systems:

$$\begin{array}{lcll} \mathcal{I}[start] & \sqsupseteq & d_0 & \\ \mathcal{I}[v] & \sqsupseteq & [\![k]\!]^\sharp\,(\mathcal{I}[u]) & k = (u, \_, v) \quad \text{edge} \end{array}$$

where $d_0 \in \mathbb{D}$ and all $[\![k]\!]^\sharp : \mathbb{D} \to \mathbb{D}$ are monotonic ...

$\Longrightarrow$ Monotonic Analysis Framework

Wanted: MOP (Merge Over all Paths)

$$\mathcal{I}^*[v] = \bigsqcup \{ [\![\pi]\!]^\sharp \, d_0 \mid \pi : start \to^* v \}$$

Wanted: MOP (Merge Over all Paths)

$$\mathcal{I}^*[v] = \bigsqcup \{ [\![\pi]\!]^\sharp \, d_0 \mid \pi : start \to^* v \}$$

Theorem Kam, Ullman 1975

Assume $\mathcal{I}$ is a solution of the constraint system. Then:

$$\mathcal{I}[v] \sqsupseteq \mathcal{I}^*[v] \qquad \text{for every} \quad v$$

Jeffrey D. Ullman, Stanford

Wanted: MOP (Merge Over all Paths)

$$\mathcal{I}^*[v] = \bigsqcup \{ [\![\pi]\!]^\sharp \, d_0 \mid \pi : start \to^* v \}$$

Theorem Kam, Ullman 1975

Assume $\mathcal{I}$ is a solution of the constraint system. Then:

$$\mathcal{I}[v] \sqsupseteq \mathcal{I}^*[v] \qquad \text{for every} \quad v$$

In particular: $\mathcal{I}[v] \sqsupseteq [\![\pi]\!]^\sharp \, d_0$ for every $\pi : start \to^* v$

Proof: Induction on the length of π.

Proof: Induction on the length of π.

Foundation: $\pi = \epsilon$ (empty path)

Proof: Induction on the length of π.

Foundation: $\pi = \epsilon$ (empty path)

Then:

$$[\![\pi]\!]^\sharp\, d_0 = [\![\epsilon]\!]^\sharp\, d_0 = d_0 \sqsubseteq \mathcal{I}[start]$$

Proof: Induction on the length of π.

Foundation: $\pi = \epsilon$ (empty path)

Then:

$$[\![\pi]\!]^\sharp\, d_0 = [\![\epsilon]\!]^\sharp\, d_0 = d_0 \sqsubseteq \mathcal{I}[start]$$

Step: $\pi = \pi' k$ for $k = (u, \_, v)$ edge.

Proof: Induction on the length of π.

Foundation: $\pi = \epsilon$ (empty path)

Then:

$$[\![\pi]\!]^\sharp\, d_0 = [\![\epsilon]\!]^\sharp\, d_0 = d_0 \sqsubseteq \mathcal{I}[start]$$

Step: $\pi = \pi' k$ for $k = (u, \_, v)$ edge.

Then:

$$\begin{array}{rlll} & [\![\pi']\!]^\sharp\, d_0 & \sqsubseteq & \mathcal{I}[u] \qquad \text{by I.H. for } \pi \\ \Longrightarrow & [\![\pi]\!]^\sharp\, d_0 & = & [\![k]\!]^\sharp\, ([\![\pi']\!]^\sharp\, d_0) \\ & & \sqsubseteq & [\![k]\!]^\sharp\, (\mathcal{I}[u]) \qquad \text{since } [\![k]\!]^\sharp \text{ monotonic} \\ & & \sqsubseteq & \mathcal{I}[v] \qquad \text{since } \mathcal{I} \text{ solution :-))} \end{array}$$

Disappointment:

Are solutions of the constraint system just upper bounds ???

Disappointment:

Are solutions of the constraint system just upper bounds ???

Answer:

In general: yes :-(

Disappointment:

Are solutions of the constraint system just upper bounds ???

Answer:

In general: yes :-(

With the notable exception when all functions $[\![k]\!]^\sharp$ are distributive ... :-)

The function $f : \mathbb{D}_1 \to \mathbb{D}_2$ is called

- distributive, if $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ for all $\emptyset \neq X \subseteq \mathbb{D}$;
- strict, if $f\,\bot = \bot$.
- totally distributive, if f is distributive and strict.

The function $f : \mathbb{D}_1 \to \mathbb{D}_2$ is called

- distributive, if $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ for all $\emptyset \neq X \subseteq \mathbb{D}$;
- strict, if $f\,\bot = \bot$.
- totally distributive, if f is distributive and strict.

Examples:

- $f\,x = x \cap a \cup b$ for $a, b \subseteq U$.

The function $f : \mathbb{D}_1 \to \mathbb{D}_2$ is called

- distributive, if $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ for all $\emptyset \neq X \subseteq \mathbb{D}$;
- strict, if $f\,\bot = \bot$.
- totally distributive, if f is distributive and strict.

Examples:

- $f\,x = x \cap a \cup b$ for $a, b \subseteq U$.

  Strictness: $f\,\emptyset = a \cap \emptyset \cup b = b = \emptyset$ whenever $b = \emptyset$:-(

The function $f : \mathbb{D}_1 \to \mathbb{D}_2$ is called

- distributive, if $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ for all $\emptyset \neq X \subseteq \mathbb{D}$;
- strict, if $f\,\bot = \bot$.
- totally distributive, if f is distributive and strict.

Examples:

- $f\,x = x \cap a \cup b$ for $a, b \subseteq U$.

  Strictness: $f\,\emptyset = a \cap \emptyset \cup b = b = \emptyset$ whenever $b = \emptyset$:-(

  Distributivity:

$$\begin{aligned} f\,(x_1 \cup x_2) &= a \cap (x_1 \cup x_2) \cup b \\ &= a \cap x_1 \cup a \cap x_2 \cup b \\ &= f\,x_1 \cup f\,x_2 \qquad \text{:-)} \end{aligned}$$

- $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad \mathsf{inc}\, x = x + 1$

- $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad \mathsf{inc}\, x = x + 1$

  Strictness: $f \perp = \mathsf{inc}\, 0 = 1 \quad \neq \quad \perp$:-(

- $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad \mathsf{inc}\, x = x + 1$

  Strictness: $f \perp = \mathsf{inc}\, 0 = 1 \neq \perp$:-(

  Distributivity: $f\,(\bigsqcup X) = \bigsqcup\{x + 1 \mid x \in X\}$ for $\emptyset \neq X$

  :-)

- $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad \mathsf{inc}\, x = x + 1$

  Strictness: $f \perp = \mathsf{inc}\, 0 = 1 \neq \perp$:-(

  Distributivity: $f\,(\bigsqcup X) = \bigsqcup\{x + 1 \mid x \in X\}$ for $\emptyset \neq X$

  :-)

- $\mathbb{D}_1 = (\mathbb{N} \cup \{\infty\})^2, \quad \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad f(x_1, x_2) = x_1 + x_2$

- $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad \mathsf{inc}\, x = x + 1$

  Strictness: $f \perp = \mathsf{inc}\, 0 = 1 \neq \perp$:-(

  Distributivity: $f\,(\bigsqcup X) = \bigsqcup\{x + 1 \mid x \in X\}$ for $\emptyset \neq X$:-)

- $\mathbb{D}_1 = (\mathbb{N} \cup \{\infty\})^2, \quad \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad f(x_1, x_2) = x_1 + x_2$:

  Strictness: $f \perp = 0 + 0 = 0$:-)

- $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad \mathsf{inc}\, x = x + 1$

  Strictness: $f \perp = \mathsf{inc}\, 0 = 1 \neq \perp$:-(

  Distributivity: $f\,(\bigsqcup X) = \bigsqcup\{x + 1 \mid x \in X\}$ for $\emptyset \neq X$

  :-)

- $\mathbb{D}_1 = (\mathbb{N} \cup \{\infty\})^2, \quad \mathbb{D}_2 = \mathbb{N} \cup \{\infty\}, \quad f(x_1, x_2) = x_1 + x_2$:

  Strictness: $f \perp = 0 + 0 = 0$:-)

  Distributivity:

$$f\,((1,4) \sqcup (4,1)) = f\,(4,4) = 8$$
$$\neq 5 = f\,(1,4) \sqcup f\,(4,1) \quad :-)$$

Remark:

If $f : \mathbb{D}_1 \to \mathbb{D}_2$ is distributive, then also monotonic :-)

Remark:

If $f : \mathbb{D}_1 \to \mathbb{D}_2$ is distributive, then also monotonic :-)

Obviously: $a \sqsubseteq b$ iff $a \sqcup b = b$.

Remark:

If $f : \mathbb{D}_1 \to \mathbb{D}_2$ is distributive, then also monotonic :-)

Obviously: $a \sqsubseteq b$ iff $a \sqcup b = b$.

From that follows:

$$\begin{aligned} f\ b &= f\ (a \sqcup b) \\ &= f\ a \sqcup f\ b \\ \Longrightarrow\ f\ a &\sqsubseteq f\ b \end{aligned}$$:-)

Assumption: all v are reachable from *start* .

Assumption: all v are reachable from *start* .
Then:

Theorem

Kildall 1972

If all effects of edges $[\![k]\!]^\sharp$ are distributive, then: $\mathcal{I}^*[v] = \mathcal{I}[v]$
for all v .

Gary A. Kildall (1942-1994).

Has developed the operating system CP/M and GUIs for PCs.

Assumption: all v are reachable from *start* .
Then:

Theorem

Kildall 1972

If all effects of edges $[\![k]\!]^\sharp$ are distributive, then: $\mathcal{I}^*[v] = \mathcal{I}[v]$
for all v .

Assumption: all v are reachable from *start* .
Then:

Theorem

Kildall 1972

If all effects of edges $[\![k]\!]^\sharp$ are distributive, then: $\mathcal{I}^*[v] = \mathcal{I}[v]$
for all v .

Proof:

It suffices to prove that $\mathcal{I}^*$ is a solution :-)

For this, we show that $\mathcal{I}^*$ satisfies all constraints :-))

(1) We prove for *start* :

$$\begin{aligned}
\mathcal{I}^*[start] &= \bigsqcup \{ [\![\pi]\!]^\sharp\, d_0 \mid \pi : start \to^* start \} \\
&\sqsupseteq [\![\epsilon]\!]^\sharp\, d_0 \\
&\sqsupseteq d_0 \qquad \text{:-)}
\end{aligned}$$

(1) We prove for $start$:

$$\begin{aligned}
\mathcal{I}^*[start] &= \bigsqcup\{[\![\pi]\!]^\sharp\, d_0 \mid \pi : start \to^* start\} \\
&\sqsupseteq [\![\epsilon]\!]^\sharp\, d_0 \\
&\sqsupseteq d_0 \qquad \text{:-)}
\end{aligned}$$

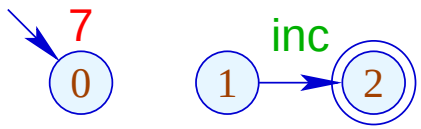
(2) For every $k = (u, \_, v)$ we prove:

$$\begin{aligned}
\mathcal{I}^*[v] &= \bigsqcup\{[\![\pi]\!]^\sharp\, d_0 \mid \pi : start \to^* v\} \\
&\sqsupseteq \bigsqcup\{[\![\pi' k]\!]^\sharp\, d_0 \mid \pi' : start \to^* u\} \\
&= \bigsqcup\{[\![k]\!]^\sharp\, ([\![\pi']\!]^\sharp\, d_0) \mid \pi' : start \to^* u\} \\
&= [\![k]\!]^\sharp\, (\bigsqcup\{[\![\pi']\!]^\sharp\, d_0 \mid \pi' : start \to^* u\}) \\
&= [\![k]\!]^\sharp\, (\mathcal{I}^*[u])
\end{aligned}$$

since $\{\pi' \mid \pi' : start \to^* u\}$ is non-empty :-)

Caveat:

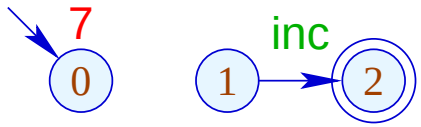
- Reachability of all program points cannot be abandoned! Consider:



where $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Caveat:

- Reachability of all program points cannot be abandoned! Consider:



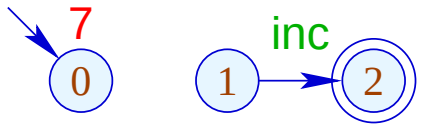
where $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Then:

$$\mathcal{I}[2] = \mathsf{inc}\,0 = 1$$
$$\mathcal{I}^*[2] = \bigsqcup \emptyset = 0$$

Caveat:

- Reachability of all program points cannot be abandoned! Consider:



where $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Then:

$$\mathcal{I}[2] = \mathsf{inc}\,0 = 1$$
$$\mathcal{I}^*[2] = \bigsqcup \emptyset = 0$$

- Unreachable program points can always be thrown away :-)

Summary and Application:

→ The effects of edges of the analysis of availability of expressions are distributive:

$$\begin{aligned}(a \cup (x_1 \cap x_2))\backslash b &= ((a \cup x_1) \cap (a \cup x_2))\backslash b \\ &= ((a \cup x_1)\backslash b) \cap ((a \cup x_2)\backslash b)\end{aligned}$$

Summary and Application:

→ The effects of edges of the analysis of availability of expressions are distributive:

$$\begin{aligned}(a \cup (x_1 \cap x_2))\backslash b &= ((a \cup x_1) \cap (a \cup x_2))\backslash b \\ &= ((a \cup x_1)\backslash b) \cap ((a \cup x_2)\backslash b)\end{aligned}$$

→ If all effects of edges are distributive, then the MOP can be computed by means of the constraint system and RR-iteration. :-)

Summary and Application:

→ The effects of edges of the analysis of availability of expressions are distributive:

$$\begin{aligned}(a \cup (x_1 \cap x_2)) \backslash b &= ((a \cup x_1) \cap (a \cup x_2)) \backslash b \\ &= ((a \cup x_1) \backslash b) \cap ((a \cup x_2) \backslash b)\end{aligned}$$

→ If all effects of edges are distributive, then the MOP can be computed by means of the constraint system and RR-iteration. :-)

→ If not all effects of edges are distributive, then RR-iteration for the constraint system at least returns a safe upper bound to the MOP :-)

1.2 Removing Assignments to Dead Variables

Example:

$$
\begin{aligned}
1: &\quad x = y + 2; \\
2: &\quad y = 5; \\
3: &\quad x = y + 3;
\end{aligned}
$$

The value of x at program points 1, 2 is over-written before it can be used.

Therefore, we call the variable x dead at these program points :-)

Note:

- → Assignments to dead variables can be removed ;-)
- → Such inefficiencies may originate from other transformations.

Note:

$\rightarrow$ Assignments to dead variables can be removed ;-)

$\rightarrow$ Such inefficiencies may originate from other transformations.

Formal Definition:

The variable x is called live at u along the path π starting at u relative to a set X of variables either:

if $x \in X$ and π does not contain a definition of x; or:

if π can be decomposed into: $\pi = \pi_1 \, k \, \pi_2$ such that:

- k is a use of x ; and
- π_1 does not contain a definition of x.