Theorem — Knaster – Tarski

Assume $\mathbb{D}$ is a complete lattice. Then every monotonic function $f : \mathbb{D} \to \mathbb{D}$ has a least fixpoint $d_0 \in \mathbb{D}$.

Let $P = \{d \in \mathbb{D} \mid f\, d \sqsubseteq d\}$.

Then $d_0 = \bigsqcap P$.

Proof:

(1) $d_0 \in P$:

Theorem — Knaster – Tarski

Assume $\mathbb{D}$ is a complete lattice. Then every monotonic function $f : \mathbb{D} \to \mathbb{D}$ has a least fixpoint $d_0 \in \mathbb{D}$.

Let $P = \{d \in \mathbb{D} \mid f\, d \sqsubseteq d\}$.

Then $d_0 = \bigsqcap P$.

Proof:

(1) $d_0 \in P$:

$$f\, d_0 \sqsubseteq f\, d \sqsubseteq d \quad \text{for all } d \in P$$

$\Longrightarrow$ $f\, d_0$ is a lower bound of P

$\Longrightarrow$ $f\, d_0 \sqsubseteq d_0$ since $d_0 = \bigsqcap P$

$\Longrightarrow$ $d_0 \in P$:-)

(2) $$f\,d_0 = d_0 :$$

(2) $\quad f\, d_0 = d_0$:

$$f\, d_0 \sqsubseteq d_0 \quad \text{by} \quad (1)$$

$$\Longrightarrow \quad f(f\, d_0) \sqsubseteq f\, d_0 \quad \text{by monotonicity of } f$$

$$\Longrightarrow \quad f\, d_0 \in P$$

$$\Longrightarrow \quad d_0 \sqsubseteq f\, d_0 \qquad \text{and the claim follows} \quad \text{:-)}$$

(2) $f\,d_0 = d_0$:

$$f\,d_0 \sqsubseteq d_0 \quad \text{by} \quad (1)$$

$$\Longrightarrow \quad f(f\,d_0) \sqsubseteq f\,d_0 \quad \text{by monotonicity of } f$$

$$\Longrightarrow \quad f\,d_0 \in P$$

$$\Longrightarrow \quad d_0 \sqsubseteq f\,d_0 \qquad \text{and the claim follows} \quad \text{:-)}$$

(3) d_0 is least fixpoint:

(2) $f\, d_0 = d_0$:

$$f\, d_0 \sqsubseteq d_0 \quad \text{by} \quad (1)$$

$$\Longrightarrow \quad f(f\, d_0) \sqsubseteq f\, d_0 \quad \text{by monotonicity of } f$$

$$\Longrightarrow \quad f\, d_0 \in P$$

$$\Longrightarrow \quad d_0 \sqsubseteq f\, d_0 \qquad \text{and the claim follows} \quad \text{:-)}$$

(3) d_0 is least fixpoint:

$$f\, d_1 = d_1 \sqsubseteq d_1 \quad \text{an other fixpoint}$$

$$\Longrightarrow \quad d_1 \in P$$

$$\Longrightarrow \quad d_0 \sqsubseteq d_1 \qquad \text{:-))}$$

Remark:

The least fixpoint d_0 is in P and a lower bound :-)

$\Longrightarrow$ d_0 is the least value x with $x \sqsupseteq f\,x$

Remark:

The least fixpoint d_0 is in P and a lower bound :-)

$\Longrightarrow$ d_0 is the least value x with $x \sqsupseteq f\,x$

Application:

Assume $x_i \sqsupseteq f_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n$ $(*)$

is a system of constraints where all $f_i : \mathbb{D}^n \to \mathbb{D}$ are monotonic.

Remark:

The least fixpoint d_0 is in P and a lower bound :-)

$\Longrightarrow$ d_0 is the least value x with $x \sqsupseteq f\,x$

Application:

Assume $x_i \sqsupseteq f_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n$ $(*)$

is a system of constraints where all $f_i : \mathbb{D}^n \to \mathbb{D}$ are monotonic.

$\Longrightarrow$ least solution of $(*)$ $==$ least fixpoint of F :-)

Example 1: $\mathbb{D} = 2^U, \quad f\, x = x \cap a \cup b$

Example 1: $\mathbb{D} = 2^U, \quad f\,x = x \cap a \cup b$

f	$f^k\,\bot$	$f^k\,\top$
0	$\emptyset$	U

Example 1: $\mathbb{D} = 2^U, \quad f\,x = x \cap a \cup b$

f	$f^k\,\bot$	$f^k\,\top$
0	$\emptyset$	U
1	b	$a \cup b$

Example 1: $\mathbb{D} = 2^U, \quad f\,x = x \cap a \cup b$

f	$f^k\,\bot$	$f^k\,\top$
0	$\emptyset$	U
1	b	$a \cup b$
2	b	$a \cup b$

Example 1: $\mathbb{D} = 2^U, \quad f\,x = x \cap a \cup b$

f	$f^k\,\bot$	$f^k\,\top$
0	$\emptyset$	U
1	b	$a \cup b$
2	b	$a \cup b$

Example 2: $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Assume $f\,x = x + 1$. Then

$$f^i\,\bot = f^i\,0 = i \quad \sqsubset \quad i + 1 = f^{i+1}\,\bot$$

Example 1: $\mathbb{D} = 2^U, \quad f\, x = x \cap a \cup b$

f	$f^k \bot$	$f^k \top$
0	$\emptyset$	U
1	b	$a \cup b$
2	b	$a \cup b$

Example 2: $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Assume $f\, x = x + 1$. Then

$$f^i \bot = f^i\, 0 = i \quad \sqsubset \quad i + 1 = f^{i+1} \bot$$

$\Longrightarrow$ Ordinary iteration will never reach a fixpoint :-(

$\Longrightarrow$ Sometimes, transfinite iteration is needed :-)

Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat: Naive fixpoint iteration is rather inefficient :-(

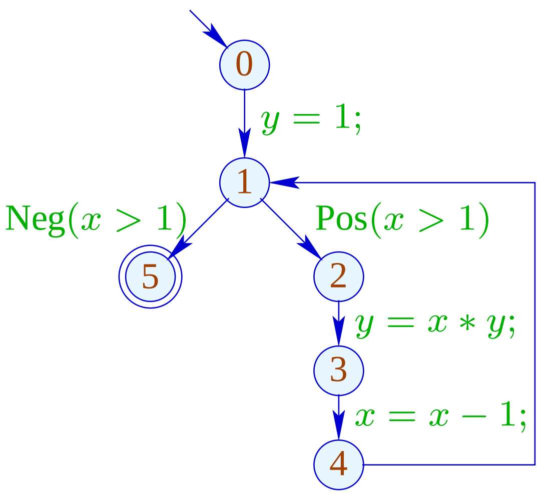
Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat:

Naive fixpoint iteration is rather inefficient :-(

Example:



0
1
2
3
4
5

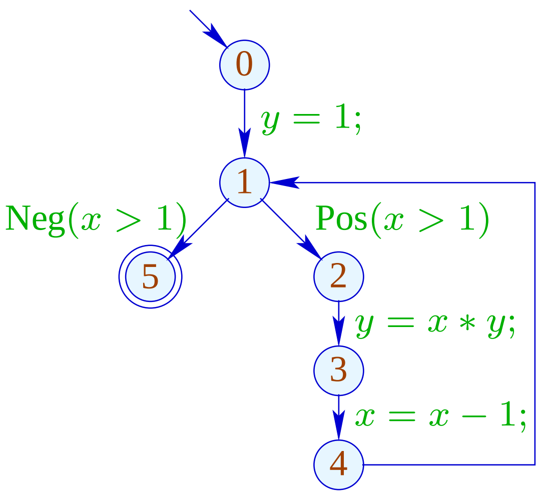
Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat:

Naive fixpoint iteration is rather inefficient :-(

Example:



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

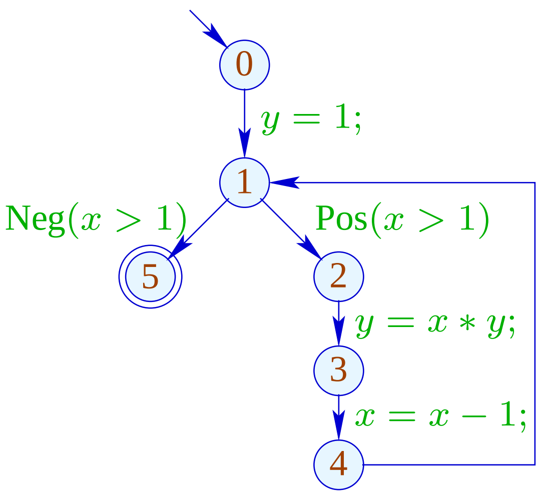
Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat:

Naive fixpoint iteration is rather inefficient :-(

Example:



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

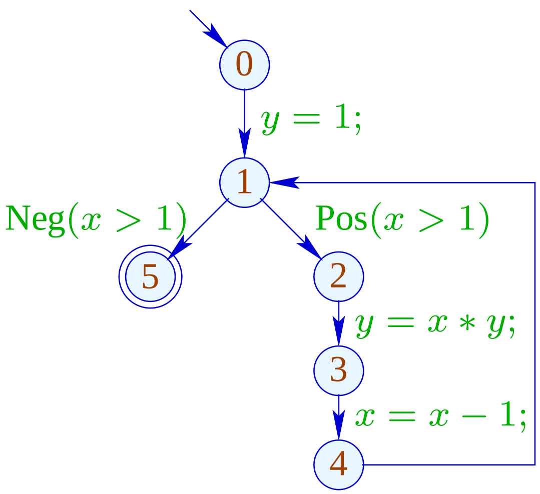
Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat:

Naive fixpoint iteration is rather inefficient :-(

Example:



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

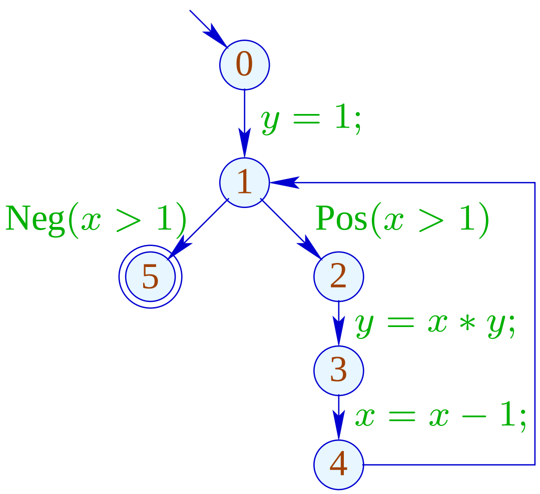
Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat:

Naive fixpoint iteration is rather inefficient :-(

Example:



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

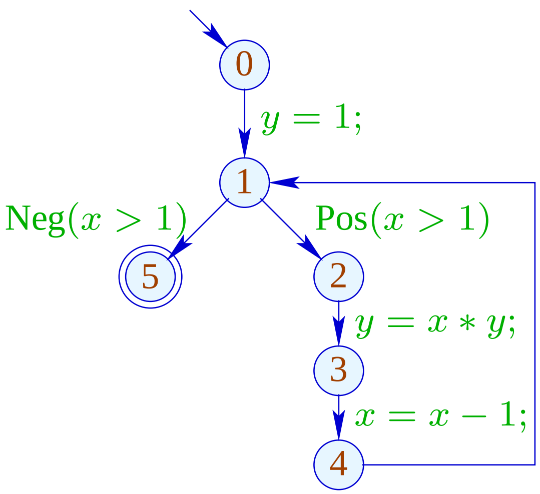
Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides :-)

Caveat:

Naive fixpoint iteration is rather inefficient :-(

Example:



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

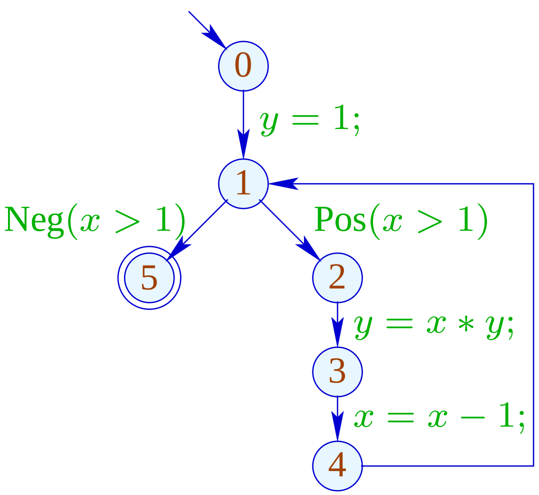
Idea: Round Robin Iteration

Instead of accessing the values of the last iteration, always use the current values of unknowns :-)

Idea: Round Robin Iteration

Instead of accessing the values of the last iteration, always use the current values of unknowns :-)

Example:

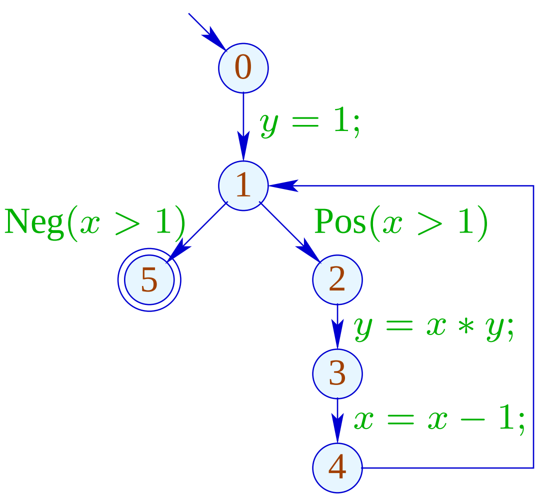


0
1
2
3
4
5

Idea: Round Robin Iteration

Instead of accessing the values of the last iteration, always use the current values of unknowns :-)

Example:

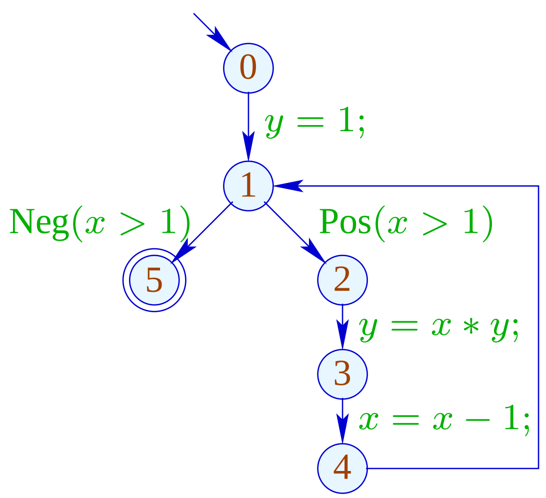


	1
0	$\emptyset$
1	$\{1\}$
2	$\{1, x > 1\}$
3	$\{1, x > 1\}$
4	$\{1\}$
5	$\{1, x > 1\}$

Idea: Round Robin Iteration

Instead of accessing the values of the last iteration, always use the current values of unknowns :-)

Example:



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

The code for Round Robin Iteration in Java looks as follows:

$$
\begin{array}{l}
\textsf{for}\ (i = 1; i \leq n; i{+}{+})\ x_i = \bot; \\
\textsf{do}\ \{ \\
\qquad \mathit{finished} = \textsf{true}; \\
\qquad \textsf{for}\ (i = 1; i \leq n; i{+}{+})\ \{ \\
\qquad\qquad \mathit{new} = f_i(x_1, \ldots, x_n); \\
\qquad\qquad \textsf{if}\ (!(x_i \sqsupseteq \mathit{new}))\ \{ \\
\qquad\qquad\qquad \mathit{finished} = \textsf{false}; \\
\qquad\qquad\qquad x_i = x_i \sqcup \mathit{new}; \\
\qquad\qquad \} \\
\qquad \} \\
\}\ \textsf{while}\ (!\mathit{finished});
\end{array}
$$

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 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 $F^d \underline{\perp}$.

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 $(z_1, \ldots, z_n)$:-)

(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:

→ u before v, if $u \rightarrow^* v$;

→ entry condition before loop body :-)

Caveat:

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

Good:

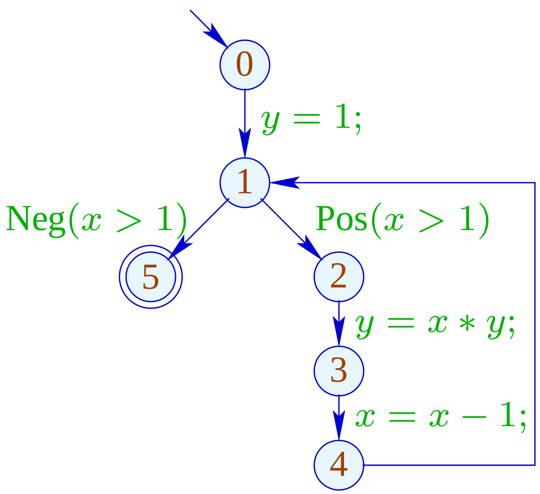
$\rightarrow$ u before v, if $u \rightarrow^* v$;

$\rightarrow$ entry condition before loop body :-)

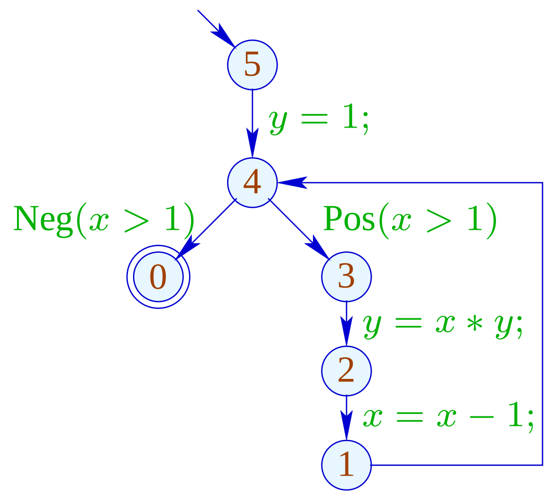
Bad:

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

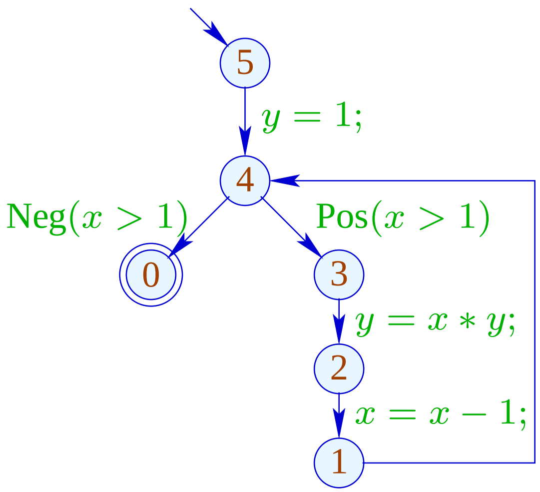
Good:



Bad:

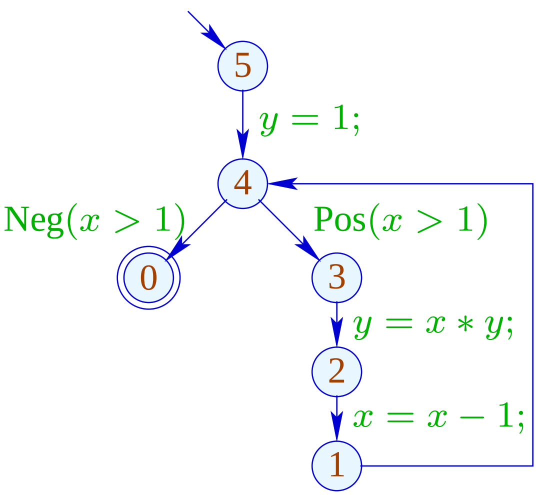


Inefficient Round Robin Iteration:



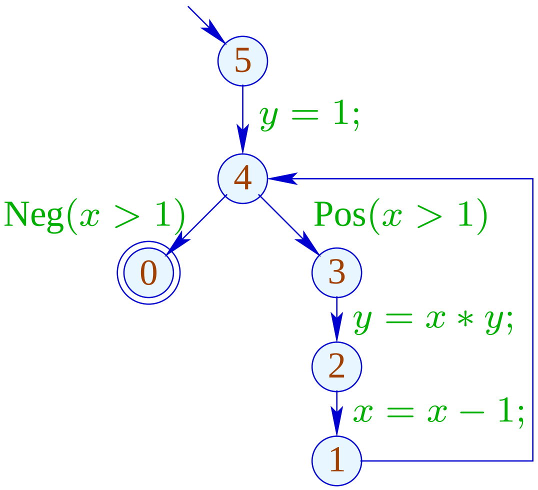
0
1
2
3
4
5

Inefficient Round Robin Iteration:



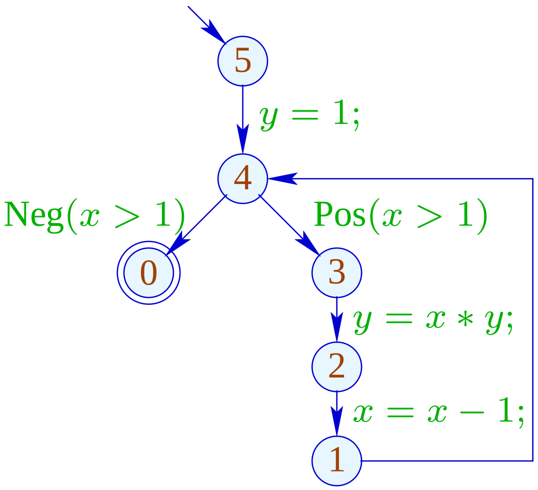
	1
0	$Expr$
1	$\{1\}$
2	$\{1, x - 1, x > 1\}$
3	$Expr$
4	$\{1\}$
5	$\emptyset$

Inefficient Round Robin Iteration:



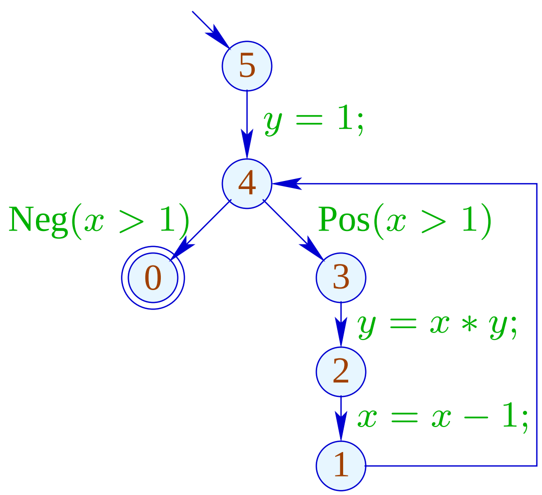
	1	2
0	*Expr*	$\{1, x > 1\}$
1	$\{1\}$	$\{1\}$
2	$\{1, x - 1, x > 1\}$	$\{1, x - 1, x > 1\}$
3	*Expr*	$\{1, x > 1\}$
4	$\{1\}$	$\{1\}$
5	$\emptyset$	$\emptyset$

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:

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

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:

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

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)

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

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:

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

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:

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

:-))

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:

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

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

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

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

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

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

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

  :-)

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

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

  Distributivity: $f\, (\bigsqcup X) \quad = \quad \bigsqcup \{x + 1 \mid x \in X\}$ 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: $\quad f\,\bot = \mathsf{inc}\, 0 = 1 \quad \neq \quad \bot$ $\quad$:-(

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

  :-)

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

  Strictness: $\quad f\,\bot = 0 + 0 \quad = \quad 0$ $\quad$:-)

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

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

  Distributivity: $f\, (\bigsqcup X) \quad = \quad \bigsqcup \{x + 1 \mid x \in X\}$ 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: $f\, \bot = 0 + 0 \quad = \quad 0$:-)

  Distributivity:

$$
\begin{array}{lllll}
f\, ((1,4) \sqcup (4,1)) & = & f\, (4,4) & = & 8 \\
 & \neq & 5 & = & f\, (1,4) \sqcup f\, (4,1)
\end{array}
$$

:-)

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:

$$\begin{array}{rrcl} & f\,b & = & f\,(a \sqcup b) \\ & & = & f\,a \sqcup f\,b \\ \Longrightarrow & f\,a & \sqsubseteq & f\,b \end{array}$$

:-)

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* :

$$\begin{array}{lll} \mathcal{I}^*[start] & = & \bigsqcup\{[\![\pi]\!]^\sharp\, d_0 \mid \pi : start \to^* start\} \\ & \sqsupseteq & [\![\epsilon]\!]^\sharp\, d_0 \\ & \sqsupseteq & d_0 \end{array}$$

:-)

(1) We prove for *start* :

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

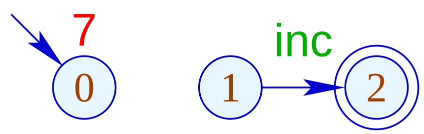
(2) For every $k = (u, \_, v)$ we prove:

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

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

Caveat:

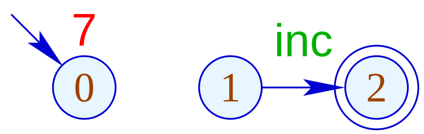
- Reachability of all program points cannot be abandoned! Consider:



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

Caveat:

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



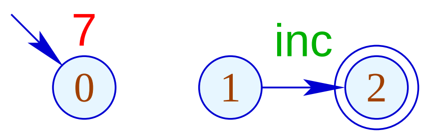
where $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Then:

$$\begin{array}{lllll} \mathcal{I}[2] & = & \mathsf{inc}\, 0 & = & 1 \\ \mathcal{I}^*[2] & = & \bigsqcup \emptyset & = & 0 \end{array}$$

Caveat:

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



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

Then:

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

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

Summary and Application:

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

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