Zur Korrektheit:

Sei $y_i^{(d)}$ die i-te Komponente von $F^d \underline{\perp}$.

Sei $x_i^{(d)}$ der Wert von x_i nach der i-ten RR-Iteration.

Zur Korrektheit:

Sei $y_i^{(d)}$ die i-te Komponente von $F^d\,\underline{\perp}$.

Sei $x_i^{(d)}$ der Wert von x_i nach der i-ten RR-Iteration.

Man zeigt:

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

Zur Korrektheit:

Sei $y_i^{(d)}$ die i-te Komponente von $F^d \underline{\perp}$.

Sei $x_i^{(d)}$ der Wert von x_i nach der i-ten RR-Iteration.

Man zeigt:

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

(2) $x_i^{(d)} \sqsubseteq z_i$ für jede Lösung $(z_1, \ldots, z_n)$:-)

Zur Korrektheit:

Sei $y_i^{(d)}$ die i-te Komponente von $F^d \underline{\perp}$.

Sei $x_i^{(d)}$ der Wert von x_i nach der i-ten RR-Iteration.

Man zeigt:

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

(2) $x_i^{(d)} \sqsubseteq z_i$ für jede Lösung $(z_1, \ldots, z_n)$:-)

(3) Terminiert RR-Iteration nach d Runden, ist $(x_1^{(d)}, \ldots, x_n^{(d)})$ eine Lösung :-))

Achtung:

Die Effizienz von RR-Iteration hängt von der Anordnung der Variablen ab !!!

Achtung:

Die Effizienz von RR-Iteration hängt von der Anordnung der Variablen ab !!!

Günstig:

$\longrightarrow$ u vor v, falls $u \rightarrow^* v$;

$\longrightarrow$ Eintrittsbedingung vor Schleifen-Rumpf :-)

Achtung:

Die Effizienz von RR-Iteration hängt von der Anordnung der Variablen ab !!!

Günstig:

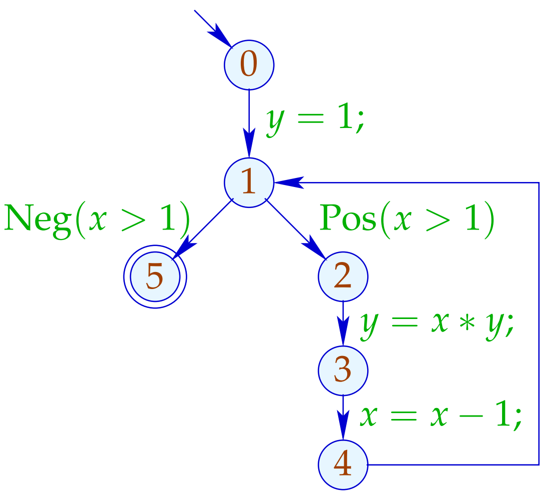
$\longrightarrow$ u vor v, falls $u \longrightarrow^* v$;

$\longrightarrow$ Eintrittsbedingung vor Schleifen-Rumpf :-)

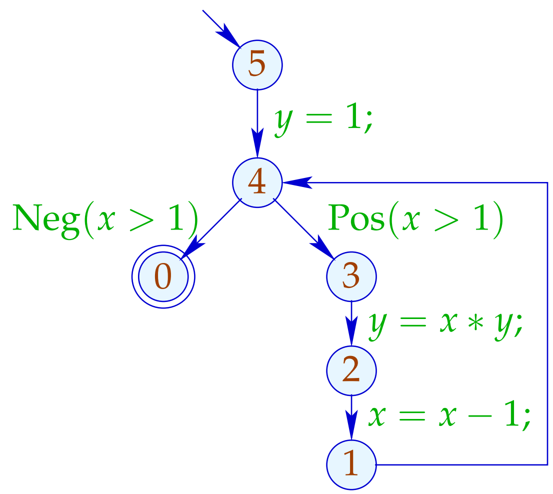
Ungünstig:

z.B. post-order DFS auf dem CFG, startend von start :-)

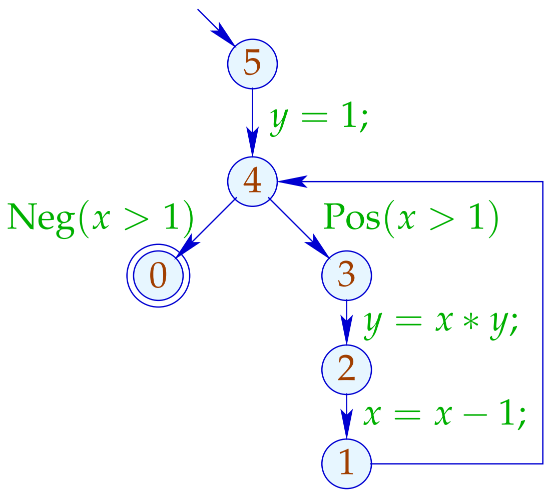
Günstig:



Ungünstig:

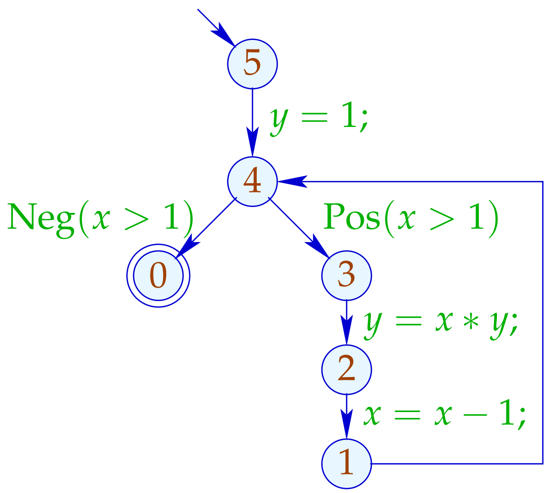


Ungünstige Round Robin Iteration:



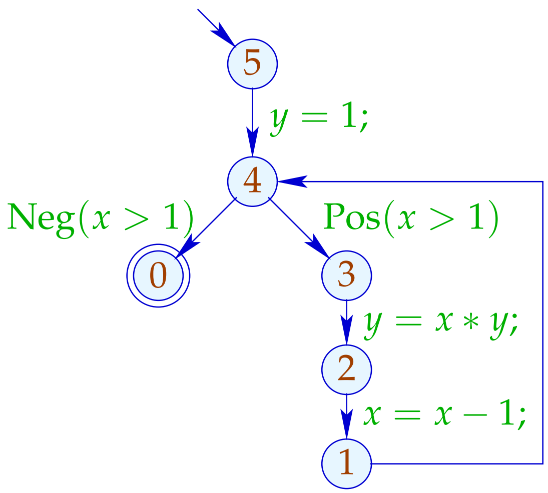
0
1
2
3
4
5

Ungünstige Round Robin Iteration:



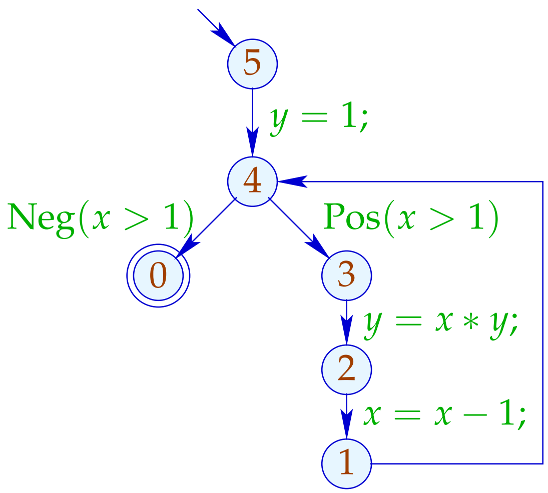
	1
0	*Expr*
1	$\{1\}$
2	$\{1, x-1, x>1\}$
3	*Expr*
4	$\{1\}$
5	$\emptyset$

Ungünstige Round Robin Iteration:



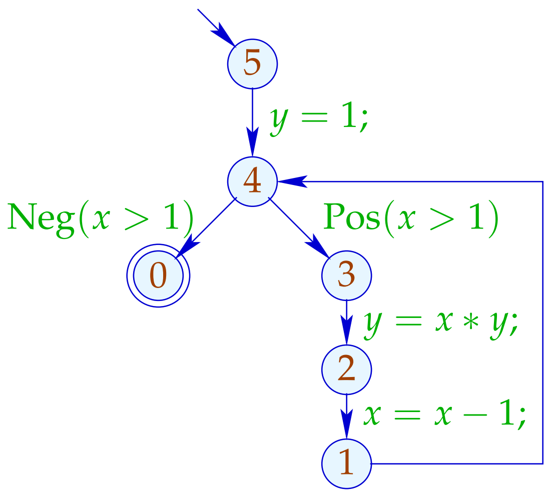
	1	2
0	*Expr*	$\{1, x > 1\}$
1	$\{1\}$	$\{1\}$
2	$\{1, x - 1, x > 1\}$	$\{1, x - 1, x > 1\}$
3	*Expr*	$\{1, x > 1\}$
4	$\{1\}$	$\{1\}$
5	$\emptyset$	$\emptyset$

Ungünstige 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	$\emptyset$	$\emptyset$	$\emptyset$

Ungünstige 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	$\emptyset$	$\emptyset$	$\emptyset$	

$\Longrightarrow$ deutlich weniger effizient :-)

... Ende des Exkurses: Vollständige Verbände

... Ende des Exkurses: Vollständige Verbände

Letzte Frage:

Wieso hilft uns eine (oder die kleinste) Lösung des Constraint-Systems weiter ???

... Ende des Exkurses: Vollständige Verbände

Letzte Frage:

Wieso hilft uns eine (oder die kleinste) Lösung des Constraint-Systems weiter ???

Betrachte für einen vollständigen Verband $\mathbb{D}$ Systeme:

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

wobei $d_0 \in \mathbb{D}$ und alle $[\![k]\!]^\sharp : \mathbb{D} \to \mathbb{D}$ monoton sind ...

... Ende des Exkurses: Vollständige Verbände

Letzte Frage:

Wieso hilft uns eine (oder die kleinste) Lösung des Constraint-Systems weiter ???

Betrachte für einen vollständigen Verband $\mathbb{D}$ Systeme:

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

wobei $d_0 \in \mathbb{D}$ und alle $[\![k]\!]^\sharp : \mathbb{D} \to \mathbb{D}$ monoton sind ...

$\Longrightarrow$ monotoner Analyse-Rahmen

Gesucht: MOP (Merge Over all Paths)

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

Gesucht: MOP (Merge Over all Paths)

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

Theorem Kam, Ullman 1975

Sei $\mathcal{I}$ die kleinste Lösung des Constraint-Systems. Dann gilt:

$$\mathcal{I}[v] \sqsupseteq \mathcal{I}^*[v] \qquad \text{für jedes} \quad v$$

Jeffrey D. Ullman, Stanford

Gesucht: MOP (Merge Over all Paths)

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

Theorem Kam, Ullman 1975

Sei $\mathcal{I}$ die kleinste Lösung des Constraint-Systems. Dann gilt:

$$\mathcal{I}[v] \sqsupseteq \mathcal{I}^*[v] \qquad \text{für jedes} \quad v$$

Insbesondere: $\mathcal{I}[v] \sqsupseteq [\![\pi]\!]^\sharp\, d_0$ für jedes $\pi : \mathit{start} \rightarrow^* v$

Beweis: Induktion nach der Länge von π.

Beweis: Induktion nach der Länge von π.

Anfang: $\pi = \epsilon$ (leerer Pfad)

Beweis: Induktion nach der Länge von π.

Anfang: $\pi = \epsilon$ (leerer Pfad)

Dann gilt:

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

Beweis: Induktion nach der Länge von π.

Anfang: $\pi = \epsilon$ (leerer Pfad)

Dann gilt:

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

Schluss: $\pi = \pi' k$ für $k = (u, \_, v)$ Kante.

Beweis: Induktion nach der Länge von π.

Anfang: $\pi = \epsilon$ (leerer Pfad)

Dann gilt:

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

Schluss: $\pi = \pi' k$ für $k = (u, \_, v)$ Kante.

Dann gilt:

$$\begin{array}{llll} & [\![\pi']\!]^\sharp\, d_0 & \sqsubseteq & \mathcal{I}[u] & \text{wegen I.H. für } \pi \\ \Longrightarrow & [\![\pi]\!]^\sharp\, d_0 & = & [\![k]\!]^\sharp\,([\![\pi']\!]^\sharp\, d_0) & \\ & & \sqsubseteq & [\![k]\!]^\sharp\,(\mathcal{I}[u]) & \text{da } [\![k]\!]^\sharp \text{ monoton} \\ & & \sqsubseteq & \mathcal{I}[v] & \text{da } \mathcal{I} \text{ Lösung} \quad \text{:-))} \end{array}$$

Enttäuschung:

Liefern Lösungen des Constraint-Systems nur obere Schranken ???

Enttäuschung:

Liefern Lösungen des Constraint-Systems nur obere Schranken ???

Antwort:

Im allgemeinen: ja :-(

Enttäuschung:

Liefern Lösungen des Constraint-Systems nur obere Schranken ???

Antwort:

Im allgemeinen: ja :-(

Es sei denn, alle Funktionen $[\![k]\!]^\sharp$ sind distributiv ... :-)

Die Funktion $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt

- distributiv, falls $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ für alle $\emptyset \neq X \subseteq \mathbb{D}$;
- strikt, falls $f\,\bot = \bot$.
- total distributiv, falls f distributiv und strikt ist.

Die Funktion $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt

- distributiv, falls $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ für alle $\emptyset \neq X \subseteq \mathbb{D}$;
- strikt, falls $f\,\bot = \bot$.
- total distributiv, falls f distributiv und strikt ist.

Beispiele:

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

Die Funktion $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt

- distributiv, falls $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ für alle $\emptyset \neq X \subseteq \mathbb{D}$;
- strikt, falls $f\,\bot = \bot$.
- total distributiv, falls f distributiv und strikt ist.

Beispiele:

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

  Striktheit: $f\,\emptyset = a \cap \emptyset \cup b = b \quad = \quad \emptyset$ sofern $b = \emptyset$:-(

Die Funktion $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt

- distributiv, falls $f\,(\bigsqcup X) = \bigsqcup\{f\,x \mid x \in X\}$ für alle $\emptyset \neq X \subseteq \mathbb{D}$;
- strikt, falls $f\,\bot = \bot$.
- total distributiv, falls f distributiv und strikt ist.

Beispiele:

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

Striktheit: $f\,\emptyset = a \cap \emptyset \cup b = b = \emptyset$ sofern $b = \emptyset$:-(

Distributivität:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

  Striktheit: $f\,\perp = 0 + 0 \quad = \quad 0 \qquad$:-)

  Distributivität:

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

Bemerkung:

Ist $f : \mathbb{D}_1 \longrightarrow \mathbb{D}_2$ distributiv, dann auch monoton :-)

Bemerkung:

Ist $f : \mathbb{D}_1 \longrightarrow \mathbb{D}_2$ distributiv, dann auch monoton :-)

Offenbar gilt: $a \sqsubseteq b$ gdw. $a \sqcup b = b$.

Bemerkung:

Ist $f : \mathbb{D}_1 \to \mathbb{D}_2$ distributiv, dann auch monoton :-)

Offenbar gilt: $a \sqsubseteq b$ gdw. $a \sqcup b = b$.

Daraus folgt:

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

:-)

Annahme: alle v sind von *start* erreichbar.

Annahme: alle v sind von *start* erreichbar.
Dann gilt:

Theorem Kildall 1972

Sind alle Kanten-Effekte $[\![k]\!]^\sharp$ distributiv, dann ist:
$\mathcal{I}^*[v] = \mathcal{I}[v]$ für alle v .

Gary A. Kildall (1942-1994).

Hat später am Betriebssystem CP/M und an GUIs für PCs gearbeitet.

Annahme: alle v sind von *start* erreichbar.
Dann gilt:

Theorem Kildall 1972

Sind alle Kanten-Effekte $[\![k]\!]^\sharp$ distributiv, dann ist:
$\mathcal{I}^*[v] = \mathcal{I}[v]$ für alle v .

Annahme: alle v sind von *start* erreichbar.
Dann gilt:

Theorem

Kildall 1972

Sind alle Kanten-Effekte $[\![k]\!]^\sharp$ distributiv, dann ist:
$\mathcal{I}^*[v] = \mathcal{I}[v]$ für alle v .

Beweis:

Offenbar genügt es zu zeigen, dass $\mathcal{I}^*$ eine Lösung ist :-)

Wir zeigen, dass $\mathcal{I}^*$ alle Ungleichungen erfüllt :-))

Wir zeigen, dass $\mathcal{I}^*$ alle Ungleichungen erfüllt :-))

(1) Für *start* zeigen wir:

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

Wir zeigen, dass $\mathcal{I}^*$ alle Ungleichungen erfüllt :-))

(1) Für *start* zeigen wir:

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

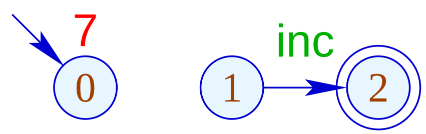
(2) Für jedes $k = (u, \_, v)$ zeigen wir:

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

da $\{\pi' \mid \pi' : \mathit{start} \rightarrow^* u\}$ nicht-leer ist :-)

Achtung:

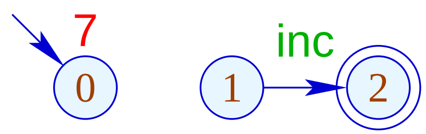
- Auf die Erreichbarkeit aller Programm-Punkt können wir nicht verzichten. Betrachte:



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

Achtung:

- Auf die Erreichbarkeit aller Programm-Punkt können wir nicht verzichten. Betrachte:



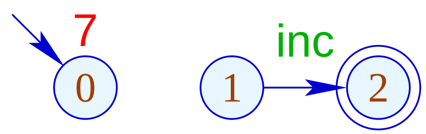
wobei $\mathbb{D} = \mathbb{N} \cup \{\infty\}$

Dann ist:

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

Achtung:

- Auf die Erreichbarkeit aller Programm-Punkt können wir nicht verzichten. Betrachte:

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

  Dann ist:

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

- Unerreichbare Programmpunkte können wir aber stets wegwerfen :-)

Zusammenfassung und Anwendung:

$\longrightarrow$ Die Kanteneffekte der Analyse zur Verfügbarkeit von Ausdrücken sind distributiv:

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

Zusammenfassung und Anwendung:

$\longrightarrow$ Die Kanteneffekte der Analyse zur Verfügbarkeit von Ausdrücken sind distributiv:

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

$\longrightarrow$ Sind alle Kanteneffekte distributiv, lässt sich der MOP mithilfe des Constraint-Systems und RR-Iteration ausrechnen :-)

Zusammenfassung und Anwendung:

→ Die Kanteneffekte der Analyse zur Verfügbarkeit von Ausdrücken sind distributiv:

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

→ Sind alle Kanteneffekte distributiv, lässt sich der MOP mithilfe des Constraint-Systems und RR-Iteration ausrechnen :-)

→ Sind nicht alle Kanteneffekte distributiv, lässt sich eine sichere obere Schranke für den MOP mithilfe des Constraint-Systems und RR-Iteration berechnen :-)