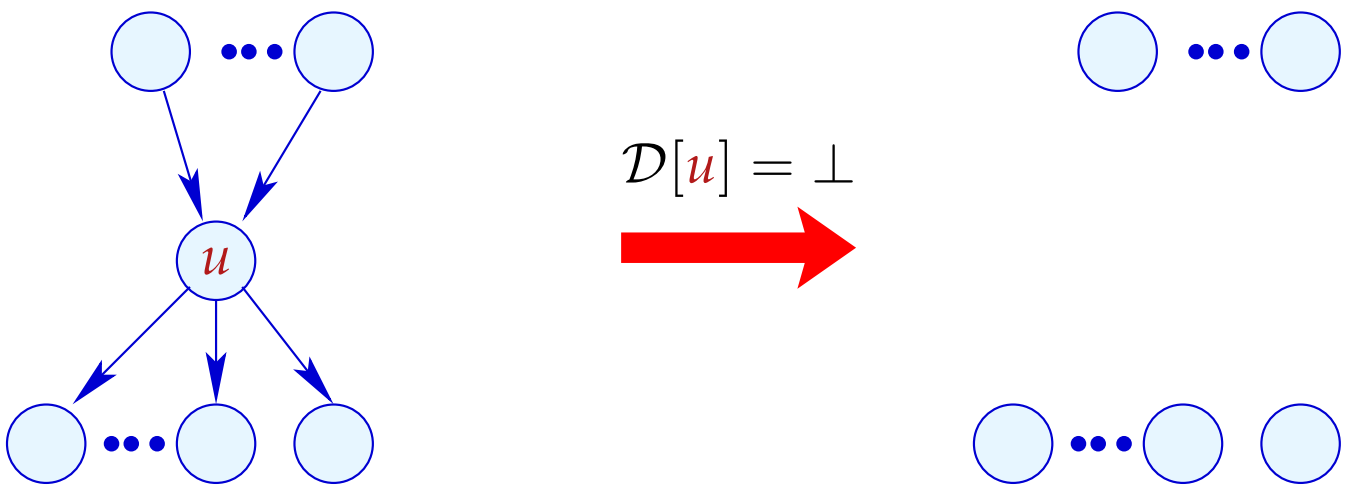
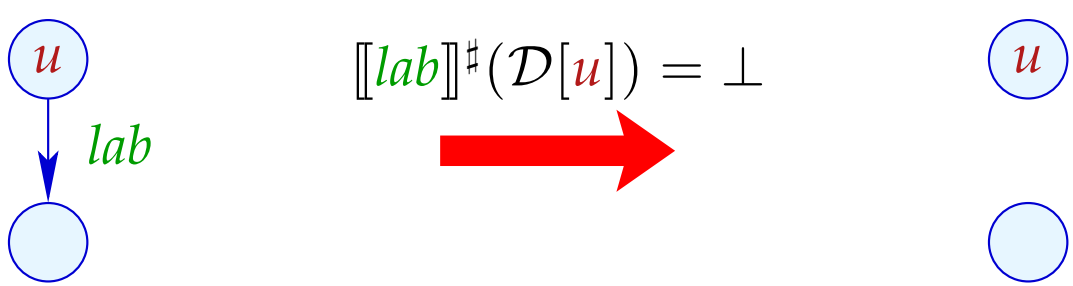
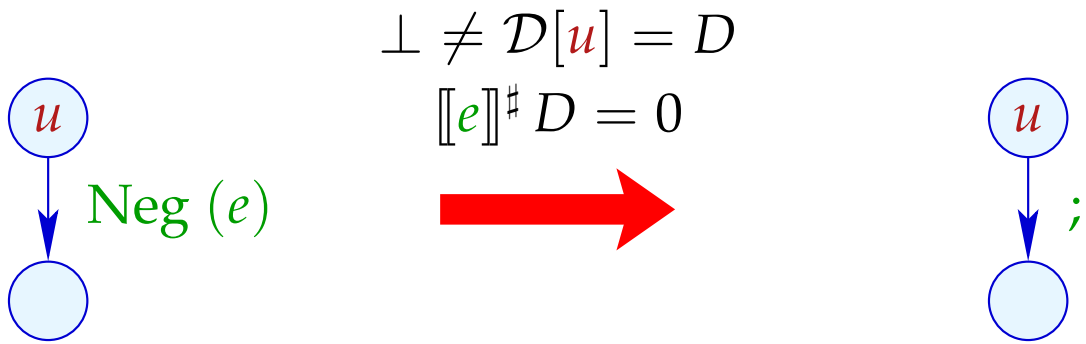
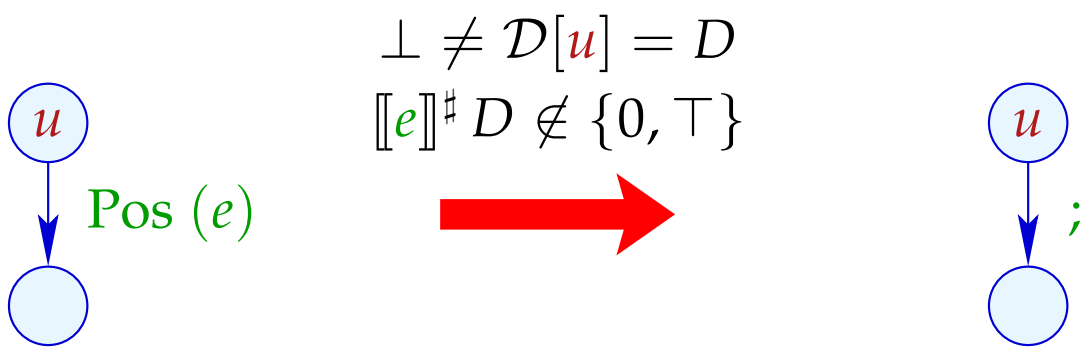
Transformation 4: Beseitigung von totem Code



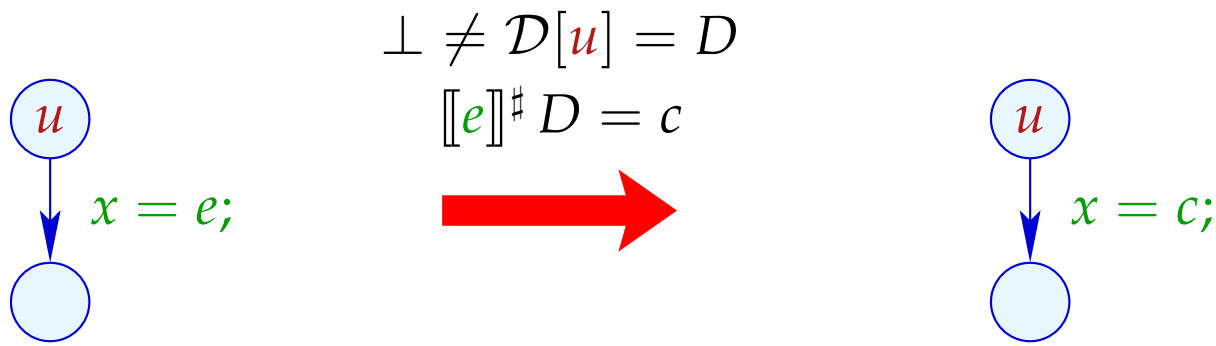


Transformation 4 (Forts.): Beseitigung von totem Code





Transformation 4 (Forts.): Vereinfachte Zuweisungen



Erweiterungen:

- Statt ganzer rechter Seiten kann man auch Teilausdrücke vereinfachen:

$$x + (3 * y) \xRightarrow{\{x \mapsto \top, y \mapsto 5\}} x + 15$$

... und weitere Vereinfachungsregeln anwenden, etwa:

$$\begin{aligned} x * 0 &\Longrightarrow 0 \\ x * 1 &\Longrightarrow x \\ x + 0 &\Longrightarrow x \\ x - 0 &\Longrightarrow x \end{aligned}$$

$$\ldots$$

- Bisher haben wir die Information von Bedingungen nicht optimal ausgenutzt:

$$\text{if } (x == 7)$$
$$y = x + 3;$$

Selbst wenn wir den Wert von x vor der if-Abfrage nicht kennen, wissen wir doch, dass bei Betreten des then-Teils x stets den Wert 7 hat :-)

Wir könnten darum definieren:

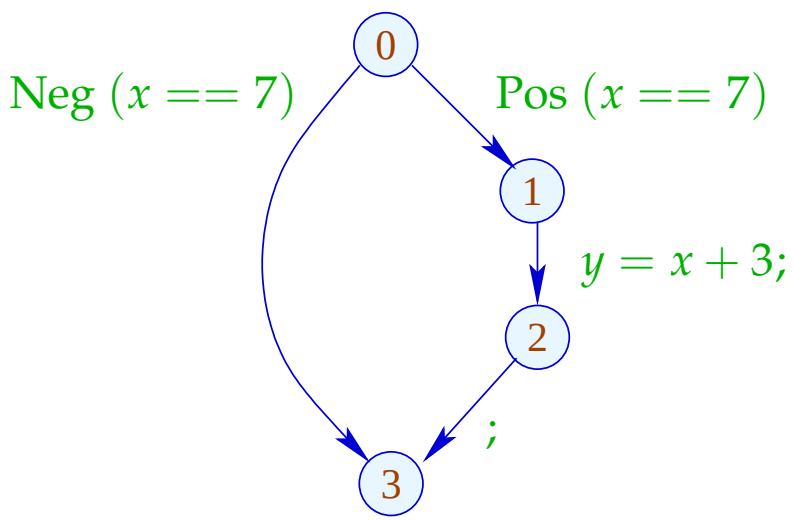
$$[\![\text{Pos}\,(x == e)]\!]^\sharp\, D = \begin{cases} D & \text{falls} \quad [\![x == e]\!]^\sharp\, D = 1 \\ \bot & \text{falls} \quad [\![x == e]\!]^\sharp\, D = 0 \\ D_1 & \text{sonst} \end{cases}$$

wobei

$$D_1 = D \oplus \{x \mapsto (D\,x \sqcap [\![e]\!]^\sharp\, D)\}$$

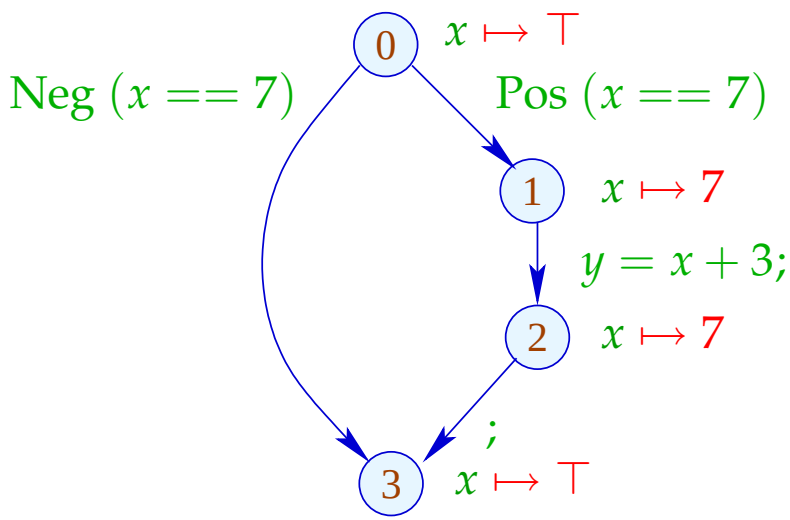
Analog sieht der Kanteneffekt für $\text{Neg}\,(x \neq e)$ aus :-)

Unser Beispiel:



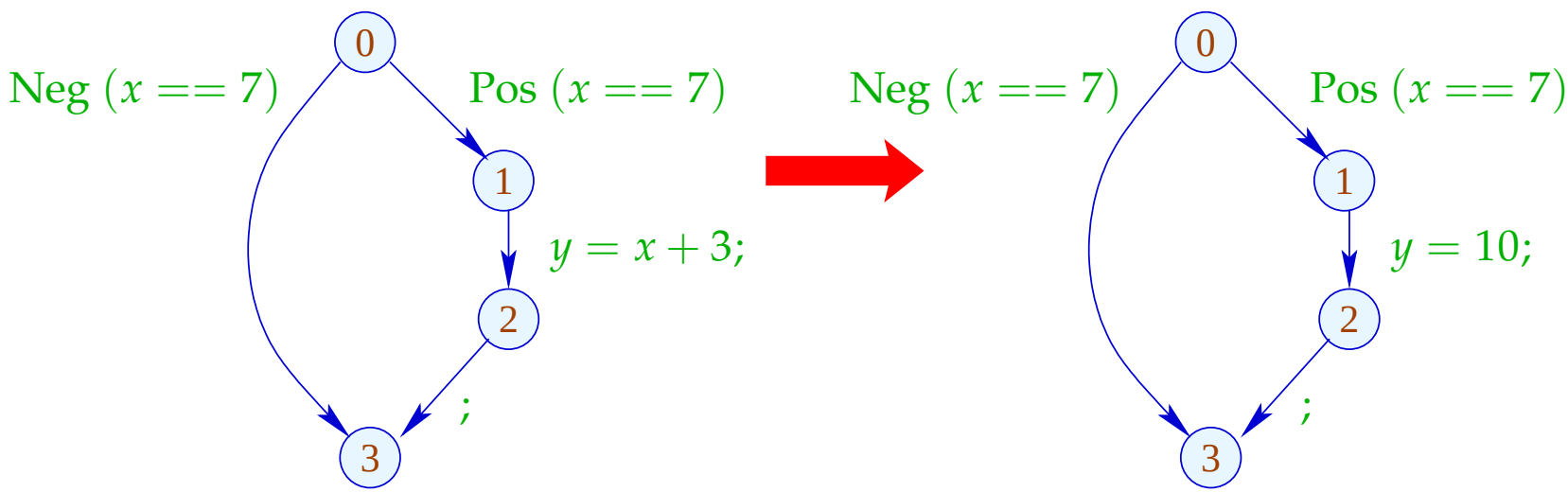
Analog sieht der Kanteneffekt für $\text{Neg}\,(x \neq e)$ aus :-)

Unser Beispiel:



Analog sieht der Kanteneffekt für $\text{Neg}\,(x \neq e)$ aus :-)

Unser Beispiel:



1.5 Intervall-Analyse

Beobachtung:

- Programmiererinnen benutzen oft globale Konstanten, um Debug-Code ein oder aus zu schalten

  $\Longrightarrow$

  Konstantenpropagation ist hilfreich :-)

- Im allgemeinen wird aber der Wert von Variablen nicht bekannt sein — möglicherweise aber ein Intervall !!!

Beispiel:

$$
\begin{array}{l}
\text{for } (i = 0; i < 42; i{+}{+}) \\
\quad \text{if } (0 \leq i \wedge i < 42)\{ \\
\qquad A_1 = A + i; \\
\qquad M[A_1] = i; \\
\quad \} \\
\quad // \quad A \text{ Anfangsadresse eines Felds} \\
\quad // \quad \text{if ist Array-Bound-Check}
\end{array}
$$

Offenbar ist die innere Abfrage überflüssig :-)

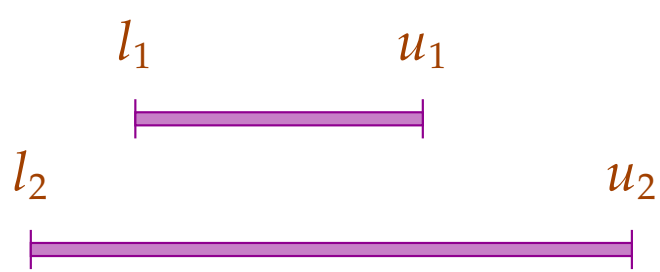
Idee 1:

Bestimme für jede Variable x ein (möglichst kleines :-) Intervall für die möglichen Werte:

$$\mathbb{I} = \{[l, u] \mid l \in \mathbb{Z} \cup \{-\infty\}, u \in \mathbb{Z} \cup \{+\infty\}, l \leq u\}$$

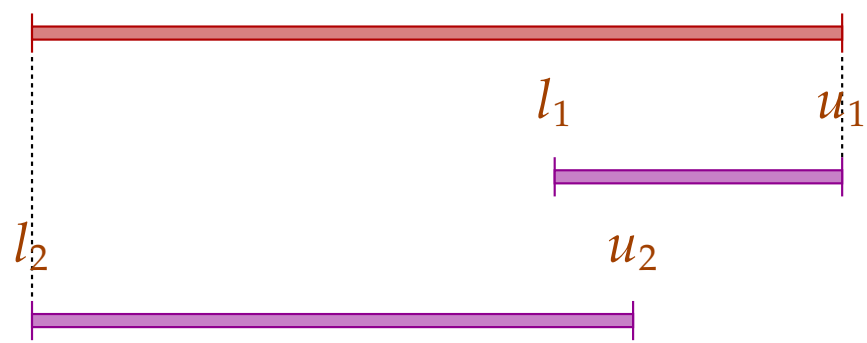
Partielle Ordnung:

$$[l_1, u_1] \sqsubseteq [l_2, u_2] \qquad \text{gdw.} \qquad l_2 \leq l_1 \wedge u_1 \leq u_2$$



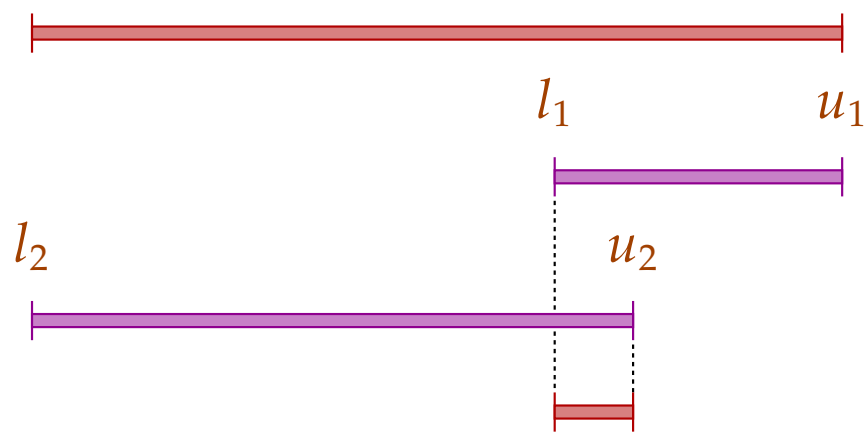
Damit:

$$[l_1, u_1] \sqcup [l_2, u_2] \quad = \quad [l_1 \sqcap l_2, u_1 \sqcup u_2]$$



Damit:

$$
\begin{aligned}
[l_1, u_1] \sqcup [l_2, u_2] &= [l_1 \sqcap l_2, u_1 \sqcup u_2] \\
[l_1, u_1] \sqcap [l_2, u_2] &= [l_1 \sqcup l_2, u_1 \sqcap u_2] \qquad \text{sofern } (l_1 \sqcup l_2) \leq (u_1 \sqcap u_2)
\end{aligned}
$$



Achtung:

$\rightarrow$ $\mathbb{I}$ ist kein vollständiger Verband :-)

$\rightarrow$ $\mathbb{I}$ besitzt unendliche aufsteigende Ketten, z.B.

$$[0,0] \sqsubset [0,1] \sqsubset [-1,1] \sqsubset [-1,2] \sqsubset \ldots$$

Achtung:

$\rightarrow$ $\mathbb{I}$ ist kein vollständiger Verband :-)

$\rightarrow$ $\mathbb{I}$ besitzt unendliche aufsteigende Ketten, z.B.

$$[0,0] \sqsubset [0,1] \sqsubset [-1,1] \sqsubset [-1,2] \sqsubset \ldots$$

Beschreibungsrelation:

$$z \;\Delta\; [l,u] \qquad \text{gdw.} \qquad l \leq z \leq u$$

Konkretisierung:

$$\gamma\,[l,u] = \{z \in \mathbb{Z} \mid l \leq z \leq u\}$$

Beispiel:

$$\begin{aligned} \gamma\,[0,7] &= \{0,\ldots,7\} \\ \gamma\,[0,\infty] &= \{0,1,2,\ldots,\} \end{aligned}$$

Rechnen mit Intervallen: Intervall-Arithmetik :-)

Addition:

$$\begin{aligned} [l_1,u_1] +^\sharp [l_2,u_2] &= [l_1+l_2, u_1+u_2] \qquad \text{wobei} \\ -\infty + _ &= -\infty \\ +\infty + _ &= +\infty \\ // \quad & -\infty + \infty \quad \text{kommt nicht vor} \quad \text{:-)} \end{aligned}$$

Negation:

$$-^\sharp [l, u] \quad = \quad [-u, -l]$$

Multiplikation:

$$\begin{aligned} [l_1, u_1] *^\sharp [l_2, u_2] &= [a, b] \qquad \text{wobei} \\ a &= l_1 l_2 \sqcap l_1 u_2 \sqcap u_1 l_2 \sqcap u_1 u_2 \\ b &= l_1 l_2 \sqcup l_1 u_2 \sqcup u_1 l_2 \sqcup u_1 u_2 \end{aligned}$$

Beispiel:

$$\begin{aligned} [0, 2] *^\sharp [3, 4] &= [0, 8] \\ [-1, 2] *^\sharp [3, 4] &= [-4, 8] \\ [-1, 2] *^\sharp [-3, 4] &= [-6, 8] \\ [-1, 2] *^\sharp [-4, -3] &= [-8, 4] \end{aligned}$$

Division: $$[l_1, u_1] \,/^\sharp\, [l_2, u_2] \;=\; [a, b]$$

- Ist 0 nicht im Nenner-Intervall enthalten, sei:

$$\begin{aligned} a &= l_1/l_2 \sqcap l_1/u_2 \sqcap u_1/l_2 \sqcap u_1/u_2 \\ b &= l_1/l_2 \sqcup l_1/u_2 \sqcup u_1/l_2 \sqcup u_1/u_2 \end{aligned}$$

- Gilt: $l_2 \leq 0 \leq u_2$, setzen wir:

$$[a, b] \;=\; [-\infty, +\infty]$$

Gleichheit:

$$[l_1, u_1] ==^\sharp [l_2, u_2] \;=\; \begin{cases} [1,1] & \text{falls} \quad l_1 = u_1 = l_2 = u_2 \\ [0,0] & \text{falls} \quad u_1 < l_2 \vee u_2 < l_1 \\ [0,1] & \text{sonst} \end{cases}$$

Gleichheit:

$$[l_1, u_1] ==^\sharp [l_2, u_2] \;=\; \begin{cases} [1,1] & \text{falls} \quad l_1 = u_1 = l_2 = u_2 \\ [0,0] & \text{falls} \quad u_1 < l_2 \vee u_2 < l_1 \\ [0,1] & \text{sonst} \end{cases}$$

Beispiel:

$$\begin{array}{rcl} [42,42] ==^\sharp [42,42] & = & [1,1] \\ [0,7] ==^\sharp [0,7] & = & [0,1] \\ [1,2] ==^\sharp [3,4] & = & [0,0] \end{array}$$

Kleiner:

$$[l_1, u_1] <^\sharp [l_2, u_2] = \begin{cases} [1,1] & \text{falls} \quad u_1 < l_2 \\ [0,0] & \text{falls} \quad u_2 \leq l_1 \\ [0,1] & \text{sonst} \end{cases}$$

Kleiner:

$$[l_1, u_1] <^\sharp [l_2, u_2] = \begin{cases} [1,1] & \text{falls} \quad u_1 < l_2 \\ [0,0] & \text{falls} \quad u_2 \leq l_1 \\ [0,1] & \text{sonst} \end{cases}$$

Beispiel:

$$\begin{aligned} [1,2] <^\sharp [9,42] &= [1,1] \\ [0,7] <^\sharp [0,7] &= [0,1] \\ [3,4] <^\sharp [1,2] &= [0,0] \end{aligned}$$

Mithilfe von $\mathbb{I}$ konstruieren wir den vollständigen Verband:

$$\mathbb{D}_{\mathbb{I}} = (\mathit{Vars} \to \mathbb{I})_{\perp}$$

Beschreibungsrelation:

$$\rho \;\; \Delta \;\; D \qquad \text{gdw.} \qquad D \neq \perp \quad \wedge \quad \forall\, x \in \mathit{Vars} : \; (\rho\, x) \;\; \Delta \;\; (D\; x)$$

Die abstrakte Ausdrucksauswertung definieren wir analog Konstantenpropagation. Wir finden:

$$(\llbracket e \rrbracket\; \rho) \;\; \Delta \;\; (\llbracket e \rrbracket^{\sharp}\; D) \qquad \text{sofern} \qquad \rho \;\; \Delta \;\; D$$

Die Kanteneffekte:

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

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

Bessere Ausnutzung von Bedingungen:

$$[\![\text{Pos}\,(e)]\!]^\sharp\, D \;=\; \begin{cases} \bot & \text{falls} \quad [0,0] = [\![e]\!]^\sharp\, D \\ D_1 & \text{sonst} \end{cases}$$

wobei :

$$D_1 \;=\; \begin{cases} D \oplus \{x \mapsto (D\,x) \sqcap ([\![e_1]\!]^\sharp\, D)\} & \text{falls } e \equiv x == e_1 \\ D \oplus \{x \mapsto (D\,x) \sqcap [-\infty, u]\} & \text{falls } e \equiv x \;\leq\; e_1, [\![e_1]\!]^\sharp\, D = [_, u] \\ D \oplus \{x \mapsto (D\,x) \sqcap [l, \infty]\} & \text{falls } e \equiv x \;\geq\; e_1, [\![e_1]\!]^\sharp\, D = [l, _] \end{cases}$$

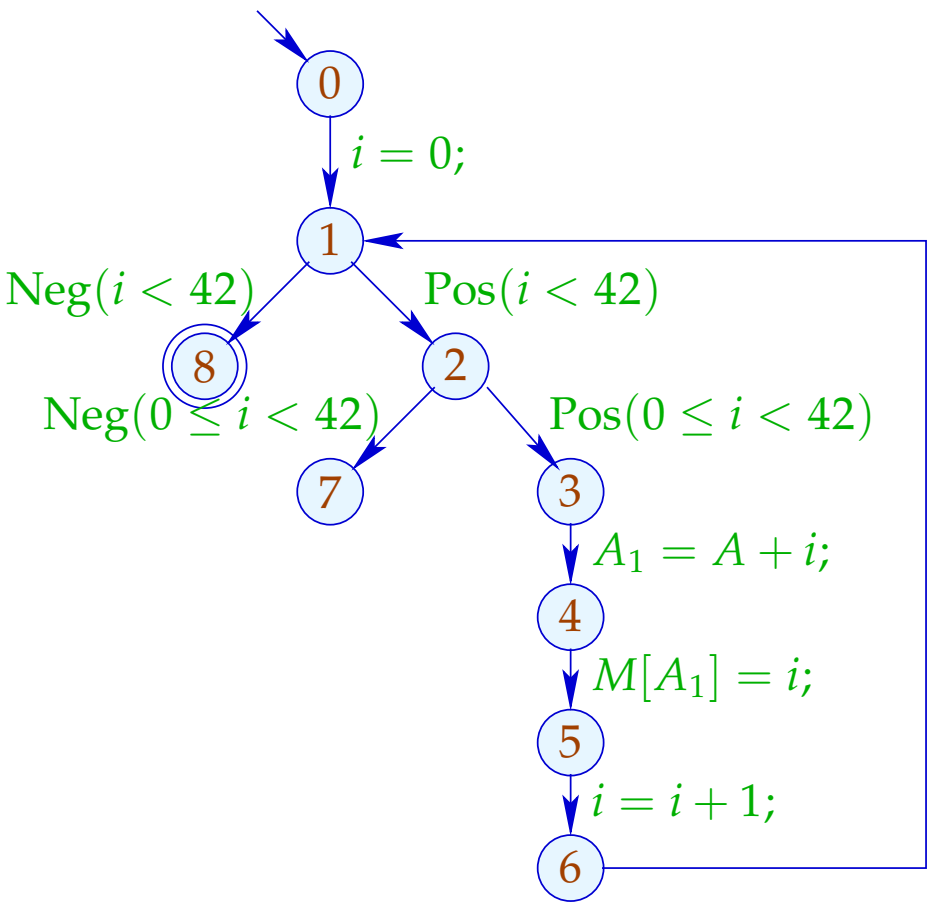
Bessere Ausnutzung von Bedingungen (Forts.):

$$[\![\mathrm{Neg}\,(e)]\!]^\sharp\, D \;=\; \begin{cases} \bot & \text{falls} \quad [0,0] \not\sqsubseteq [\![e]\!]^\sharp\, D \\ D_1 & \text{sonst} \end{cases}$$

wobei :

$$D_1 \;=\; \begin{cases} D \oplus \{x \mapsto (D\,x) \sqcap ([\![e_1]\!]^\sharp\, D)\} & \text{falls } e \equiv x \;\neq\; e_1 \\ D \oplus \{x \mapsto (D\,x) \sqcap [-\infty, u]\} & \text{falls } e \equiv x \;>\; e_1, [\![e_1]\!]^\sharp\, D = [_, u] \\ D \oplus \{x \mapsto (D\,x) \sqcap [l, \infty]\} & \text{falls } e \equiv x \;<\; e_1, [\![e_1]\!]^\sharp\, D = [l, _] \end{cases}$$

Beispiel:



	i	
	l	u
0	$-\infty$	$+\infty$
1	0	42
2	0	41
3	0	41
4	0	41
5	0	41
6	1	42
7	$\perp$	
8	42	42

Problem:

$\rightarrow$ Die Lösung lässt sich mit RR-Iteration berechnen — nach ca. 42 Runden :-(

$\rightarrow$ Auf manchen Programmen terminiert die Iteration nie :-((

Idee 1: Widening

- Iteriere beschleunigt — unter Preisgabe von Präzision :-)
- Erlaube nur beschränkt oft die Modifikation eines Werts !!!

  ... im Beispiel:
- verbiete Updates von Intervall-Grenzen in $\mathbb{Z}$ ...

  $\Longrightarrow$ eine maximale Kette:

$$[3, 17] \sqsubset [3, +\infty] \sqsubset [-\infty, +\infty]$$

Formalisierung dieses Vorgehens:

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

ein Ungleichungsssystem über $\mathbb{D}$, wobei die f_i nicht notwendigerweise monoton sind.

Trotzdem können wir eine akkumulierende Iteration definieren. Betrachte das Gleichungssystem:

$$x_i \;=\; x_i \sqcup f_i\,(x_1, \ldots, x_n)\,, \quad i = 1, \ldots, n \qquad (2)$$

Offenbar gilt:

(a) $\underline{x}$ ist Lösung von (1) gdw. $\underline{x}$ Lösung von (2) ist.

(b) Die Funktion $G : \mathbb{D}^n \to \mathbb{D}^n$ mit
$G\,(x_1, \ldots, x_n) = (y_1, \ldots, y_n)\,, \quad y_i = x_i \sqcup f_i\,(x_1, \ldots, x_n)$
ist vergrößernd, d.h. $\underline{x} \sqsubseteq G\,\underline{x}$ für alle $\underline{x} \in \mathbb{D}^n$.

(c) Die Folge $G^k \underline{\perp}$, $k \geq 0$, ist eine aufsteigende Kette:

$$\underline{\perp} \sqsubseteq G\underline{\perp} \sqsubseteq \ldots \sqsubseteq G^k \underline{\perp} \sqsubseteq \ldots$$

(d) Gilt $G^k \underline{\perp} = G^{k+1} \underline{\perp} = \underline{y}$ ist $\underline{y}$ eine Lösung von (1).

(e) Hat $\mathbb{D}$ unendliche aufsteigende Ketten, ist uns mit (d) noch nicht viel gedient ...

aber: wir könnten statt Gleichungssystem (2) ein Gleichungssystem:

$$x_i = x_i \uplus f_i(x_1, \ldots, x_n) \;, \quad i = 1, \ldots, n \tag{3}$$

betrachten für eine binäre Operation Widening:

$$\uplus : \mathbb{D}^2 \to \mathbb{D} \qquad \text{mit} \qquad v_1 \sqcup v_2 \sqsubseteq v_1 \uplus v_2$$

Dann berechnet (RR)-Iteration für (3) immer noch eine Lösung von (1) :-)

... für die Intervall-Analyse:

- Der vollständige Verband ist: $\mathbb{D}_{\mathbb{I}} \;=\; (\mathit{Vars} \to \mathbb{I})_\perp$
- Das Widening $\sqcup\!\!\!\sqcup$ definieren wir als:

$$\perp \sqcup\!\!\!\sqcup D \;=\; D \sqcup\!\!\!\sqcup \perp \;=\; D \qquad \text{und für} \quad D_1 \neq \perp \neq D_2\text{:}$$

$$\begin{array}{rcll}
(D_1 \sqcup\!\!\!\sqcup D_2)\, x & = & (D_1\, x) \sqcup\!\!\!\sqcup (D_2\, x) & \text{wobei} \\
[l_1, u_1] \sqcup\!\!\!\sqcup [l_2, u_2] & = & [l, u] & \text{mit} \\
l & = & \begin{cases} l_1 & \text{falls} \quad l_1 \leq l_2 \\ -\infty & \text{sonst} \end{cases} & \\
u & = & \begin{cases} u_1 & \text{falls} \quad u_1 \geq u_2 \\ +\infty & \text{sonst} \end{cases} &
\end{array}$$

$\Longrightarrow$ $\sqcup\!\!\!\sqcup$ ist nicht kommutativ !!!

Beispiel:

$$
\begin{array}{rcl}
[0,2] \sqcup [1,2] & = & [0,2] \\
[1,2] \sqcup [0,2] & = & [-\infty,2] \\
[1,5] \sqcup [3,7] & = & [1,+\infty]
\end{array}
$$

$\rightarrow$ Widening liefert schneller größere Werte.

$\rightarrow$ Es sollte so gewählt werden, dass es die Terminierung der Iteration garantiert :-)

$\rightarrow$ Bei Intervall-Analyse begrenzt es die Anzahl der Iterationen auf:

$$\#\mathit{Punkte} \cdot (1 + 2 \cdot \#\mathit{Vars})$$

Fazit:

- Um eine Lösung von (1) über einem vollständigen Verband mit unendlichen aufsteigenden Ketten zu bestimmen, definieren wir ein geeignetes Widening und lösen dann (3) :-)

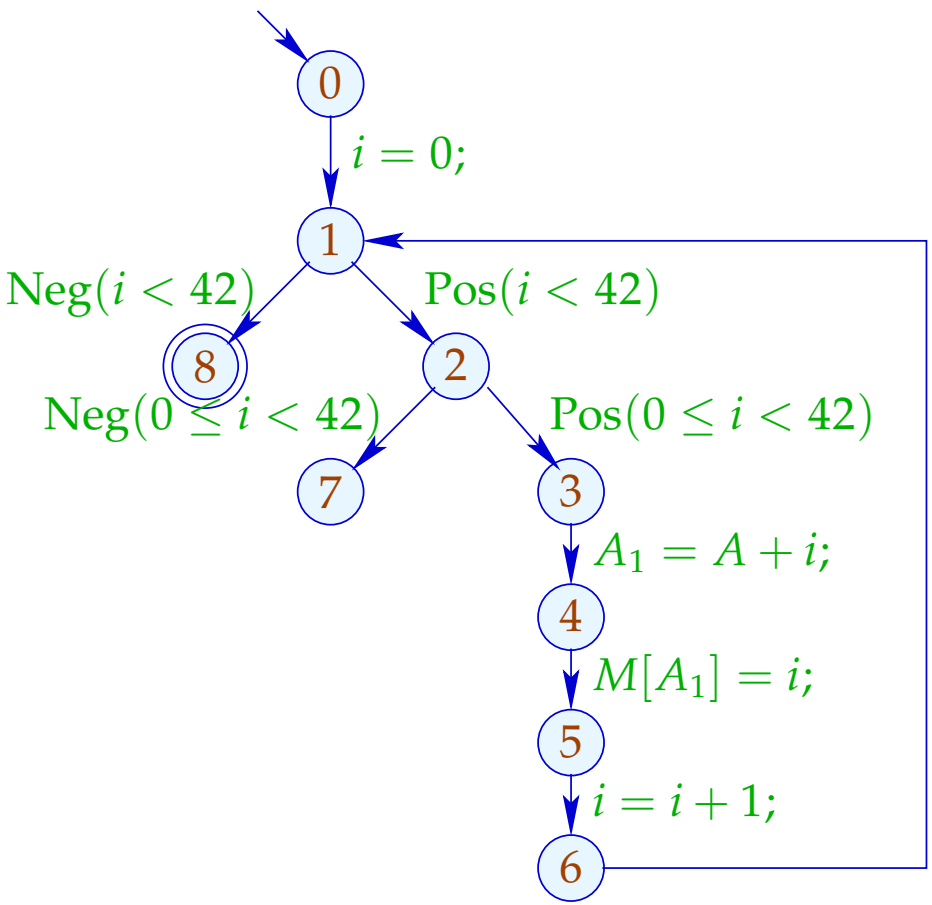
- Achtung: Die Konstruktion geeigneter Widenings ist eine schwarze Kunst !!!

  Oft wählt man $\sqcup$ ganz pragmatisch dynamisch während der Iteration, so dass

  $\rightarrow$ die abstrakten Werte nicht zu kompliziert werden;

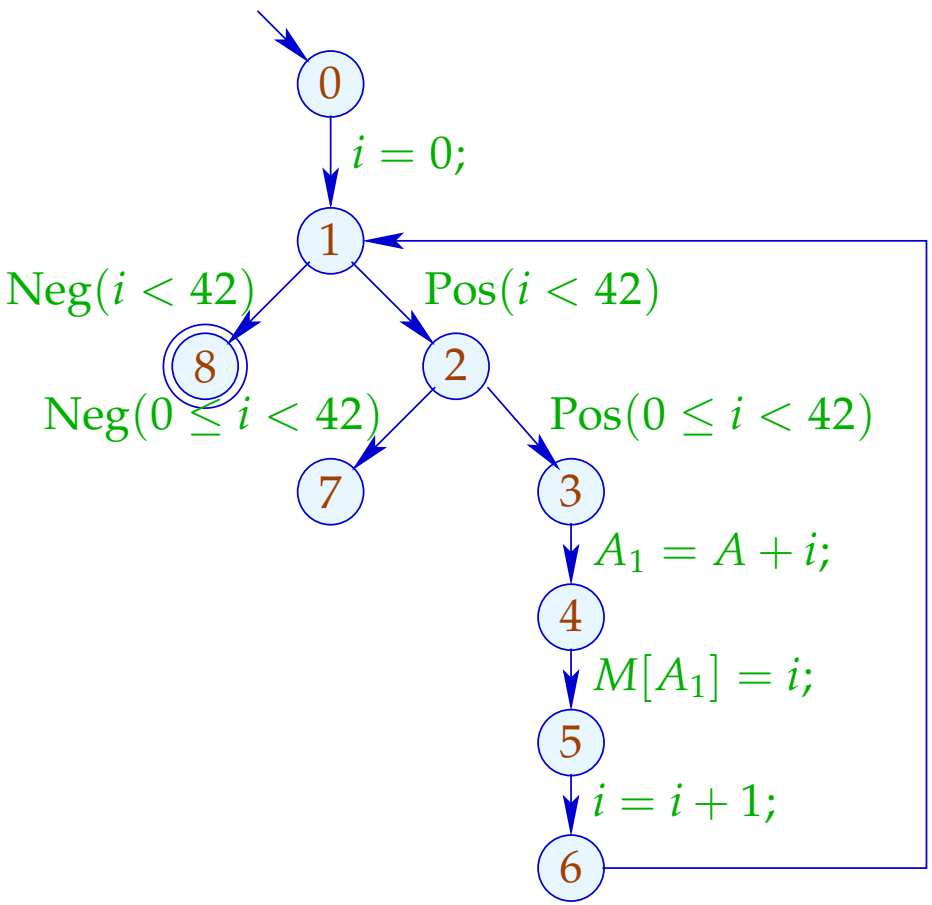
  $\rightarrow$ die Anzahl der Updates fest beschränkt bleibt ...

Unser Beispiel:



	1	
	l	u
0	$-\infty$	$+\infty$
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	1	1
7	$\perp$	
8	$\perp$	

Unser Beispiel:



	1		2		3	
	l	u	l	u	l	u
0	$-\infty$	$+\infty$	$-\infty$	$+\infty$		
1	0	0	0	$+\infty$		
2	0	0	0	$+\infty$		
3	0	0	0	$+\infty$		
4	0	0	0	$+\infty$	dito	
5	0	0	0	$+\infty$		
6	1	1	1	$+\infty$		
7	$\perp$		42	$+\infty$		
8	$\perp$		42	$+\infty$		

... offenbar ist das Ergebnis enttäuschend :-(

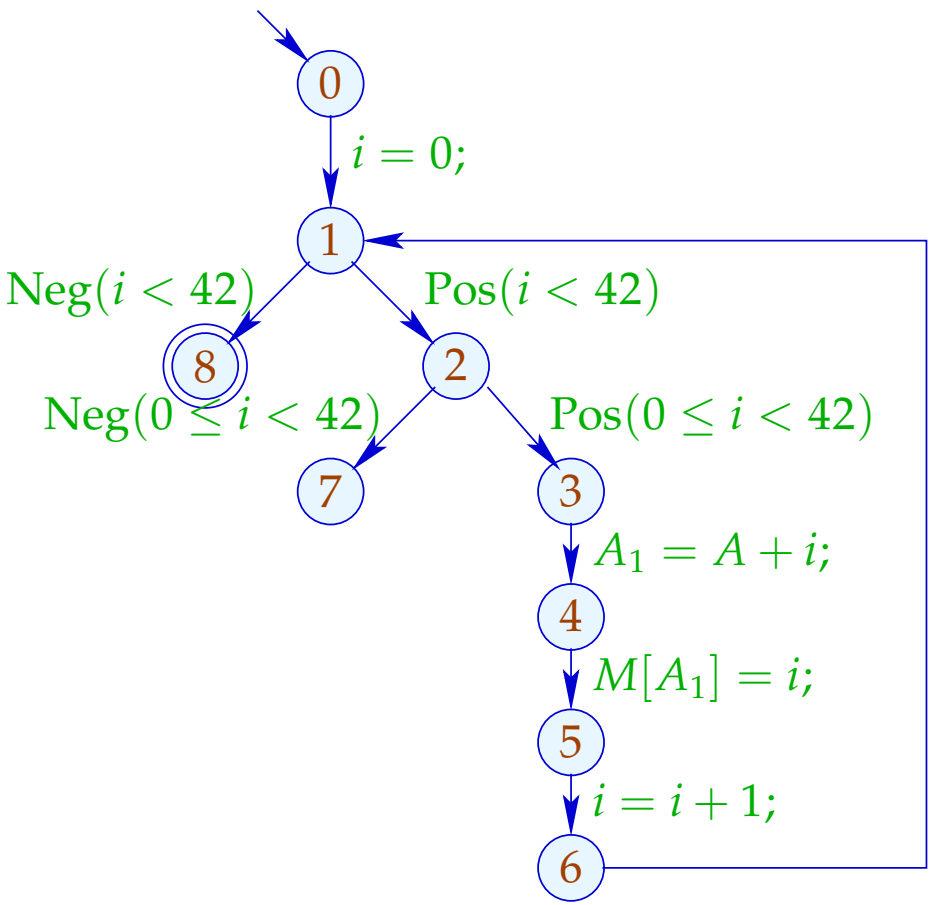
Idee 2:

Eigentlich reicht es, die Beschleunigung mittels $\sqcup$ nur an genügend vielen Stellen anzuwenden!

Eine Menge I heißt Loop Separator (Kreis-Trenner), falls jeder Kreis mindestens einen Punkt aus I enthält :-)

Wenden wir Widening nur an den Punkten aus einer solchen Menge I, terminiert RR-Iteration immer noch !!!

In unserem Beispiel:

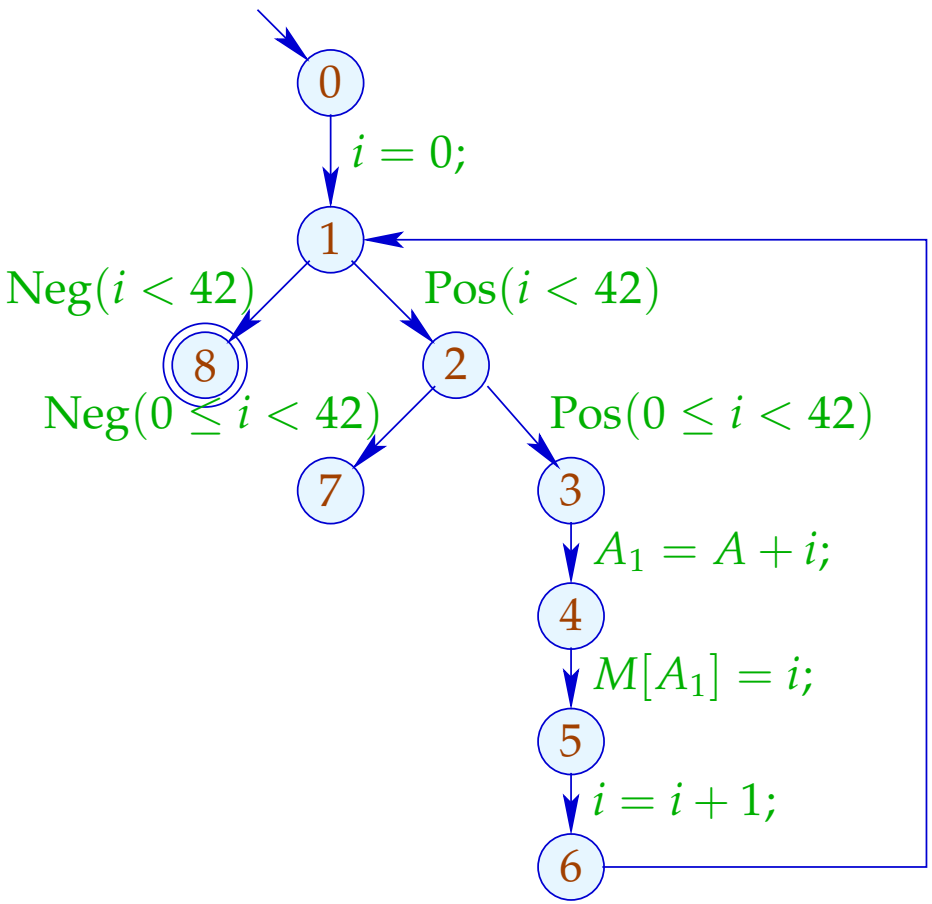


$I_1 = \{1\}$ oder auch:

$I_2 = \{2\}$ oder auch:

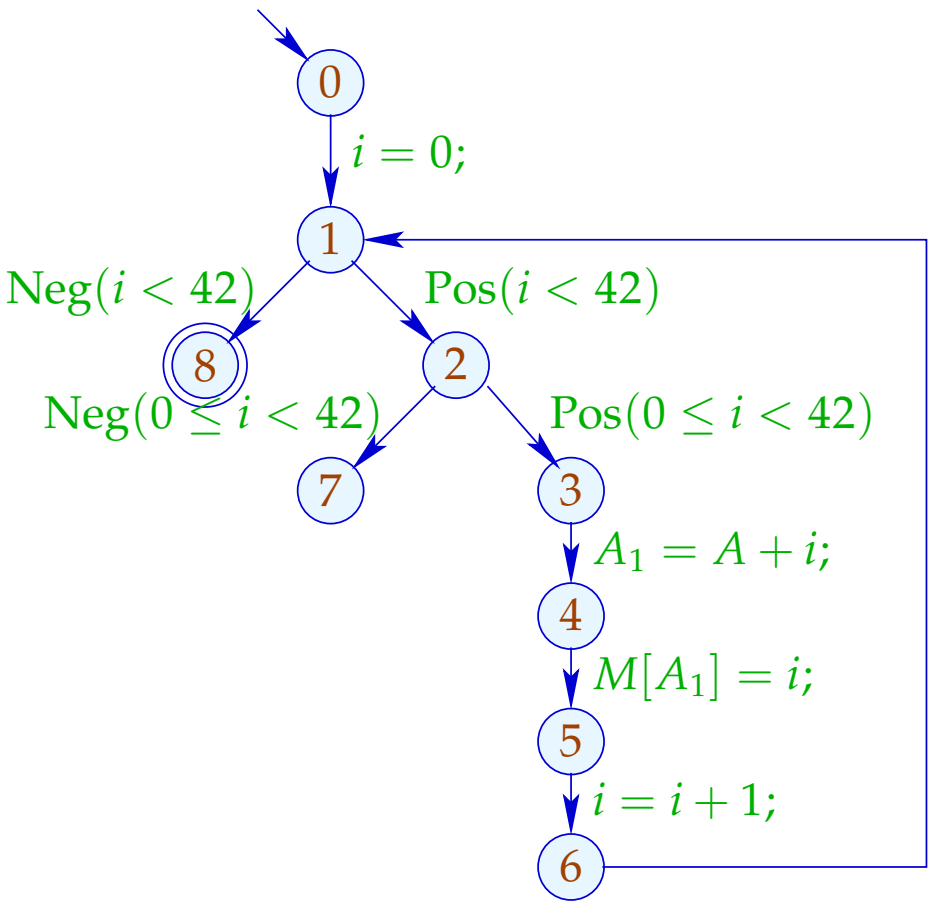
$I_3 = \{3\}$

Die Analyse mit $I = \{1\}$:



	1		2		3	
	l	u	l	u	l	u
0	$-\infty$	$+\infty$	$-\infty$	$+\infty$		
1	0	0	0	$+\infty$		
2	0	0	0	41		
3	0	0	0	41		
4	0	0	0	41	dito	
5	0	0	0	41		
6	1	1	1	42		
7	$\perp$		$\perp$			
8	$\perp$		42	$+\infty$		

Die Analyse mit $I = \{2\}$:



	1		2		3	
	l	u	l	u	l	u
0	$-\infty$	$+\infty$	$-\infty$	$+\infty$		
1	0	0	0	42		
2	0	0	0	$+\infty$		
3	0	0	0	41		
4	0	0	0	41	dito	
5	0	0	0	41		
6	1	1	1	42		
7	$\perp$		42	$+\infty$		
8	$\perp$		42	42		

Diskussion:

- Beide Analysen-Läufe berechnen interessante Informationen :-)
- Der Lauf mit $I = \{2\}$ belegt, dass nach Verlassen der Schleife stets $i = 42$ gilt.
- Nur der Lauf mit $I = \{1\}$ belegt aber, dass der äußere Test den inneren überflüssig macht :-(

Wie findet man einen geeigneten Loop Separator I ???

Idee 3: Narrowing

Sei $\underline{x}$ irgend eine Lösung von (1) , d.h.

$$x_i \sqsupseteq f_i\,\underline{x} \;, \qquad i = 1, \ldots, n$$

Dann gilt für monotone f_i ,

$$\underline{x} \sqsupseteq F\,\underline{x} \sqsupseteq F^2\,\underline{x} \sqsupseteq \ldots \sqsupseteq F^k\,\underline{x} \sqsupseteq \ldots$$

// Narrowing Iteration

Idee 3: Narrowing

Sei $\underline{x}$ irgend eine Lösung von (1), d.h.

$$x_i \sqsupseteq f_i\,\underline{x}\ , \qquad i = 1, \ldots, n$$

Dann gilt für monotone f_i,

$$\underline{x} \sqsupseteq F\,\underline{x} \sqsupseteq F^2\,\underline{x} \sqsupseteq \ldots \sqsupseteq F^k\,\underline{x} \sqsupseteq \ldots$$

// Narrowing Iteration

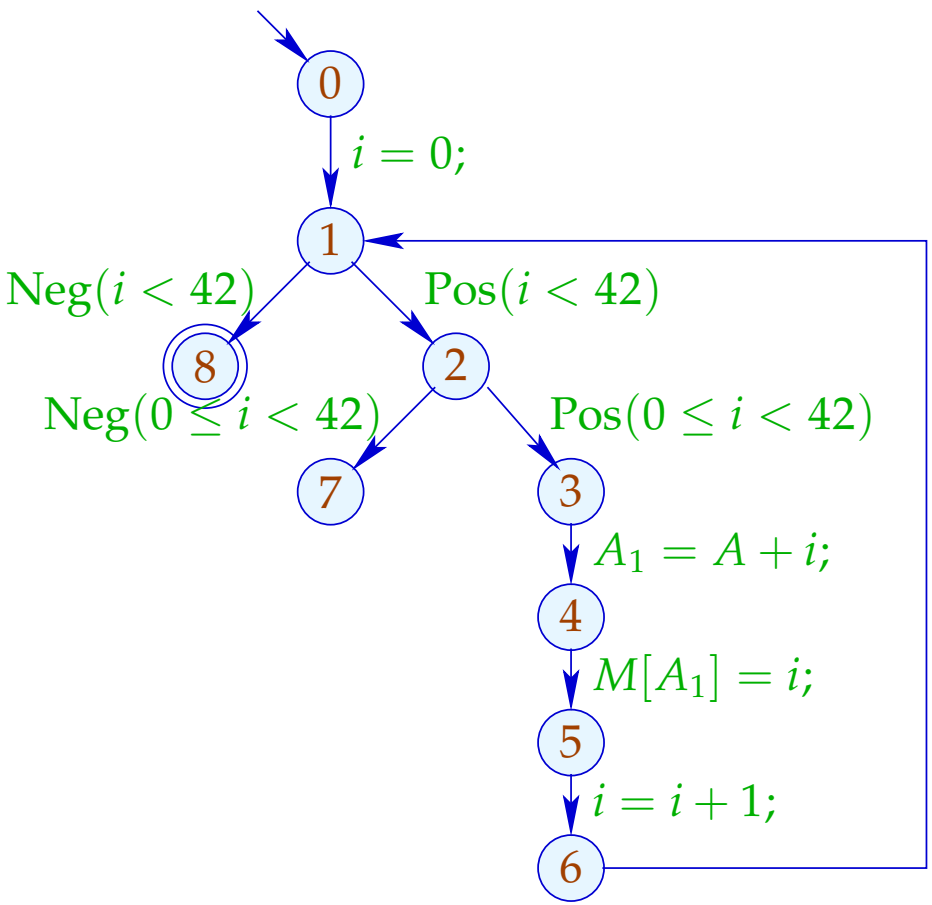
Jeder der Tupel $F^k\,\underline{x}$ ist eine Lösung von (1) :-)

$\Longrightarrow$

Terminierung ist kein Problem mehr:
wir stoppen, wenn wir keine Lust mehr haben :-))

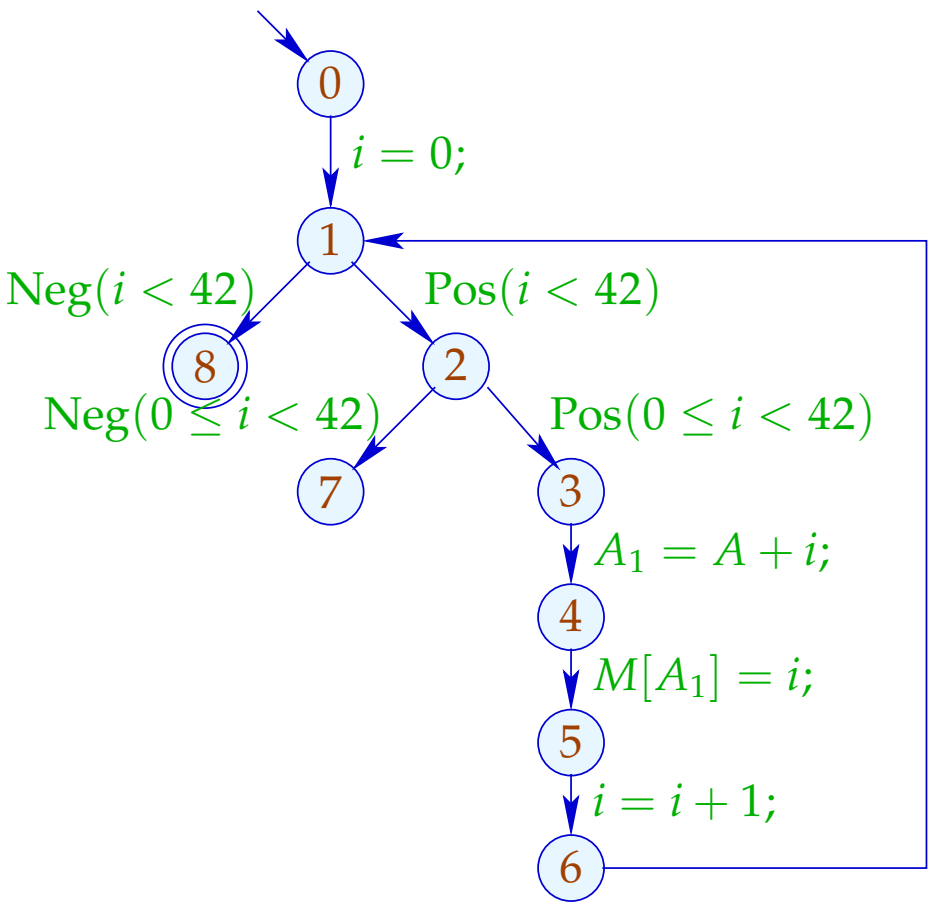
// Analoges gilt für RR-Iteration.

Narrowing Iteration im Beispiel:



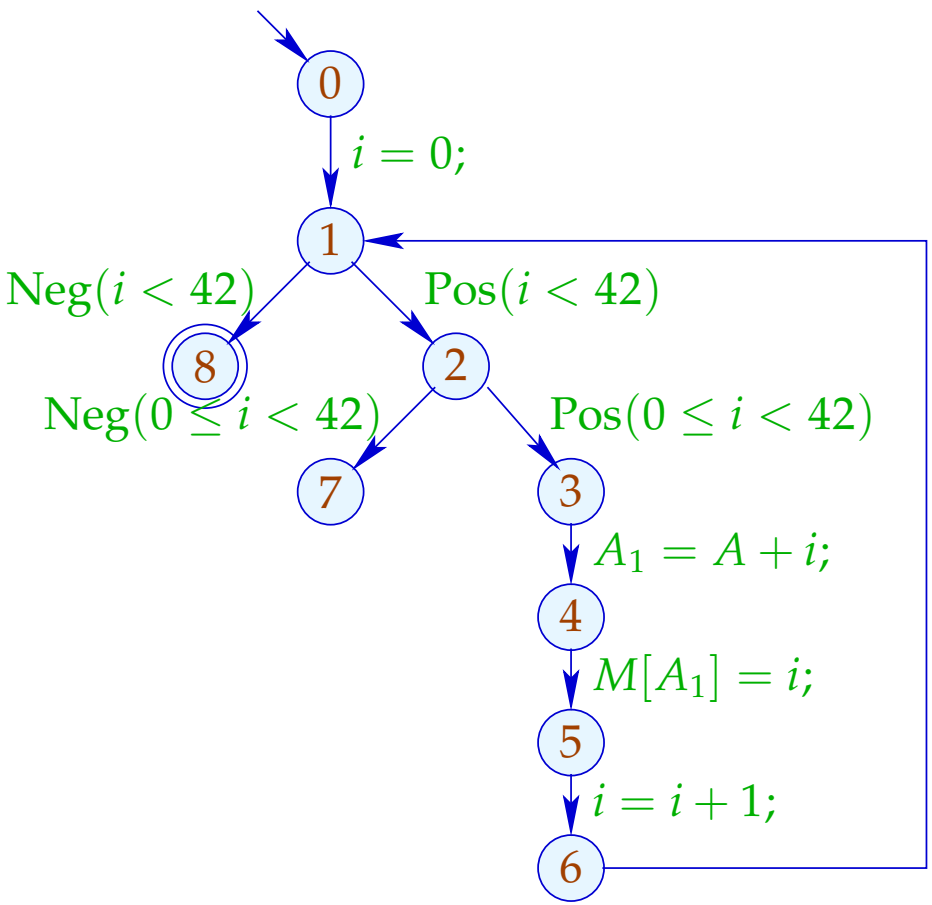
	0	
	l	u
0	$-\infty$	$+\infty$
1	0	$+\infty$
2	0	$+\infty$
3	0	$+\infty$
4	0	$+\infty$
5	0	$+\infty$
6	1	$+\infty$
7	42	$+\infty$
8	42	$+\infty$

Narrowing Iteration im Beispiel:



	0		1	
	l	u	l	u
0	$-\infty$	$+\infty$	$-\infty$	$+\infty$
1	0	$+\infty$	0	$+\infty$
2	0	$+\infty$	0	41
3	0	$+\infty$	0	41
4	0	$+\infty$	0	41
5	0	$+\infty$	0	41
6	1	$+\infty$	1	42
7	42	$+\infty$	$\perp$	
8	42	$+\infty$	42	$+\infty$

Narrowing Iteration im Beispiel:



	0		1		2	
	l	u	l	u	l	u
0	$-\infty$	$+\infty$	$-\infty$	$+\infty$	$-\infty$	$+\infty$
1	0	$+\infty$	0	$+\infty$	0	42
2	0	$+\infty$	0	41	0	41
3	0	$+\infty$	0	41	0	41
4	0	$+\infty$	0	41	0	41
5	0	$+\infty$	0	41	0	41
6	1	$+\infty$	1	42	1	42
7	42	$+\infty$	$\perp$		$\perp$	
8	42	$+\infty$	42	$+\infty$	42	42

Diskussion:

$\rightarrow$ Wir beginnen mit einer sicheren Approximation.

$\rightarrow$ Wir finden, dass die innere Abfrage redundant ist :-)

$\rightarrow$ Wir finden, dass nach der Iteration gilt: $i = 42$:-))

$\rightarrow$ Dazu war nicht erforderlich, einen optimalen Loop Separator zu berechnen :-)))

Letzte Frage:

Müssen wir hinnehmen, dass Narrowing möglicherweise nicht terminiert ???

4. Idee: Beschleunigtes Narrowing

Nehmen wir an, wir hätten eine Lösung $\underline{x} = (x_1, \ldots, x_n)$ des Ungleichungssystems:

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

Dann schreiben betrachten wir das Gleichungssystem:

$$x_i \;=\; x_i \sqcap f_i\,(x_1, \ldots, x_n)\,, \quad i = 1, \ldots, n \qquad (4)$$

Offenbar gilt für monotone f_i : $H^k\,\underline{x} \;=\; F^k\,\underline{x}$:-)

wobei $H\,(x_1, \ldots, x_n) = (y_1, \ldots, y_n)\,, \quad y_i = x_i \sqcap f_i\,(x_1, \ldots, x_n)$.

In (4) ersetzen wir $\sqcap$ durch den neuen Operator $\overline{\sqcap}$ mit:

$$a_1 \sqcap a_2 \;\sqsubseteq\; a_1 \,\overline{\sqcap}\, a_2 \;\sqsubseteq\; a_1$$

... für die Intervall-Analyse:

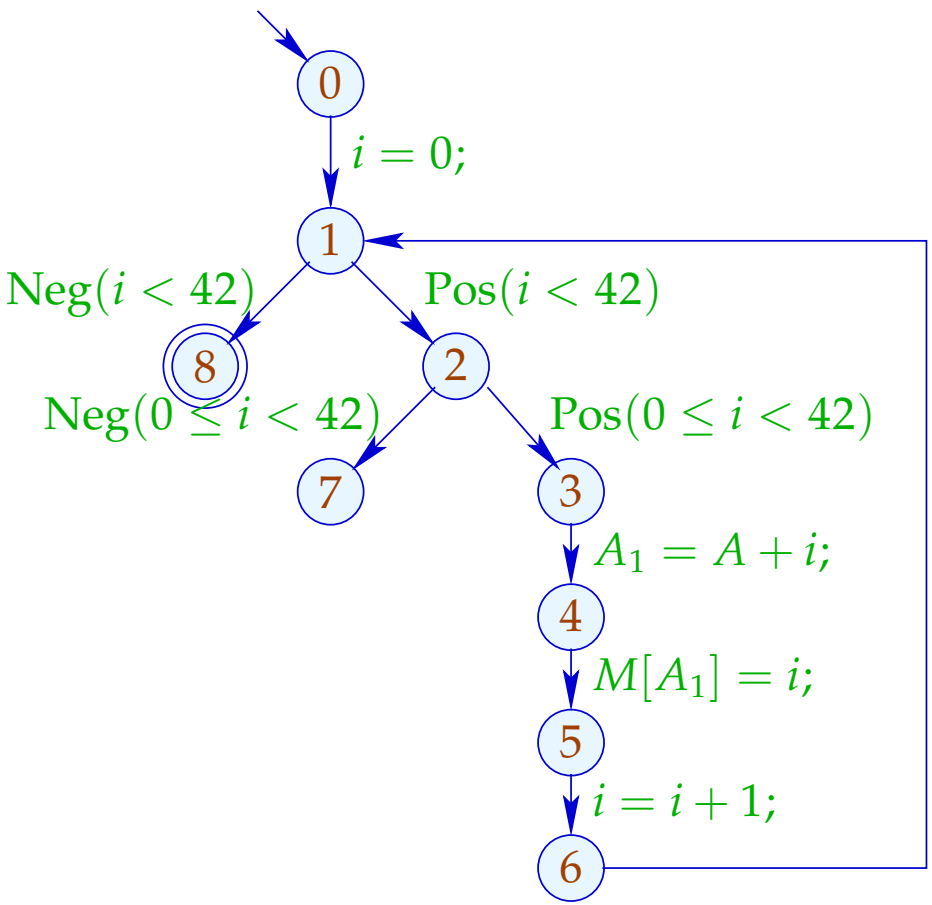
Wir konservieren endliche Intervall-Grenzen :-)

Deshalb $\bot \sqcap D = D \sqcap \bot = \bot$ und für $D_1 \neq \bot \neq D_2$:

$$
\begin{aligned}
(D_1 \sqcap D_2)\, x &= (D_1\, x) \sqcap (D_2\, x) \qquad \text{wobei} \\
[l_1, u_1] \sqcap [l_2, u_2] &= [l, u] \qquad \text{mit} \\
l &= \begin{cases} l_2 & \text{falls} \quad l_1 = -\infty \\ l_1 & \text{sonst} \end{cases} \\
u &= \begin{cases} u_2 & \text{falls} \quad u_1 = \infty \\ u_1 & \text{sonst} \end{cases}
\end{aligned}
$$

$\Longrightarrow$ $\sqcap$ ist nicht kommutativ !!!

Beschleunigtes Narrowing im Beispiel:



	0		1		2	
	l	u	l	u	l	u
0	$-\infty$	$+\infty$	$-\infty$	$+\infty$	$-\infty$	$+\infty$
1	0	$+\infty$	0	$+\infty$	0	42
2	0	$+\infty$	0	41	0	41
3	0	$+\infty$	0	41	0	41
4	0	$+\infty$	0	41	0	41
5	0	$+\infty$	0	41	0	41
6	1	$+\infty$	1	42	1	42
7	42	$+\infty$	$\perp$		$\perp$	
8	42	$+\infty$	42	$+\infty$	42	42

Diskussion:

→ Achtung: Widening liefert für nicht-monotone f_i eine Lösung. Narrowing liefert dagegen nur für monotone f_i eine Lösung!!

→ Das beschleunigte Narrowing liefert (im Beispiel) das richtige Ergebnis :-)

→ Erlaubt der neue Operator $\sqcap$ nur endlich viele Verbesserungen bei jedem Wert, kann Narrowing bis zur Stabilisierung durchgeführt werden.

→ Für die Intervall-Analyse sind das maximal

$$\#\mathit{Punkte} \cdot (1 + 2 \cdot \#\mathit{Vars})$$