Ein Pfad $\pi = k_1 k_2 \ldots k_m$ ist eine Berechnung für den Zustand s falls:

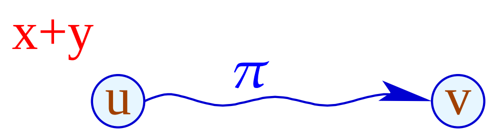
$$s \in def\ (\llbracket k_m \rrbracket \circ \ldots \circ \llbracket k_1 \rrbracket)$$

Das Ergebnis der Berechnung ist:

$$\llbracket \pi \rrbracket\, s = (\llbracket k_m \rrbracket \circ \ldots \circ \llbracket k_1 \rrbracket)\, s$$

Anwendung:

Nehmen wir an, wir hätten am Punkt u den Wert von $x + y$ berechnet:



Wir führen eine Berechnung entland des Pfades π aus und erreichen v, wo wir erneut $x + y$ berechnen sollen ...

Idee:

Wenn x und y in π nicht verändert werden, dann muss $x + y$ in v den gleichen Wert liefern wie in u :-)

Diese Eigenschaft können wir an jeder Kante in π überprüfen   :-}

Idee:

Wenn x und y in π nicht verändert werden, dann muss $x + y$ in v den gleichen Wert liefern wie in u :-)

Diese Eigenschaft können wir an jeder Kante in π überprüfen   :-}

Allgemeiner:

Nehmen wir an, in u hätten wir die Werte der Ausdrücke aus $A = \{e_1, \ldots, e_r\}$ zur Verfügung.

Idee:

Wenn x und y in π nicht verändert werden, dann muss $x + y$ in v den gleichen Wert liefern wie in u :-)

Diese Eigenschaft können wir an jeder Kante in π überprüfen :-}

Allgemeiner:

Nehmen wir an, in u hätten wir die Werte der Ausdrücke aus $A = \{e_1, \ldots, e_r\}$ zur Verfügung.

Jede Kante k transformiert diese Menge in eine Menge $[\![k]\!]^\sharp\, A$ von Ausdrücken, die nach Ausführung von k verfügbar sind ...

... die wir zur Ermittlung des Effekts eines Pfads $\pi = k_1 \ldots k_r$ zusammen setzen können:

$$[\![\pi]\!]^\sharp = [\![k_r]\!]^\sharp \circ \ldots \circ [\![k_1]\!]^\sharp$$

... die wir zur Ermittlung des Effekts eines Pfads $\pi = k_1 \ldots k_r$ zusammen setzen können:

$$[\![\pi]\!]^\sharp = [\![k_r]\!]^\sharp \circ \ldots \circ [\![k_1]\!]^\sharp$$

Der Effekt $[\![k]\!]^\sharp$ einer Kante $k = (u, lab, v)$ hängt nur vom Label lab ab, d.h. $[\![k]\!]^\sharp = [\![lab]\!]^\sharp$

... die wir zur Ermittlung des Effekts eines Pfads $\pi = k_1 \ldots k_r$ zusammen setzen können:

$$[\![\pi]\!]^\sharp = [\![k_r]\!]^\sharp \circ \ldots \circ [\![k_1]\!]^\sharp$$

Der Effekt $[\![k]\!]^\sharp$ einer Kante $k = (u, lab, v)$ hängt nur vom Label lab ab, d.h. $[\![k]\!]^\sharp = [\![lab]\!]^\sharp$ wobei:

$$\begin{array}{lllll}
[\![;]\!]^\sharp\, A & = & A & & \\
[\![Pos(e)]\!]^\sharp\, A & = & [\![Neg(e)]\!]^\sharp\, A & = & A \cup \{e\} \\
[\![R = e;]\!]^\sharp\, A & = & (A \cup \{e\}) \backslash Expr_R & & \text{wobei} \\
& & \multicolumn{3}{l}{Expr_R \text{ alle Ausdrücke sind, die } R \text{ enthalten}}
\end{array}$$

$$
\begin{aligned}
[\![R = M[e];]\!]^\sharp\, A &= (A \cup \{e\}) \backslash Expr_R \\
[\![M[e_1] = e_2;]\!]^\sharp\, A &= A \cup \{e_1, e_2\}
\end{aligned}
$$

$$\begin{array}{lll} [\![R = M[e];]\!]^\sharp\, A & = & (A \cup \{e\}) \backslash Expr_R \\ [\![M[e_1] = e_2;]\!]^\sharp\, A & = & A \cup \{e_1, e_2\} \end{array}$$

Damit können wir jeden Pfad untersuchen :-)

In einem Programm kann es mehrere Pfade geben :-(

Bei jeder Eingabe kann ein anderer gewählt werden :-((

$$
\begin{array}{lll}
[\![R = M[e];]\!]^\sharp\, A & = & (A \cup \{e\}) \backslash Expr_R \\
[\![M[e_1] = e_2;]\!]^\sharp\, A & = & A \cup \{e_1, e_2\}
\end{array}
$$

Damit können wir jeden Pfad untersuchen :-)

In einem Programm kann es mehrere Pfade geben :-(

Bei jeder Eingabe kann ein anderer gewählt werden :-((

$\Longrightarrow$ Wir benötigen die Menge:

$$
\mathcal{A}[v] \quad = \quad \bigcap \{ [\![\pi]\!]^\sharp \emptyset \mid \pi : start \rightarrow^* v \}
$$

Im Klartext:

$\longrightarrow$ Wir betrachten sämtliche Pfade, die v erreichen.

$\longrightarrow$ Für jeden Pfad π bestimmen wir die Menge der entlang π verfügbaren Ausdrücke.

$\longrightarrow$ Vor Programm-Ausführung ist nichts verfügbar :-)

$\longrightarrow$ Wir bilden den Durchschnitt $\Longrightarrow$ sichere Information

Im Klartext:

$\longrightarrow$ Wir betrachten sämtliche Pfade, die v erreichen.

$\longrightarrow$ Für jeden Pfad π bestimmen wir die Menge der entlang π verfügbaren Ausdrücke.

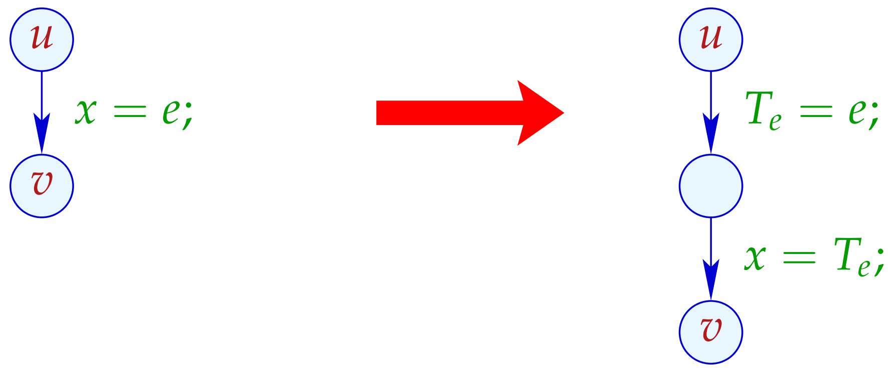
$\longrightarrow$ Vor Programm-Ausführung ist nichts verfügbar :-)

$\longrightarrow$ Wir bilden den Durchschnitt $\Longrightarrow$ sichere Information

Wie nutzen wir diese Information aus ???

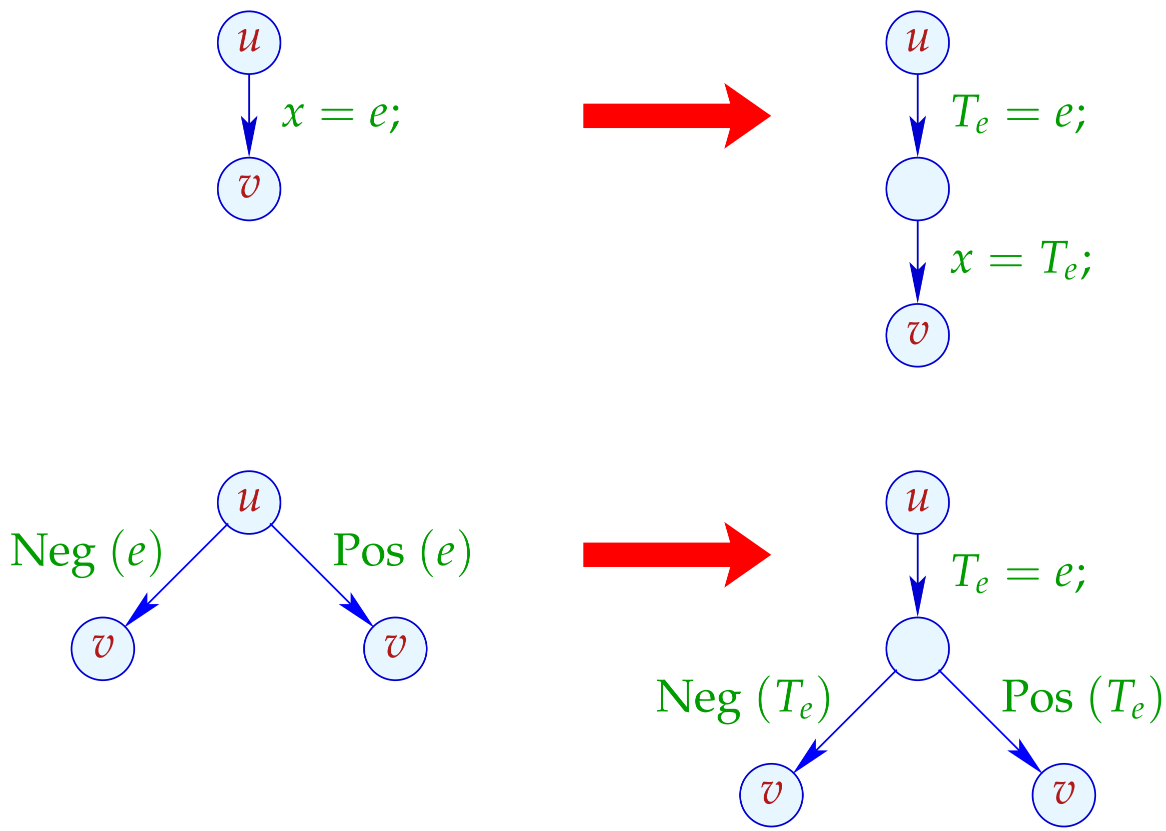
Transformation 1.1:

Wir stellen neue Register T_e als Speicherplatz für die e bereit:



Transformation 1.1:

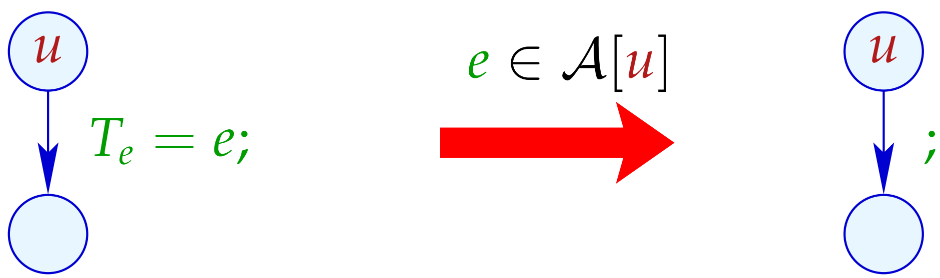
Wir stellen neue Register T_e als Speicherplatz für die e bereit:



... analog für $R = M[e];$ und $M[e_1] = e_2;$.

Transformation 1.2:

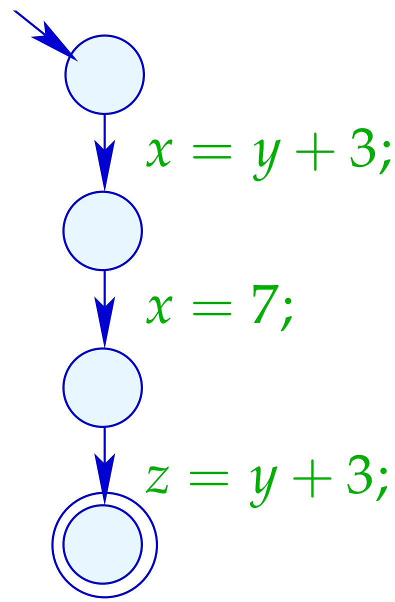
Falls e am Punkt u verfügbar ist, wird e nicht neu berechnet:



Wir ersetzen dann die Zuweisung durch Nop :-)

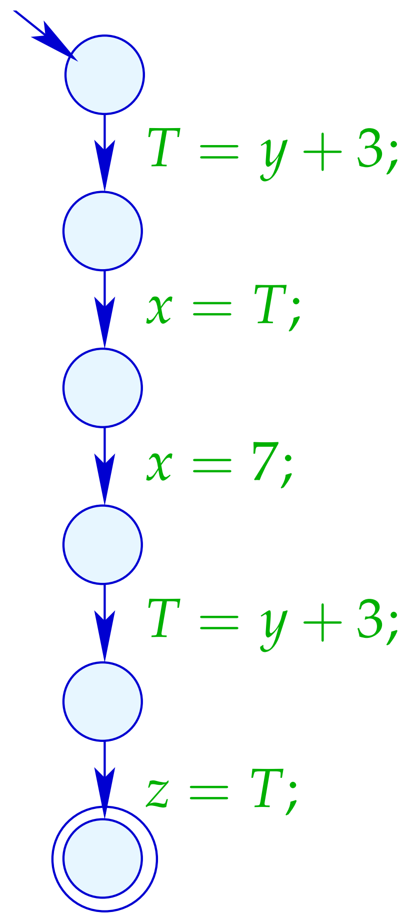
Beispiel:

$$
\begin{array}{lll}
x & = & y+3; \\
x & = & 7; \\
z & = & y+3;
\end{array}
$$



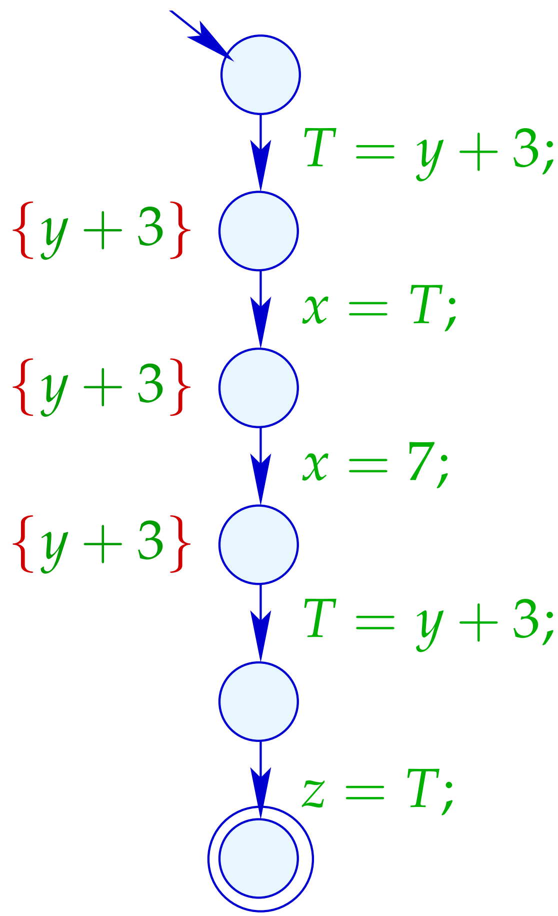
Beispiel:

$$\begin{array}{lll} x & = & y + 3; \\ x & = & 7; \\ z & = & y + 3; \end{array}$$



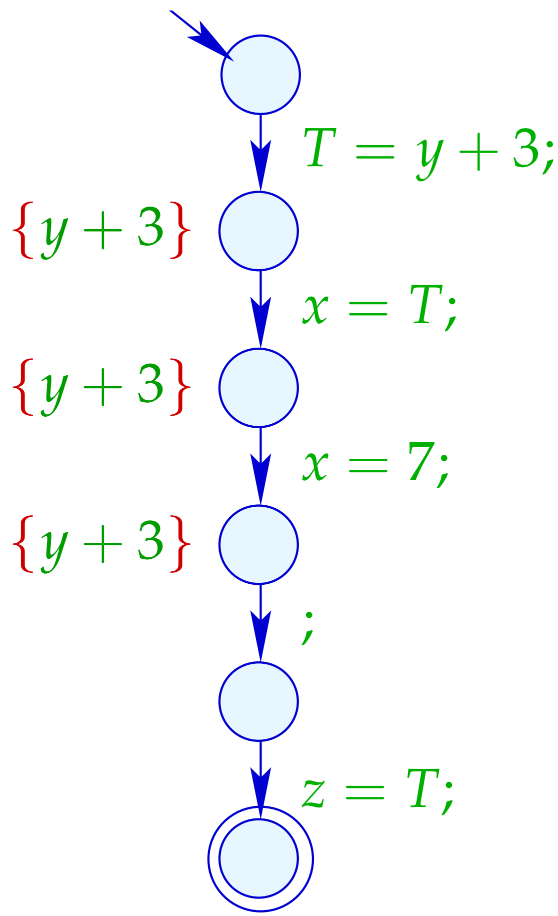
Beispiel:

$$\begin{aligned} x &= y+3; \\ x &= 7; \\ z &= y+3; \end{aligned}$$



Beispiel:

$$\begin{aligned} x &= y+3; \\ x &= 7; \\ z &= y+3; \end{aligned}$$



Korrektheit: (Idee)

Transformation 1.1 erhält offenbar die Bedeutung und $\mathcal{A}[u]$ für alle Knoten u :-)

Sei $\pi : start \rightarrow^* u$ der Pfad, den eine Berechnung nimmt.

Ist $e \in \mathcal{A}[u]$, dann auch $e \in [\![\pi]\!]^\sharp\, \emptyset$.

Dann muss es eine Zerlegung von π geben:

$$\text{start} \overset{\pi_1}{\rightsquigarrow} u_1 \xrightarrow{k} u_2 \overset{\pi_2}{\rightsquigarrow} u$$

mit den folgenden Eigenschaften:

- Der Ausdruck e wird an der Kante k berechnet;
- Der Ausdruck e wird an keiner Kante in π_2 aus der Menge der verfügbaren Ausdrücke entfernt, d.h. keine Variable von e erhält einen neuen Wert :-)

- Der Ausdruck e wird an der Kante k berechnet;
- Der Ausdruck e wird an keiner Kante in π_2 aus der Menge der verfügbaren Ausdrücke entfernt, d.h. keine Variable von e erhält einen neuen Wert :-)

$$\Longrightarrow$$

Wird u erreicht, enthält das Register T_e den Wert von e :-))

Achtung:

Die Transformation 1.1 ist nur sinnvoll an Zuweisungen $x = e;$, wobei:

$\longrightarrow$ $x \not\in Vars(e)$;

$\longrightarrow$ $e \not\in Vars$;

$\longrightarrow$ sich die Berechnung von e lohnt :-}

Achtung:

Die Transformation 1.1 ist nur sinnvoll an Zuweisungen $x = e;$, wobei:

$\rightarrow$ $x \notin Vars(e)$;

$\rightarrow$ $e \notin Vars$;

$\rightarrow$ sich die Berechnung von e lohnt :-}

Bleibt die Preisfrage ...

Preisfrage:

Wie berechnen wir $\mathcal{A}[u]$ für jeden Programmpunkt u ??

Preisfrage:

Wie berechnen wir $\mathcal{A}[u]$ für jeden Programmpunkt u ??

Idee:

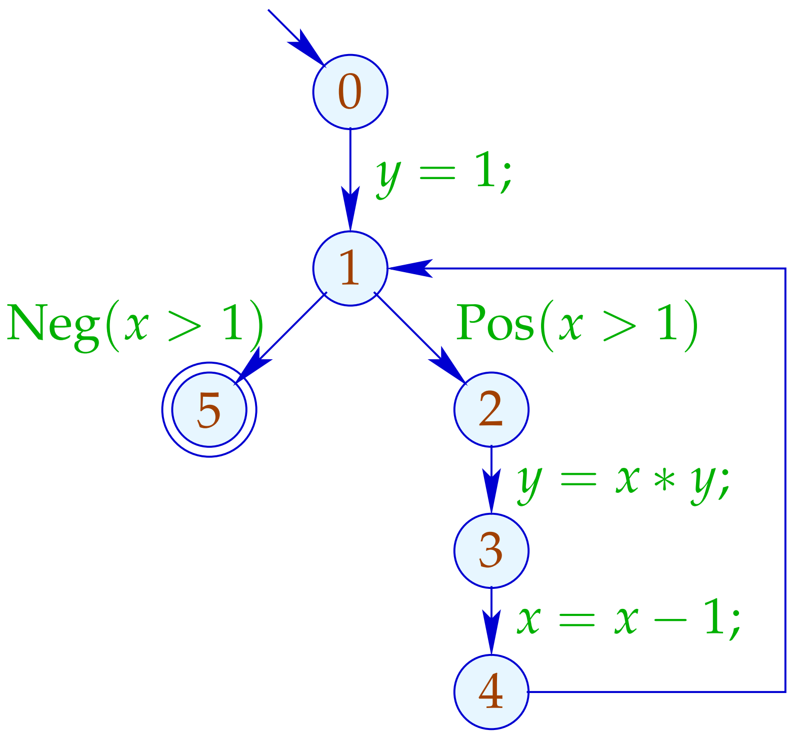
Wir stellen ein Ungleichungssystem auf, das alle Bedingungen an die Werte $\mathcal{A}[u]$ sammelt:

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

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

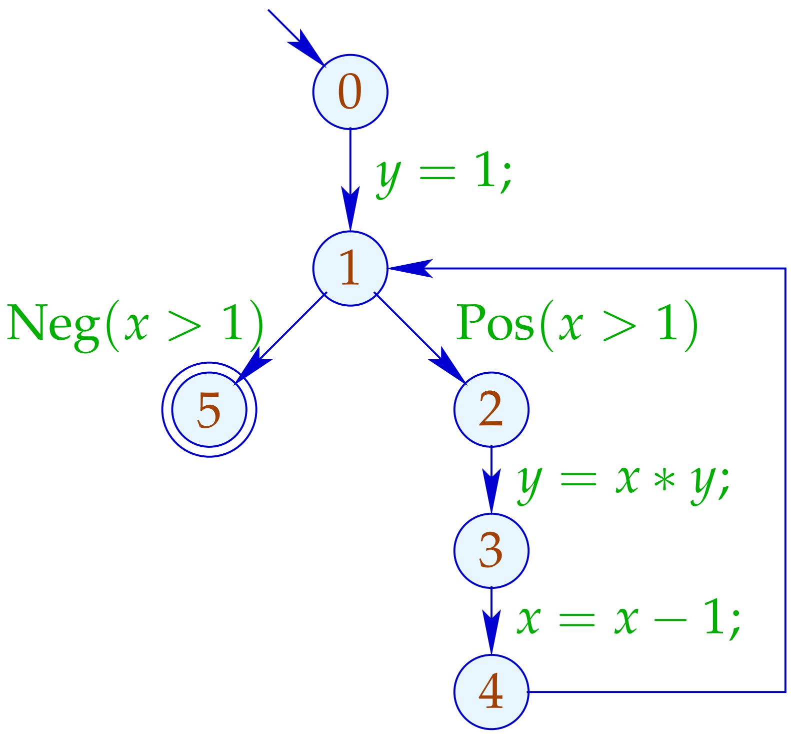
Beispiel:



Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:

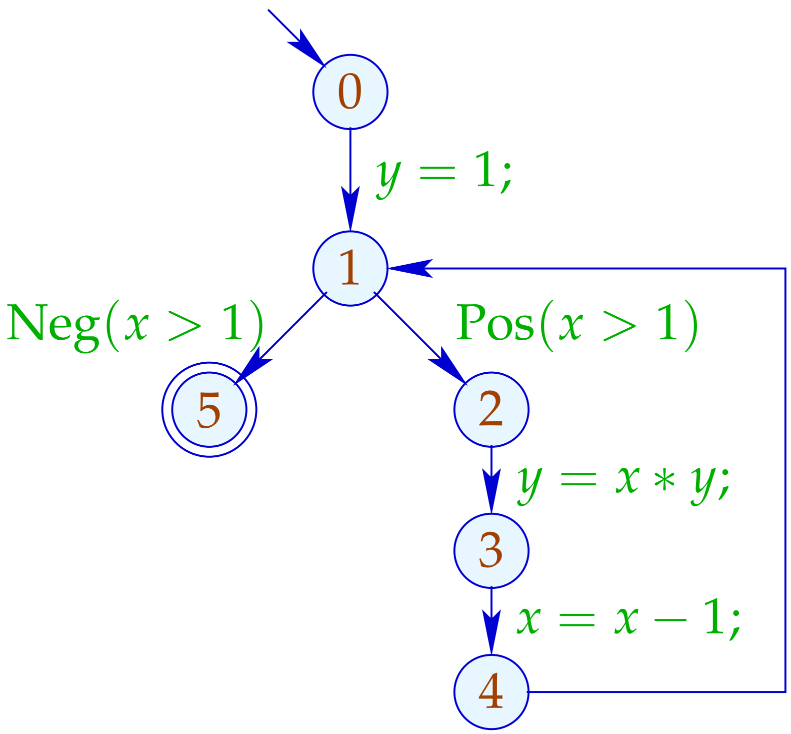


$$\mathcal{A}[0] \quad \subseteq \quad \emptyset$$

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:

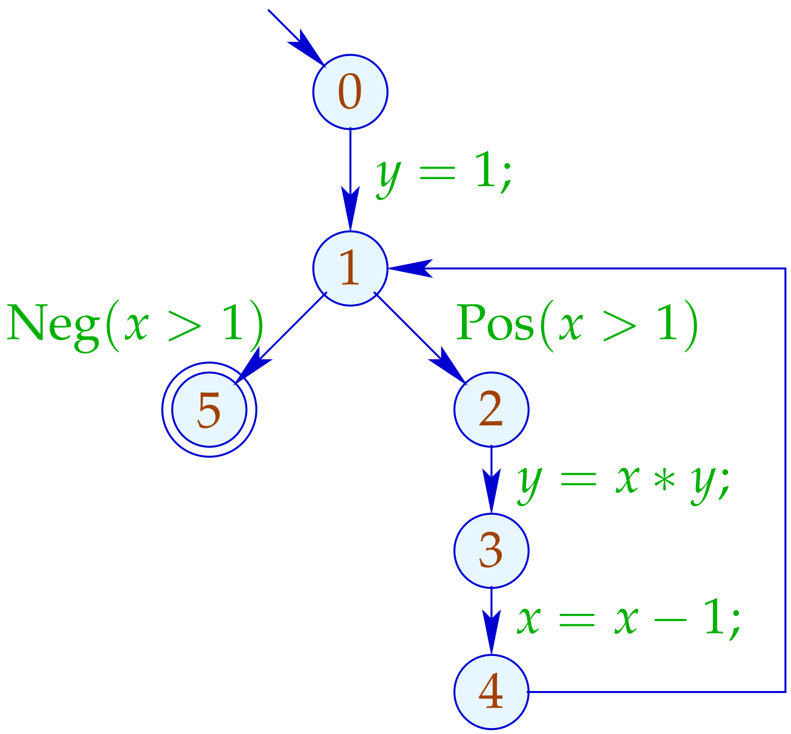


$$\begin{array}{lll} \mathcal{A}[0] & \subseteq & \emptyset \\ \mathcal{A}[1] & \subseteq & (\mathcal{A}[0] \cup \{1\}) \backslash Expr_y \\ \mathcal{A}[1] & \subseteq & \mathcal{A}[4] \end{array}$$

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:

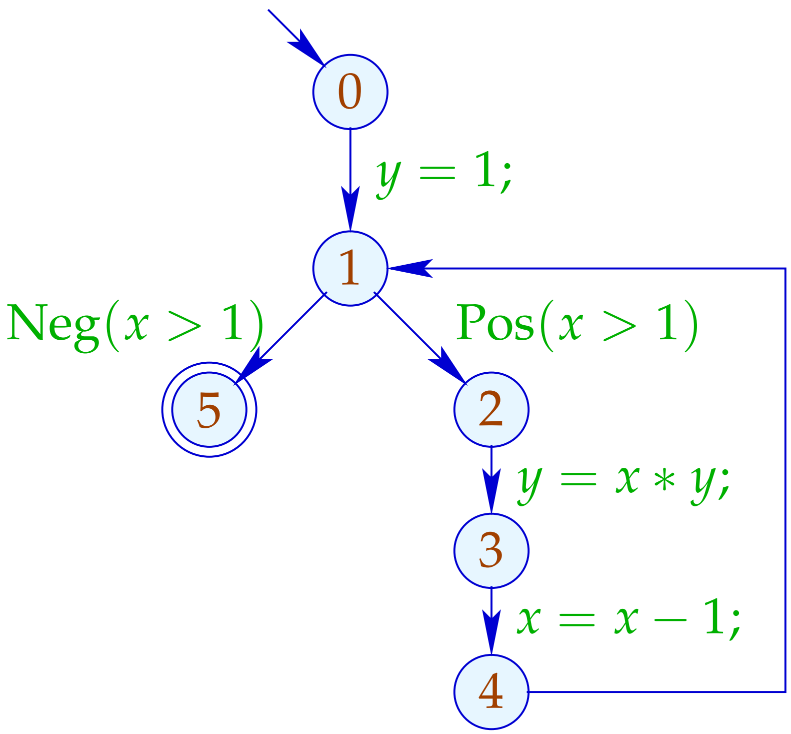


$$\begin{array}{lll} \mathcal{A}[0] & \subseteq & \emptyset \\ \mathcal{A}[1] & \subseteq & (\mathcal{A}[0] \cup \{1\}) \backslash Expr_y \\ \mathcal{A}[1] & \subseteq & \mathcal{A}[4] \\ \mathcal{A}[2] & \subseteq & \mathcal{A}[1] \cup \{x > 1\} \end{array}$$

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:

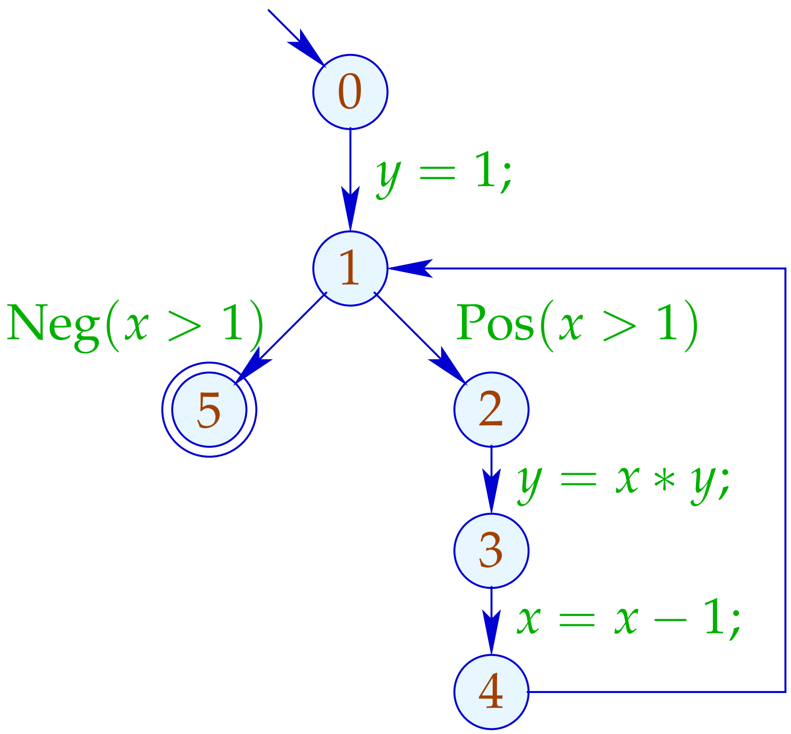


$$
\begin{array}{lll}
\mathcal{A}[0] & \subseteq & \emptyset \\
\mathcal{A}[1] & \subseteq & (\mathcal{A}[0] \cup \{1\}) \backslash Expr_y \\
\mathcal{A}[1] & \subseteq & \mathcal{A}[4] \\
\mathcal{A}[2] & \subseteq & \mathcal{A}[1] \cup \{x > 1\} \\
\mathcal{A}[3] & \subseteq & (\mathcal{A}[2] \cup \{x * y\}) \backslash Expr_y
\end{array}
$$

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:

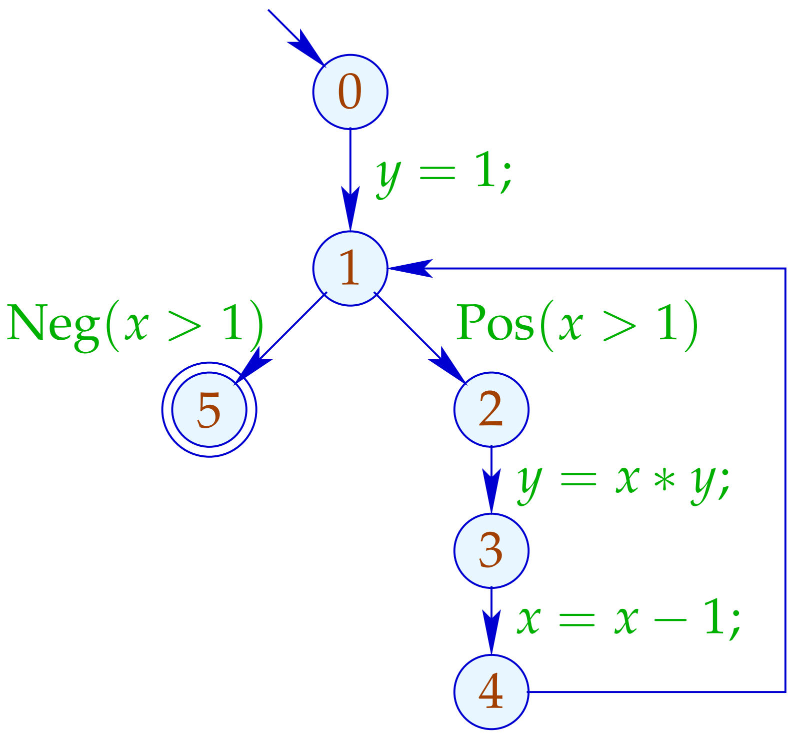


$$
\begin{array}{lcl}
\mathcal{A}[0] & \subseteq & \emptyset \\
\mathcal{A}[1] & \subseteq & (\mathcal{A}[0] \cup \{1\}) \backslash Expr_y \\
\mathcal{A}[1] & \subseteq & \mathcal{A}[4] \\
\mathcal{A}[2] & \subseteq & \mathcal{A}[1] \cup \{x > 1\} \\
\mathcal{A}[3] & \subseteq & (\mathcal{A}[2] \cup \{x * y\}) \backslash Expr_y \\
\mathcal{A}[4] & \subseteq & (\mathcal{A}[3] \cup \{x - 1\}) \backslash Expr_x
\end{array}
$$

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:

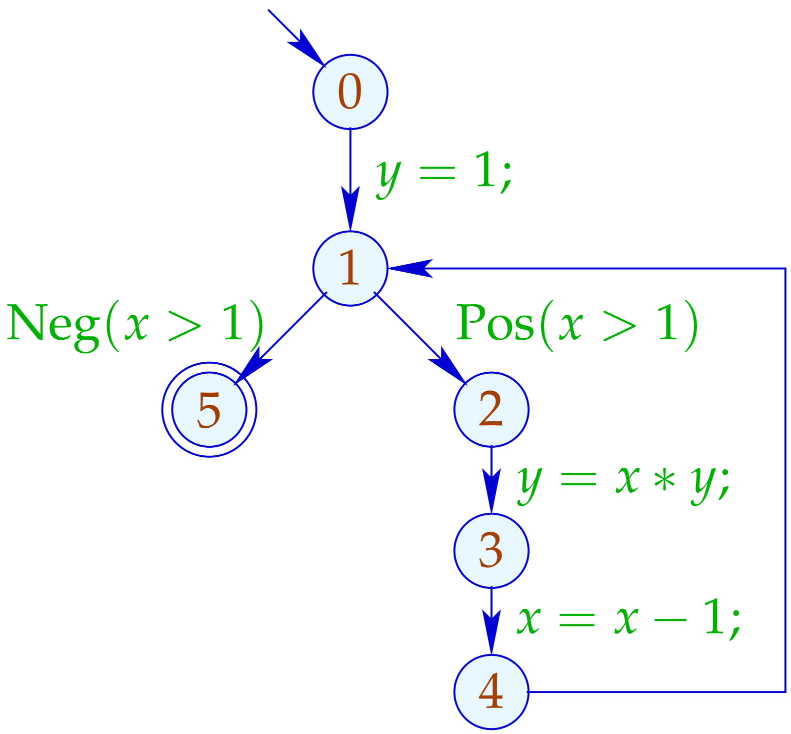


$$
\begin{array}{lll}
\mathcal{A}[0] & \subseteq & \emptyset \\
\mathcal{A}[1] & \subseteq & (\mathcal{A}[0] \cup \{1\}) \backslash Expr_y \\
\mathcal{A}[1] & \subseteq & \mathcal{A}[4] \\
\mathcal{A}[2] & \subseteq & \mathcal{A}[1] \cup \{x > 1\} \\
\mathcal{A}[3] & \subseteq & (\mathcal{A}[2] \cup \{x * y\}) \backslash Expr_y \\
\mathcal{A}[4] & \subseteq & (\mathcal{A}[3] \cup \{x - 1\}) \backslash Expr_x \\
\mathcal{A}[5] & \subseteq & \mathcal{A}[1] \cup \{x > 1\}
\end{array}
$$

Gesucht:

- möglichst große Lösung (??)
- Algorithmus, der diese berechnet :-)

Beispiel:



Lösung:

$$
\begin{array}{lll}
\mathcal{A}[0] & = & \emptyset \\
\mathcal{A}[1] & = & \{1\} \\
\mathcal{A}[2] & = & \{1, x > 1\} \\
\mathcal{A}[3] & = & \{1, x > 1\} \\
\mathcal{A}[4] & = & \{1\} \\
\mathcal{A}[5] & = & \{1, x > 1\}
\end{array}
$$

Beobachtung:

- Die möglichen Werte für $\mathcal{A}[u]$ bilden einen vollständigen Verband:

$$\mathbb{D} = 2^{Expr} \quad \text{mit} \quad B_1 \sqsubseteq B_2 \quad \text{gdw.} \quad B_1 \supseteq B_2$$

Beobachtung:

- Die möglichen Werte für $\mathcal{A}[u]$ bilden einen vollständigen Verband:

$$\mathbb{D} = 2^{Expr} \quad \text{mit} \quad B_1 \sqsubseteq B_2 \quad \text{gdw.} \quad B_1 \supseteq B_2$$

- Die Funktionen $\quad [\![k]\!]^\sharp : \mathbb{D} \to \mathbb{D} \quad$ sind monoton, d.h.

$$[\![k]\!]^\sharp(B_1) \sqsubseteq [\![k]\!]^\sharp(B_2) \quad \text{gdw.} \quad B_1 \sqsubseteq B_2$$

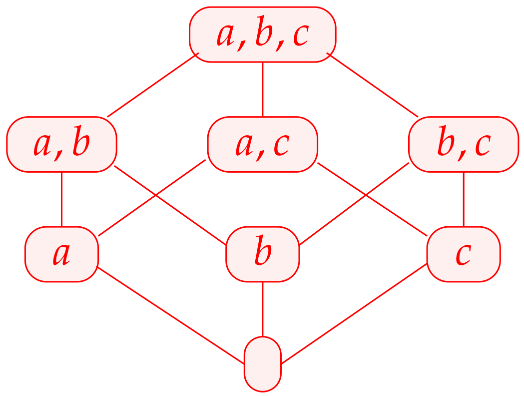
Exkurs 2: Vollständige Verbände

Eine Menge $\mathbb{D}$ mit einer Relation $\sqsubseteq \; \subseteq \; \mathbb{D} \times \mathbb{D}$ ist eine partielle Ordnung falls für alle $a, b, c \in \mathbb{D}$ gilt:

$$
\begin{array}{lll}
a \sqsubseteq a & & \textit{Reflexivität} \\
a \sqsubseteq b \land b \sqsubseteq a & \Longrightarrow \quad a = b & \textit{Anti-Symmetrie} \\
a \sqsubseteq b \land b \sqsubseteq c & \Longrightarrow \quad a \sqsubseteq c & \textit{Transitivität}
\end{array}
$$

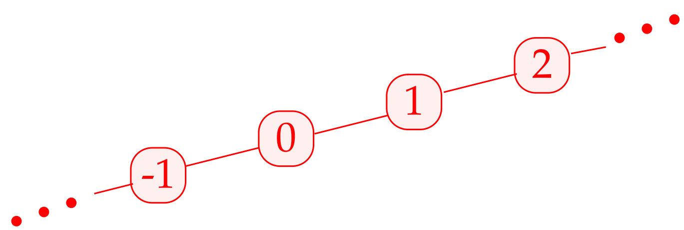
Beispiele:

1. $\mathbb{D} = 2^{\{a,b,c\}}$ mit der Relation "$\subseteq$" :

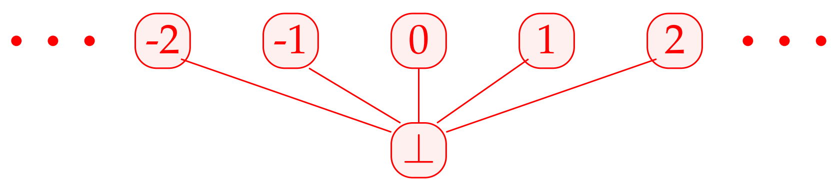


3. $\mathbb{Z}$ mit der Relation "$=$" :

3. $\mathbb{Z}$ mit der Relation "$\leq$" :



4. $\mathbb{Z}_\perp = \mathbb{Z} \cup \{\perp\}$ mit der Ordnung:



$d \in \mathbb{D}$ heißt obere Schranke für $X \subseteq \mathbb{D}$ falls

$$x \sqsubseteq d \qquad \text{für alle } x \in X$$

$d \in \mathbb{D}$ heißt obere Schranke für $X \subseteq \mathbb{D}$ falls

$$x \sqsubseteq d \qquad \text{für alle } x \in X$$

d heißt kleinste obere Schranke (lub) falls

1. d eine obere Schranke ist und
2. $d \sqsubseteq y$ für jede obere Schranke y für X.

$d \in \mathbb{D}$ heißt obere Schranke für $X \subseteq \mathbb{D}$ falls

$$x \sqsubseteq d \qquad \text{für alle } x \in X$$

d heißt kleinste obere Schranke (lub) falls

1. d eine obere Schranke ist und
2. $d \sqsubseteq y$ für jede obere Schranke y für X.

Achtung:

- $\{0, 2, 4, \ldots\} \subseteq \mathbb{Z}$ besitzt keine obere Schranke!
- $\{0, 2, 4\} \subseteq \mathbb{Z}$ besitzt die oberen Schranken $4, 5, 6, \ldots$

Ein vollständiger Verband (cl) $\mathbb{D}$ ist eine partielle Ordnung, in der jede Teilmenge $X \subseteq \mathbb{D}$ eine kleinste obere Schranke $\bigsqcup X \in \mathbb{D}$ besitzt.

Beachte:

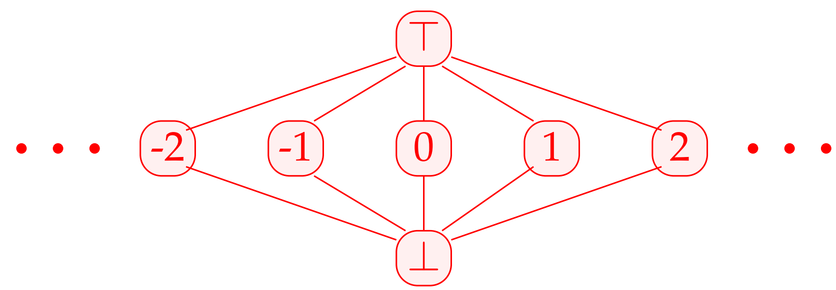
Jeder vollständige Verband besitzt

$\longrightarrow$ ein kleinstes Element $\perp = \bigsqcup \emptyset \quad \in \mathbb{D}$;

$\longrightarrow$ ein größtes Element $\top = \bigsqcup \mathbb{D} \quad \in \mathbb{D}$.

Beispiele:

1. $\mathbb{D} = 2^{\{a,b,c\}}$ ist ein cl :-)
2. $\mathbb{D} = \mathbb{Z}$ mit "$=$" ist keiner.
3. $\mathbb{D} = \mathbb{Z}$ mit "$\leq$" ebenfalls nicht.
4. $\mathbb{D} = \mathbb{Z}_\perp$ auch nicht :-(
5. Mit einem zusätzlichen Symbol $\top$ erhalten wir den flachen Verband $\mathbb{Z}_\perp^\top = \mathbb{Z} \cup \{\perp, \top\}$:



Es gilt:

Satz:

In jedem vollständigen Verband $\mathbb{D}$ besitzt jede Teilmenge $X \subseteq \mathbb{D}$ eine größte untere Schranke $\bigsqcap X$.

Es gilt:

Satz:

In jedem vollständigen Verband $\mathbb{D}$ besitzt jede Teilmenge $X \subseteq \mathbb{D}$ eine größte untere Schranke $\bigsqcap X$.

Beweis:

Konstruiere $U = \{u \in \mathbb{D} \mid \forall\, x \in X : \; u \sqsubseteq x\}$.

// die Menge der unteren Schranken von X :-)

Es gilt:

Satz:

In jedem vollständigen Verband $\mathbb{D}$ besitzt jede Teilmenge $X \subseteq \mathbb{D}$ eine größte untere Schranke $\bigsqcap X$.

Beweis:

Konstruiere $\quad U = \{u \in \mathbb{D} \mid \forall\, x \in X : \; u \sqsubseteq x\}$.

// die Menge der unteren Schranken von X :-)

Setze: $\quad g := \bigsqcup U$

Behauptung: $\quad g = \bigsqcap X$

(1) $\quad g$ ist eine untere Schranke von X :

Für $\quad x \in X \quad$ gilt:

$$u \sqsubseteq x \text{ für alle } u \in U$$

$$\Longrightarrow \quad x \text{ ist obere Schranke von } U$$

$$\Longrightarrow \quad g \sqsubseteq x \qquad \text{:-)}$$

(1) g ist eine untere Schranke von X :

Für $x \in X$ gilt:

$$u \sqsubseteq x \text{ für alle } u \in U$$

$\Longrightarrow$ x ist obere Schranke von U

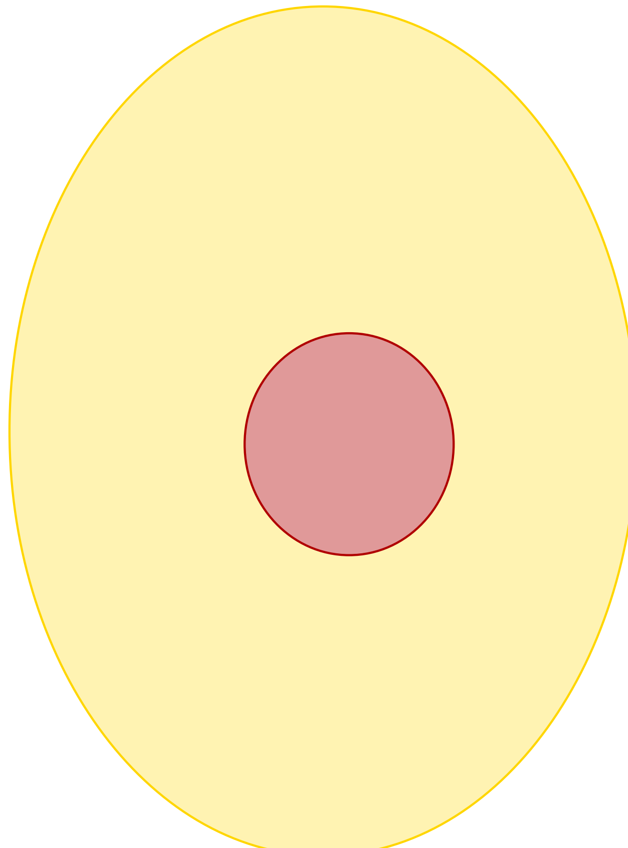
$\Longrightarrow$ $g \sqsubseteq x$:-)

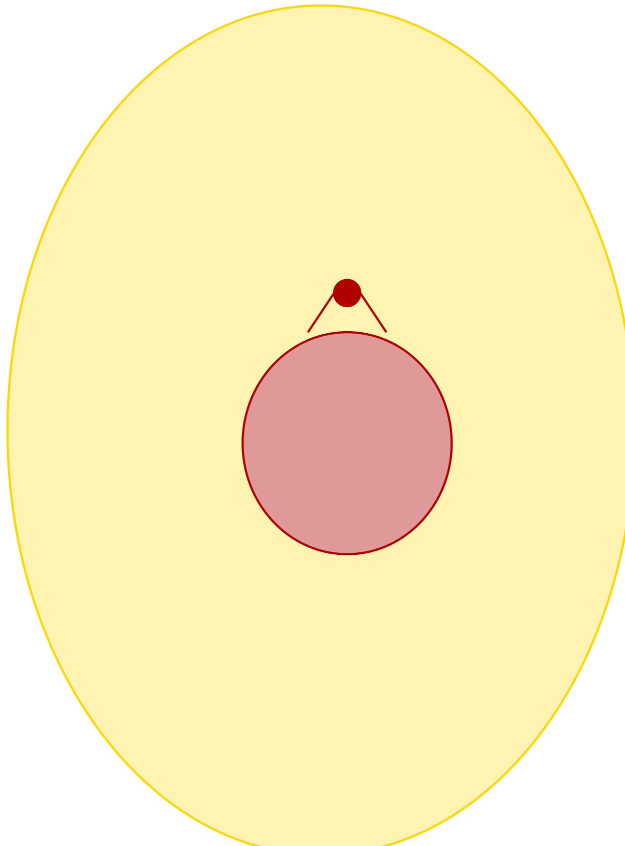
(2) g ist größte untere Schranke von X :

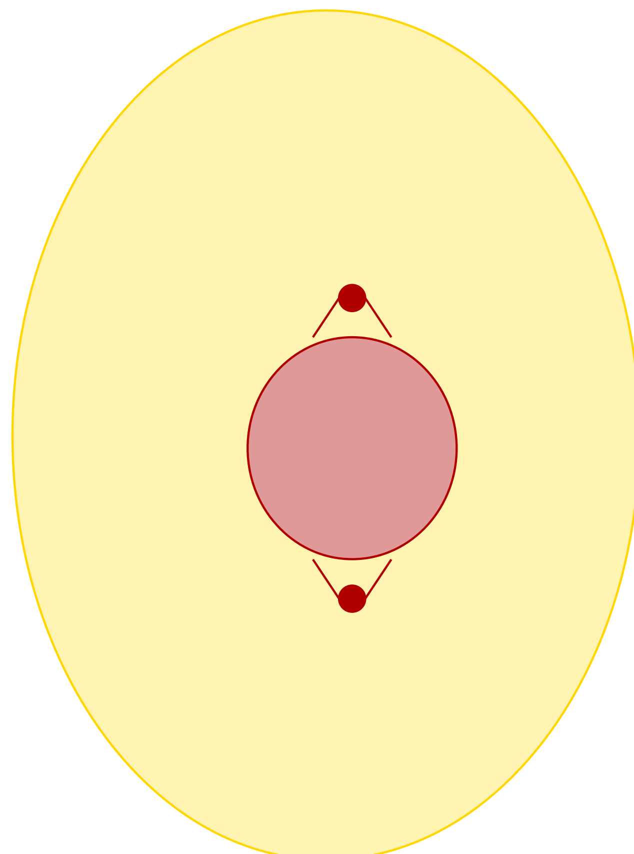
Für jede untere Schranke u von X gilt:

$$u \in U$$

$\Longrightarrow$ $u \sqsubseteq g$:-))







Wir suchen Lösungen für Ungleichungssysteme der Form:

$$x_i \quad \sqsupseteq \quad f_i(x_1, \ldots, x_n) \qquad\qquad (*)$$

Wir suchen Lösungen für Ungleichungssysteme der Form:

$$x_i \quad \sqsupseteq \quad f_i(x_1, \ldots, x_n) \qquad\qquad (*)$$

wobei:

x_i	Unbekannte	hier:	$\mathcal{A}[u]$
$\mathbb{D}$	Werte	hier:	2^{Expr}
$\sqsubseteq \quad \subseteq \quad \mathbb{D} \times \mathbb{D}$	Ordnungsrelation	hier:	$\supseteq$
$f_i\colon \mathbb{D}^n \longrightarrow \mathbb{D}$	Bedingung	hier:	...

Wir suchen Lösungen für Ungleichungssysteme der Form:

$$x_i \quad \sqsupseteq \quad f_i(x_1, \ldots, x_n) \qquad\qquad (*)$$

wobei:

x_i	Unbekannte	hier:	$\mathcal{A}[u]$
$\mathbb{D}$	Werte	hier:	2^{Expr}
$\sqsubseteq \quad \subseteq \quad \mathbb{D} \times \mathbb{D}$	Ordnungsrelation	hier:	$\supseteq$
$f_i \colon \mathbb{D}^n \longrightarrow \mathbb{D}$	Bedingung	hier:	...

Ungleichung für $\mathcal{A}[v]$:

$$\mathcal{A}[v] \quad \subseteq \quad \bigcap\{[\![k]\!]^\sharp\,(\mathcal{A}[u]) \mid k = (u, \_, v) \text{ Kante}\}$$

Wir suchen Lösungen für Ungleichungssysteme der Form:

$$x_i \quad \sqsupseteq \quad f_i(x_1, \ldots, x_n) \qquad\qquad (*)$$

wobei:

x_i	Unbekannte	hier:	$\mathcal{A}[u]$
$\mathbb{D}$	Werte	hier:	2^{Expr}
$\sqsubseteq \quad \subseteq \quad \mathbb{D} \times \mathbb{D}$	Ordnungsrelation	hier:	$\supseteq$
$f_i\colon \mathbb{D}^n \longrightarrow \mathbb{D}$	Bedingung	hier:	...

Ungleichung für $\mathcal{A}[v]$:

$$\mathcal{A}[v] \quad \subseteq \quad \bigcap\{[\![k]\!]^\sharp\,(\mathcal{A}[u]) \mid k = (u, \_, v) \;\; \text{Kante}\}$$

Denn:

$$x \sqsupseteq d_1 \wedge \ldots \wedge x \sqsupseteq d_k \quad \text{gdw.} \quad x \sqsupseteq \bigsqcup\{d_1, \ldots, d_k\} \qquad :\!\text{-})$$

Eine Abbildung $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt monoton, falls $f(a) \sqsubseteq f(b)$ für alle $a \sqsubseteq b$.

Eine Abbildung $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt monoton, falls
$f(a) \sqsubseteq f(b)$ für alle $a \sqsubseteq b$.

Beispiele:

(1) $\mathbb{D}_1 = \mathbb{D}_2 = 2^U$ für eine Menge U und $f\,x = (x \cap a) \cup b$.
Offensichtlich ist jedes solche f monoton :-)

Eine Abbildung $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt monoton, falls
$f(a) \sqsubseteq f(b)$ für alle $a \sqsubseteq b$.

Beispiele:

(1) $\mathbb{D}_1 = \mathbb{D}_2 = 2^U$ für eine Menge U und $f\,x = (x \cap a) \cup b$.
Offensichtlich ist jedes solche f monoton :-)

(2) $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{Z}$ (mit der Ordnung "$\leq$"). Dann gilt:

- $\mathsf{inc}\,x = x + 1$ ist monoton.
- $\mathsf{dec}\,x = x - 1$ ist monoton.

Eine Abbildung $f : \mathbb{D}_1 \to \mathbb{D}_2$ heißt monoton, falls $f(a) \sqsubseteq f(b)$ für alle $a \sqsubseteq b$.

Beispiele:

(1) $\mathbb{D}_1 = \mathbb{D}_2 = 2^U$ für eine Menge U und $f\,x = (x \cap a) \cup b$.
Offensichtlich ist jedes solche f monoton :-)

(2) $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{Z}$ (mit der Ordnung "$\leq$"). Dann gilt:

- $\mathsf{inc}\,x = x + 1$ ist monoton.
- $\mathsf{dec}\,x = x - 1$ ist monoton.
- $\mathsf{inv}\,x = -x$ ist nicht monoton :-)