Principles of Program Analysis:

Data Flow Analysis

Transparencies based on Chapter 2 of the book: Flemming Nielson, Hanne Riis Nielson and Chris Hankin: Principles of Program Analysis. Springer Verlag 2005.

Example Language

Syntax of While-programs

$$\begin{array}{lcl} a & ::= & x \mid n \mid a_1 \; op_a \; a_2 \\ b & ::= & \texttt{true} \mid \texttt{false} \mid \texttt{not}\; b \mid b_1 \; op_b \; b_2 \mid a_1 \; op_r \; a_2 \\ S & ::= & [x := a]^\ell \mid [\texttt{skip}]^\ell \mid S_1; S_2 \mid \\ & & \texttt{if}\; [b]^\ell \;\texttt{then}\; S_1 \;\texttt{else}\; S_2 \mid \texttt{while}\; [b]^\ell \;\texttt{do}\; S \end{array}$$

Example: $[\texttt{z:=1}]^1; \texttt{while}\; [\texttt{x>0}]^2 \;\texttt{do}\; ([\texttt{z:=z*y}]^3; [\texttt{x:=x-1}]^4)$

Abstract syntax – parentheses are inserted to disambiguate the syntax

Building an "Abstract Flowchart"

Example: $[\texttt{z:=1}]^1; \texttt{while}\ [\texttt{x>0}]^2\ \texttt{do}\ ([\texttt{z:=z*y}]^3; [\texttt{x:=x-1}]^4)$

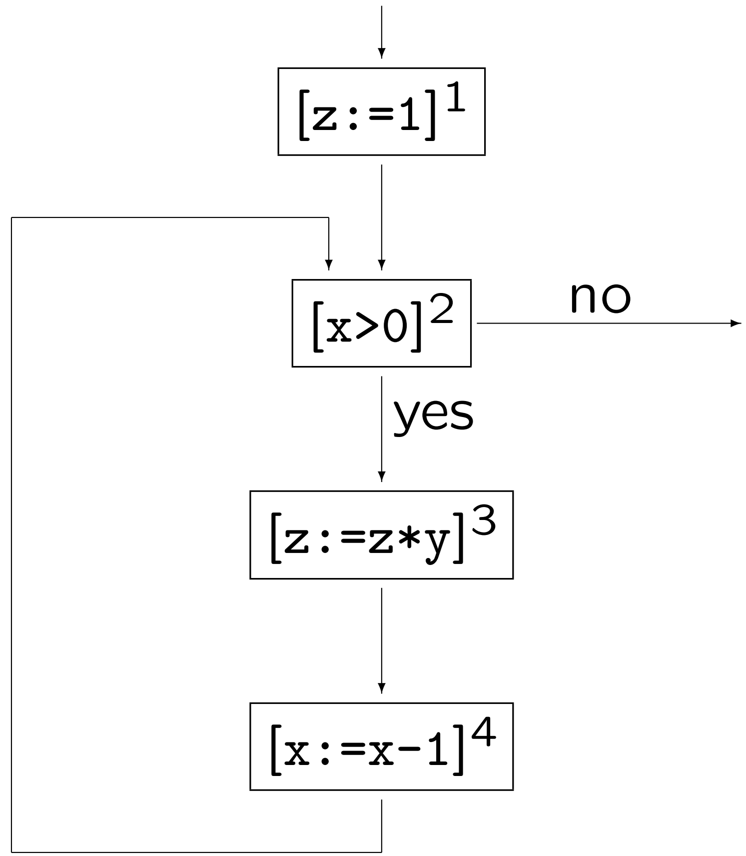
$$\mathit{init}(\cdots) = 1$$

$$\mathit{final}(\cdots) = \{2\}$$

$$\mathit{labels}(\cdots) = \{1, 2, 3, 4\}$$

$$\mathit{flow}(\cdots) = \{(1,2), (2,3), (3,4), (4,2)\}$$

$$\mathit{flow}^R(\cdots) = \{(2,1), (2,4), (3,2), (4,3)\}$$



Initial labels

$\mathit{init}(S)$ is the label of the first elementary block of S:

$$\mathit{init} : \mathbf{Stmt} \to \mathbf{Lab}$$

$$\begin{aligned}
\mathit{init}([x := a]^\ell) &= \ell \\
\mathit{init}([\mathtt{skip}]^\ell) &= \ell \\
\mathit{init}(S_1; S_2) &= \mathit{init}(S_1) \\
\mathit{init}(\mathtt{if}\ [b]^\ell\ \mathtt{then}\ S_1\ \mathtt{else}\ S_2) &= \ell \\
\mathit{init}(\mathtt{while}\ [b]^\ell\ \mathtt{do}\ S) &= \ell
\end{aligned}$$

Example:

$$\mathit{init}([\mathtt{z:=1}]^1; \mathtt{while}\ [\mathtt{x>0}]^2\ \mathtt{do}\ ([\mathtt{z:=z*y}]^3; [\mathtt{x:=x-1}]^4)) = 1$$

Final labels

final(S) is the set of labels of the last elementary blocks of S:

$$\mathit{final} : \mathbf{Stmt} \to \mathcal{P}(\mathbf{Lab})$$

$$\begin{aligned}
\mathit{final}([x := a]^\ell) &= \{\ell\} \\
\mathit{final}([\mathtt{skip}]^\ell) &= \{\ell\} \\
\mathit{final}(S_1; S_2) &= \mathit{final}(S_2) \\
\mathit{final}(\mathtt{if}\ [b]^\ell\ \mathtt{then}\ S_1\ \mathtt{else}\ S_2) &= \mathit{final}(S_1) \cup \mathit{final}(S_2) \\
\mathit{final}(\mathtt{while}\ [b]^\ell\ \mathtt{do}\ S) &= \{\ell\}
\end{aligned}$$

Example:

$$\mathit{final}([\mathtt{z:=1}]^1; \mathtt{while}\ [\mathtt{x>0}]^2\ \mathtt{do}\ ([\mathtt{z:=z*y}]^3; [\mathtt{x:=x-1}]^4)) = \{2\}$$

Labels

$\mathit{labels}(S)$ is the entire set of labels in the statement S:

$$\mathit{labels} : \mathbf{Stmt} \to \mathcal{P}(\mathbf{Lab})$$

$$\begin{aligned}
\mathit{labels}([x := a]^\ell) &= \{\ell\} \\
\mathit{labels}([\texttt{skip}]^\ell) &= \{\ell\} \\
\mathit{labels}(S_1; S_2) &= \mathit{labels}(S_1) \cup \mathit{labels}(S_2) \\
\mathit{labels}(\texttt{if } [b]^\ell \texttt{ then } S_1 \texttt{ else } S_2) &= \{\ell\} \cup \mathit{labels}(S_1) \cup \mathit{labels}(S_2) \\
\mathit{labels}(\texttt{while } [b]^\ell \texttt{ do } S) &= \{\ell\} \cup \mathit{labels}(S)
\end{aligned}$$

Example

$$\mathit{labels}([\texttt{z:=1}]^1; \texttt{while } [\texttt{x>0}]^2 \texttt{ do } ([\texttt{z:=z*y}]^3; [\texttt{x:=x-1}]^4)) = \{1, 2, 3, 4\}$$

Flows and reverse flows

$\mathit{flow}(S)$ and $\mathit{flow}^R(S)$ are representations of how control flows in S:

$$\mathit{flow}, \mathit{flow}^R : \mathbf{Stmt} \to \mathcal{P}(\mathbf{Lab} \times \mathbf{Lab})$$

$$\begin{aligned}
\mathit{flow}([x := a]^\ell) &= \emptyset \\
\mathit{flow}([\texttt{skip}]^\ell) &= \emptyset \\
\mathit{flow}(S_1; S_2) &= \mathit{flow}(S_1) \cup \mathit{flow}(S_2) \\
&\quad \cup \{(\ell, \mathit{init}(S_2)) \mid \ell \in \mathit{final}(S_1)\} \\
\mathit{flow}(\texttt{if } [b]^\ell \texttt{ then } S_1 \texttt{ else } S_2) &= \mathit{flow}(S_1) \cup \mathit{flow}(S_2) \\
&\quad \cup \{(\ell, \mathit{init}(S_1)), (\ell, \mathit{init}(S_2))\} \\
\mathit{flow}(\texttt{while } [b]^\ell \texttt{ do } S) &= \mathit{flow}(S) \cup \{(\ell, \mathit{init}(S))\} \\
&\quad \cup \{(\ell', \ell) \mid \ell' \in \mathit{final}(S)\}
\end{aligned}$$

$$\mathit{flow}^R(S) = \{(\ell, \ell') \mid (\ell', \ell) \in \mathit{flow}(S)\}$$

Elementary blocks

A statement consists of a set of *elementary blocks*

$$\mathit{blocks} : \mathbf{Stmt} \rightarrow \mathcal{P}(\mathbf{Blocks})$$

$$\begin{aligned}
\mathit{blocks}([\mathtt{x} := a]^\ell) &= \{[\mathtt{x} := a]^\ell\} \\
\mathit{blocks}([\mathtt{skip}]^\ell) &= \{[\mathtt{skip}]^\ell\} \\
\mathit{blocks}(S_1; S_2) &= \mathit{blocks}(S_1) \cup \mathit{blocks}(S_2) \\
\mathit{blocks}(\mathtt{if}\ [b]^\ell\ \mathtt{then}\ S_1\ \mathtt{else}\ S_2) &= \{[b]^\ell\} \cup \mathit{blocks}(S_1) \cup \mathit{blocks}(S_2) \\
\mathit{blocks}(\mathtt{while}\ [b]^\ell\ \mathtt{do}\ S) &= \{[b]^\ell\} \cup \mathit{blocks}(S)
\end{aligned}$$

A statement S is *label consistent* if and only if any two elementary statements $[S_1]^\ell$ and $[S_2]^\ell$ with the same label in S are equal: $S_1 = S_2$

A statement *where all labels are unique* is automatically label consistent

Intraprocedural Analysis

Classical analyses:

- Available Expressions Analysis
- Reaching Definitions Analysis
- Very Busy Expressions Analysis
- Live Variables Analysis

Derived analysis:

- Use-Definition and Definition-Use Analysis

Available Expressions Analysis

The aim of the *Available Expressions Analysis* is to determine

> For each program point, which expressions must have already been computed, and not later modified, on all paths to the program point.

Example:

point of interest ⇓

$[\texttt{x:=a+b}]^1; [\texttt{y:=a*b}]^2; \texttt{while } [\texttt{y>a+b}]^3 \texttt{ do } ([\texttt{a:=a+1}]^4; [\texttt{x:=a+b}]^5)$

The analysis enables a transformation into

$[\texttt{x:= a+b}]^1; [\texttt{y:=a*b}]^2; \texttt{while } [\texttt{y>x}]^3 \texttt{ do } ([\texttt{a:=a+1}]^4; [\texttt{x:= a+b}]^5)$

Available Expressions Analysis – the basic idea

X_1 $\qquad$ X_2

$N = X_1 \cap X_2$

$x := a$

$$X = (N \setminus \underbrace{\{\text{expressions with an } x\}}_{\text{kill}}) \cup \underbrace{\{\text{subexpressions of } a \text{ without an } x\}}_{\text{gen}}$$

Available Expressions Analysis

kill and *gen* functions

$$
\begin{aligned}
\mathit{kill}_{\mathsf{AE}}([x := a]^\ell) &= \{a' \in \mathbf{AExp}_\star \mid x \in \mathit{FV}(a')\} \\
\mathit{kill}_{\mathsf{AE}}([\texttt{skip}]^\ell) &= \emptyset \\
\mathit{kill}_{\mathsf{AE}}([b]^\ell) &= \emptyset
\end{aligned}
$$

$$
\begin{aligned}
\mathit{gen}_{\mathsf{AE}}([x := a]^\ell) &= \{a' \in \mathbf{AExp}(a) \mid x \notin \mathit{FV}(a')\} \\
\mathit{gen}_{\mathsf{AE}}([\texttt{skip}]^\ell) &= \emptyset \\
\mathit{gen}_{\mathsf{AE}}([b]^\ell) &= \mathbf{AExp}(b)
\end{aligned}
$$

data flow equations: $\mathsf{AE}^{=}$

$$
\mathsf{AE}_{entry}(\ell) = \begin{cases} \emptyset & \text{if } \ell = \mathit{init}(S_\star) \\ \bigcap\{\mathsf{AE}_{exit}(\ell') \mid (\ell', \ell) \in \mathit{flow}(S_\star)\} & \text{otherwise} \end{cases}
$$

$$
\mathsf{AE}_{exit}(\ell) = (\mathsf{AE}_{entry}(\ell) \backslash \mathit{kill}_{\mathsf{AE}}(B^\ell)) \cup \mathit{gen}_{\mathsf{AE}}(B^\ell)
$$

where $B^\ell \in \mathit{blocks}(S_\star)$

Example:

$[\texttt{x:=a+b}]^1; [\texttt{y:=a*b}]^2; \texttt{while } [\texttt{y>a+b}]^3 \texttt{ do } ([\texttt{a:=a+1}]^4; [\texttt{x:=a+b}]^5)$

kill and *gen* functions:

ℓ	$kill_{AE}(\ell)$	$gen_{AE}(\ell)$
1	$\emptyset$	{a+b}
2	$\emptyset$	{a*b}
3	$\emptyset$	{a+b}
4	{a+b, a*b, a+1}	$\emptyset$
5	$\emptyset$	{a+b}

Example (cont.):

$$[\texttt{x:=a+b}]^1; [\texttt{y:=a*b}]^2; \texttt{while } [\texttt{y>a+b}]^3 \texttt{ do } ([\texttt{a:=a+1}]^4; [\texttt{x:=a+b}]^5)$$

Equations:

$$
\begin{aligned}
\mathsf{AE}_{entry}(1) &= \emptyset \\
\mathsf{AE}_{entry}(2) &= \mathsf{AE}_{exit}(1) \\
\mathsf{AE}_{entry}(3) &= \mathsf{AE}_{exit}(2) \cap \mathsf{AE}_{exit}(5) \\
\mathsf{AE}_{entry}(4) &= \mathsf{AE}_{exit}(3) \\
\mathsf{AE}_{entry}(5) &= \mathsf{AE}_{exit}(4)
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{AE}_{exit}(1) &= \mathsf{AE}_{entry}(1) \cup \{\texttt{a+b}\} \\
\mathsf{AE}_{exit}(2) &= \mathsf{AE}_{entry}(2) \cup \{\texttt{a*b}\} \\
\mathsf{AE}_{exit}(3) &= \mathsf{AE}_{entry}(3) \cup \{\texttt{a+b}\} \\
\mathsf{AE}_{exit}(4) &= \mathsf{AE}_{entry}(4) \backslash \{\texttt{a+b}, \texttt{a*b}, \texttt{a+1}\} \\
\mathsf{AE}_{exit}(5) &= \mathsf{AE}_{entry}(5) \cup \{\texttt{a+b}\}
\end{aligned}
$$

Example (cont.):

$[\texttt{x:=a+b}]^1; [\texttt{y:=a*b}]^2; \texttt{while } [\texttt{y>a+b}]^3 \texttt{ do } ([\texttt{a:=a+1}]^4; [\texttt{x:=a+b}]^5)$

Largest solution:

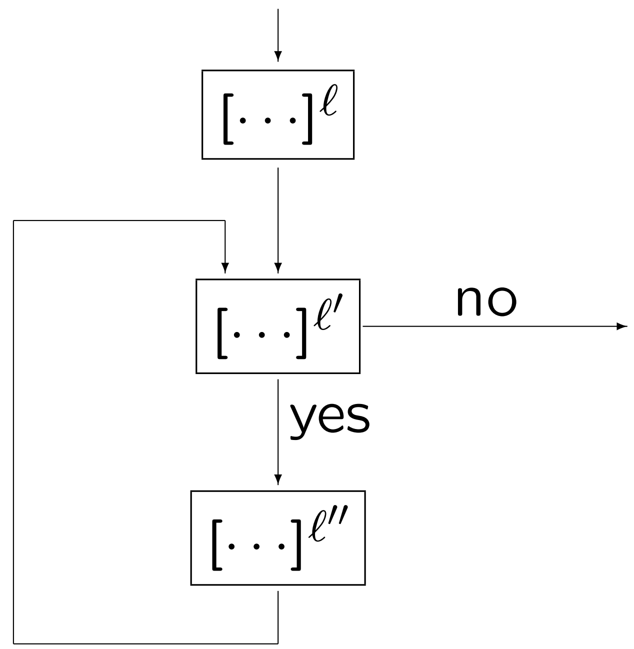
ℓ	$\mathsf{AE}_{entry}(\ell)$	$\mathsf{AE}_{exit}(\ell)$
1	$\emptyset$	{a+b}
2	{a+b}	{a+b, a*b}
3	{a+b}	{a+b}
4	{a+b}	$\emptyset$
5	$\emptyset$	{a+b}

Why largest solution?

$$[\texttt{z:=x+y}]^{\ell};\texttt{while}\ [\texttt{true}]^{\ell'}\ \texttt{do}\ [\texttt{skip}]^{\ell''}$$

Equations:

$$\begin{aligned}
\mathsf{AE}_{entry}(\ell) &= \emptyset \\
\mathsf{AE}_{entry}(\ell') &= \mathsf{AE}_{exit}(\ell) \ \cap\ \mathsf{AE}_{exit}(\ell'') \\
\mathsf{AE}_{entry}(\ell'') &= \mathsf{AE}_{exit}(\ell') \\
\mathsf{AE}_{exit}(\ell) &= \mathsf{AE}_{entry}(\ell) \cup \{\texttt{x+y}\} \\
\mathsf{AE}_{exit}(\ell') &= \mathsf{AE}_{entry}(\ell') \\
\mathsf{AE}_{exit}(\ell'') &= \mathsf{AE}_{entry}(\ell'')
\end{aligned}$$



After some simplification: $\mathsf{AE}_{entry}(\ell') = \{\texttt{x+y}\} \cap \mathsf{AE}_{entry}(\ell')$

Two solutions to this equation: $\{\texttt{x+y}\}$ and $\emptyset$

Reaching Definitions Analysis

The aim of the *Reaching Definitions Analysis* is to determine

> For each program point, which assignments may have been made and not overwritten, when program execution reaches this point along some path.

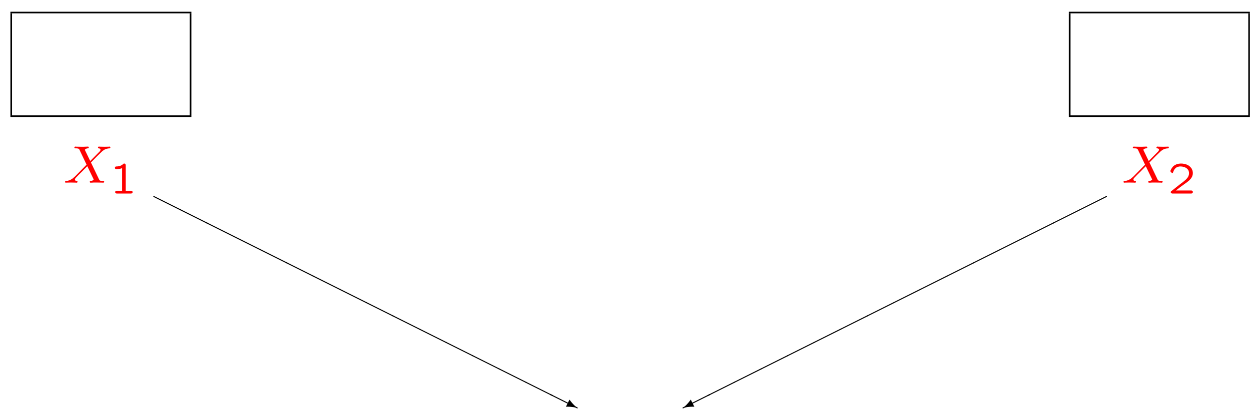
Example:

point of interest $\Downarrow$

$[\texttt{x:=5}]^1; [\texttt{y:=1}]^2; \texttt{while } [\texttt{x>1}]^3 \texttt{ do } ([\texttt{y:=x*y}]^4; [\texttt{x:=x-1}]^5)$

useful for definition-use chains and use-definition chains

Reaching Definitions Analysis – the basic idea



$$N = X_1 \cup X_2$$

$$[x := a]^\ell$$

$$X = (N \setminus \overbrace{\{(x,?),(x,1),\cdots\}}^{\text{kill}}) \cup \underbrace{\{(x,\ell)\}}_{\text{gen}}$$

Reaching Definitions Analysis

kill and *gen* functions

$$
\begin{aligned}
\mathit{kill}_{\mathsf{RD}}([x := a]^\ell) &= \{(x, ?)\} \\
&\quad \cup \{(x, \ell') \mid B^{\ell'} \text{ is an assignment to } x \text{ in } S_\star\} \\
\mathit{kill}_{\mathsf{RD}}([\mathtt{skip}]^\ell) &= \emptyset \\
\mathit{kill}_{\mathsf{RD}}([b]^\ell) &= \emptyset \\
\\
\mathit{gen}_{\mathsf{RD}}([x := a]^\ell) &= \{(x, \ell)\} \\
\mathit{gen}_{\mathsf{RD}}([\mathtt{skip}]^\ell) &= \emptyset \\
\mathit{gen}_{\mathsf{RD}}([b]^\ell) &= \emptyset
\end{aligned}
$$

data flow equations: $\mathsf{RD}^{=}$

$$
\mathsf{RD}_{entry}(\ell) = \begin{cases} \{(x, ?) \mid x \in \mathit{FV}(S_\star)\} & \text{if } \ell = \mathit{init}(S_\star) \\ \bigcup\{\mathsf{RD}_{exit}(\ell') \mid (\ell', \ell) \in \mathit{flow}(S_\star)\} & \text{otherwise} \end{cases}
$$

$$
\begin{aligned}
\mathsf{RD}_{exit}(\ell) &= (\mathsf{RD}_{entry}(\ell) \backslash \mathit{kill}_{\mathsf{RD}}(B^\ell)) \cup \mathit{gen}_{\mathsf{RD}}(B^\ell) \\
&\quad \text{where } B^\ell \in \mathit{blocks}(S_\star)
\end{aligned}
$$

Example:

$$[\texttt{x:=5}]^1; [\texttt{y:=1}]^2; \texttt{while } [\texttt{x>1}]^3 \texttt{ do } ([\texttt{y:=x*y}]^4; [\texttt{x:=x-1}]^5)$$

kill and *gen* functions:

ℓ	$kill_{RD}(\ell)$	$gen_{RD}(\ell)$
1	$\{(x,?),(x,1),(x,5)\}$	$\{(x,1)\}$
2	$\{(y,?),(y,2),(y,4)\}$	$\{(y,2)\}$
3	$\emptyset$	$\emptyset$
4	$\{(y,?),(y,2),(y,4)\}$	$\{(y,4)\}$
5	$\{(x,?),(x,1),(x,5)\}$	$\{(x,5)\}$

Example (cont.):

$$[\texttt{x:=5}]^1; [\texttt{y:=1}]^2; \texttt{while } [\texttt{x>1}]^3 \texttt{ do } ([\texttt{y:=x*y}]^4; [\texttt{x:=x-1}]^5)$$

Equations:

$$\begin{aligned}
RD_{entry}(1) &= \{(x,?),(y,?)\} \\
RD_{entry}(2) &= RD_{exit}(1) \\
RD_{entry}(3) &= RD_{exit}(2) \cup RD_{exit}(5) \\
RD_{entry}(4) &= RD_{exit}(3) \\
RD_{entry}(5) &= RD_{exit}(4)
\end{aligned}$$

$$\begin{aligned}
RD_{exit}(1) &= (RD_{entry}(1) \backslash \{(x,?),(x,1),(x,5)\}) \cup \{(x,1)\} \\
RD_{exit}(2) &= (RD_{entry}(2) \backslash \{(y,?),(y,2),(y,4)\}) \cup \{(y,2)\} \\
RD_{exit}(3) &= RD_{entry}(3) \\
RD_{exit}(4) &= (RD_{entry}(4) \backslash \{(y,?),(y,2),(y,4)\}) \cup \{(y,4)\} \\
RD_{exit}(5) &= (RD_{entry}(5) \backslash \{(x,?),(x,1),(x,5)\}) \cup \{(x,5)\}
\end{aligned}$$

Example (cont.):

$[\mathtt{x:=5}]^1; [\mathtt{y:=1}]^2; \mathtt{while}\ [\mathtt{x>1}]^3\ \mathtt{do}\ ([\mathtt{y:=x*y}]^4; [\mathtt{x:=x-1}]^5)$

Smallest solution:

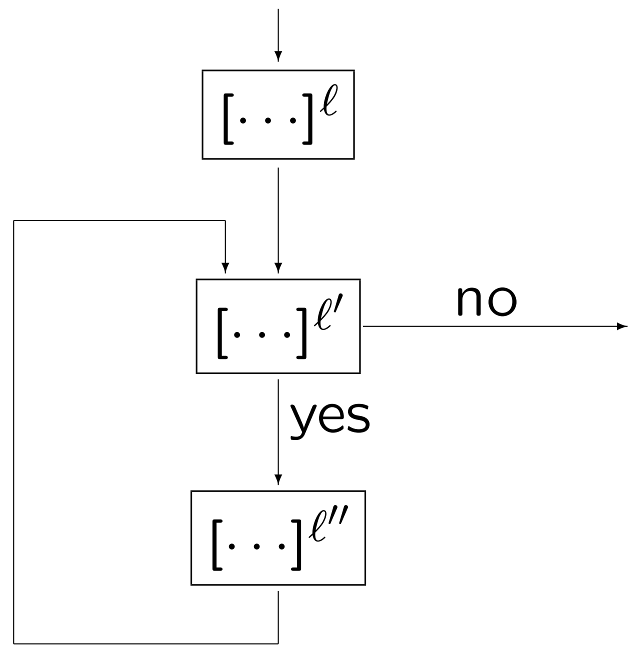
ℓ	$\mathsf{RD}_{entry}(\ell)$	$\mathsf{RD}_{exit}(\ell)$
1	$\{(x,?),(y,?)\}$	$\{(y,?),(x,1)\}$
2	$\{(y,?),(x,1)\}$	$\{(x,1),(y,2)\}$
3	$\{(x,1),(y,2),(y,4),(x,5)\}$	$\{(x,1),(y,2),(y,4),(x,5)\}$
4	$\{(x,1),(y,2),(y,4),(x,5)\}$	$\{(x,1),(y,4),(x,5)\}$
5	$\{(x,1),(y,4),(x,5)\}$	$\{(y,4),(x,5)\}$

Why smallest solution?

$$[\mathtt{z{:=}x{+}y}]^{\ell};\mathtt{while}\ [\mathtt{true}]^{\ell'}\ \mathtt{do}\ [\mathtt{skip}]^{\ell''}$$

Equations:

$$\begin{aligned}
\mathsf{RD}_{entry}(\ell) &= \{(\mathtt{x},?),(\mathtt{y},?),(\mathtt{z},?)\}\\
\mathsf{RD}_{entry}(\ell') &= \mathsf{RD}_{exit}(\ell)\cup\mathsf{RD}_{exit}(\ell'')\\
\mathsf{RD}_{entry}(\ell'') &= \mathsf{RD}_{exit}(\ell')\\
\mathsf{RD}_{exit}(\ell) &= (\mathsf{RD}_{entry}(\ell)\setminus\{(\mathtt{z},?)\})\cup\{(\mathtt{z},\ell)\}\\
\mathsf{RD}_{exit}(\ell') &= \mathsf{RD}_{entry}(\ell')\\
\mathsf{RD}_{exit}(\ell'') &= \mathsf{RD}_{entry}(\ell'')
\end{aligned}$$



After some simplification: $\mathsf{RD}_{entry}(\ell') = \{(\mathtt{x},?),(\mathtt{y},?),(\mathtt{z},\ell)\}\cup\mathsf{RD}_{entry}(\ell')$

Many solutions to this equation: any superset of $\{(\mathtt{x},?),(\mathtt{y},?),(\mathtt{z},\ell)\}$

Very Busy Expressions Analysis

An expression is *very busy* at the exit from a label if, no matter what path is taken from the label, the expression is always used before any of the variables occurring in it are redefined.

The aim of the *Very Busy Expressions Analysis* is to determine

> For each program point, which expressions must be very busy at the exit from the point.

Example:

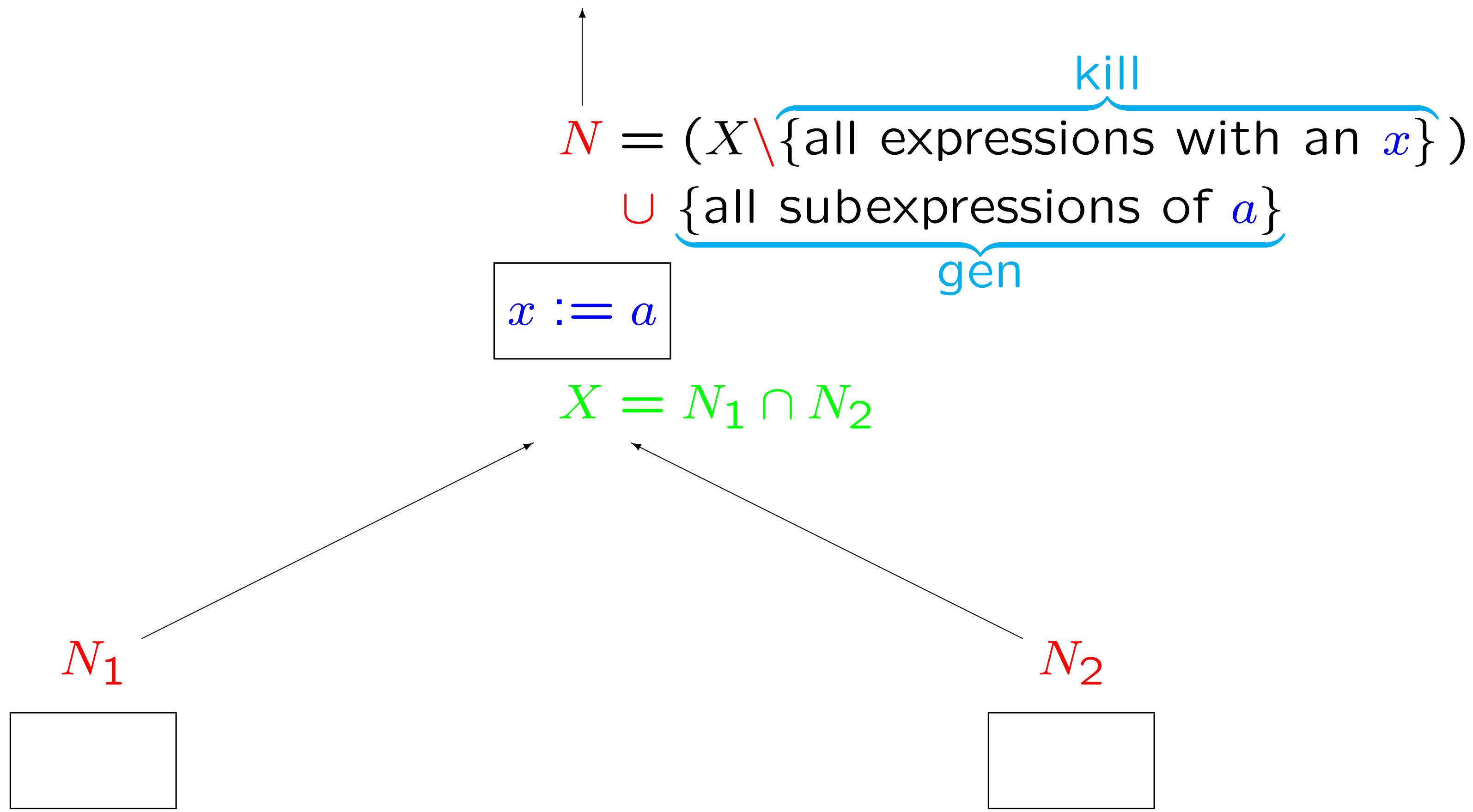
point of interest

⇓ `if` $[\texttt{a>b}]^1$ `then` $([\texttt{x:=b-a}]^2; [\texttt{y:=a-b}]^3)$ `else` $([\texttt{y:=b-a}]^4; [\texttt{x:=a-b}]^5)$

The analysis enables a transformation into

$[\texttt{t1:=b-a}]^A; [\texttt{t2:=b-a}]^B;$
`if` $[\texttt{a>b}]^1$ `then` $([\texttt{x:=t1}]^2; [\texttt{y:=t2}]^3)$ `else` $([\texttt{y:=t1}]^4; [\texttt{x:=t2}]^5)$

Very Busy Expressions Analysis – the basic idea



Very Busy Expressions Analysis

kill and *gen* functions

$$
\begin{aligned}
\mathit{kill}_{\mathsf{VB}}([x := a]^\ell) &= \{a' \in \mathbf{AExp}_\star \mid x \in \mathit{FV}(a')\} \\
\mathit{kill}_{\mathsf{VB}}([\mathtt{skip}]^\ell) &= \emptyset \\
\mathit{kill}_{\mathsf{VB}}([b]^\ell) &= \emptyset \\
\\
\mathit{gen}_{\mathsf{VB}}([\mathtt{x} := a]^\ell) &= \mathbf{AExp}(a) \\
\mathit{gen}_{\mathsf{VB}}([\mathtt{skip}]^\ell) &= \emptyset \\
\mathit{gen}_{\mathsf{VB}}([b]^\ell) &= \mathbf{AExp}(b)
\end{aligned}
$$

data flow equations: VB$^{=}$

$$
\mathsf{VB}_{exit}(\ell) = \begin{cases} \emptyset & \text{if } \ell \in \mathit{final}(S_\star) \\ \bigcap\{\mathsf{VB}_{entry}(\ell') \mid (\ell', \ell) \in \mathit{flow}^R(S_\star)\} & \text{otherwise} \end{cases}
$$

$$
\mathsf{VB}_{entry}(\ell) = (\mathsf{VB}_{exit}(\ell) \backslash \mathit{kill}_{\mathsf{VB}}(B^\ell)) \cup \mathit{gen}_{\mathsf{VB}}(B^\ell)
$$

where $B^\ell \in \mathit{blocks}(S_\star)$

Example:

$\texttt{if [a>b]}^1 \texttt{ then ([x:=b-a]}^2\texttt{; [y:=a-b]}^3\texttt{) else ([y:=b-a]}^4\texttt{; [x:=a-b]}^5\texttt{)}$

kill and *gen* function:

ℓ	$kill_{\mathsf{VB}}(\ell)$	$gen_{\mathsf{VB}}(\ell)$
1	$\emptyset$	$\emptyset$
2	$\emptyset$	{b-a}
3	$\emptyset$	{a-b}
4	$\emptyset$	{b-a}
5	$\emptyset$	{a-b}

Example (cont.):

$\texttt{if [a>b]}^1\ \texttt{then ([x:=b-a]}^2\texttt{; [y:=a-b]}^3\texttt{) else ([y:=b-a]}^4\texttt{; [x:=a-b]}^5\texttt{)}$

Equations:

$$\begin{aligned}
\mathsf{VB}_{entry}(1) &= \mathsf{VB}_{exit}(1)\\
\mathsf{VB}_{entry}(2) &= \mathsf{VB}_{exit}(2) \cup \{\texttt{b-a}\}\\
\mathsf{VB}_{entry}(3) &= \{\texttt{a-b}\}\\
\mathsf{VB}_{entry}(4) &= \mathsf{VB}_{exit}(4) \cup \{\texttt{b-a}\}\\
\mathsf{VB}_{entry}(5) &= \{\texttt{a-b}\}
\end{aligned}$$

$$\begin{aligned}
\mathsf{VB}_{exit}(1) &= \mathsf{VB}_{entry}(2) \cap \mathsf{VB}_{entry}(4)\\
\mathsf{VB}_{exit}(2) &= \mathsf{VB}_{entry}(3)\\
\mathsf{VB}_{exit}(3) &= \emptyset\\
\mathsf{VB}_{exit}(4) &= \mathsf{VB}_{entry}(5)\\
\mathsf{VB}_{exit}(5) &= \emptyset
\end{aligned}$$

Example (cont.):

`if [a>b]`^{1} `then ([x:=b-a]`^{2}`; [y:=a-b]`^{3}`) else ([y:=b-a]`^{4}`; [x:=a-b]`^{5}`)`

Largest solution:

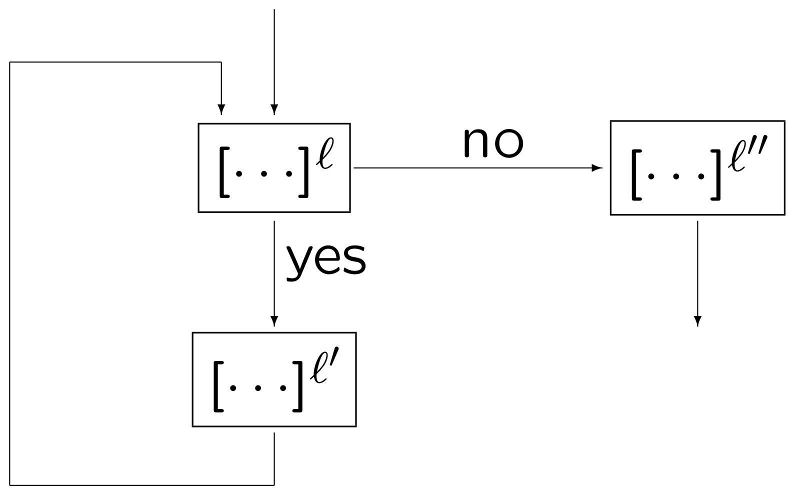
ℓ	$\mathsf{VB}_{entry}(\ell)$	$\mathsf{VB}_{exit}(\ell)$
1	{a-b, b-a}	{a-b, b-a}
2	{a-b, b-a}	{a-b}
3	{a-b}	$\emptyset$
4	{a-b, b-a}	{a-b}
5	{a-b}	$\emptyset$

Why largest solution?

$$(\texttt{while } [\texttt{x>1}]^{\ell} \texttt{ do } [\texttt{skip}]^{\ell'}); [\texttt{x:=x+1}]^{\ell''}$$

Equations:

$$\begin{aligned}
\mathsf{VB}_{entry}(\ell) &= \mathsf{VB}_{exit}(\ell)\\
\mathsf{VB}_{entry}(\ell') &= \mathsf{VB}_{exit}(\ell')\\
\mathsf{VB}_{entry}(\ell'') &= \{\texttt{x+1}\}\\
\mathsf{VB}_{exit}(\ell) &= \mathsf{VB}_{entry}(\ell') \cap \mathsf{VB}_{entry}(\ell'')\\
\mathsf{VB}_{exit}(\ell') &= \mathsf{VB}_{entry}(\ell)\\
\mathsf{VB}_{exit}(\ell'') &= \emptyset
\end{aligned}$$



After some simplifications: $\mathsf{VB}_{exit}(\ell) = \mathsf{VB}_{exit}(\ell) \cap \{\texttt{x+1}\}$

Two solutions to this equation: $\{\texttt{x+1}\}$ and $\emptyset$

Live Variables Analysis

A variable is *live* at the exit from a label if there is a path from the label to a use of the variable that does not re-define the variable.

The aim of the *Live Variables Analysis* is to determine

> For each program point, which variables may be live at the exit from the point.

Example:

point of interest

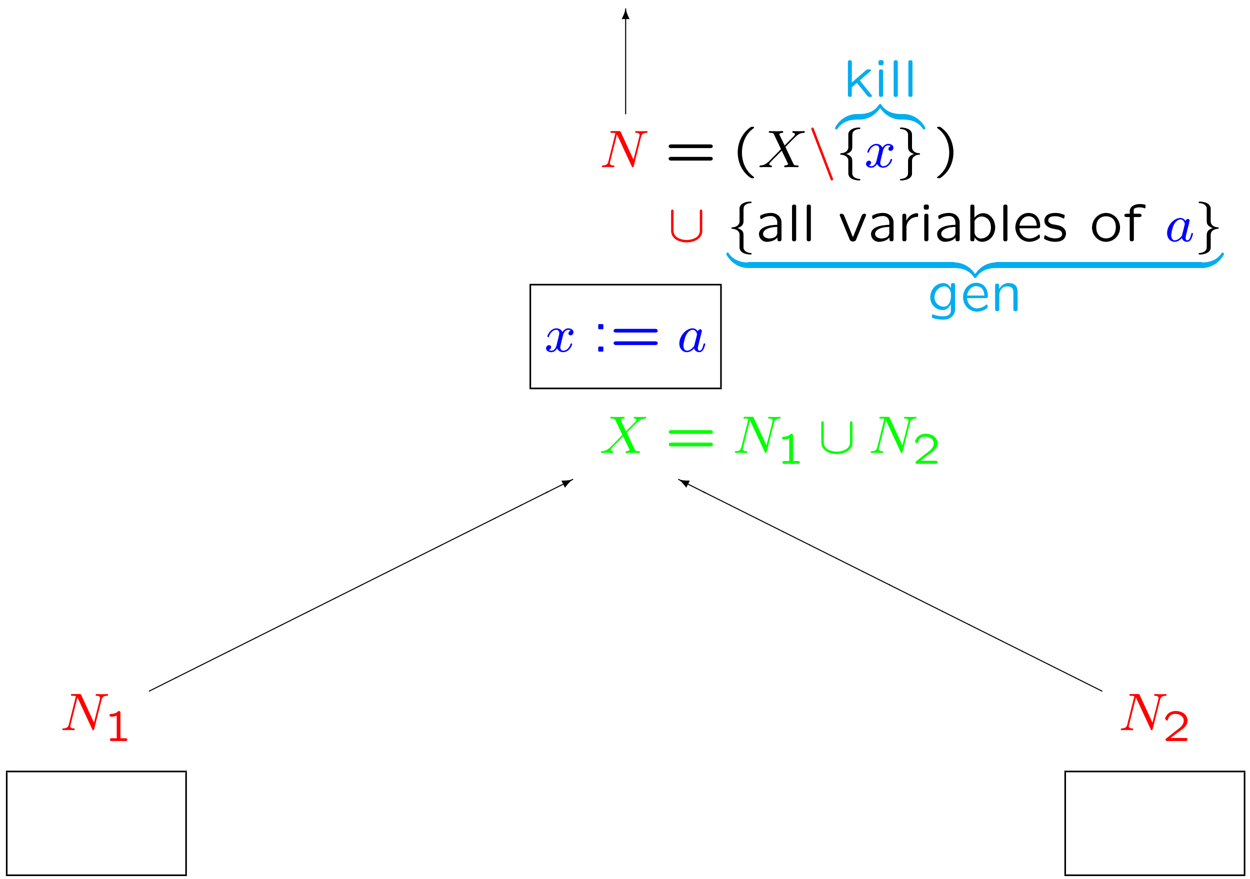
$\Downarrow$

$[\texttt{x}:=2]^1; [\texttt{y}:=4]^2; [\texttt{x}:=1]^3; (\texttt{if } [\texttt{y}>\texttt{x}]^4 \texttt{ then } [\texttt{z}:=\texttt{y}]^5 \texttt{ else } [\texttt{z}:=\texttt{y}*\texttt{y}]^6); [\texttt{x}:=\texttt{z}]^7$

The analysis enables a transformation into

$$[\texttt{y}:=4]^2; [\texttt{x}:=1]^3; (\texttt{if } [\texttt{y}>\texttt{x}]^4 \texttt{ then } [\texttt{z}:=\texttt{y}]^5 \texttt{ else } [\texttt{z}:=\texttt{y}*\texttt{y}]^6); [\texttt{x}:=\texttt{z}]^7$$

Live Variables Analysis – the basic idea



Live Variables Analysis

kill and *gen* functions

$$
\begin{aligned}
\mathit{kill}_{\mathsf{LV}}([x := a]^\ell) &= \{x\} \\
\mathit{kill}_{\mathsf{LV}}([\mathtt{skip}]^\ell) &= \emptyset \\
\mathit{kill}_{\mathsf{LV}}([b]^\ell) &= \emptyset
\end{aligned}
$$

$$
\begin{aligned}
\mathit{gen}_{\mathsf{LV}}([x := a]^\ell) &= \mathit{FV}(a) \\
\mathit{gen}_{\mathsf{LV}}([\mathtt{skip}]^\ell) &= \emptyset \\
\mathit{gen}_{\mathsf{LV}}([b]^\ell) &= \mathit{FV}(b)
\end{aligned}
$$

data flow equations: $\mathsf{LV}^{=}$

$$
\mathsf{LV}_{exit}(\ell) = \begin{cases} \emptyset & \text{if } \ell \in \mathit{final}(S_\star) \\ \bigcup\{\mathsf{LV}_{entry}(\ell') \mid (\ell', \ell) \in \mathit{flow}^R(S_\star)\} & \text{otherwise} \end{cases}
$$

$$
\mathsf{LV}_{entry}(\ell) = (\mathsf{LV}_{exit}(\ell) \backslash \mathit{kill}_{\mathsf{LV}}(B^\ell)) \cup \mathit{gen}_{\mathsf{LV}}(B^\ell)
$$

where $B^\ell \in \mathit{blocks}(S_\star)$

Example:

$[\texttt{x:=2}]^1; [\texttt{y:=4}]^2; [\texttt{x:=1}]^3; (\texttt{if } [\texttt{y>x}]^4 \texttt{ then } [\texttt{z:=y}]^5 \texttt{ else } [\texttt{z:=y*y}]^6); [\texttt{x:=z}]^7$

kill and *gen* functions:

ℓ	$kill_{LV}(\ell)$	$gen_{LV}(\ell)$
1	$\{\texttt{x}\}$	$\emptyset$
2	$\{\texttt{y}\}$	$\emptyset$
3	$\{\texttt{x}\}$	$\emptyset$
4	$\emptyset$	$\{\texttt{x},\texttt{y}\}$
5	$\{\texttt{z}\}$	$\{\texttt{y}\}$
6	$\{\texttt{z}\}$	$\{\texttt{y}\}$
7	$\{\texttt{x}\}$	$\{\texttt{z}\}$

Example (cont.):

$[\texttt{x:=2}]^1; [\texttt{y:=4}]^2; [\texttt{x:=1}]^3; (\texttt{if } [\texttt{y>x}]^4 \texttt{ then } [\texttt{z:=y}]^5 \texttt{ else } [\texttt{z:=y*y}]^6); [\texttt{x:=z}]^7$

Equations:

$$
\begin{array}{rclrcl}
\mathsf{LV}_{entry}(1) & = & \mathsf{LV}_{exit}(1)\backslash\{\texttt{x}\} & \mathsf{LV}_{exit}(1) & = & \mathsf{LV}_{entry}(2) \\
\mathsf{LV}_{entry}(2) & = & \mathsf{LV}_{exit}(2)\backslash\{\texttt{y}\} & \mathsf{LV}_{exit}(2) & = & \mathsf{LV}_{entry}(3) \\
\mathsf{LV}_{entry}(3) & = & \mathsf{LV}_{exit}(3)\backslash\{\texttt{x}\} & \mathsf{LV}_{exit}(3) & = & \mathsf{LV}_{entry}(4) \\
\mathsf{LV}_{entry}(4) & = & \mathsf{LV}_{exit}(4) \cup \{\texttt{x},\texttt{y}\} & \mathsf{LV}_{exit}(4) & = & \mathsf{LV}_{entry}(5) \cup \mathsf{LV}_{entry}(6) \\
\mathsf{LV}_{entry}(5) & = & (\mathsf{LV}_{exit}(5)\backslash\{\texttt{z}\}) \cup \{\texttt{y}\} & \mathsf{LV}_{exit}(5) & = & \mathsf{LV}_{entry}(7) \\
\mathsf{LV}_{entry}(6) & = & (\mathsf{LV}_{exit}(6)\backslash\{\texttt{z}\}) \cup \{\texttt{y}\} & \mathsf{LV}_{exit}(6) & = & \mathsf{LV}_{entry}(7) \\
\mathsf{LV}_{entry}(7) & = & \{\texttt{z}\} & \mathsf{LV}_{exit}(7) & = & \emptyset
\end{array}
$$

Example (cont.):

$[\mathtt{x:=2}]^1; [\mathtt{y:=4}]^2; [\mathtt{x:=1}]^3; (\mathtt{if}\ [\mathtt{y>x}]^4\ \mathtt{then}\ [\mathtt{z:=y}]^5\ \mathtt{else}\ [\mathtt{z:=y*y}]^6); [\mathtt{x:=z}]^7$

Smallest solution:

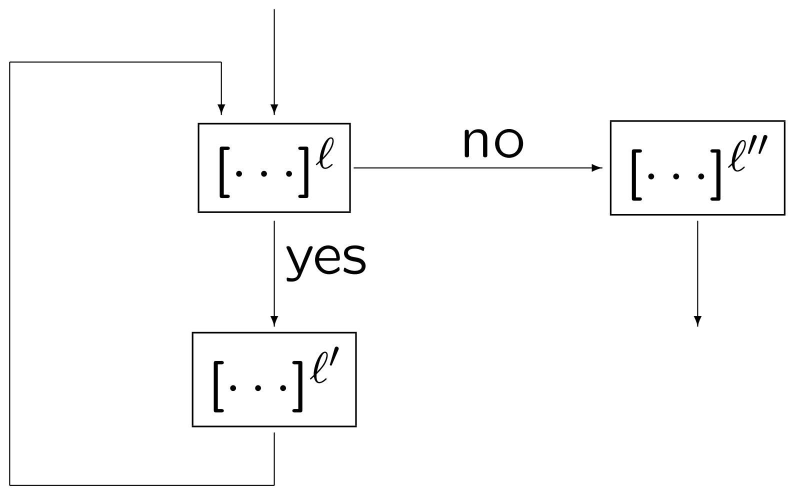
ℓ	$\mathsf{LV}_{entry}(\ell)$	$\mathsf{LV}_{exit}(\ell)$
1	$\emptyset$	$\emptyset$
2	$\emptyset$	$\{\mathtt{y}\}$
3	$\{\mathtt{y}\}$	$\{\mathtt{x}, \mathtt{y}\}$
4	$\{\mathtt{x}, \mathtt{y}\}$	$\{\mathtt{y}\}$
5	$\{\mathtt{y}\}$	$\{\mathtt{z}\}$
6	$\{\mathtt{y}\}$	$\{\mathtt{z}\}$
7	$\{\mathtt{z}\}$	$\emptyset$

Why smallest solution?

$$(\texttt{while } [\texttt{x>1}]^{\ell} \texttt{ do } [\texttt{skip}]^{\ell'}); [\texttt{x:=x+1}]^{\ell''}$$

Equations:

$$
\begin{aligned}
\mathsf{LV}_{entry}(\ell) &= \mathsf{LV}_{exit}(\ell) \cup \{\texttt{x}\} \\
\mathsf{LV}_{entry}(\ell') &= \mathsf{LV}_{exit}(\ell') \\
\mathsf{LV}_{entry}(\ell'') &= \{\texttt{x}\} \\
\mathsf{LV}_{exit}(\ell) &= \mathsf{LV}_{entry}(\ell') \cup \mathsf{LV}_{entry}(\ell'') \\
\mathsf{LV}_{exit}(\ell') &= \mathsf{LV}_{entry}(\ell) \\
\mathsf{LV}_{exit}(\ell'') &= \emptyset
\end{aligned}
$$



After some calculations: $\mathsf{LV}_{exit}(\ell) = \mathsf{LV}_{exit}(\ell) \cup \{\texttt{x}\}$

Many solutions to this equation: any superset of $\{\texttt{x}\}$

Derived Data Flow Information

- *Use-Definition chains* or *ud* chains:

  each use of a variable is linked to all assignments that reach it

  $[\texttt{x:=0}]^1; [\texttt{x:=3}]^2; (\texttt{if } [\texttt{z=x}]^3 \texttt{ then } [\texttt{z:=0}]^4 \texttt{ else } [\texttt{z:=x}]^5); [\texttt{y:=x}]^6; [\texttt{x:=y+z}]^7$

- *Definition-Use chains* or *du* chains:

  each assignment to a variable is linked to all uses of it

  $[\texttt{x:=0}]^1; [\texttt{x:=3}]^2; (\texttt{if } [\texttt{z=x}]^3 \texttt{ then } [\texttt{z:=0}]^4 \texttt{ else } [\texttt{z:=x}]^5); [\texttt{y:=x}]^6; [\texttt{x:=y+z}]^7$

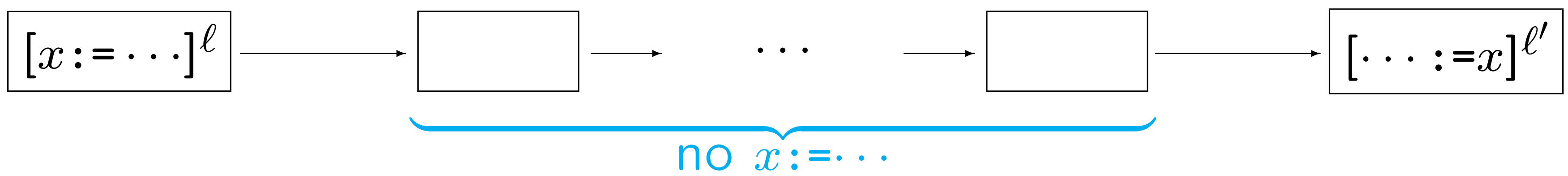
ud chains

$$ud : \mathbf{Var}_\star \times \mathbf{Lab}_\star \to \mathcal{P}(\mathbf{Lab}_\star)$$

given by

$$\begin{array}{rcl} ud(x,\ell') & = & \{\ell \mid def(x,\ell) \wedge \exists \ell'' : (\ell,\ell'') \in flow(S_\star) \wedge clear(x,\ell'',\ell')\} \\ & \cup & \{? \mid clear(x, init(S_\star), \ell')\} \end{array}$$

where



- $def(x,\ell)$ means that the block ℓ assigns a value to x
- $clear(x,\ell,\ell')$ means that none of the blocks on a path from ℓ to ℓ' contains an assignments to x but that the block ℓ' uses x (in a test or on the right hand side of an assignment)

ud chains - an alternative definition

$$\mathsf{UD} : \mathbf{Var}_\star \times \mathbf{Lab}_\star \to \mathcal{P}(\mathbf{Lab}_\star)$$

is defined by:

$$\mathsf{UD}(x,\ell) = \begin{cases} \{\ell' \mid (x,\ell') \in \mathsf{RD}_{entry}(\ell)\} & \text{if } x \in \mathit{gen}_{\mathsf{LV}}(B^\ell) \\ \emptyset & \text{otherwise} \end{cases}$$

One can show that:

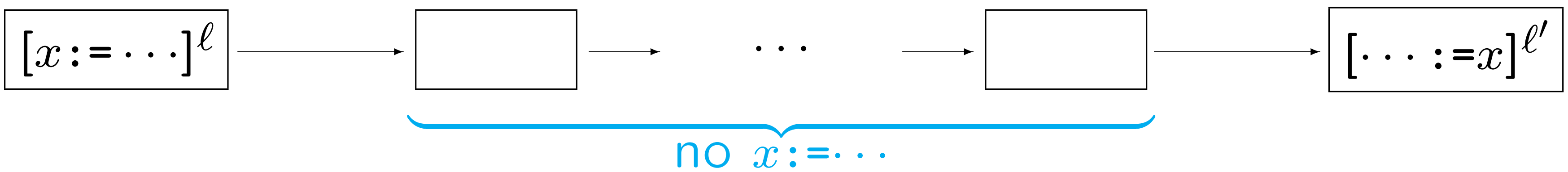
$$\mathit{ud}(x,\ell) = \mathsf{UD}(x,\ell)$$

du chains

$$du : \mathbf{Var}_\star \times \mathbf{Lab}_\star \to \mathcal{P}(\mathbf{Lab}_\star)$$

given by

$$du(x,\ell) = \begin{cases} \{\ell' \mid def(x,\ell) \wedge \exists \ell'' : (\ell,\ell'') \in flow(S_\star) \wedge clear(x,\ell'',\ell')\} \\ \qquad \text{if } \ell \neq ? \\ \{\ell' \mid clear(x, init(S_\star), \ell')\} \\ \qquad \text{if } \ell = ? \end{cases}$$



One can show that:

$$du(x,\ell) = \{\ell' \mid \ell \in ud(x,\ell')\}$$

Example:

[x:=0]^1; [x:=3]^2; (if [z=x]^3 then [z:=0]^4 else [z:=x]^5); [y:=x]^6; [x:=y+z]^7

$ud(x,\ell)$	x	y	z
1	$\emptyset$	$\emptyset$	$\emptyset$
2	$\emptyset$	$\emptyset$	$\emptyset$
3	$\{2\}$	$\emptyset$	$\{?\}$
4	$\emptyset$	$\emptyset$	$\emptyset$
5	$\{2\}$	$\emptyset$	$\emptyset$
6	$\{2\}$	$\emptyset$	$\emptyset$
7	$\emptyset$	$\{6\}$	$\{4,5\}$

$du(x,\ell)$	x	y	z
1	$\emptyset$	$\emptyset$	$\emptyset$
2	$\{3,5,6\}$	$\emptyset$	$\emptyset$
3	$\emptyset$	$\emptyset$	$\emptyset$
4	$\emptyset$	$\emptyset$	$\{7\}$
5	$\emptyset$	$\emptyset$	$\{7\}$
6	$\emptyset$	$\{7\}$	$\emptyset$
7	$\emptyset$	$\emptyset$	$\emptyset$
?	$\emptyset$	$\emptyset$	$\{3\}$