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

$$a \;::=\; x \mid n \mid a_1 \; op_a \; a_2$$

$$b \;::=\; \texttt{true} \mid \texttt{false} \mid \texttt{not}\; b \mid b_1 \; op_b \; b_2 \mid a_1 \; op_r \; a_2$$

$$S \;::=\; [x := a]^\ell \mid [\texttt{skip}]^\ell \mid S_1; S_2 \mid \texttt{if}\; [b]^\ell \;\texttt{then}\; S_1 \;\texttt{else}\; S_2 \mid \texttt{while}\; [b]^\ell \;\texttt{do}\; S$$

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

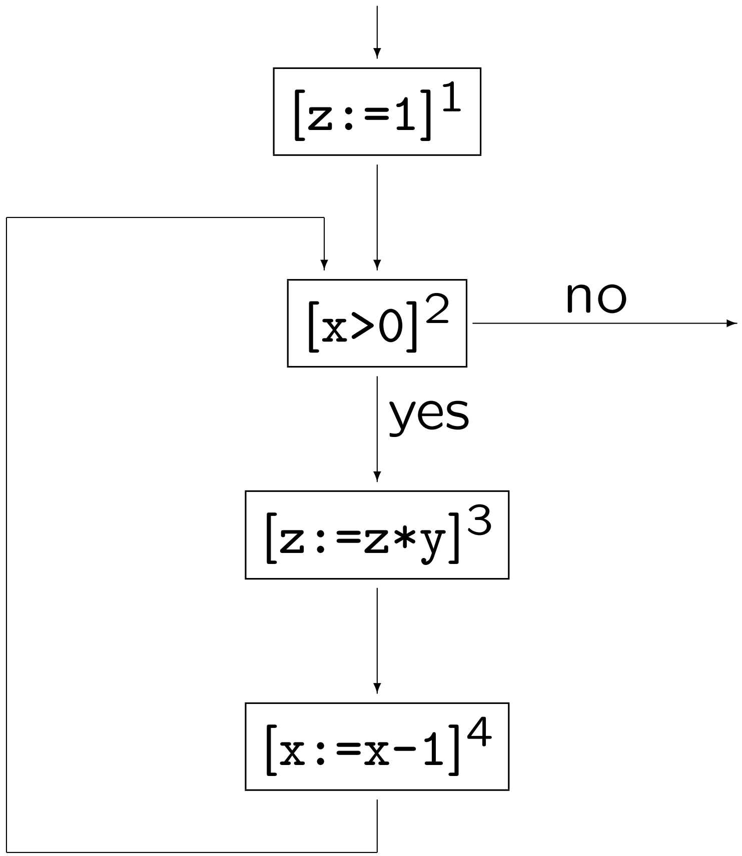
$$init(\cdots) = 1$$

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

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

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

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



Initial labels

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

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

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

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:

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

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

Example:

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

Labels

labels(S) is the entire set of labels in the statement S:

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

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

Example

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

Flows and reverse flows

flow(S) and *flow*$^R(S)$ are representations of how control flows in S:

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

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

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

Elementary blocks

A statement consists of a set of *elementary blocks*

$$\textit{blocks} : \mathbf{Stmt} \to \mathcal{P}(\mathbf{Blocks})$$

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

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

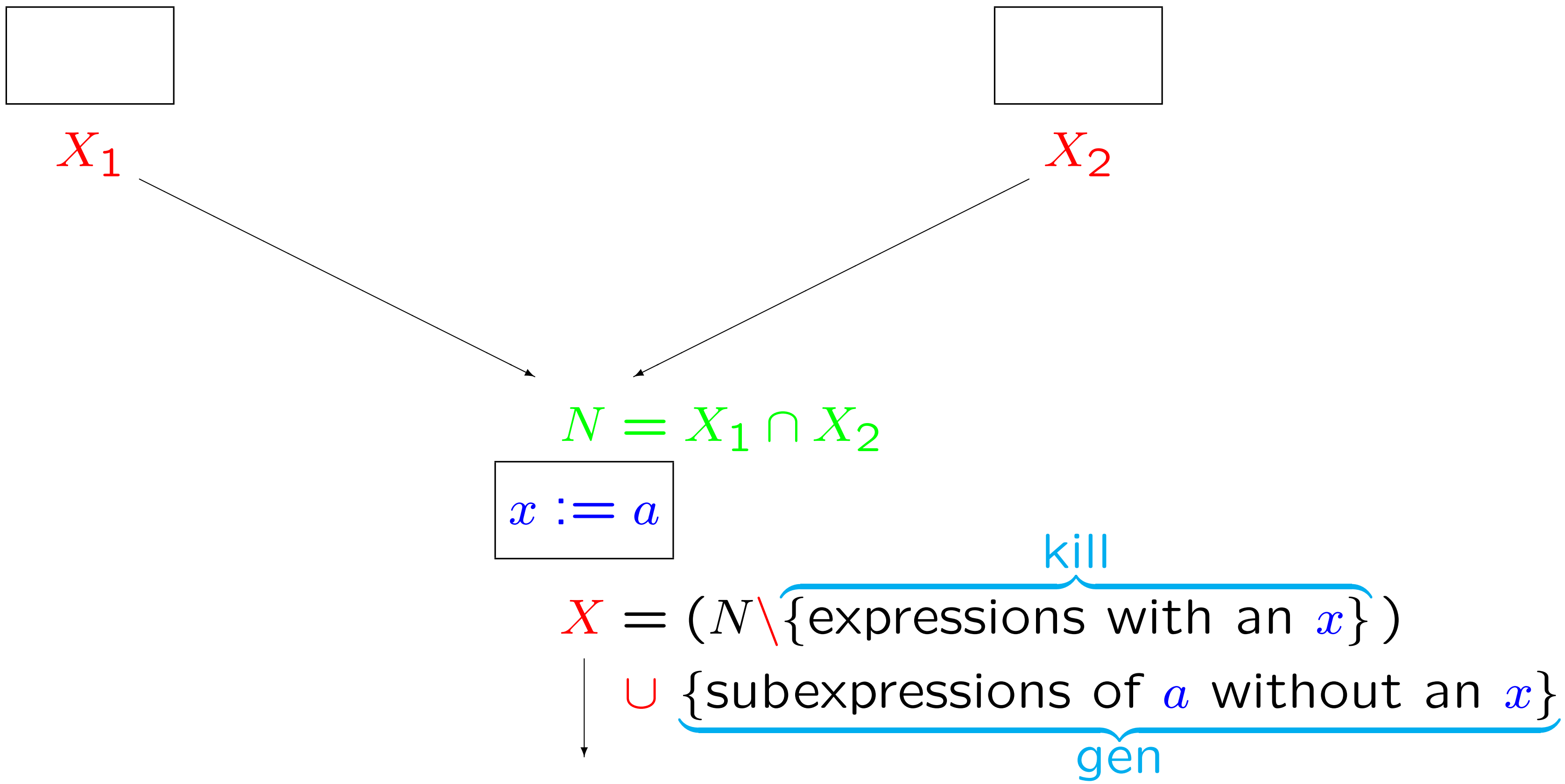
$\Downarrow$

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

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



Available Expressions Analysis

kill and *gen* functions

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

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

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:

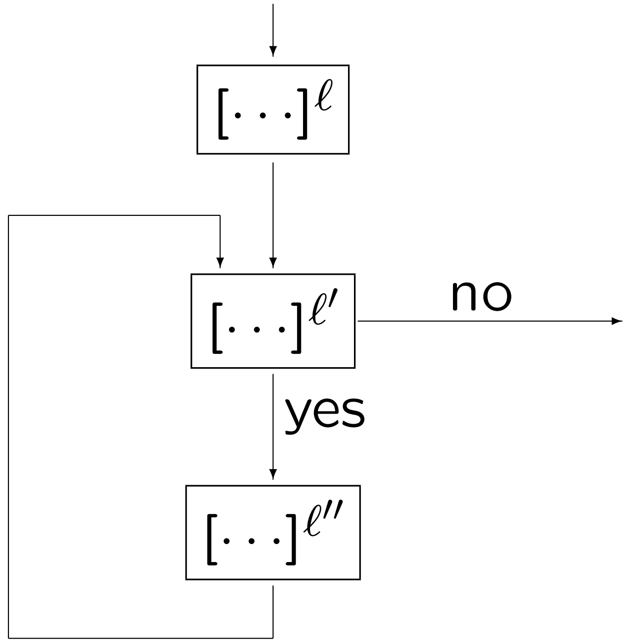
ℓ	$\mathsf{AE}_{entry}(\ell)$	$\mathsf{AE}_{exit}(\ell)$
1	$\emptyset$	{a+b}
2	{a+b}	{a+b, a*b}
3	{a+b}	{a+b}
4	{a+b}	$\emptyset$
5	$\emptyset$	{a+b}

Why largest solution?

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

Equations:

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



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:

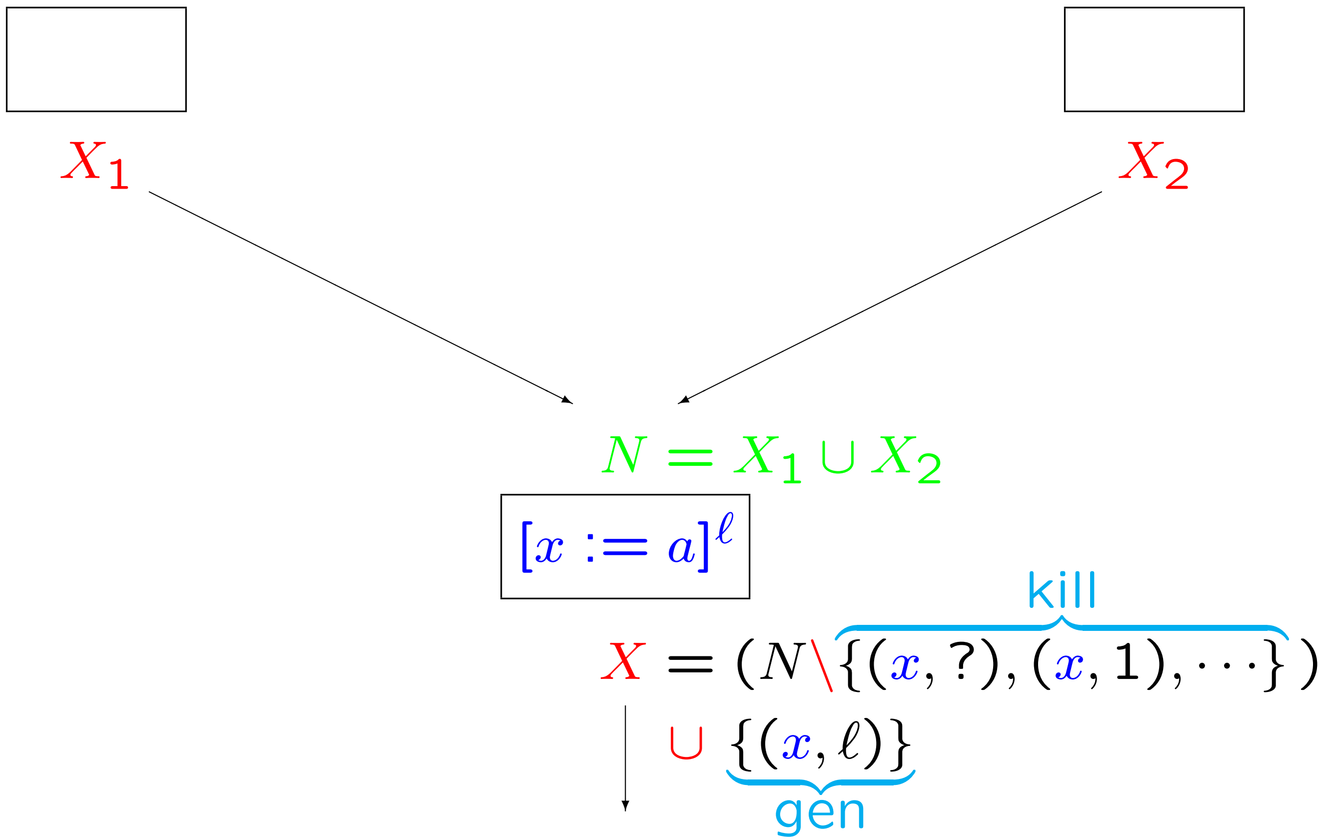
point of interest

$\Downarrow$

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

useful for definition-use chains and use-definition chains

Reaching Definitions Analysis – the basic idea



Reaching Definitions Analysis

kill and *gen* functions

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

$$
\begin{aligned}
gen_{RD}([x := a]^\ell) &= \{(x, \ell)\} \\
gen_{RD}([\texttt{skip}]^\ell) &= \emptyset \\
gen_{RD}([b]^\ell) &= \emptyset
\end{aligned}
$$

data flow equations: $RD^=$

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

$$
RD_{exit}(\ell) = (RD_{entry}(\ell) \backslash kill_{RD}(B^\ell)) \cup gen_{RD}(B^\ell) \quad \text{where } B^\ell \in blocks(S_\star)
$$

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:

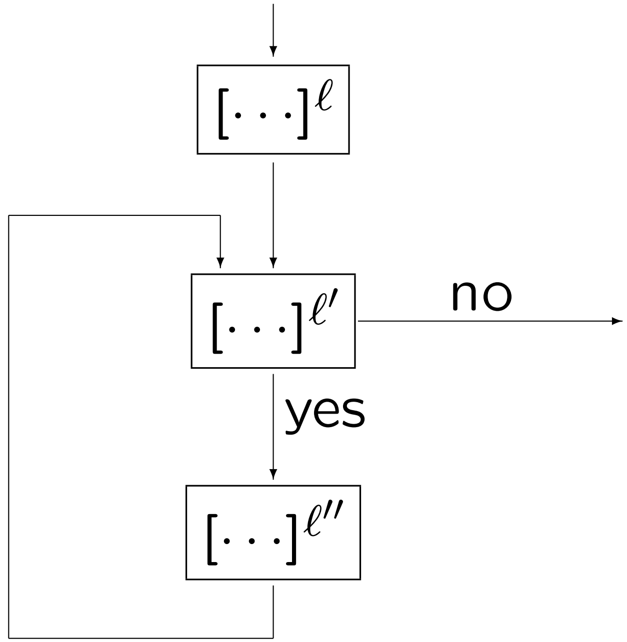
ℓ	$\mathsf{RD}_{entry}(\ell)$	$\mathsf{RD}_{exit}(\ell)$
1	$\{(x,?),(y,?)\}$	$\{(y,?),(x,1)\}$
2	$\{(y,?),(x,1)\}$	$\{(x,1),(y,2)\}$
3	$\{(x,1),(y,2),(y,4),(x,5)\}$	$\{(x,1),(y,2),(y,4),(x,5)\}$
4	$\{(x,1),(y,2),(y,4),(x,5)\}$	$\{(x,1),(y,4),(x,5)\}$
5	$\{(x,1),(y,4),(x,5)\}$	$\{(y,4),(x,5)\}$

Why smallest solution?

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

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:

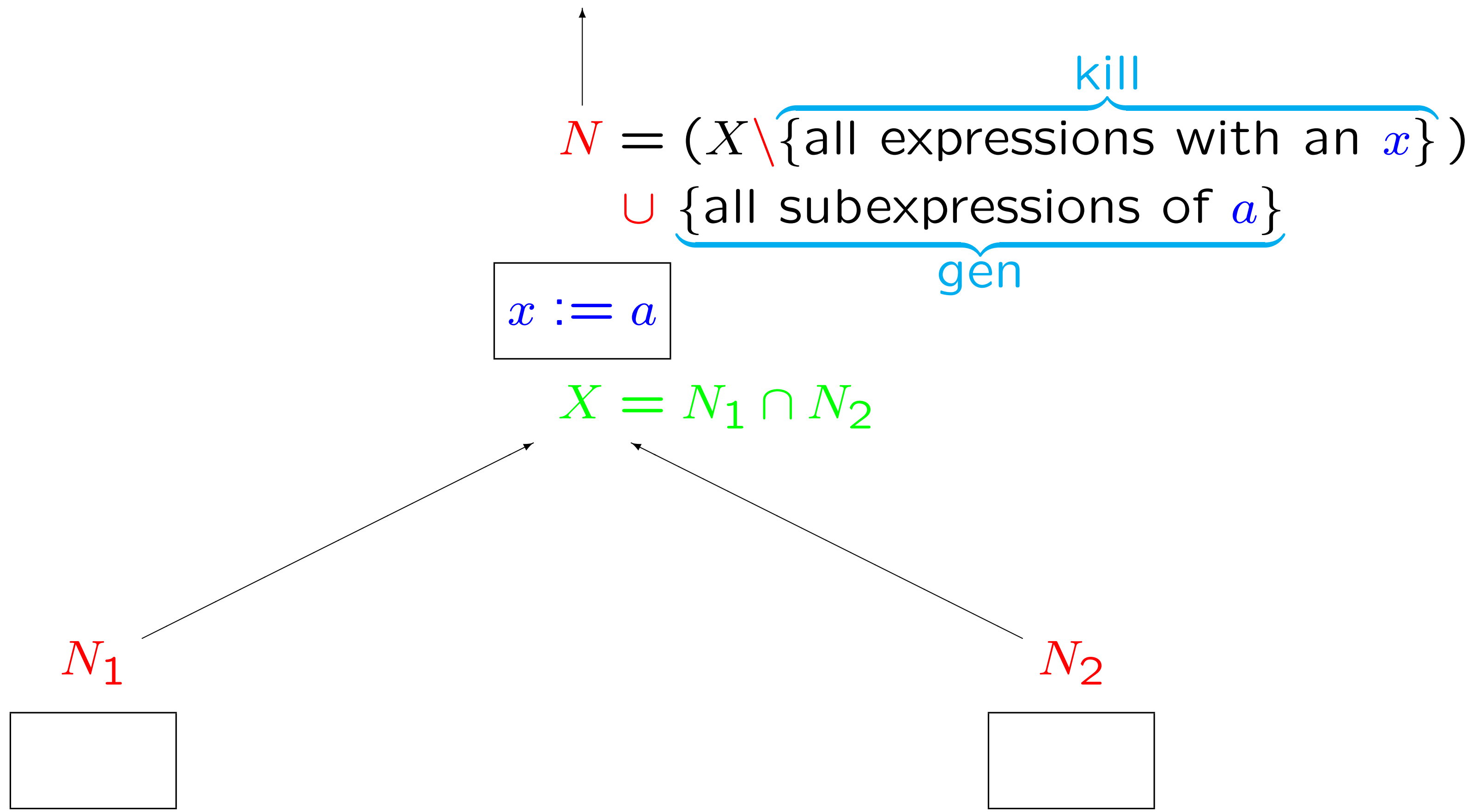
point of interest

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

The analysis enables a transformation into

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

Very Busy Expressions Analysis – the basic idea



Very Busy Expressions Analysis

kill and *gen* functions

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

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

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

$$\begin{aligned}
\mathsf{VB}_{entry}(\ell) &= (\mathsf{VB}_{exit}(\ell) \backslash kill_{\mathsf{VB}}(B^\ell)) \cup gen_{\mathsf{VB}}(B^\ell) \\
&\quad \text{where } B^\ell \in blocks(S_\star)
\end{aligned}$$

Example:

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

kill and *gen* function:

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

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.):

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

Largest solution:

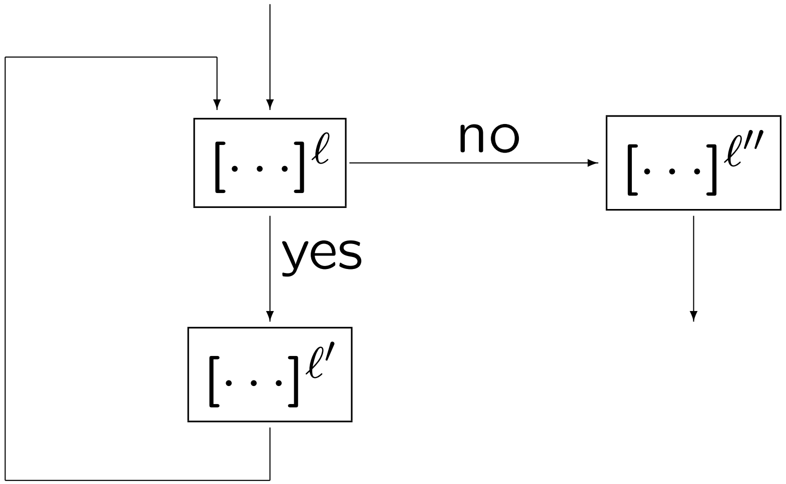
ℓ	$\mathsf{VB}_{entry}(\ell)$	$\mathsf{VB}_{exit}(\ell)$
1	$\{\texttt{a-b}, \texttt{b-a}\}$	$\{\texttt{a-b}, \texttt{b-a}\}$
2	$\{\texttt{a-b}, \texttt{b-a}\}$	$\{\texttt{a-b}\}$
3	$\{\texttt{a-b}\}$	$\emptyset$
4	$\{\texttt{a-b}, \texttt{b-a}\}$	$\{\texttt{a-b}\}$
5	$\{\texttt{a-b}\}$	$\emptyset$

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

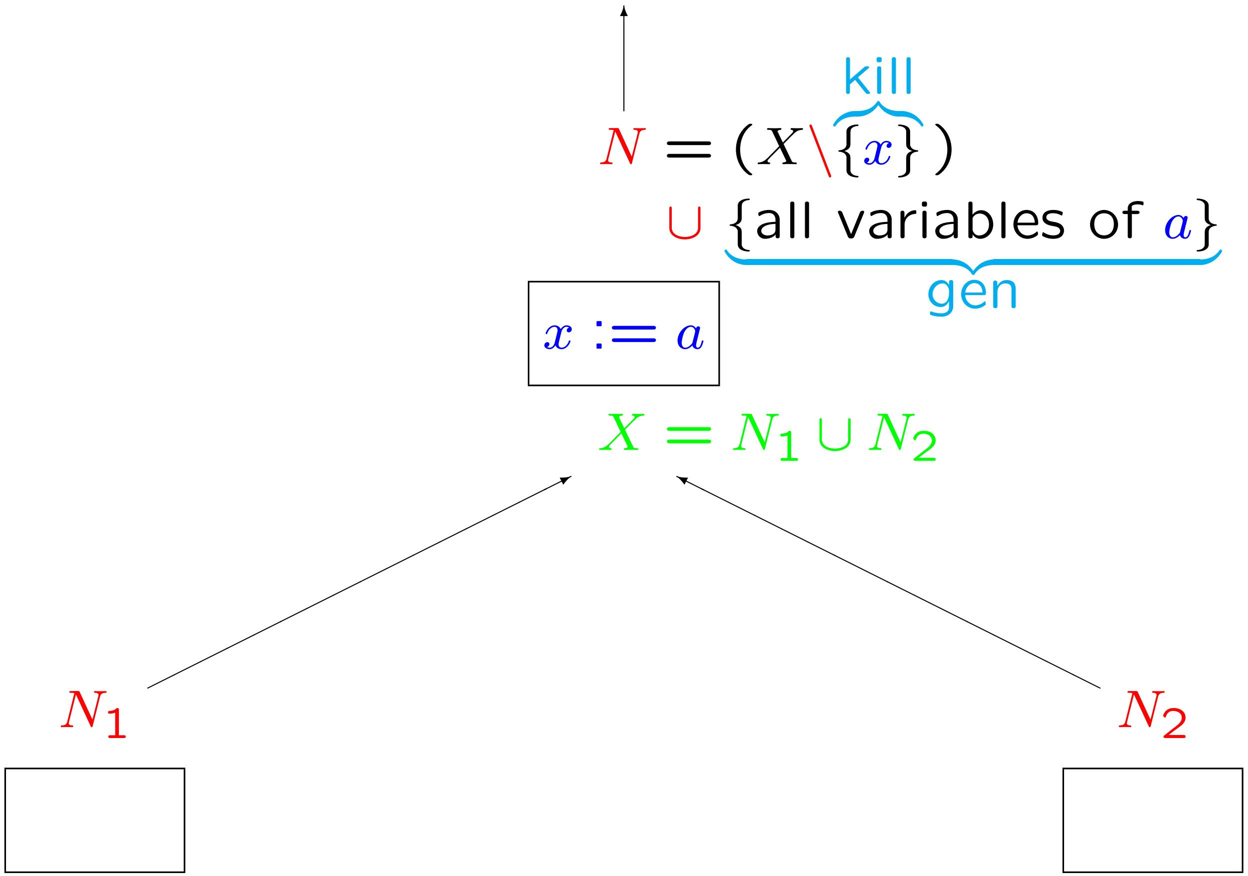
$\Downarrow$

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

The analysis enables a transformation into

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

Live Variables Analysis – the basic idea



Live Variables Analysis

kill and *gen* functions

$$\begin{aligned}
kill_{LV}([x := a]^{\ell}) &= \{x\}\\
kill_{LV}([\texttt{skip}]^{\ell}) &= \emptyset\\
kill_{LV}([b]^{\ell}) &= \emptyset\\
\\
gen_{LV}([x := a]^{\ell}) &= FV(a)\\
gen_{LV}([\texttt{skip}]^{\ell}) &= \emptyset\\
gen_{LV}([b]^{\ell}) &= FV(b)
\end{aligned}$$

data flow equations: $LV^{=}$

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

$$LV_{entry}(\ell) = (LV_{exit}(\ell) \backslash kill_{LV}(B^{\ell})) \cup gen_{LV}(B^{\ell})$$
where $B^{\ell} \in blocks(S_\star)$

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:

$$
\begin{aligned}
LV_{entry}(1) &= LV_{exit}(1) \backslash \{\texttt{x}\} \\
LV_{entry}(2) &= LV_{exit}(2) \backslash \{\texttt{y}\} \\
LV_{entry}(3) &= LV_{exit}(3) \backslash \{\texttt{x}\} \\
LV_{entry}(4) &= LV_{exit}(4) \cup \{\texttt{x}, \texttt{y}\} \\
LV_{entry}(5) &= (LV_{exit}(5) \backslash \{\texttt{z}\}) \cup \{\texttt{y}\} \\
LV_{entry}(6) &= (LV_{exit}(6) \backslash \{\texttt{z}\}) \cup \{\texttt{y}\} \\
LV_{entry}(7) &= \{\texttt{z}\}
\end{aligned}
$$

$$
\begin{aligned}
LV_{exit}(1) &= LV_{entry}(2) \\
LV_{exit}(2) &= LV_{entry}(3) \\
LV_{exit}(3) &= LV_{entry}(4) \\
LV_{exit}(4) &= LV_{entry}(5) \cup LV_{entry}(6) \\
LV_{exit}(5) &= LV_{entry}(7) \\
LV_{exit}(6) &= LV_{entry}(7) \\
LV_{exit}(7) &= \emptyset
\end{aligned}
$$

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:

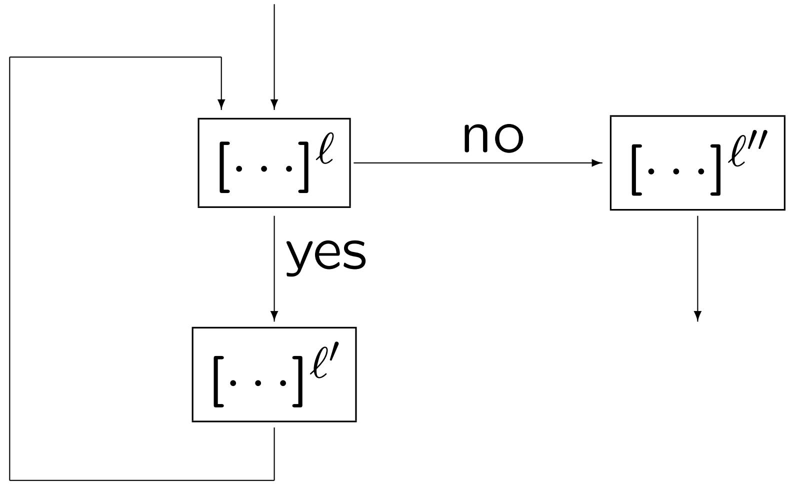
ℓ	$\mathsf{LV}_{entry}(\ell)$	$\mathsf{LV}_{exit}(\ell)$
1	$\emptyset$	$\emptyset$
2	$\emptyset$	$\{\mathtt{y}\}$
3	$\{\mathtt{y}\}$	$\{\mathtt{x}, \mathtt{y}\}$
4	$\{\mathtt{x}, \mathtt{y}\}$	$\{\mathtt{y}\}$
5	$\{\mathtt{y}\}$	$\{\mathtt{z}\}$
6	$\{\mathtt{y}\}$	$\{\mathtt{z}\}$
7	$\{\mathtt{z}\}$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

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 $\{\texttt{x}\}$

Monotone Frameworks

- Monotone and Distributive Frameworks

- Instances of Frameworks

- Constant Propagation Analysis

The Overall Pattern

Each of the four classical analyses take the form

$$\mathit{Analysis}_\circ(\ell) = \begin{cases} \iota & \text{if } \ell \in E \\ \bigsqcup\{\mathit{Analysis}_\bullet(\ell') \mid (\ell', \ell) \in F\} & \text{otherwise} \end{cases}$$

$$\mathit{Analysis}_\bullet(\ell) = f_\ell(\mathit{Analysis}_\circ(\ell))$$

where

- $\bigsqcup$ is $\bigcap$ or $\bigcup$ (and $\sqcup$ is $\cup$ or $\cap$),
- F is either $\mathit{flow}(S_\star)$ or $\mathit{flow}^R(S_\star)$,
- E is $\{\mathit{init}(S_\star)\}$ or $\mathit{final}(S_\star)$,
- ι specifies the initial or final analysis information, and
- f_ℓ is the transfer function associated with $B^\ell \in \mathit{blocks}(S_\star)$.

The Principle: forward versus backward

- The *forward analyses* have F to be *flow*$(S_\star)$ and then *Analysis*$_\circ$ concerns entry conditions and *Analysis*$_\bullet$ concerns exit conditions; the equation system presupposes that $S_\star$ has isolated entries.

- The *backward analyses* have F to be *flow*$^R(S_\star)$ and then *Analysis*$_\circ$ concerns exit conditions and *Analysis*$_\bullet$ concerns entry conditions; the equation system presupposes that $S_\star$ has isolated exits.

The Principle: union versus intersecton

- When $\bigsqcup$ is $\bigcap$ we require the greatest sets that solve the equations and we are able to detect properties satisfied by *all execution paths* reaching (or leaving) the entry (or exit) of a label; the analysis is called a must-analysis.

- When $\bigsqcup$ is $\bigcup$ we require the smallest sets that solve the equations and we are able to detect properties satisfied by *at least one execution path* to (or from) the entry (or exit) of a label; the analysis is called a may-analysis.

Property Spaces

The *property space*, L, is used to represent the data flow information, and the *combination operator*, $\bigsqcup$: $\mathcal{P}(L) \rightarrow L$, is used to combine information from different paths.

- L is a *complete lattice*, that is, a partially ordered set, $(L, \sqsubseteq)$, such that each subset, Y, has a least upper bound, $\bigsqcup Y$.

- L satisfies the *Ascending Chain Condition*; that is, each ascending chain eventually stabilises (meaning that if $(l_n)_n$ is such that $l_1 \sqsubseteq l_2 \sqsubseteq l_3 \sqsubseteq \cdots$,then there exists n such that $l_n = l_{n+1} = \cdots$).

Example: Reaching Definitions

- $L = \mathcal{P}(\mathbf{Var}_\star \times \mathbf{Lab}_\star)$ is partially ordered by subset inclusion so $\sqsubseteq$ is $\subseteq$
- the least upper bound operation $\bigsqcup$ is $\bigcup$ and the least element $\bot$ is $\emptyset$
- L satisfies the Ascending Chain Condition because $\mathbf{Var}_\star \times \mathbf{Lab}_\star$ is finite (unlike $\mathbf{Var} \times \mathbf{Lab}$)

Example: Available Expressions

- $L = \mathcal{P}(\mathbf{AExp}_\star)$ is partially ordered by superset inclusion so $\sqsubseteq$ is $\supseteq$
- the least upper bound operation $\bigsqcup$ is $\bigcap$ and the least element $\bot$ is $\mathbf{AExp}_\star$
- L satisfies the Ascending Chain Condition because $\mathbf{AExp}_\star$ is finite (unlike $\mathbf{AExp}$)

Transfer Functions

The set of transfer functions, $\mathcal{F}$, is a set of monotone functions over L, meaning that

$$l \sqsubseteq l' \text{ implies } f_\ell(l) \sqsubseteq f_\ell(l')$$

and furthermore they fulfil the following conditions:

- $\mathcal{F}$ contains *all* the transfer functions $f_\ell : L \rightarrow L$ in question (for $\ell \in \mathbf{Lab}_\star$)
- $\mathcal{F}$ contains the *identity function*
- $\mathcal{F}$ is *closed under composition* of functions

Frameworks

A *Monotone Framework* consists of:

- a complete lattice, L, that satisfies the Ascending Chain Condition; we write $\bigsqcup$ for the least upper bound operator

- a set $\mathcal{F}$ of monotone functions from L to L that contains the identity function and that is closed under function composition

A *Distributive Framework* is a Monotone Framework where additionally all functions f in $\mathcal{F}$ are required to be distributive:

$$f(l_1 \sqcup l_2) = f(l_1) \sqcup f(l_2)$$

Instances

An *instance* of a Framework consists of:

– the complete lattice, L, of the framework

– the space of functions, $\mathcal{F}$, of the framework

– a finite flow, F (typically *flow*$(S_\star)$ or *flow*$^R(S_\star)$)

– a finite set of *extremal labels*, E (typically $\{$*init*$(S_\star)\}$ or *final*$(S_\star)$)

– an *extremal value*, $\iota \in L$, for the extremal labels

– a mapping, $f_\cdot$, from the labels $\mathbf{Lab}_\star$ to transfer functions in $\mathcal{F}$

Equations of the Instance:

$$\mathit{Analysis}_\circ(\ell) = \bigsqcup\{\mathit{Analysis}_\bullet(\ell') \mid (\ell',\ell) \in F\} \sqcup \iota_E^\ell$$

$$\text{where } \iota_E^\ell = \begin{cases} \iota & \text{if } \ell \in E \\ \bot & \text{if } \ell \notin E \end{cases}$$

$$\mathit{Analysis}_\bullet(\ell) = f_\ell(\mathit{Analysis}_\circ(\ell))$$

Constraints of the Instance:

$$\mathit{Analysis}_\circ(\ell) \sqsupseteq \bigsqcup\{\mathit{Analysis}_\bullet(\ell') \mid (\ell',\ell) \in F\} \sqcup \iota_E^\ell$$

$$\text{where } \iota_E^\ell = \begin{cases} \iota & \text{if } \ell \in E \\ \bot & \text{if } \ell \notin E \end{cases}$$

$$\mathit{Analysis}_\bullet(\ell) \sqsupseteq f_\ell(\mathit{Analysis}_\circ(\ell))$$

The Examples Revisited

	Available Expressions	Reaching Definitions	Very Busy Expressions	Live Variables
L	$\mathcal{P}(\mathbf{AExp}_\star)$	$\mathcal{P}(\mathbf{Var}_\star \times \mathbf{Lab}_\star)$	$\mathcal{P}(\mathbf{AExp}_\star)$	$\mathcal{P}(\mathbf{Var}_\star)$
$\sqsubseteq$	$\supseteq$	$\subseteq$	$\supseteq$	$\subseteq$
$\bigsqcup$	$\bigcap$	$\bigcup$	$\bigcap$	$\bigcup$
$\bot$	$\mathbf{AExp}_\star$	$\emptyset$	$\mathbf{AExp}_\star$	$\emptyset$
ι	$\emptyset$	$\{(x,?) \mid x \in \mathit{FV}(S_\star)\}$	$\emptyset$	$\emptyset$
E	$\{\mathit{init}(S_\star)\}$	$\{\mathit{init}(S_\star)\}$	$\mathit{final}(S_\star)$	$\mathit{final}(S_\star)$
F	$\mathit{flow}(S_\star)$	$\mathit{flow}(S_\star)$	$\mathit{flow}^R(S_\star)$	$\mathit{flow}^R(S_\star)$
$\mathcal{F}$	$\{f : L \to L \mid \exists l_k, l_g : f(l) = (l \setminus l_k) \cup l_g\}$			
f_ℓ	$f_\ell(l) = (l \setminus \mathit{kill}(B^\ell)) \cup \mathit{gen}(B^\ell)$ where $B^\ell \in \mathit{blocks}(S_\star)$			

Bit Vector Frameworks

A *Bit Vector Framework* has

- $L = \mathcal{P}(D)$ for D finite
- $\mathcal{F} = \{f \mid \exists l_k, l_g : f(l) = (l \setminus l_k) \cup l_g\}$

Examples:

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

Lemma: Bit Vector Frameworks are always Distributive Frameworks

Proof

$$f(l_1 \sqcup l_2) = \begin{cases} f(l_1 \cup l_2) \\ f(l_1 \cap l_2) \end{cases} = \begin{cases} ((l_1 \cup l_2) \setminus l_k) \cup l_g \\ ((l_1 \cap l_2) \setminus l_k) \cup l_g \end{cases}$$

$$= \begin{cases} ((l_1 \setminus l_k) \cup (l_2 \setminus l_k)) \cup l_g \\ ((l_1 \setminus l_k) \cap (l_2 \setminus l_k)) \cup l_g \end{cases} = \begin{cases} ((l_1 \setminus l_k) \cup l_g) \cup ((l_2 \setminus l_k) \cup l_g) \\ ((l_1 \setminus l_k) \cup l_g) \cap ((l_2 \setminus l_k) \cup l_g) \end{cases}$$

$$= \begin{cases} f(l_1) \cup f(l_2) \\ f(l_1) \cap f(l_2) \end{cases} = f(l_1) \sqcup f(l_2)$$

- $id(l) = (l \setminus \emptyset) \cup \emptyset$
- $f_2(f_1(l)) = (((l \setminus l_k^1) \cup l_g^1) \setminus l_k^2) \cup l_g^2 = (l \setminus (l_k^1 \cup l_k^2)) \cup ((l_g^1 \setminus l_k^2) \cup l_g^2)$
- monotonicity follows from distributivity
- $\mathcal{P}(D)$ satisfies the Ascending Chain Condition because D is finite