COMP0174 Practical Program Analysis
Very Busy Expressions

Sergey Mechtaev
s.mechtaev@ucl.ac.uk
UCL Computer Science

Four Classic Analyses

	Forward	Backward
Must	Available Expressions	**Very Busy Expressions**
May	Reaching Definitions	Live Variables

Very Busy Expressions

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

Very busy expressions analysis determines for each program point, which expressions must be busy at the exit from the point.

It is *backward must* analysis.

Applications: Optimization (evaluate the expression in the block and store its value for later use, aka *hoisting* the expression)

Example

$if\ [a > b]^1\ then$
$\quad [x \coloneqq b - a]^2;$
$\quad [y \coloneqq a - b]^3;$
$else$
$\quad [y \coloneqq b - a]^4;$
$\quad [x \coloneqq a - b]^5;$

$a - b$ and $b - a$ are both very busy at the start of the conditional, can be hoisted to reduce the size of generated code.

Killed Expressions

An expression is killed in a block if any of the variables used in the expression are modified in the block:

$$kill_{VB}: Blocks_* \rightarrow P(AExp_*)$$

$$kill_{VB}\left([x := a]^l\right) = \{a' \in AExp_* \mid x \in Vars(a')\}$$
$$kill_{VB}\left([skip]^l\right) = \emptyset$$
$$kill_{VB}\left([b]^l\right) = \emptyset$$

where $AExp_*$ are all expressions in the program.

Killed Expressions

An expression is killed in a block if any of the variables used in the expression are modified in the block:

$$kill_{VB}: Blocks_* \rightarrow P(AExp_*)$$

$$\boxed{kill_{VB}([x := a]^l) = \{a' \in AExp_* \mid x \in Vars(a')\}}$$

$$kill_{VB}([skip]^l) = \emptyset$$

$$kill_{VB}([b]^l) = \emptyset$$

For **assignment statements**: all the variables that hold expressions using the variables that has been assigned need to be killed

where $AExp_*$ are all expressions in the program.

Generated Expressions

A very busy expression is generated:

$$gen_{VB} : Blocks_* \to P(AExp_*)$$

$$gen_{VB}([x := a]^l) = AExp(a)$$
$$gen_{VB}([skip]^l) = \emptyset$$
$$gen_{VB}([b]^l) = AExp(b)$$

if it appears on the **right-hand side** of an assignment or inside an **if or loop condition**.

Analysis

The goal of the analysis is to compute the largest set satisfying the equation for VB_{exit}:

$$VB_{exit}(l) = \begin{cases} \emptyset & if\ l = init(program) \\ \cap \{VB_{entry}(l') \mid (l', l) \in flow^R\ (program)\} & otherwise \end{cases}$$

$$VB_{entry}(l) = \left(VB_{exit}(l) \setminus kill_{VB}\left(B^l\right)\right) \cup gen_{VB}\left(B^l\right)$$

where $B^l \in blocks(program)$

Example

$$
\begin{aligned}
&if\ [a > b]^1\ then\\
&\quad [x := b - a]^2;\\
&\quad [y := a - b]^3;\\
&else\\
&\quad [y := b - a]^4;\\
&\quad [x := a - b]^5;
\end{aligned}
$$

$\boldsymbol{l}$	$\boldsymbol{kill_{RD}(l)}$	$\boldsymbol{gen_{RD}(l)}$
1	$\emptyset$	$\emptyset$
2	$\emptyset$	$\{b - a\}$
3	$\emptyset$	$\{a - b\}$
4	$\emptyset$	$\{b - a\}$
5	$\emptyset$	$\{a - b\}$

Example

$$VB_{entry}(1) = VB_{exit}(1)$$
$$VB_{entry}(2) = VB_{exit}(2) \cup \{b - a\}$$
$$VB_{entry}(3) = \{a - b\}$$
$$VB_{entry}(4) = VB_{exit}(4) \cup \{b - a\}$$
$$VB_{entry}(5) = \{a - b\}$$

$$VB_{exit}(1) = VB_{entry}(2) \cap VB_{entry}(4)$$
$$VB_{exit}(2) = VB_{entry}(3)$$
$$VB_{exit}(3) = \emptyset$$
$$VB_{exit}(4) = VB_{entry}(5)$$
$$VB_{exit}(5) = \emptyset$$

Example

$$
\begin{aligned}
&if\ [a > b]^1\ then\\
&\quad [x := b - a]^2;\\
&\quad [y := a - b]^3;\\
&else\\
&\quad [y := b - a]^4;\\
&\quad [x := a - b]^5;
\end{aligned}
$$

$\boldsymbol{l}$	$\boldsymbol{VB_{entry}}\ (\boldsymbol{l})$	$\boldsymbol{VB_{exit}}\ (\boldsymbol{l})$
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	$\{a-b\}$	$\emptyset$