Workshop on Essential Abstractions in GCC

Introduction to Data Flow Analysis

GCC Resource Center
(www.cse.iitb.ac.in/grc)

Department of Computer Science and Engineering,
Indian Institute of Technology, Bombay

1 July 2012
Outline

- Motivation
- Live Variables Analysis
- Available Expressions Analysis
- Pointer Analysis
Part 2

Motivation
Dead Code Elimination

- No uses for variables a_3, b_4, c_5, and n_6
Dead Code Elimination

- No uses for variables \( a_3, b_4, c_5, \) and \( n_6 \)

\[
\begin{align*}
B2 & : \quad & a_1 = \phi (1, a_7) \\
B4 & : \quad & \text{if } a_1 \leq 6 \\
B3 & : \quad & a_7 = a_1 + 1 \\
B5 & : \quad & \text{if } a_1 \leq 11 \\
B6 & : \quad & D.1200.8 = a_1 + 2 \\
& & a_9 = D.1200.8 + 3 \\
B7 & : \quad & a_2 = \phi (a_1, a_9) \\
& & \text{return } a_2
\end{align*}
\]
Dead Code Elimination

- No uses for variables a_3, b_4, c_5, and n_6
- Assignments to these variables can be deleted

How can we conclude this systematically?
Find out at each program point $p$, the variables that are used beyond $p$.

```
B2
a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6

B4
a_1 = \phi (1, a_7)
if a_1 \leq 6

B3
a_7 = a_1 + 1

B5
if a_1 \leq 11

B6
D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3

B7
a_2 = \phi (a_1, a_9)
return a_2
```
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

\[
\begin{align*}
B2 & \quad a_3 = 1; \ b_4 = 2 \\
& \quad c_5 = 3; \ n_6 = 6 \\
B4 & \quad a_1 = \phi (1, a_7) \\
& \quad \text{if } a_1 \leq 6 \\
B3 & \quad a_7 = a_1 + 1 \\
B5 & \quad \text{if } a_1 \leq 11 \\
B6 & \quad D.1200_8 = a_1 + 2 \\
& \quad a_9 = D.1200_8 + 3 \\
B7 & \quad a_2 = \phi (a_1, a_9) \\
& \quad \text{return } a_2
\end{align*}
\]

Which variables are used beyond this point?

\( \emptyset \)
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

```plaintext
B2
a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6

B4
a_1 = φ (1, a_7)
if a_1 ≤ 6

B3
a_7 = a_1 + 1

B5
if a_1 ≤ 11

B6
D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3

B7
a_2 = φ (a_1, a_9)
return a_2
```

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?

$\{a_1, a_9\}$
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

B2

\[ a_3 = 1; b_4 = 2; c_5 = 3; n_6 = 6 \]

B4

\[ a_1 = \phi(1, a_7) \]

if $a_1 \leq 6$

B3

\[ a_7 = a_1 + 1 \]

B5

if $a_1 \leq 11$

B6

\[ D.1200_8 = a_1 + 2; a_9 = D.1200_8 + 3 \]

B7

\[ a_2 = \phi(a_1, a_9) \]

return $a_2$

Which variables are used beyond this point?

\{a_1, a_9\}

What about $a_2$?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

B2
\[
a_3 = 1; \ b_4 = 2 \\
c_5 = 3; \ n_6 = 6
\]

B4
\[
a_1 = \phi(1, a_7) \\
\text{if } a_1 \leq 6
\]

B3
\[
a_7 = a_1 + 1
\]

B5
\[
\text{if } a_1 \leq 11
\]

B6
\[
D.1200_8 = a_1 + 2 \\
a_9 = D.1200_8 + 3
\]

B7
\[
a_2 = \phi(a_1, a_9) \\
\text{return } a_2
\]

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$.

Which variables are used beyond this point?

$\{a_1, a_9\}$
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

```
B2
a_3 = 1; b_4 = 2
  c_5 = 3; n_6 = 6

B4
a_1 = \phi (1, a_7)
  if a_1 \leq 6

B3
a_7 = a_1 + 1

B5
if a_1 \leq 11

B6
D.1200_8 = a_1 + 2
  a_9 = D.1200_8 + 3

B7
a_2 = \phi (a_1, a_9)
  return a_2
```

Which variables are used beyond this point?

\{a_1\}
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

\[
\begin{align*}
\text{B2:} & \quad a_3 = 1; \ b_4 = 2 \\
& \quad c_5 = 3; \ n_6 = 6
\end{align*}
\]

\[
\begin{align*}
\text{B4:} & \quad a_1 = \phi (1, a_7) \\
& \quad \text{if } a_1 \leq 6
\end{align*}
\]

\[
\begin{align*}
\text{B3:} & \quad a_7 = a_1 + 1
\end{align*}
\]

\[
\begin{align*}
\text{B5:} & \quad \text{if } a_1 \leq 11
\end{align*}
\]

\[
\begin{align*}
\text{B6:} & \quad D.1200_8 = a_1 + 2 \\
& \quad a_9 = D.1200_8 + 3
\end{align*}
\]

\[
\begin{align*}
\text{B7:} & \quad a_2 = \phi (a_1, a_9) \\
& \quad \text{return } a_2
\end{align*}
\]

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?

\{a\_1, a\_9\}
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

\[
\begin{align*}
\text{B2} & : \quad a_3 = 1; \quad b_4 = 2; \quad c_5 = 3; \quad n_6 = 6 \\
\text{B4} & : \quad a_1 = \phi(1, a_7) \\
\text{B3} & : \quad a_7 = a_1 + 1 \\
\text{B5} & : \quad \text{if } a_1 \leq 11 \\
\text{B6} & : \quad D.1200_8 = a_1 + 2; \quad a_9 = D.1200_8 + 3 \\
\text{B7} & : \quad a_2 = \phi(a_1, a_9); \quad \text{return } a_2
\end{align*}
\]

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?

$\{a_1, a_9\}$
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$.

```
B2
a_3 = 1; b_4 = 2
   c_5 = 3; n_6 = 6
```

```
B4
a_1 = \phi (1, a_7)
if a_1 \leq 6
```

```
B5
if a_1 \leq 11
```

```
B3
a_7 = a_1 + 1
```

```
B6
D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3
```

```
B7
a_2 = \phi (a_1, a_9)
return a_2
```

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

```
B2

\[
a_3 = 1; \ b_4 = 2
\[
c_5 = 3; \ n_6 = 6
\]

B4

\[
a_1 = \phi (1, a_7)
\]

if \( a_1 \leq 6 \)

B3

\[
a_7 = a_1 + 1
\]

Which variables are used beyond this point?

\( \emptyset \) (Conservative assumption)

B5

if \( a_1 \leq 11 \)

B6

\[
D.1200_8 = a_1 + 2
\]

\[
a_9 = D.1200_8 + 3
\]

B7

\[
a_2 = \phi (a_1, a_9)
\]

return \( a_2 \)
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

\[
\begin{align*}
B2 & : a_{3} = 1; b_{4} = 2 \\
& \quad \quad c_{5} = 3; n_{6} = 6
\end{align*}
\]

\[
\begin{align*}
B4 & : a_{1} = \phi (1, a_{7}) \\
& \quad \text{if } a_{1} \leq 6
\end{align*}
\]

\[
\begin{align*}
B3 & : a_{7} = a_{1} + 1 \\
B5 & : \text{if } a_{1} \leq 11
\end{align*}
\]

\[
\begin{align*}
B6 & : D.1200_{8} = a_{1} + 2 \\
& \quad \quad a_{9} = D.1200_{8} + 3
\end{align*}
\]

\[
\begin{align*}
B7 & : a_{2} = \phi (a_{1}, a_{9}) \\
& \quad \quad \text{return } a_{2}
\end{align*}
\]

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

```
B2
\( a_{3} = 1; b_{4} = 2 \)
\( c_{5} = 3; n_{6} = 6 \)

B4
\( a_{1} = \phi (1, a_{7}) \)
if \( a_{1} \leq 6 \)

B3
\( a_{7} = a_{1} + 1 \)

B5
if \( a_{1} \leq 11 \)

B6
\( D.1200.8 = a_{1} + 2 \)
\( a_{9} = D.1200.8 + 3 \)

B7
\( a_{2} = \phi (a_{1}, a_{9}) \)
return \( a_{2} \)
```

Which variables are used beyond this point?

\{ \( a_{1} \) \}
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

```
B2
a_3 = 1; b_4 = 2
  c_5 = 3; n_6 = 6

B4
a_1 = \phi (1, a_7)
  if a_1 \leq 6

B5
if a_1 \leq 11

B3
a_7 = a_1 + 1

B6
D.1200_8 = a_1 + 2
  a_9 = D.1200_8 + 3

B7
a_2 = \phi (a_1, a_9)
  return a_2
```

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?

\{a_1, a_9\}
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

Which variables are used beyond this point?

```
B2
a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6

B4
a_1 = \phi (1, a_7)
if a_1 \leq 6

B3
a_7 = a_1 + 1

B5
if a_1 \leq 11

B6
D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3

B7
a_2 = \phi (a_1, a_9)
return a_2
```
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

Which variables are used beyond this point?

\( \{a_7, a_9\} \)

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

Which variables are used beyond this point?
Liveness Analysis of Variables

Find out at each program point \( p \), the variables that are used beyond \( p \)

\[
\begin{align*}
\text{B2:} & \quad a_3 = 1; \ b_4 = 2 \\
& \quad c_5 = 3; \ n_6 = 6
\end{align*}
\]

\[
\begin{align*}
\text{B4:} & \quad a_1 = \phi (1, a_7) \\
& \quad \text{if } a_1 \leq 6
\end{align*}
\]

\[
\begin{align*}
\text{B5:} & \quad \text{if } a_1 \leq 11
\end{align*}
\]

\[
\begin{align*}
\text{B6:} & \quad D.1200 \_8 = a_1 + 2 \\
& \quad a_9 = D.1200 \_8 + 3
\end{align*}
\]

\[
\begin{align*}
\text{B7:} & \quad a_2 = \phi (a_1, a_9) \\
& \quad \text{return } a_2
\end{align*}
\]

Which variables are used beyond this point?

\( \{a_7, a_9\} \)
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

{a_7, a_9}

Essential Abstractions in GCC  
GCC Resource Center, IIT Bombay
Liveness Analysis of Variables

Find out at each program point $p$, the variables that are used beyond $p$

```plaintext
B2
a_3 = 1; b_4 = 2
  c_5 = 3; n_6 = 6

B4
a_1 = \phi (1, a_7)
  if $a_1 \leq 6$

B3
a_7 = a_1 + 1

B5
if $a_1 \leq 11$

B6
D.1200_8 = a_1 + 2
  a_9 = D.1200_8 + 3

B7
a_2 = \phi (a_1, a_9)
  return $a_2$

{a_7, a_9}

{a_7, a_9}

F

{a_1, a_9}

∅ (Conservative assumption)

{a_1, a_9}

{a_1, a_9}

∅
Liveness Analysis of Variables: Iteration 2

Find out at each program point $p$, the variables that are used beyond $p$

```
B2
a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6
B4
a_1 = φ (1, a_7)
if a_1 ≤ 6

B3
a_7 = a_1 + 1
F
B5
if a_1 ≤ 11
B6
D.1200.8 = a_1 + 2
a_9 = D.1200.8 + 3
F
B7
a_2 = φ (a_1, a_9)
return a_2
∅ (Conservative assumption)
{a_7, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
∅
Liveness Analysis of Variables: Iteration 2

Find out at each program point $p$, the variables that are used beyond $p$.

- **B2**: $a_3 = 1; b_4 = 2$
  $c_5 = 3; n_6 = 6$

- **B4**: $a_1 = \phi(1, a_7)$
  if $a_1 \leq 6$

- **B3**: $a_7 = a_1 + 1$

- **B5**: if $a_1 \leq 11$

- **B6**: $D.1200.8 = a_1 + 2$
  $a_9 = D.1200.8 + 3$

- **B7**: $a_2 = \phi(a_1, a_9)$
  return $a_2$

- Variables used beyond $p$:
  - $\{a_7, a_9\}$
  - $\{a_1, a_9\}$
  - $\{a_7, a_9\}$
  - $\{a_1, a_9\}$
  - $\{a_7, a_9\}$
  - $\{a_7, a_9\}$
  - $\{a_1, a_9\}$
  - $\emptyset$
Using Liveness Analysis for Dead Code Elimination

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Using Liveness Analysis for Dead Code Elimination

- Values of \(a_3, a_4, c_5, \) and \(n_6\) are guaranteed not to be used.
Using Liveness Analysis for Dead Code Elimination

- Values of \( a_3, a_4, c_5, \) and \( n_6 \) are guaranteed not to be used.
Using Liveness Analysis for Dead Code Elimination

- Values of \( a_3, a_4, c_5, \) and \( n_6 \) are guaranteed not to be used
- Why are the values of \( a_7 \) and \( a_9 \) meaningful at the exit of B2?
Using Liveness Analysis for Dead Code Elimination

- Values of $a_3$, $a_4$, $c_5$, and $n_6$ are guaranteed not to be used.
- Why are the values of $a_7$ and $a_9$ meaningful at the exit of B2?
- We have assumed a $\phi$ function to be an ordinary expression in which operands are computed along every path reaching the computation.
Part 3

Live Variables Analysis
Defining Live Variables Analysis

A variable \( v \) is live at a program point \( p \), if some path from \( p \) to program exit contains an r-value occurrence of \( v \) which is not preceded by an l-value occurrence of \( v \).

\[
\begin{align*}
\text{Start} & \quad v = a \ast b \\
& \quad a = v + 2 \\
& \quad \text{End}
\end{align*}
\]

\[
\begin{align*}
\text{Start} & \quad v = a \ast b \\
& \quad v = a + 2 \\
& \quad \text{End}
\end{align*}
\]

\[
\begin{align*}
\text{Start} & \quad v = v + 2 \\
& \quad \text{End}
\end{align*}
\]
Defining Live Variables Analysis

A variable $v$ is live at a program point $p$, if some path from $p$ to program exit contains an r-value occurrence of $v$ which is not preceded by an l-value occurrence of $v$. 

$v$ is live at $p$
Defining Live Variables Analysis

A variable $v$ is live at a program point $p$, if some path from $p$ to program exit contains an r-value occurrence of $v$ which is not preceded by an l-value occurrence of $v$. 

$v$ is live at $p$

$v$ is not live at $p$

$v = a \times b$

$a = v + 2$

End

$v = a \times b$

$v = a + 2$

End

$v = v + 2$

End
Defining Live Variables Analysis

A variable $v$ is live at a program point $p$, if some path from $p$ to program exit contains an r-value occurrence of $v$ which is not preceded by an l-value occurrence of $v$.

$v$ is live at $p$

$v$ is not live at $p$

$v$ is live at $p$
Defining Live Variables Analysis

A variable \( v \) is live at a program point \( p \), if some path from \( p \) to program exit contains an r-value occurrence of \( v \) which is not preceded by an l-value occurrence of \( v \).

Path based specification

\[
v \text{ is live at } p
\]

\[
\begin{align*}
v &= a \ast b \\
a &= v + 2
\end{align*}
\]

\[
\begin{align*}
v &= a \ast b \\
v &= a + 2
\end{align*}
\]

\[
\begin{align*}
v &= v + 2
\end{align*}
\]
Defining Data Flow Analysis for Live Variables Analysis

\[
\begin{align*}
\text{Gen}_i &= \text{In}_i \cup \text{Gen}_j \cup \left( \text{Out}_k \setminus \text{Kill}_k \right) \\
\text{Gen}_k &= \text{In}_i \cup \text{In}_j \\
\text{Kill}_k &= \text{Out}_k \\
\text{Out}_k &= \text{In}_i \cup \text{In}_j
\end{align*}
\]
Defining Data Flow Analysis for Live Variables Analysis

Basic Blocks ≡ Single statements or Maximal groups of sequentially executed statements
Defining Data Flow Analysis for Live Variables Analysis

Basic Blocks ≡ Single statements or Maximal groups of sequentially executed statements

Control Transfer
Defining Data Flow Analysis for Live Variables Analysis

\[ \text{Gen}_k, \text{Kill}_k \]

\[ \text{Gen}_i, \text{Kill}_i \]

\[ \text{Gen}_j, \text{Kill}_j \]
Defining Data Flow Analysis for Live Variables Analysis

Local Data Flow Properties

Gen_k, Kill_k

Gen_i, Kill_i

Gen_j, Kill_j
Local Data Flow Properties for Live Variables Analysis

\[ Gen_n = \{ v \mid \text{variable } v \text{ is used in basic block } n \text{ and is not preceded by a definition of } v \} \]

\[ Kill_n = \{ v \mid \text{basic block } n \text{ contains a definition of } v \} \]
Local Data Flow Properties for Live Variables Analysis

Gen
\textsubscript{n} = \{ v \mid \text{variable } v \text{ is used in basic block } n \text{ and is not preceded by a definition of } v \}\}

Kill\textsubscript{n} = \{ v \mid \text{basic block } n \text{ contains a definition of } v \}\}

---

r-value occurrence

Value is only read, e.g. x,y,z in

\texttt{x.sum = y.data + z.data}
Local Data Flow Properties for Live Variables Analysis

- **r-value occurrence**: Value is only read, e.g., x, y, z in
  \[ x \text{.sum} = y \text{.data} + z \text{.data} \]

- **Value is modified**, e.g., y in
  \[ y = x \text{.lptr} \]

- **Gen** \(_n\) = \{ \(v\) | variable \(v\) is **used** in basic block \(n\) and is not **preceded** by a **definition** of \(v\) \}

- **Kill** \(_n\) = \{ \(v\) | basic block \(n\) **contains** a definition of \(v\) \}

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Local Data Flow Properties for Live Variables Analysis

**Gen**$_n$ = \{ $v$ | variable $v$ is used in basic block $n$ and is not preceded by a definition of $v$ \} \\
**Kill**$_n$ = \{ $v$ | basic block $n$ contains a definition of $v$ \}

Value is only read, e.g. $x,y,z$ in $x$.sum = $y$.data + $z$.data

Value is modified e.g. $y$ in $y = x$.lptr

r-value occurrence

l-value occurrence

within $n$
Local Data Flow Properties for Live Variables Analysis

- **r-value occurrence**: Value is only read, e.g., \( x, y, z \) in
  \[ x.\text{sum} = y.\text{data} + z.\text{data} \]

- **l-value occurrence**: Value is modified, e.g., \( y \) in
  \[ y = x.\text{lptr} \]

- **\( Gen_n \)**: Variable \( v \) is used in basic block \( n \) and is not preceded by a definition of \( v \)

- **\( Kill_n \)**: Basic block \( n \) contains a definition of \( v \)

\[
\begin{align*}
Gen_n &= \{ v \mid \text{variable } v \text{ is used in basic block } n \text{ and is not preceded by a definition of } v \} \\
Kill_n &= \{ v \mid \text{basic block } n \text{ contains a definition of } v \}
\end{align*}
\]
Local Data Flow Properties for Live Variables Analysis

- $Gen_n$: Use not preceded by definition

- $Kill_n$: Definition anywhere in a block
Local Data Flow Properties for Live Variables Analysis

- $Gen_n$: Use not preceded by definition
  - Upwards exposed use

- $Kill_n$: Definition anywhere in a block
  - Stop the effect from being propagated across a block
Defining Data Flow Analysis for Live Variables Analysis

\[ \text{In}_k = \text{Gen}_k \cup (\text{Out}_k - \text{Kill}_k) \]

\[ \text{Out}_k = \text{In}_i \cup \text{In}_j \]
Defining Data Flow Analysis for Live Variables Analysis

Global Data Flow Properties

\[ \text{\(In_k = Gen_k \cup (Out_k - Kill_k)\)} \]

\[ \text{\(Out_k = In_i \cup In_j\)} \]

\(\text{\(In_i\)}\)

\(\text{\(In_j\)}\)
Defining Data Flow Analysis for Live Variables Analysis

Global Data Flow Properties

\[ \text{In}_k = \text{Gen}_k \cup (\text{Out}_k - \text{Kill}_k) \]

\[ \text{Out}_k = \text{In}_i \cup \text{In}_j \]

Edge based specifications

- \( \text{In}_i \)
- \( \text{In}_j \)
Data Flow Equations For Live Variables Analysis

\[ \begin{align*}
In_n &= (Out_n - Kill_n) \cup Gen_n \\
Out_n &= \begin{cases} 
  BI & \text{n is End block} \\
  \bigcup_{s \in succ(n)} In_s & \text{otherwise}
\end{cases}
\end{align*} \]
Data Flow Equations For Live Variables Analysis

\[
\begin{align*}
\text{In}_n &= (\text{Out}_n - \text{Kill}_n) \cup \text{Gen}_n \\
\text{Out}_n &= \begin{cases} 
\text{BI} & \text{if } n \text{ is End block} \\
\bigcup_{s \in \text{succ}(n)} \text{In}_s & \text{otherwise}
\end{cases}
\end{align*}
\]

\text{In}_n \text{ and } \text{Out}_n \text{ are sets of variables.}
Performing Live Variables Analysis

\[
a_3 = 1; \ b_4 = 2 \\
c_5 = 3; \ n_6 = 6
\]

\[
a_1 = \phi \ (1, a_7) \\
\text{if} \ a_1 \leq 6
\]

\[
a_7 = a_1 + 1
\]

\[
\text{if} \ a_1 \leq 11
\]

\[
D.1200.8 = a_1 + 2 \\
a_9 = D.1200.8 + 3
\]

\[
a_2 = \phi \ (a_1, a_9) \\
\text{return} \ a_2
\]

---

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>$\emptyset$</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B3</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

```
a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6

a_1 = \phi (1, a_7)
if a_1 \leq 6

F  a_7 = a_1 + 1
B3

T  if a_1 \leq 11
B5

F  D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3
B6

T  a_2 = \phi (a_1, a_9)
return a_2
B7

F

Gen    Kill
B2    \emptyset    \{a_3, b_4, c_5, n_6\}
B4    \{a_7\}    \{a_1\}
B3    \{a_1\}    \{a_7\}
B5    \{a_1\}    \emptyset
B6    \{a_1\}    \{a_9\}
B7    \{a_1, a_9\}    \{a_2\}
```
Performing Live Variables Analysis

**Code Example:**

```c
a_3 = 1; b_4 = 2
a_1 = \phi (1, a_7)
if a_1 \leq 6
B2

B4

if a_1 \leq 11
B5

B3

a_7 = a_1 + 1
T

F

D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3
T

F

B6

B7

a_2 = \phi (a_1, a_9)
\{a_1, a_9\}
return a_2

\emptyset
```

**Live Variables Table:**

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>{}</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

\[
a_3 = 1; b_4 = 2 \\
c_5 = 3; n_6 = 6
\]

\[
a_1 = \phi (1, a_7) \\
\text{if } a_1 \leq 6
\]

\[
a_7 = a_1 + 1
\]

\[
\text{if } a_1 \leq 11
\]

\[
\text{T}
\]

\[
\text{F}
\]

\[
D.1200.8 = a_1 + 2 \\
a_9 = D.1200.8 + 3
\]

\[
\{a_1, a_9\}
\]

\[
a_2 = \phi (a_1, a_9) \\
\text{return } a_2
\]

\[
\{a_1, a_9\}
\]

\[
\emptyset
\]

---

### Table: Gen and Kill

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>\emptyset</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B3</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>\emptyset</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

```
1 = 1; b_4 = 2  
c_5 = 3; n_6 = 6
```

```
a_1 = \phi (1, a_7)  
if a_1 \leq 6
```

```
a_7 = a_1 + 1
```

```
if a_1 \leq 11
```

```
D.1200_8 = a_1 + 2  
a_9 = D.1200_8 + 3
```

```
a_2 = \phi (a_1, a_9)  
return a_2
```

---

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>\emptyset</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B3</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>\emptyset</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Performing Live Variables Analysis

```
a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6

a_1 = \phi (1, a_7)
if a_1 \leq 6

B4

if a_1 \leq 11
B5

{a_1, a_9}

T

F

B3

a_7 = a_1 + 1

{a_1, a_9}

B2

\emptyset

B4

\{a_7\}

\{a_1\}

B3

\{a_1\}

\{a_7\}

B5

\{a_1\}

\emptyset

B6

\{a_1\}

\{a_9\}

B7

\{a_1, a_9\}

\{a_2\}

\emptyset

D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3

{a_1}

B6

\{a_1\}

\{a_9\}

B7

\{a_1, a_9\}

\{a_2\}

\emptyset

return a_2

a_2 = \phi (a_1, a_9)

{a_1, a_9}

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Performing Live Variables Analysis

**Source Code:**

```c
a_3 = 1; b_4 = 2

if a_1 ≤ 6
    a_1 = φ(1, a_7)
    a_7 = a_1 + 1
else
    a_1 = φ(1, a_7)
    a_7 = a_1 + 1

if a_1 ≤ 11
    D.1200_8 = a_1 + 2
    a_9 = D.1200_8 + 3
else
    a_9 = D.1200_8 + 3

a_2 = φ(a_1, a_9)
return a_2
```

**Live Variables Analysis:**

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>∅</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B3</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>∅</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

\[ a_3 = 1; \quad b_4 = 2 \]
\[ c_5 = 3; \quad n_6 = 6 \]

\[ a_1 = \phi (1, a_7) \]
if \( a_1 \leq 6 \)

\[ a_7 = a_1 + 1 \]

\[ \{ a_1, a_9 \} \]

if \( a_1 \leq 11 \)

\[ D.1200.8 = a_1 + 2 \]
\[ a_9 = D.1200.8 + 3 \]

\[ a_2 = \phi (a_1, a_9) \]
return \( a_2 \)

\[ \{ a_1, a_9 \} \]

Gen | Kill
--- | ---
B2 | \( \emptyset \) | \( \{ a_3, b_4, c_5, n_6 \} \)
B4 | \( \{ a_7 \} \) | \( \{ a_1 \} \)
B3 | \( \{ a_1 \} \) | \( \{ a_7 \} \)
B5 | \( \{ a_1 \} \) | \( \emptyset \)
B6 | \( \{ a_1 \} \) | \( \{ a_9 \} \)
B7 | \( \{ a_1, a_9 \} \) | \( \{ a_2 \} \)
Performing Live Variables Analysis

\[ a_3 = 1; b_4 = 2 \]
\[ c_5 = 3; n_6 = 6 \]

\[ a_1 = \phi(1, a_7) \]
\[ \text{if } a_1 \leq 6 \]

\[ a_7 = a_1 + 1 \]
\[ \{a_1, a_9\} \]

\[ \{a_1, a_9\} \]
\[ \text{if } a_1 \leq 11 \]

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]

\[ a_2 = \phi(a_1, a_9) \]
\[ \text{return } a_2 \]

\[ \{a_1, a_9\} \]

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>{}</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B3</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>{}</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

```
a_3 = 1; b_4 = 2
b_4 = 2
```

```
c_5 = 3; n_6 = 6
```

```
a_1 = \phi (1, a_7)
if a_1 \leq 6
```

```
a_7 = a_1 + 1
```

```
a_1 \leq 11
```

```
a_7 = a_1 + 1
```

```
a_1 \leq 11
```

```
D.1200_8 = a_1 + 2
a_9 = D.1200_8 + 3
```

```
a_2 = \phi (a_1, a_9)
return a_2
```

```
\{a_7, a_9\}
\{a_1, a_9\}
\{a_1, a_9\}
\{a_1, a_9\}
\{a_1, a_9\}
\{a_1, a_9\}
\{a_1, a_9\}
```

```
B2
B3
B4
B5
B6
B7
```

```
\{a_3, b_4, c_5, n_6\}
\{a_1\}
\{a_7\}
\{a_1\}
\{a_1\}
\{a_9\}
\{a_2\}
```

---

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Performing Live Variables Analysis

\[
\begin{align*}
\{a_7, a_9\} & \quad B2 \\
\{a_7, a_9\} & \quad B4 \\
\{a_1, a_9\} & \quad if \ a_1 \leq 6 \\
\{a_1, a_9\} & \quad B3 \\
\{a_1, a_9\} & \quad if \ a_1 \leq 11 \\
\{a_1, a_9\} & \quad B5 \\
\{a_1, a_9\} & \quad F \\
\{a_1, a_9\} & \quad B6 \\
\{a_1, a_9\} & \quad T \\
\{a_1, a_9\} & \quad B7
\end{align*}
\]

\[
\begin{align*}
a_3 &= 1; \ b_4 = 2 \\
c_5 &= 3; \ n_6 = 6 \\
a_1 &= \phi (1, a_7) \\
if \ a_1 \leq 6 & \quad B4
\end{align*}
\]

\[
\begin{align*}
a_7 &= a_1 + 1 \\
if \ a_1 \leq 11 & \quad B5
\end{align*}
\]

\[
\begin{align*}
D.1200.8 &= a_1 + 2 \\
a_9 &= D.1200.8 + 3 \\
a_2 &= \phi (a_1, a_9) & \quad B7
\end{align*}
\]

\[
\begin{array}{c|c|c}
& Gen & Kill \\
B2 & \emptyset & \{a_3, b_4, c_5, n_6\} \\
B4 & \{a_7\} & \{a_1\} \\
B3 & \{a_1\} & \{a_7\} \\
B5 & \{a_1\} & \emptyset \\
B6 & \{a_1\} & \{a_9\} \\
B7 & \{a_1, a_9\} & \{a_2\}
\end{array}
\]
Performing Live Variables Analysis

\[
\begin{align*}
\{a_7, a_9\} & \quad a_3 = 1; b_4 = 2 \\
\{a_7, a_9\} & \quad c_5 = 3; n_6 = 6 \\
\{a_7, a_9\} & \quad a_1 = \phi (1, a_7) \\
\{a_1, a_9\} & \quad \text{if } a_1 \leq 6 \\
\{a_1, a_9\} & \quad a_7 = a_1 + 1 \\
\{a_1, a_9\} & \quad \text{if } a_1 \leq 11 \\
\{a_1, a_9\} & \quad D.1200.8 = a_1 + 2 \\
\{a_1, a_9\} & \quad a_9 = D.1200.8 + 3 \\
\{a_1, a_9\} & \quad a_2 = \phi (a_1, a_9) \\
\{a_1, a_9\} & \quad \text{return } a_2
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th>Kill</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>\emptyset</td>
<td>{a_3, b_4, c_5, n_6}</td>
</tr>
<tr>
<td>B4</td>
<td>{a_7}</td>
<td>{a_1}</td>
</tr>
<tr>
<td>B3</td>
<td>{a_1}</td>
<td>{a_7}</td>
</tr>
<tr>
<td>B5</td>
<td>{a_1}</td>
<td>\emptyset</td>
</tr>
<tr>
<td>B6</td>
<td>{a_1}</td>
<td>{a_9}</td>
</tr>
<tr>
<td>B7</td>
<td>{a_1, a_9}</td>
<td>{a_2}</td>
</tr>
</tbody>
</table>
Performing Live Variables Analysis

\{a_7, a_9\} \quad a_3 = 1; b_4 = 2
c_5 = 3; n_6 = 6

B2

\{a_7, a_9\} \quad a_1 = \phi (1, a_7)
if \ a_1 \leq 6

B4

\{a_7, a_9\} \quad a_7 = a_1 + 1

\{a_1\}

B3

\{a_1, a_9\} \quad \text{if } a_1 \leq 11

B5

\{a_7, a_9\} \quad D.1200_8 = a_1 + 2
\quad a_9 = D.1200_8 + 3

\{a_1\}

B6

\{a_1, a_9\} \quad a_2 = \phi (a_1, a_9)
return a_2

B7

\{a_1, a_9\}

\{a_1\}

\{a_1, a_9\}

\{a_7\}

\{a_1\}

\{a_7\}

\{a_1\}

\{a_9\}

\{a_2\}

\{a_3, b_4, c_5, n_6\}

\{a_1\}

\{a_7\}

\{a_9\}

\{a_2\}

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Performing Live Variables Analysis

\[ \{a_7, a_9\} \quad a_3 = 1; b_4 = 2 \]
\[ c_5 = 3; n_6 = 6 \]

\[ \{a_7, a_9\} \quad a_1 = \phi (1, a_7) \]
if \( a_1 \leq 6 \)

\[ \{a_1, a_9\} \quad a_7 = a_1 + 1 \]
if \( a_1 \leq 11 \)

\[ \{a_1, a_9\} \quad D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]

\[ \{a_1, a_9\} \quad a_2 = \phi (a_1, a_9) \]
return \( a_2 \)

\[
\begin{array}{|c|c|c|}
\hline
\text{Gen} & \text{Kill} \\
\hline
\text{B2} & \emptyset & \{a_3, b_4, c_5, n_6\} \\
\hline
\text{B4} & \{a_7\} & \{a_1\} \\
\hline
\text{B3} & \{a_1\} & \{a_7\} \\
\hline
\text{B5} & \{a_1\} & \emptyset \\
\hline
\text{B6} & \{a_1\} & \{a_9\} \\
\hline
\text{B7} & \{a_1, a_9\} & \{a_2\} \\
\hline
\end{array}
\]
Strongly Live Variables Analysis

A variable \( v \) is strongly live if it is used in

- in statement other than assignment statement, or
  (this case is same as simple liveness analysis)

- in defining other strongly live variables in an assignment statement
  (this case is different from simple liveness analysis)
Understanding Strong Liveness

```
y = x
print (x)
```

```
y = x
print (y)
```

```
y = x
print (z)
```
Understanding Strong Liveness

Strong Liveness

\[ y = x \]

print \((x)\)

\[ y = x \]

print \((y)\)

\[ y = x \]

print \((z)\)
Understanding Strong Liveness

Strong Liveness

\[
y = x
\]

\[
\text{print} (x)
\]

\[
\emptyset
\]
Understanding Strong Liveness

\[ y = x \]

\{ x \}

print (x)

\[ y = x \]

print (y)

\[ y = x \]

print (z)

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Understanding Strong Liveness

\[ y = x \]
\[ \{x\} \]
\[ \text{print (x)} \]
\[ \emptyset \]

\[ y = x \]
\[ \{x\} \]
\[ \text{print (y)} \]

\[ y = x \]
\[ \text{print (z)} \]
Understanding Strong Liveness

Simple Liveness  Strong Liveness

\{x\} \downarrow \quad \{x\} \downarrow
\quad y = x \quad y = x

\{x\} \downarrow \quad \{x\} \downarrow
\quad \text{print}(x) \quad \text{print}(y)

\emptyset \downarrow \quad \emptyset \downarrow

y = x
\text{print}(z)
Understanding Strong Liveness

**Simple Liveness**
- Initial state: \{x\}
- Transition: \( y = x \)
- Final state: \{x\}
- Print: \( \text{print}(x) \)
- Final state: \emptyset

**Strong Liveness**
- Initial state: \{x\}
- Transition: \( y = x \)
- Final state: \{x\}
- Print: \( \text{print}(x) \)
- Final state: \emptyset

**Strong Liveness**
- Initial state: \{x\}
- Transition: \( y = x \)
- Final state: \{x\}
- Print: \( \text{print}(y) \)
- Final state: \emptyset

**Strong Liveness**
- Initial state: \{x\}
- Transition: \( y = x \)
- Final state: \{x\}
- Print: \( \text{print}(z) \)
- Final state: \emptyset
Understanding Strong Liveness

Simple Liveness

\{x\} \rightarrow y = x \rightarrow \{x\} \rightarrow \emptyset

\{x\} \rightarrow \text{print}(x) \rightarrow \emptyset

Strong Liveness

\{x\} \rightarrow y = x \rightarrow \{x\} \rightarrow \emptyset

\{x\} \rightarrow \text{print}(y) \rightarrow \emptyset

Strong Liveness

\{x\} \rightarrow y = x \rightarrow \{x\} \rightarrow \emptyset

\{x\} \rightarrow \text{print}(z) \rightarrow \emptyset

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Understanding Strong Liveness

Simple Liveness

\{x\} \downarrow \{x\}

\text{\(y = x\)}

\{x\} \downarrow \{x\}

\text{print (x)}

\emptyset \downarrow \emptyset

Strong Liveness

\emptyset \downarrow \emptyset

\text{\(y = x\)}

\emptyset \downarrow \emptyset

\text{print (y)}

\{y\} \downarrow \emptyset

Strong Liveness

\text{\(y = x\)}

\emptyset \downarrow \emptyset

\text{print (z)}
Understanding Strong Liveness

Simple Liveness

\[ \{x\} \rightarrow \{x\} \rightarrow \text{print}(x) \rightarrow \emptyset \]

\[ \{x\} \rightarrow \{x\} \rightarrow \text{print}(x) \rightarrow \emptyset \]

Strong Liveness

\[ \{x\} \rightarrow \{x\} \rightarrow \text{print}(y) \rightarrow \emptyset \]

\[ \{x\} \rightarrow \{x\} \rightarrow \text{print}(y) \rightarrow \emptyset \]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Understanding Strong Liveness

Simple Liveness | Strong Liveness
---|---
\{x\} | \{x\}
\{x\} | \{x\}
\text{print (x)} | \text{print (x)}
\emptyset | \emptyset

Simple Liveness | Strong Liveness
---|---
\{x\} | \{x\}
\{x\} | \{y\}
\text{print (y)} | \text{print (y)}
\emptyset | \emptyset

\text{y = x} | \text{y = x}
\text{y = x} | \text{y = x}
\text{y = x} | \text{y = x}
\text{print (z)} | \text{print (z)}
Understanding Strong Liveness

Simple Liveness  |  Strong Liveness  |  Simple Liveness  |  Strong Liveness  |  Strong Liveness
\{x\}  |  \{x\}  |  \{x\}  |  \{x\}  |  \{y\}  |  \{y\}  
\(y = x\)  |  \(y = x\)  |  \(y = x\)  |  \(y = x\)  |  
\{x\}  |  \{x\}  |  \{x\}  |  \{x\}  |  \{y\}  |  \{y\}  
print (x)  |  print (y)  |  print (z)  |  
\(\emptyset\)  |  \(\emptyset\)  |  \(\emptyset\)  |  

Essential Abstractions in GCC  GCC Resource Center, IIT Bombay
Understanding Strong Liveness

Simple Liveness  |  Strong Liveness
\{x\}  |  \{x\}
\{x\}  |  \{x\}
\text{print (x)}  |  \text{print (x)}
\emptyset  |  \emptyset

Simple Liveness  |  Strong Liveness
\{x\}  |  \{x\}
\{y\}  |  \{y\}
\text{print (y)}  |  \text{print (y)}
\emptyset  |  \emptyset

Strong Liveness
\text{print (z)}
\emptyset

Essential Abstractions in GCC  |  GCC Resource Center, IIT Bombay
Understanding Strong Liveness

Simple Liveness | Strong Liveness
---|---
$\{x\}$ | $\{x\}$
$\{x\}$ | $\{x\}$
$\text{print}(x)$ | $\text{print}(y)$
$\emptyset$ | $\emptyset$

Simple Liveness | Strong Liveness
---|---
$\{x\}$ | $\{x\}$
$\{y\}$ | $\{y\}$
$\text{print}(y)$ | $\text{print}(z)$
$\emptyset$ | $\emptyset$

Strong Liveness
$y = x$
$\{z\}$
$\emptyset$
Understanding Strong Liveness

Simple Liveness | Strong Liveness
---|---
{x} | {x}
\( y = x \) | \( y = x \)
\( \{x\} \) | \( \{x\} \)
\( \text{print} (x) \) | \( \text{print} (y) \)
\( \emptyset \) | \( \emptyset \)

Simple Liveness | Strong Liveness
---|---
\( y = x \) | \( y = x \)
\( \{x\} \) | \( \{x\} \)
\( \{y\} \) | \( \{y\} \)
\( \emptyset \) | \( \emptyset \)

Strong Liveness
\( y = x \)
\( \{z\} \)
\( \text{print} (z) \)
Understanding Strong Liveness

Simple Liveness | Strong Liveness
---|---
\{x\} | \{x\}
\{x\} | \{x\}
\(y = x\) | \(y = x\)
\(\text{print } (x)\) | \(\text{print } (y)\)
\(\emptyset\) | \(\emptyset\)

Simple Liveness | Strong Liveness
---|---
\{x\} | \{x\}
\{y\} | \{y\}
\(y = x\) | \(y = x\)
\(\text{print } (y)\) | \(\text{print } (z)\)
\(\emptyset\) | \(\emptyset\)

Simple Liveness | Strong Liveness
---|---
\{z, x\} | \{z\}
\{z\} | \{z\}
\(y = x\) | \(\text{print } (z)\)
\(\emptyset\) | \(\emptyset\)
Understanding Strong Liveness

Simple Liveness
\{x\} → \{x\}
\{x\} → \{x\}

Strong Liveness
\{x\} → \{x\}
\{x\} → \{x\}

Simple Liveness
\{x\} → \{y\}
\{y\} → \{y\}

Strong Liveness
\{x\} → \{x\}
\{x\} → \{x\}

Simple Liveness
\{z, x\} → \{z\}
\{z\} → \{z\}

Strong Liveness
\{z\} → \{z\}
\{z\} → \{z\}

Same

\( y = x \)

\( \text{print} \ (x) \)

\( y = x \)

\( \text{print} \ (y) \)

\( y = x \)

\( \text{print} \ (z) \)
Understanding Strong Liveness

Simple Liveness | Strong Liveness
---|---
{\(x\)} | {\(x\)}
\(y = x\) | \(y = x\)
{\(x\)} | {\(x\)}
print (\(x\)) | print (\(y\))
\(\emptyset\) | \(\emptyset\)

Same

Simple Liveness | Strong Liveness
---|---
{\(x\)} | {\(x\)}
\(y = x\) | \(y = x\)
{\(y\)} | {\(y\)}
print (\(y\)) | print (\(z\))
\(\emptyset\) | \(\emptyset\)

Same

Simple Liveness | Strong Liveness
---|---
{\(z, x\)} | {\(z\)}
\(y = x\) | \(y = x\)
{\(z\)} | {\(z\)}
print (\(z\)) | print (\(z\))
\(\emptyset\) | \(\emptyset\)
Understanding Strong Liveness

<table>
<thead>
<tr>
<th>Simple Liveness</th>
<th>Strong Liveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>{x}</td>
<td>{x}</td>
</tr>
<tr>
<td>{x}</td>
<td>{x}</td>
</tr>
<tr>
<td>{x}</td>
<td>{x}</td>
</tr>
<tr>
<td>{x}</td>
<td>{x}</td>
</tr>
<tr>
<td>y = x</td>
<td>y = x</td>
</tr>
<tr>
<td>print (x)</td>
<td>print (y)</td>
</tr>
<tr>
<td>∅</td>
<td>∅</td>
</tr>
<tr>
<td>∅</td>
<td>∅</td>
</tr>
</tbody>
</table>

Same

<table>
<thead>
<tr>
<th>Simple Liveness</th>
<th>Strong Liveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>{x}</td>
<td>{x}</td>
</tr>
<tr>
<td>{x}</td>
<td>{x}</td>
</tr>
<tr>
<td>{y}</td>
<td>{y}</td>
</tr>
<tr>
<td>{y}</td>
<td>{y}</td>
</tr>
<tr>
<td>y = x</td>
<td>y = x</td>
</tr>
<tr>
<td>print (y)</td>
<td>print (z)</td>
</tr>
<tr>
<td>∅</td>
<td>∅</td>
</tr>
<tr>
<td>∅</td>
<td>∅</td>
</tr>
</tbody>
</table>

Same

<table>
<thead>
<tr>
<th>Simple Liveness</th>
<th>Strong Liveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>{z, x}</td>
<td>{z}</td>
</tr>
<tr>
<td>{z}</td>
<td>{z}</td>
</tr>
<tr>
<td>{z}</td>
<td>{z}</td>
</tr>
<tr>
<td>{z}</td>
<td>{z}</td>
</tr>
<tr>
<td>y = x</td>
<td>y = x</td>
</tr>
<tr>
<td>print (z)</td>
<td>print (z)</td>
</tr>
<tr>
<td>∅</td>
<td>∅</td>
</tr>
<tr>
<td>∅</td>
<td>∅</td>
</tr>
</tbody>
</table>

Different
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

- \( \{a_7, a_9\} \)
  - \( a_3 = 1; b_4 = 2 \)
  - \( c_5 = 3; n_6 = 6 \)

- \( \{a_7, a_9\} \)
  - \( a_1 = \phi(1, a_7) \)
  - if \( a_1 \leq 6 \)

- \( \{a_7, a_9\} \)
  - \( a_7 = a_1 + 1 \)
  - if \( a_1 \leq 11 \)

- \( \{a_1, a_9\} \)
  - \( D.1200_8 = a_1 + 2 \)
  - \( a_9 = D.1200_8 + 3 \)
  - print "Hello"

- \( \{a_1, a_9\} \)
  - \( a_2 = \phi(a_1, a_9) \)
  - print "Hi"

**Strong Liveness**

- \( \{a_7, a_9\} \)
  - \( a_3 = 1; b_4 = 2 \)
  - \( c_5 = 3; n_6 = 6 \)

- \( \{a_7, a_9\} \)
  - \( a_1 = \phi(1, a_7) \)
  - if \( a_1 \leq 6 \)

- \( \{a_7, a_9\} \)
  - \( a_7 = a_1 + 1 \)
  - if \( a_1 \leq 11 \)

- \( \{a_1, a_9\} \)
  - \( D.1200_8 = a_1 + 2 \)
  - \( a_9 = D.1200_8 + 3 \)
  - print "Hello"

- \( \{a_1, a_9\} \)
  - \( a_2 = \phi(a_1, a_9) \)
  - print "Hi"
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[
\begin{align*}
\{a_7, a_9\} & : \quad a_3 = 1; \quad b_4 = 2 \\
& \quad c_5 = 3; \quad n_6 = 6 \\
\{a_7, a_9\} & : \quad a_1 = \phi (1, a_7) \\
& \quad \text{if } a_1 \leq 6 \\
\{a_1, a_9\} & : \quad a_7 = a_1 + 1 \\
\{a_1, a_9\} & : \quad \text{if } a_1 \leq 11 \\
\{a_1, a_9\} & : \quad D.1200_8 = a_1 + 2 \\
& \quad a_9 = D.1200_8 + 3 \\
& \quad \text{print } "Hello" \\
\{a_1, a_9\} & : \quad a_2 = \phi (a_1, a_9) \\
& \quad \text{print } "Hi" \\
\{a_1, a_9\} & : \quad \emptyset
\end{align*}
\]

**Strong Liveness**

\[
\begin{align*}
\{a_7, a_9\} & : \quad a_3 = 1; \quad b_4 = 2 \\
& \quad c_5 = 3; \quad n_6 = 6 \\
\{a_7, a_9\} & : \quad a_1 = \phi (1, a_7) \\
& \quad \text{if } a_1 \leq 6 \\
\{a_7, a_9\} & : \quad a_7 = a_1 + 1 \\
\{a_1, a_9\} & : \quad \text{if } a_1 \leq 11 \\
\{a_1, a_9\} & : \quad D.1200_8 = a_1 + 2 \\
& \quad a_9 = D.1200_8 + 3 \\
& \quad \text{print } "Hello" \\
\{a_1, a_9\} & : \quad a_2 = \phi (a_1, a_9) \\
& \quad \text{print } "Hi" \\
\{a_1, a_9\} & : \quad \emptyset
\end{align*}
\]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[ \{a_7, a_9\} \]

\[ a_3 = 1; \ b_4 = 2 \]
\[ c_5 = 3; \ n_6 = 6 \]

\[ \{a_7, a_9\} \]

\[ a_1 = \phi (1, a_7) \]
\[ \text{if } a_1 \leq 6 \]

\[ \{a_1, a_9\} \]

\[ a_7 = a_1 + 1 \]

\[ \{a_7, a_9\} \]

\[ \text{if } a_1 \leq 11 \]

\[ \{a_1, a_9\} \]

\[ \{a_1, a_9\} \]

\[ a_2 = \phi (a_1, a_9) \]
\[ \text{print } "Hi" \]

\[ \{a_1, a_9\} \]

**Strong Liveness**

\[ \{a_7, a_9\} \]

\[ a_3 = 1; \ b_4 = 2 \]
\[ c_5 = 3; \ n_6 = 6 \]

\[ \{a_7, a_9\} \]

\[ a_1 = \phi (1, a_7) \]
\[ \text{if } a_1 \leq 6 \]

\[ \{a_1, a_9\} \]

\[ a_7 = a_1 + 1 \]

\[ \{a_7, a_9\} \]

\[ \text{if } a_1 \leq 11 \]

\[ \{a_1, a_9\} \]

\[ a_2 = \phi (a_1, a_9) \]
\[ \text{print } "Hi" \]

\[ \{a_1, a_9\} \]

\[ \{a_1, a_9\} \]

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]
\[ \text{print } "Hello" \]

\[ \{a_1\} \]

\[ \{a_1, a_9\} \]

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]
\[ \text{print } "Hello" \]

\[ \{a_1\} \]

\[ \{a_1, a_9\} \]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

- \{a_7, a_9\}
  - if a_1 \leq 6
    - a_7 = a_1 + 1
    - \{a_7, a_9\}

- \{a_7, a_9\}
  - if a_1 \leq 6
    - a_7 = a_1 + 1
    - \{a_7, a_9\}

- \{a_1, a_9\}
  - D.1200_8 = a_1 + 2
  - a_9 = D.1200_8 + 3
  - print "Hello"
  - \{a_1, a_9\}

- \{a_1, a_9\}
  - a_2 = \phi (a_1, a_9)
  - print "Hi"
  - \{a_1, a_9\}

**Strong Liveness**

- \{a_7, a_9\}
  - if a_1 \leq 6
    - a_7 = a_1 + 1
    - \{a_7, a_9\}

- \{a_7, a_9\}
  - if a_1 \leq 6
    - a_7 = a_1 + 1
    - \{a_7, a_9\}

- \{a_1, a_9\}
  - D.1200_8 = a_1 + 2
  - a_9 = D.1200_8 + 3
  - print "Hello"
  - \{a_1, a_9\}

- \{a_1, a_9\}
  - a_2 = \phi (a_1, a_9)
  - print "Hi"
  - \{a_1, a_9\}
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[ \{a_7, a_9\} \]

\[ a_3 = 1; \, b_4 = 2 \]
\[ c_5 = 3; \, n_6 = 6 \]

\[ a_1 = \phi (1, a_7) \]

if \( a_1 \leq 6 \)

\[ B4 \]

\[ \{a_1, a_9\} \]

\[ a_7 = a_1 + 1 \]

if \( a_1 \leq 11 \)

\[ B3 \]

\[ a_1 \leq 11 \]

\[ B5 \]

\[ \{a_7, a_9\} \]

\[ \{a_1, a_9\} \]

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]

print "Hello"

\[ B6 \]

\[ \{a_1\} \]

\[ \{a_1, a_9\} \]

\[ a_2 = \phi (a_1, a_9) \]

print "Hi"

\[ B7 \]

\[ \{a_1, a_9\} \]

\[ \emptyset \]

**Strong Liveness**

\[ \{a_7, a_9\} \]

\[ a_3 = 1; \, b_4 = 2 \]
\[ c_5 = 3; \, n_6 = 6 \]

\[ a_1 = \phi (1, a_7) \]

if \( a_1 \leq 6 \)

\[ B4 \]

\[ \{a_1, a_9\} \]

\[ a_7 = a_1 + 1 \]

if \( a_1 \leq 11 \)

\[ B3 \]

\[ a_1 \leq 11 \]

\[ B5 \]

\[ \{a_7, a_9\} \]

\[ \{a_1, a_9\} \]

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]

print "Hello"

\[ B6 \]

\[ \{a_1\} \]

\[ \{a_1, a_9\} \]

\[ a_2 = \phi (a_1, a_9) \]

print "Hi"

\[ B7 \]

\[ \{a_1, a_9\} \]

\[ \emptyset \]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

```
{a_7, a_9}  a_3 = 1; b_4 = 2
{a_7, a_9}  c_5 = 3; n_6 = 6

{a_7, a_9}  a_1 = \phi (1, a_7)
            if a_1 \leq 6

F  B3  a_7 = a_1 + 1
T  B3  \{a_1, a_9\}
            {a_7, a_9}

if a_1 \leq 11
B5  \{a_1, a_9\}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
B7  a_2 = \phi (a_1, a_9)
    print "Hi"
\{a_1, a_9\}
```

**Strong Liveness**

```
{a_7, a_9}  a_3 = 1; b_4 = 2
{a_7, a_9}  c_5 = 3; n_6 = 6

{a_7, a_9}  a_1 = \phi (1, a_7)
            if a_1 \leq 6

F  B3  a_7 = a_1 + 1
T  B4  \{a_1, a_9\}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
{a_1, a_9}
B7  a_2 = \phi (a_1, a_9)
    print "Hi"
\{a_1, a_9\}
```

Essential Abstractions in GCC  
GCC Resource Center, IIT Bombay
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[ \{a_7, a_9\} \]

\[ a_3 = 1; b_4 = 2; c_5 = 3; n_6 = 6 \]

**B2**

\[ \{a_7, a_9\} \]

\[ a_1 = \phi (1, a_7) \]

if \( a_1 \leq 6 \)

**B4**

\[ \{a_1, a_9\} \]

\[ a_7 = a_1 + 1 \]

**B3**

\[ \{a_7, a_9\} \]

if \( a_1 \leq 11 \)

**B5**

\[ \{a_1, a_9\} \]

\[ D.1200_8 = a_1 + 2 \]

\[ a_9 = D.1200_8 + 3 \]

print "Hello"

**B6**

\[ \{a_1, a_9\} \]

\[ a_2 = \phi (a_1, a_9) \]

\[ \{a_1, a_9\} \]

print "Hi"

**B7**

\[ \{a_1, a_9\} \]

\[ \{a_7, a_9\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1, a_9\} \]

\[ \{a_7, a_9\} \]

\[ \{a_1, a_9\} \]

\[ \{a_1, a_9\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

**Strong Liveness**

\[ \{a_7, a_9\} \]

\[ a_3 = 1; b_4 = 2; c_5 = 3; n_6 = 6 \]

**B2**

\[ \{a_7, a_9\} \]

\[ a_1 = \phi (1, a_7) \]

if \( a_1 \leq 6 \)

**B4**

\[ \{a_1, a_9\} \]

\[ a_7 = a_1 + 1 \]

**B3**

\[ \{a_7, a_9\} \]

if \( a_1 \leq 11 \)

**B5**

\[ \{a_1, a_9\} \]

\[ D.1200_8 = a_1 + 2 \]

\[ a_9 = D.1200_8 + 3 \]

print "Hello"

**B6**

\[ \{a_1, a_9\} \]

\[ a_2 = \phi (a_1, a_9) \]

\[ \{a_1, a_9\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]

\[ \{a_1\} \]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[
\{a_7, a_9\} \quad a_3 = 1; \ b_4 = 2 \\
c_5 = 3; \ n_6 = 6
\]

\[
\{a_7, a_9\} \quad a_1 = \phi (1, a_7) \\
\text{if } a_1 \leq 6
\]

\[
\{a_7, a_9\} \quad a_7 = a_1 + 1
\]

\[
\text{if } a_1 \leq 11
\]

\[
\{a_7, a_9\} \quad D.1200.8 = a_1 + 2 \\
a_9 = D.1200.8 + 3 \\
\text{print } "Hello"
\]

\[
\{a_1, a_9\} \quad a_2 = \phi (a_1, a_9) \\
\text{print } "Hi"
\]

**Strong Liveness**

\[
\{a_7, a_9\} \quad a_3 = 1; \ b_4 = 2 \\
c_5 = 3; \ n_6 = 6
\]

\[
\{a_7, a_9\} \quad a_1 = \phi (1, a_7) \\
\text{if } a_1 \leq 6
\]

\[
\{a_7, a_9\} \quad a_7 = a_1 + 1
\]

\[
\text{if } a_1 \leq 11
\]

\[
\{a_7, a_9\} \quad D.1200.8 = a_1 + 2 \\
a_9 = D.1200.8 + 3 \\
\text{print } "Hello"
\]

\[
\{a_1, a_9\} \quad a_2 = \phi (a_1, a_9) \\
\text{print } "Hi"
\]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[
\begin{align*}
\{a_{7}, a_{9}\} & \quad a_{3} = 1; b_{4} = 2 \\
\{a_{7}, a_{9}\} & \quad c_{5} = 3; n_{6} = 6 \\
\{a_{7}, a_{9}\} & \quad a_{1} = \phi (1, a_{7}) \\
\{a_{1}, a_{9}\} & \quad \text{if } a_{1} \leq 6 \\
\{a_{1}, a_{9}\} & \quad a_{7} = a_{1} + 1 \\
{a_{1}, a_{9}\} & \quad \text{if } a_{1} \leq 11 \\
{a_{1}, a_{9}\} & \quad D.1200\_8 = a_{1} + 2 \\
{a_{1}, a_{9}\} & \quad a_{9} = D.1200\_8 + 3 \\
{a_{1}, a_{9}\} & \quad \text{print "Hello"} \\
{a_{1}, a_{9}\} & \quad a_{2} = \phi (a_{1}, a_{9}) \\
{a_{1}, a_{9}\} & \quad \text{print "Hi"}
\end{align*}
\]

**Strong Liveness**

\[
\begin{align*}
\{a_{7}, a_{9}\} & \quad a_{3} = 1; b_{4} = 2 \\
\{a_{7}, a_{9}\} & \quad c_{5} = 3; n_{6} = 6 \\
\{a_{7}, a_{9}\} & \quad a_{1} = \phi (1, a_{7}) \\
\{a_{1}, a_{9}\} & \quad \text{if } a_{1} \leq 6 \\
\{a_{1}, a_{9}\} & \quad a_{7} = a_{1} + 1 \\
{a_{1}, a_{9}\} & \quad \text{if } a_{1} \leq 11 \\
{a_{1}, a_{9}\} & \quad D.1200\_8 = a_{1} + 2 \\
{a_{1}, a_{9}\} & \quad a_{9} = D.1200\_8 + 3 \\
{a_{1}, a_{9}\} & \quad \text{print "Hello"} \\
{a_{1}, a_{9}\} & \quad a_{2} = \phi (a_{1}, a_{9}) \\
{a_{1}, a_{9}\} & \quad \text{print "Hi"}
\end{align*}
\]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

\[
\begin{align*}
\{a_7, a_9\} & \quad a_3 = 1; b_4 = 2 \\
\{a_7, a_9\} & \quad c_5 = 3; n_6 = 6 \\
\{a_7, a_9\} & \quad a_1 = \phi (1, a_7) \\
\{a_1, a_9\} & \quad \text{if } a_1 \leq 6 \\
\{a_1, a_9\} & \quad a_7 = a_1 + 1 \\
\{a_1, a_9\} & \quad \text{if } a_1 \leq 11 \\
\{a_1, a_9\} & \quad D.1200_8 = a_1 + 2 \\
\{a_1, a_9\} & \quad a_9 = D.1200_8 + 3 \\
\{a_1, a_9\} & \quad \text{print } "Hello" \\
\{a_1, a_9\} & \quad a_2 = \phi (a_1, a_9) \\
\{a_1, a_9\} & \quad \text{print } "Hi" \\
\{a_7, a_9\} & \quad B7
\end{align*}
\]

**Strong Liveness**

\[
\begin{align*}
\{a_7, a_9\} & \quad a_3 = 1; b_4 = 2 \\
\{a_7, a_9\} & \quad c_5 = 3; n_6 = 6 \\
\{a_7, a_9\} & \quad a_1 = \phi (1, a_7) \\
\{a_1\} & \quad \text{if } a_1 \leq 6 \\
\{a_1\} & \quad a_7 = a_1 + 1 \\
\{a_1\} & \quad \text{if } a_1 \leq 11 \\
\{a_1\} & \quad D.1200_8 = a_1 + 2 \\
\{a_1\} & \quad a_9 = D.1200_8 + 3 \\
\{a_1\} & \quad \text{print } "Hello" \\
\{a_1\} & \quad a_2 = \phi (a_1, a_9) \\
\{a_1\} & \quad \text{print } "Hi" \\
\{a_1\} & \quad B7
\end{align*}
\]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

- \( \{a_7, a_9\} \)
- \( a_3 = 1; b_4 = 2 \)
- \( c_5 = 3; n_6 = 6 \)

\[ a_1 = \phi (1, a_7) \]

if \( a_1 \leq 6 \)

- \( \{a_7, a_9\} \)

\[ a_7 = a_1 + 1 \]

if \( a_1 \leq 11 \)

- \( \{a_1, a_9\} \)

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]

print "Hello"

- \( \{a_1, a_9\} \)

\[ a_2 = \phi (a_1, a_9) \]

print "Hi"

**Strong Liveness**

- \( \{a_7\} \)
- \( a_3 = 1; b_4 = 2 \)
- \( c_5 = 3; n_6 = 6 \)

\[ a_1 = \phi (1, a_7) \]

if \( a_1 \leq 6 \)

- \( \{a_7\} \)

\[ a_7 = a_1 + 1 \]

if \( a_1 \leq 11 \)

- \( \{a_1\} \)

\[ D.1200_8 = a_1 + 2 \]
\[ a_9 = D.1200_8 + 3 \]

print "Hello"

- \( \{a_1, a_9\} \)

\[ a_2 = \phi (a_1, a_9) \]

print "Hi"
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

- Initial state: \( \{a_7, a_9\} \)
- Transition 1: \( a_3 = 1; b_4 = 2 \) if \( a_1 \leq 6 \)
- Transition 2: \( a_7 = a_1 + 1 \) if \( a_1 \leq 11 \)
- Transition 3: \( D.1200_8 = a_1 + 2 \) if \( a_1 \leq 6 \)
- Transition 4: \( a_9 = D.1200_8 + 3 \) if \( a_1 \leq 6 \)
- Final state: \( \{a_1, a_9\} \)

**Strong Liveness**

- Initial state: \( \{a_7\} \)
- Transition 1: \( a_3 = 1; b_4 = 2 \) if \( a_1 \leq 6 \)
- Transition 2: \( a_7 = a_1 + 1 \) if \( a_1 \leq 11 \)
- Transition 3: \( D.1200_8 = a_1 + 2 \) if \( a_1 \leq 6 \)
- Transition 4: \( a_9 = D.1200_8 + 3 \) if \( a_1 \leq 6 \)
- Final state: \( \emptyset \)
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

1. \{a_7, a_9\}
2. \{a_7, a_9\}
3. \{a_7, a_9\}
4. \{a_7, a_9\}
5. \{a_1, a_9\}
6. \{a_1, a_9\}

\[
\begin{align*}
a_3 &= 1; \ b_4 = 2 \\
c_5 &= 3; \ n_6 = 6
\end{align*}
\]

\[
\begin{align*}
a_1 &= \phi(1, a_7) \\
\text{if } a_1 &\leq 6
\end{align*}
\]

\[
\begin{align*}
a_7 &= a_1 + 1 \\
\text{if } a_1 &\leq 11
\end{align*}
\]

\[
\begin{align*}
D.1200_8 &= a_1 + 2 \\
a_9 &= D.1200_8 + 3 \\
\text{print } "Hello"
\end{align*}
\]

\[
\begin{align*}
a_2 &= \phi(a_1, a_9) \\
\text{print } "Hi"
\end{align*}
\]

**Strong Liveness**

1. \{a_7\}
2. \{a_7\}
3. \{a_7\}
4. \{a_7\}
5. \{a_7\}
6. \{a_7\}

\[
\begin{align*}
a_3 &= 1; \ b_4 = 2 \\
c_5 &= 3; \ n_6 = 6
\end{align*}
\]

\[
\begin{align*}
a_1 &= \phi(1, a_7) \\
\text{if } a_1 &\leq 6
\end{align*}
\]

\[
\begin{align*}
a_7 &= a_1 + 1 \\
\text{if } a_1 &\leq 11
\end{align*}
\]

\[
\begin{align*}
D.1200_8 &= a_1 + 2 \\
a_9 &= D.1200_8 + 3 \\
\text{print } "Hello"
\end{align*}
\]

\[
\begin{align*}
a_2 &= \phi(a_1, a_9) \\
\text{print } "Hi"
\end{align*}
\]
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_1, a_9}`

\[
\begin{align*}
  a_3 &= 1; b_4 = 2 \\
  c_5 &= 3; n_6 = 6
\end{align*}
\]

- `{a_1, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`

\[
\begin{align*}
  a_1 &= \phi (1, a_7) \\
  \text{if } a_1 &\leq 6
\end{align*}
\]

- `{a_1, a_9}`
- `{a_1, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`

\[
\begin{align*}
  a_7 &= a_1 + 1
\end{align*}
\]

- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`

\[
\begin{align*}
  \text{D.1200}_8 &= a_1 + 2 \\
  a_9 &= \text{D.1200}_8 + 3 \\
  \text{print } "\text{Hello}" \\
  a_2 &= \phi (a_1, a_9)
\end{align*}
\]

- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`
- `{a_7, a_9}`

**Strong Liveness**

- `{a_7}`
- `{a_7}`
- `{a_7}`
- `{a_7}`

\[
\begin{align*}
  a_3 &= 1; b_4 = 2 \\
  c_5 &= 3; n_6 = 6
\end{align*}
\]

- `{a_7}`
- `{a_7}`
- `{a_7}`
- `{a_7}`

\[
\begin{align*}
  a_1 &= \phi (1, a_7) \\
  \text{if } a_1 &\leq 6
\end{align*}
\]

- `{a_7}`
- `{a_7}`
- `{a_7}`
- `{a_7}`

\[
\begin{align*}
  a_7 &= a_1 + 1
\end{align*}
\]

- `{a_7}`
- `{a_7}`
- `{a_7}`
- `{a_7}`

\[
\begin{align*}
  \text{D.1200}_8 &= a_1 + 2 \\
  a_9 &= \text{D.1200}_8 + 3 \\
  \text{print } "\text{Hello}" \\
  a_2 &= \phi (a_1, a_9)
\end{align*}
\]

- `{a_7}`
- `{a_7}`
- `{a_7}`
- `{a_7}`
Comparision of Simple and Strong Liveness for our Example

**Simple Liveness**

{a₇, a₉}

\[
\begin{align*}
a_3 &= 1; \ b_4 &= 2 \\
c_5 &= 3; \ n_6 &= 6
\end{align*}
\]

{a₇, a₉}

\[
a_1 = \phi (1, a_7)
\]

if \(a_1 \leq 6\)

{a₇, a₉}

\[
a_7 = a_1 + 1
\]

\[\{a_7, a_9\}\]

if \(a_1 \leq 11\)

{a₇, a₉}

\[
D.1200.8 = a_1 + 2 \\
a_9 = D.1200.8 + 3
\]

print ‘Hello’

{a₇, a₉}

\[
a_2 = \phi (a_1, a_9)
\]

print ‘Hi’

{a₇, a₉}

{a₇, a₉}

B2

B4

B3

B5

B6

B7

**Strong Liveness**

{a₇}

\[
\begin{align*}
a_3 &= 1; \ b_4 &= 2 \\
c_5 &= 3; \ n_6 &= 6
\end{align*}
\]

{a₇}

\[
a_1 = \phi (1, a_7)
\]

if \(a_1 \leq 6\)

{a₇}

\[
a_7 = a_1 + 1
\]

\[\{a_1\}\]

if \(a_1 \leq 11\)

{a₇}

\[
D.1200.8 = a_1 + 2 \\
a_9 = D.1200.8 + 3
\]

print ‘Hello’

{a₇}

\[
a_2 = \phi (a_1, a_9)
\]

print ‘Hi’

{a₇}

{a₇}

B2

B4

B3

B5

B6

B7
Using Data Flow Information of Live Variables Analysis

- Used for register allocation.
  
  If variable $x$ is live in a basic block $b$, it is a potential candidate for register allocation.
Using Data Flow Information of Live Variables Analysis

- Used for register allocation.
  If variable $x$ is live in a basic block $b$, it is a potential candidate for register allocation.

- Used for dead code elimination.
  If variable $x$ is not live after an assignment $x = \ldots$, then the assignment is redundant and can be deleted as dead code.
Part 4

Available Expressions Analysis
Defining Available Expressions Analysis

An expression $e$ is available at a program point $p$, if every path from program entry to $p$ contains an evaluation of $e$ which is not followed by a definition of any operand of $e$. 
Defining Available Expressions Analysis

An expression \( e \) is available at a program point \( p \), if every path from program entry to \( p \) contains an evaluation of \( e \) which is not followed by a definition of any operand of \( e \).
Defining Available Expressions Analysis

An expression $e$ is available at a program point $p$, if every path from program entry to $p$ contains an evaluation of $e$ which is not followed by a definition of any operand of $e$.

$\text{a} \times \text{b}$ is available at $p$

$\text{a} \times \text{b}$ is not available at $p$

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Defining Available Expressions Analysis

An expression $e$ is available at a program point $p$, if every path from program entry to $p$ contains an evaluation of $e$ which is not followed by a definition of any operand of $e$. 

- $a \times b$ is available at $p$
- $a \times b$ is not available at $p$
- $a \times b$ is not available at $p$
Local Data Flow Properties for Available Expressions Analysis

\[ \text{Gen}_n = \{ e | \text{expression } e \text{ is evaluated in basic block } n \text{ and this evaluation is not followed by a definition of any operand of } e \} \]

\[ \text{Kill}_n = \{ e | \text{basic block } n \text{ contains a definition of an operand of } e \} \]

<table>
<thead>
<tr>
<th>Entity</th>
<th>Manipulation</th>
<th>Exposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gen$_n$</td>
<td>Expression</td>
<td>Use</td>
</tr>
<tr>
<td>Kill$_n$</td>
<td>Expression</td>
<td>Modification</td>
</tr>
</tbody>
</table>
Data Flow Equations For Available Expressions Analysis

\[
In_n = \begin{cases} 
Bl & \text{if } n \text{ is Start block} \\
\bigcap_{p \in \text{pred}(n)} Out_p & \text{otherwise}
\end{cases}
\]

\[
Out_n = Gen_n \cup (In_n - Kill_n)
\]
Data Flow Equations For Available Expressions Analysis

\[ \ln_n = \begin{cases} B I & \text{if } n \text{ is Start block} \\ \bigcap_{p \in \text{pred}(n)} \text{Out}_p & \text{otherwise} \end{cases} \]

\[ \text{Out}_n = \text{Gen}_n \cup (\ln_n - \text{Kill}_n) \]

Alternatively,

\[ \text{Out}_n = f_n(\ln_n), \quad \text{where} \]

\[ f_n(X) = \text{Gen}_n \cup (X - \text{Kill}_n) \]
Data Flow Equations For Available Expressions Analysis

\[ In_n = \begin{cases} \mathit{Bl} & \text{n is Start block} \\ \bigcap_{p \in \text{pred}(n)} \mathit{Out}_p & \text{otherwise} \end{cases} \]

\[ Out_n = \mathit{Gen}_n \cup (\mathit{In}_n - \mathit{Kill}_n) \]

Alternatively,

\[ Out_n = f_n(\mathit{In}_n), \quad \text{where} \]

\[ f_n(X) = \mathit{Gen}_n \cup (X - \mathit{Kill}_n) \]

\[ \mathit{In}_n \text{ and } \mathit{Out}_n \text{ are sets of expressions.} \]
Using Data Flow Information of Available Expressions Analysis

- Common subexpression elimination
Using Data Flow Information of Available Expressions Analysis

• Common subexpression elimination
  ▶ If an expression is available at the entry of a block $b$ and
Using Data Flow Information of Available Expressions Analysis

- Common subexpression elimination
  - If an expression is available at the entry of a block $b$ and
  - a computation of the expression exists in $b$ such that
Using Data Flow Information of Available Expressions Analysis

- Common subexpression elimination
  - If an expression is available at the entry of a block \( b \) and
  - a computation of the expression exists in \( b \) such that
  - it is not preceded by a definition of any of its operands
Using Data Flow Information of Available Expressions Analysis

- Common subexpression elimination
  - If an expression is available at the entry of a block $b$ and
  - a computation of the expression exists in $b$ such that
  - it is not preceded by a definition of any of its operands

  Then the expression is redundant
Common subexpression elimination

- If an expression is available at the entry of a block \( b \) and
  - a computation of the expression exists in \( b \) such that
  - it is not preceded by a definition of any of its operands

Then the expression is redundant

- Redundant expression must be upwards exposed
Using Data Flow Information of Available Expressions Analysis

- Common subexpression elimination
  - If an expression is available at the entry of a block $b$ and
  - a computation of the expression exists in $b$ such that
  - it is not preceded by a definition of any of its operands

  Then the expression is redundant

- Redundant expression must be upwards exposed

- Expressions in $Gen_n$ are downwards exposed
Part 5

Introduction to Pointer Analysis
Code Optimization In Presence of Pointers

Program

1. \( q = p \);
2. \( \text{while (\ldots)} \) {
3. \( q = q \rightarrow \text{next} \);
4. }
5. \( p \rightarrow \text{data} = r1 \);
6. \( \text{print} (q \rightarrow \text{data}); \)
7. \( p \rightarrow \text{data} = r2 \);

Memory graph at statement 5

- Is \( p \rightarrow \text{data} \) live at the exit of line 5? Can we delete line 5?
Code Optimization In Presence of Pointers

Program

1. q = p;
2. do {
3.     q = q->next;
4. while (…)
5. p->data = r1;
6. print (q->data);
7. p->data = r2;

Memory graph at statement 5

- Is p->data live at the exit of line 5? Can we delete line 5?
Code Optimization In Presence of Pointers

Program

1. q = p;
2. do {
3.     q = q→next;
4.   while (…)
5.     p→data = r1;
6.     print (q→data);
7.     p→data = r2;

Memory graph at statement 5

- Is p→data live at the exit of line 5? Can we delete line 5?
- No, if p and q can be possibly aliased
  (while loop or do-while loop with a circular list)
Code Optimization In Presence of Pointers

<table>
<thead>
<tr>
<th>Program</th>
<th>Memory graph at statement 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. (q = p);</td>
<td><img src="image" alt="Memory graph at statement 5" /></td>
</tr>
<tr>
<td>2. do {</td>
<td></td>
</tr>
<tr>
<td>3. (q = q\rightarrow\text{next});</td>
<td></td>
</tr>
<tr>
<td>4. while (...)</td>
<td></td>
</tr>
<tr>
<td>5. (p\rightarrow\text{data} = r1);</td>
<td></td>
</tr>
<tr>
<td>6. print ((q\rightarrow\text{data}));</td>
<td></td>
</tr>
<tr>
<td>7. (p\rightarrow\text{data} = r2);</td>
<td></td>
</tr>
</tbody>
</table>

- Is \(p\rightarrow\text{data}\) live at the exit of line 5? Can we delete line 5?
- No, if \(p\) and \(q\) can be possibly aliased
  - (while loop or do-while loop with a circular list)
- Yes, if \(p\) and \(q\) are definitely not aliased
  - (do-while loop without a circular list)
Code Optimization In Presence of Pointers

Original Program

\[
\begin{align*}
& a = 5 \\
& x = &a \\
& b = *x
\end{align*}
\]
Code Optimization In Presence of Pointers

Original Program

```
a = 5
x = &a
b = *x
```

Constant Propagation

without aliasing

```
a = 5
x = &a
b = *x
```
Code Optimization In Presence of Pointers

Original Program

\[
\begin{align*}
    a &= 5 \\
    x &= \&a \\
    b &= \ast x
\end{align*}
\]

Constant Propagation without aliasing

\[
\begin{align*}
    a &= 5 \\
    x &= \&a \\
    b &= \ast x
\end{align*}
\]

Constant Propagation with aliasing

\[
\begin{align*}
    a &= 5 \\
    x &= \&a \\
    b &= 5
\end{align*}
\]
The World of Pointer Analysis

Alias Analysis

- Alias analysis of reference parameters, fields of unions, array indices

- Alias analysis of data pointers

Pointer Analysis

- Points-to analysis of data and function pointers
Alias Information Vs. Points-To Information

1. \( x = \&a \)

2. \( b = x \)
Alias Information Vs. Points-To Information

\[ a \]
\[ x \]
\[ b \]

```
1. x = &a
2. b = x
```

“\( x \) Points-To a”
denoted \( x \rightarrow a \)
**Alias Information Vs. Points-To Information**

```
1. $x = \&a$

"x Points-To a"

denoted $x \rightarrow a$

2. $b = x$

"x and b are Aliases"

denoted $x \triangleq b$
```
Alias Information Vs. Points-To Information

1. \( x = \&a \) denoted \( x \rightarrow a \)

2. \( b = x \) denoted \( x \triangleq b \)

\( x \) and \( b \) are Aliases

Symmetric and Reflexive
**Alias Information Vs. Points-To Information**

1. \( x = \& a \)  
   - Denoted \( x \rightarrow a \)  
   - \( x \) Points-To \( a \)  
   - Neither Symmetric Nor Reflexive

2. \( b = x \)  
   - \( x \) and \( b \) are Aliases  
   - Denoted \( x \uparrow b \)  
   - Symmetric and Reflexive
Alias Information Vs. Points-To Information

1. \( x = \&a \) denoted \( x \rightarrow a \)
   - "\( x \) Points-To \( a \)"

2. \( b = x \)
   - "\( x \) and \( b \) are Aliases"
   - denoted \( x \bowtie b \)

- What about transitivity?

Neither Symmetric Nor Reflexive

Symmetric and Reflexive
Alias Information Vs. Points-To Information

1. $x = \&a$
   - "$x$ Points-To $a$"
   - denoted $x \rightarrow a$

2. $b = x$
   - "$x$ and $b$ are Aliases"
   - denoted $x \bowtie b$

- What about transitivity?
  - Points-To: No.

Neither Symmetric Nor Reflexive

Symmetric and Reflexive
Alias Information Vs. Points-To Information

1. $x = \&a$
   - "$x$ Points-To $a$" denoted $x \rightarrow a$

2. $b = x$
   - "$x$ and $b$ are Aliases" denoted $x \bowtie b$

- What about transitivity?
  - Points-To: No.
  - Alias: Depends.

Neither Symmetric Nor Reflexive

Symmetric and Reflexive
Two important dimensions for precise pointer analysis are

- Flow Sensitivity
- Context Sensitivity
A flow-sensitive analysis computes the data flow information at each program point according to the control-flow of a program.

**Flow Sensitive analysis**

At the exit of node $n_4$

Flow insensitive information:

\{a\rightarrow b, a\rightarrow c, a\rightarrow d\}

Flow sensitive information:

\{a\rightarrow d\}
Context Sensitivity in Interprocedural Analysis

Diagram:

- Start\(_s\)
- a = &b
- \(c_i\)
- \(C_i\)
- \(R_i\)
- End\(_s\)

- Start\(_r\)
- a\(\rightarrow\)b
- \(f_r\)
- C\(_r\)
- \(R_i\)
- End\(_r\)

- Start\(_t\)
- c = &d
- \(c_j\)
- \(C_j\)
- \(R_j\)
- End\(_t\)

Essential Abstractions in GCC
GCC Resource Center, IIT Bombay
Context Sensitivity in Interprocedural Analysis

- `Start_s`: `a = &b`
- `C_i`
- `R_i`: `a = b`
- `End_s`

- `Start_r`: `Start_r`
- `C_j`
- `R_j`: `c = d`
- `End_r`

- `Start_t`: `c = &d`
- `C_j`
- `R_j`: `c = d`
- `End_t`
Context Sensitivity in Interprocedural Analysis

\[ \text{Start}_s \]
\[ a = \& b \]
\[ c_i \]
\[ C_i \]
\[ R_i \]
\[ \text{End}_s \]
\[ a \rightarrow b \]

\[ \text{Start}_r \]
\[ \text{End}_r \]
\[ f_r \]
\[ \times \]
\[ a \rightarrow b \]
\[ c \rightarrow d \]

\[ \text{Start}_t \]
\[ c = \& d \]
\[ c_j \]
\[ C_j \]
\[ R_j \]
\[ \text{End}_t \]
\[ c \rightarrow d \]
Context Sensitivity in Interprocedural Analysis

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Context Sensitivity in Interprocedural Analysis

\[ \text{Start}_s \]
\[ a = \& b \]
\[ C_i \]
\[ R_i \]
\[ \text{End}_s \]

\[ \text{Start}_r \]
\[ a \to b \]
\[ C_j \]
\[ R_j \]
\[ \text{End}_r \]

\[ \text{Start}_t \]
\[ c = \& d \]
\[ C_j \]
\[ R_j \]
\[ \text{End}_t \]

\[ a \to b \quad c \to d \]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Issues with Pointer Analysis

- For precise pointer information, we require flow and context sensitive pointer analysis.

- Flow and context sensitive pointer analysis computes a large size of information.
Example of Points-to Analysis

```
x = &y
y = &z
z = &u

*Z = Y  n2   Z = Y  n3

Use u  n4
Use x
```
Example of Points-to Analysis

\[
\begin{align*}
  n_1: & \quad x = \&y \\
  & \quad y = \&z \\
  & \quad z = \&u \\
  n_2: & \quad *z = y \\
  n_3: & \quad z = y \\
  n_4: & \quad \text{use } u \\
  & \quad \text{use } x
\end{align*}
\]
Example of Points-to Analysis

\[ x = &y \\
y = &z \\
z = &u \]

\[ \{x\to y, y\to z, z\to u\} \]

\[ *z = y \]

\[ z = y \]

\[ \text{use } u \]

\[ \text{use } x \]
Example of Points-to Analysis

\[ n_1 \]
\[ x = &y \]
\[ y = &z \]
\[ z = &u \]
\[ \emptyset \]
\[ \{ x \rightarrow y, y \rightarrow z, z \rightarrow u \} \]

\[ n_2 \]
\[ *z = y \]
\[ \{ x \rightarrow y, y \rightarrow z, z \rightarrow u \} \]

\[ n_3 \]
\[ z = y \]
\[ \{ x \rightarrow y, y \rightarrow z, z \rightarrow u \} \]

\[ n_4 \]
\[ \text{use } u \]
\[ \text{use } x \]
Example of Points-to Analysis

1. $x = \& y$
2. $y = \& z$
3. $z = \& u$

$n_1$

$n_2$

$n_3$

$n_4$

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Example of Points-to Analysis

```
x = &y
y = &z
z = &u
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u, u\to z\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```

```
\{x\to y, y\to z, z\to u\}
```
**Example of Points-to Analysis**

```
x = &y
y = &z
z = &u

n_1

\{x→y, y→z, z→u\}

n_2

*z = y

\{x→y, y→z, z→u, u→z\}

n_3

z = y

\{x→y, y→z, z→u\}

\{x→y, y→z, z→z\}

n_4

use u
use x

\{x→y, y→z, z→u\}

\{x→y, y→z, z→u\}

\{x→y, y→z, z→u\}

\{x→y, y→z, z→u\}
```
Example of Points-to Analysis

```
x = &y  
y = &z  
z = &u  
```

```
\{ x \rightarrow y, y \rightarrow z, z \rightarrow u \} 
```

```
*z = y  
```

```
\{ x \rightarrow y, y \rightarrow z, z \rightarrow u, u \rightarrow z \} 
```

```
z = y  
```

```
\{ x \rightarrow y, y \rightarrow z, z \rightarrow z \} 
```

```
use u  
use x  
```

```
\{ x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z \} 
```
Example of Points-to Analysis

\[ \begin{align*}
  n_1 & : \begin{cases}
    x = & y \\
    y = & z \\
    z = & u
  \end{cases} & \Rightarrow & \emptyset \\
  n_2 & : *z = y & \Rightarrow & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
  n_3 & : z = y & \Rightarrow & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
  n_4 & : \text{use } u \text{ use } x & \Rightarrow & \{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z\}
\end{align*} \]
Is All This Information Useful?

\[
\begin{align*}
 & \text{Essential Abstractions in GCC} \\
\end{align*}
\]
Is All This Information Useful?

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Is All This Information Useful?

Diagram:

- **n1**: $x = \& y$
  - $y = \& z$
  - $z = \& u$
  - $\emptyset$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u\}$

- **n2**: $z = y$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u\}$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u, u \rightarrow z\}$

- **n3**: $\ast z = y$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u\}$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u, u \rightarrow z\}$

- **n4**: $\text{use } u$
  - $\text{use } x$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z\}$
  - $\{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z\}$
Is All This Information Useful?

\[
\begin{align*}
\begin{array}{c}
\text{Is All This Information Useful?} \\
\begin{array}{c}
\text{n}_1 \quad x = \& y \\
\text{y = \&z} \\
\text{z = \&u} \\
\text{\{}x\rightarrow y, y\rightarrow z, z\rightarrow u\}\}
\end{array} \\
\begin{array}{c}
\text{n}_2 \quad *z = y \\
\text{\{}x\rightarrow y, y\rightarrow z, z\rightarrow u\text{, }u\rightarrow z\}\}
\end{array} \\
\begin{array}{c}
\text{n}_3 \quad z = y \\
\text{\{}x\rightarrow y, y\rightarrow z, z\rightarrow u\text{, }z\rightarrow z\}\}
\end{array} \\
\begin{array}{c}
\text{n}_4 \quad \text{use } u \\
\text{use } x \\
\text{\{}x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\}\}
\end{array}
\end{array}
\end{align*}
\]
Is All This Information Useful?

\[
\begin{align*}
&n_1: x = &y \\
&y = &z \\
z = &u \\
&\emptyset \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u\} \\
&*z = y \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, u\rightarrow z\} \\
&z = y \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow z\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\} \\
&\{x\rightarrow y, y\rightarrow z, z\rightarrow u, z\rightarrow z, u\rightarrow z\} \\
\end{align*}
\]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Is All This Information Useful?

\[
\begin{align*}
& x = & y \\
& y = & z \\
& z = & u \\
& n_1 & & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
& \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} & n_2 & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
& \{x \rightarrow y, y \rightarrow z, z \rightarrow u, u \rightarrow z\} & \ast z = y & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
& n_2 & & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
& z = y & n_3 & \{x \rightarrow y, y \rightarrow z, z \rightarrow u\} \\
& \{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z\} & \text{use } u & \{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z\} \\
& & n_4 & \{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z\} \\
& \text{use } x & & \{x \rightarrow y, y \rightarrow z, z \rightarrow u, z \rightarrow z, u \rightarrow z\}
\end{align*}
\]
Improving pointer analysis

For a fast flow and context sensitive pointer analysis, we can reduce the number of computations done at a program point. This can be done in following ways:

- Computing pointer information for only those variables that are being used at some later program point.
- Propagating only the new data flow values obtained in current iteration to the next iteration.
Liveness Based Pointer analysis (L-FCPA)

- A flow and context sensitive pointer analysis
Liveness Based Pointer analysis (L-FCPA)

- A flow and context sensitive pointer analysis
- Pointer information is not computed unless a variable becomes live.
Liveness Based Pointer analysis (L-FCPA)

- A flow and context sensitive pointer analysis

- Pointer information is not computed unless a variable becomes live.

- Strong liveness is used for computing liveness information.

  If basic block contains statement like $x = y$, then $y$ is said to be live, if $x$ is live at the exit of basic block.
Liveness Based Pointer analysis (L-FCPA)

- A flow and context sensitive pointer analysis
- Pointer information is not computed unless a variable becomes live.
- Strong liveness is used for computing liveness information.
  If basic block contains statement like $x = y$, then $y$ is said to be live, if $x$ is live at the exit of basic block.
- Pointer information is propagated only in live range of the pointer
First Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
x &= \& y \\
y &= \& z \\
z &= \& u \\
x^* &= y \\
z &= y \\
use\ u \\
use\ x
\end{align*}
\]
First Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
  x &= \& y \\
  y &= \& z \\
  z &= \& u
\end{align*}
\]

\[
\begin{align*}
  *z &= y \\
  z &= y
\end{align*}
\]

use u
use x
First Round of Liveness Analysis and Points-to Analysis

\[ x = \& y \]
\[ y = \& z \]
\[ z = \& u \]

\[ *z = y \] \( n_2 \)
\[ z = y \] \( n_3 \)

\( \{ u, x \} \) \( n_4 \)

Essential Abstractions in GCC
First Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
  x &= \& y \\
  y &= \& z \\
  z &= \& u
\end{align*}
\]

\[
\begin{align*}
  *z &= y \\
  z &= y \\
  \{u, x\} \\
  \{u, x\} \\
  \{u, x\} \\
  \{u, x\}
\end{align*}
\]

\[
\begin{align*}
  use u \\
  use x
\end{align*}
\]
First Round of Liveness Analysis and Points-to Analysis

\[ x = \& y \]
\[ y = \& z \]
\[ z = \& u \]

\[ *z = y \]
\[ z = y \]

\[ \{ u, x \} \]

\[ \{ z \} \]
\[ \{ u, x \} \]
\[ \{ u, x \} \]

\[ use u \]
\[ use x \]

Essential Abstractions in GCC

 GCC Resource Center, IIT Bombay
First Round of Liveness Analysis and Points-to Analysis

\[
x = &y \\
y = &z \\
z = &u
\]

\[
\{u, x, z\} \\
\{u, x\}
\]

\[
\{z\} \\
\{u, x\}
\]

\[
\{u, x\} \\
\{u, x\}
\]

\[
\{u, x\}
\]

Essential Abstractions in GCC
GCC Resource Center, IIT Bombay
First Round of Liveness Analysis and Points-to Analysis

\begin{align*}
  x &= \& y \\
  y &= \& z \\
  z &= \& u \\
  *z &= y \\
  z &= y \\
  use u \\
  use x
\end{align*}
First Round of Liveness Analysis and Points-to Analysis

\[
x = &y \\
y = &z \\
z = &u
\]

\[\{u\}\]

\[n_1\]

\[\{u, x, z\}\]

\[n_2\]

\[\{z\}\]

\[\{u, x\}\]

\[*z = y\]

\[n_3\]

\[z = y\]

\[\{u, x\}\]

\[\{u, x\}\]

\[\{u, x\}\]

\[\{u, x\}\]

\[\{u, x\}\]

\[use u\]

\[use x\]

\[n_4\]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
First Round of Liveness Analysis and Points-to Analysis

\[ x = \& y \]
\[ y = \& z \]
\[ z = \& u \]

\[ \{ u \} \{ u \rightarrow ? \} \]
\[ n_1 \]
\[ \{ u, x, z \} \]

\[ \{ z \} \]
\[ n_2 \]
\[ \{ u, x \} \]

\[ *z = y \]

\[ \{ u, x \} \]

\[ \{ u, x \} \]
\[ n_3 \]
\[ \{ u, x \} \]

\[ use u \]
\[ use x \]

\[ \{ u, x \} \]
\[ n_4 \]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
First Round of Liveness Analysis and Points-to Analysis

\begin{align*}
x &= \&y \\
y &= \&z \\
z &= \&u
\end{align*}

\begin{align*}
\{u\} & \{u \rightarrow ? \} \\
n_1
\end{align*}

\begin{align*}
\{u, x, z\} & \{u \rightarrow ?, x \rightarrow y, z \rightarrow u\}
\end{align*}

\begin{align*}
\{z\}
\end{align*}

\begin{align*}
\{u, x\}
\end{align*}

\begin{align*}
\{z\} & \{u, x\} \\
n_2
\end{align*}

\begin{align*}
\{u, x\}
\end{align*}

\begin{align*}
\{z = y\} & \{u, x\} \\
n_3
\end{align*}

\begin{align*}
\{u, x\}
\end{align*}

\begin{align*}
use \ u \\
use \ x
\end{align*}

\begin{align*}
\{u, x\} & \{u, x\} \\
n_4
\end{align*}
First Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
x &= &\& y \\
y &= &\& z \\
z &= &\& u
\end{align*}
\]

\[
\begin{align*}
\{u\} & \{u\rightarrow\}\quad n_1 \\
\{u, x, z\} & \{u\rightarrow?, x\rightarrow y, z\rightarrow u\}
\end{align*}
\]

\[
\begin{align*}
\{z\rightarrow u\} & \{z\} \\
\{u, x\} & \quad n_2
\end{align*}
\]

\[
\begin{align*}
\{u, x\} & \quad \{u\rightarrow?, x\rightarrow y\} \\
\{u, x\} & \\
\{u, x\} & \\
\{u, x\} & \quad \{u\rightarrow?, x\rightarrow y\}
\end{align*}
\]

\[
\begin{align*}
\text{use } u \\
\text{use } x 
\end{align*}
\]

\[
\begin{align*}
\{u, x\} & \\
\{u, x\} & \quad \{u\rightarrow?, x\rightarrow y\} \\
\{u, x\} & \\
\{u, x\} & \\
\{u, x\} & \{u\rightarrow?, x\rightarrow y\}
\end{align*}
\]
First Round of Liveness Analysis and Points-to Analysis

\[ x = \& y \]
\[ y = \& z \]
\[ z = \& u \]

\[ \{ z \rightarrow u \} \quad \{ u \} \]
\[ \{ u \rightarrow ? \} \quad \{ u \rightarrow ?, x \rightarrow y \} \]

\[ *z = y \quad n_2 \]
\[ \{ u, x \} \quad \{ u \rightarrow ?, x \rightarrow y \} \]

\[ n_1 \]
\[ \{ u, x, z \} \quad \{ u \rightarrow ?, x \rightarrow y, z \rightarrow u \} \]

\[ z = y \quad n_3 \]
\[ \{ u, x \} \quad \{ u \rightarrow ?, x \rightarrow y \} \]

\[ use u \]
\[ use x \quad n_4 \]
First Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
&x = \&y \\
y = \&z \\
z = \&u \\
\end{align*}
\]

\[
\begin{align*}
x = \&y & \{u\} \{u \to ?\} \\
y = \&z & \{u, x, z\} \{u \to ?, x \to y, z \to u\} \\
z = \&u & \{u, x\} \{u \to ?, x \to y\} \\
\end{align*}
\]

\[
\begin{align*}
*z = y & \{z \to u\} \{z\} \\
z = y & \{u, x\} \{u \to ?, x \to y\} \\
use u, use x & \{u, x\} \{u \to ?, x \to y\} \\
\end{align*}
\]
Second Round of Liveness Analysis and Points-to Analysis

\[
x = \&y \\
y = \&z \\
z = \&u
\]

\[
z \rightarrow u
\]

\[
{z} & {u, x} \\
{u, x} & {u, x} \\
{u, x} & {u, x}
\]

\[
{u} & {u, x, z} \\
{u, x} & {u, x, z}
\]

Essential Abstractions in GCC
Second Round of Liveness Analysis and Points-to Analysis

\[ \begin{align*}
  x &= \&y \\
  y &= \&z \\
  z &= \&u \\
  z \rightarrow u \\
  *z &= y \\
  z &= y \\
  \text{use } u \\
  \text{use } x
\end{align*} \]
Second Round of Liveness Analysis and Points-to Analysis

\[ x = \& y \]
\[ y = \& z \]
\[ z = \& u \]

\[ z \rightarrow u \]
\[ *z = y \]

\[ \{ u, x \} \]
\[ \{ u, x, y \} \]

\[ n_2 \]

\[ z = y \]

\[ \{ u, x \} \]
\[ \{ u, x \} \]

\[ n_3 \]

\[ use u \]
\[ use x \]

\[ n_4 \]
Second Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
\text{x} &= \& \text{y} \\
\text{y} &= \& \text{z} \\
\text{z} &= \& \text{u}
\end{align*}
\]

\begin{align*}
\text{n}_1 &: \{u\} \\
\text{n}_2 &: \{u, x, z, y\} \\
\text{n}_3 &: \{u, x\} \\
\text{n}_4 &: \{u, x\}
\end{align*}

\text{Essential Abstractions in GCC}
Second Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
x &= & & \& y \\
y &= & & \& z \\
z &= & & \& u \\
\end{align*}
\]

\[
\begin{align*}
\{z, x, y\} \\
\{u, x\} \\
\{u, x, z, y\} \\
\{u\} \\
\end{align*}
\]

\[
\begin{align*}
\{z, x, y\} \\
\{u, x\} \\
\{u, x\} \\
\{u, x\} \\
\end{align*}
\]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Second Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
  x &= \& y \\
  y &= \& z \\
  z &= \& u \\
  *z &= y \\
  z &= y \\
  \text{use } u \\
  \text{use } x
\end{align*}
\]
Second Round of Liveness Analysis and Points-to Analysis

- \( x = &y \)
- \( y = &z \)
- \( z = &u \)

\[ \{ u \} \{ u \rightarrow ? \} \]

\[ n_1 \]

\[ \{ u \rightarrow ?, x \rightarrow y, z \rightarrow u \} \cup \{ y \rightarrow z \} \]

\[ \{ u, x, z, y \} \]

\[ z = y \]

\[ \{ u, x \} \]

\[ n_3 \]

\[ \{ u, x \} \]

\[ \{ u, x \} \]

\[ n_4 \]

\[ \{ u, x \} \]
Second Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
&x = &y \\
y = &z \\
z = &u \\
\end{align*}
\]

\[
\begin{align*}
{n_1} &:\{u\to?, x\to y, z\to u\} \cup \{y\to z\} \\
{n_2} &:\{z, x, y\} \\
{n_3} &:\{u, x\} \\
{n_4} &:\{u, x\} \\
\end{align*}
\]

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Second Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
x &= \& y \\
y &= \& z \\
z &= \& u
\end{align*}
\]

- \( n_1 \): \( \{u\} \cup \{u\rightarrow?\} \)

- \( n_2 \): \( \{u\rightarrow?, x\rightarrow y, z\rightarrow u\} \cup \{y\rightarrow z\} \)
  - \( \{u, x, z, y\} \)

- \( n_3 \): \( \{u, x\} \cup \{u\rightarrow?, x\rightarrow y\} \)

- \( n_4 \): \( \{u, x\} \cup \{u\rightarrow?, x\rightarrow y\} \)

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay
Second Round of Liveness Analysis and Points-to Analysis

\[
x = \&y \\
y = \&z \\
z = \&u
\]

\[
\{u\} \{u\rightarrow?\}
\]

\[
n_1 \\
\{u\rightarrow?, x\rightarrow y, z\rightarrow u\} \cup \{y\rightarrow z\} \\
\{u, x, z, y\}
\]

\[
\{z\rightarrow u\} \cup \{y\rightarrow z, x\rightarrow y\} \\
\{z, x, y\}
\]

\[
n_2 \\
\{u, x\}
\]

\[
\{u\rightarrow?, x\rightarrow y\}
\]

\[
\{z = y\} \\
\{u, x\}
\]

\[
\{u\rightarrow?, x\rightarrow y\}
\]

\[
\ast z = y \\
\{u, x\}
\]

\[
n_3 \\
\{u, x\}
\]

\[
\{u\rightarrow?, x\rightarrow y\}
\]

\[
\ use u \\
\ use x \\
\]

\[
n_4 \\
\{u, x\}
\]

\[
\{u\rightarrow?, x\rightarrow y\}
\]
Second Round of Liveness Analysis and Points-to Analysis

\[
x = \& y \\
y = \& z \\
z = \& u
\]

\[
\{z \mapsto u\} \cup \{y \mapsto z, x \mapsto y\} \\
\{z, x, y\}
\]

\[
x = \& y \\
\{u\} \{u \mapsto ?\}
\]

\[
z = y \\
\{u\} \{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
use u \\
use x
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]

\[
\{u, x\} \\
\{u \mapsto ?, x \mapsto y\}
\]
Second Round of Liveness Analysis and Points-to Analysis

\[
\begin{align*}
  x &= & \&y \\
  y &= & \&z \\
  z &= & \&u \\
  n_1 &= & \{u\} \{u \rightarrow ?\} \\
  n_2 &= & \{u \rightarrow ?, x \rightarrow y, z \rightarrow u\} \cup \{y \rightarrow z\} \\
  n_3 &= & \{u, x\} \{u \rightarrow ?, x \rightarrow y\} \\
  n_4 &= & \{u, x\} \{u \rightarrow ?, x \rightarrow y\} \cup \{u \rightarrow z\}
\end{align*}
\]
Observation

- L-FCPA has 2 fixed point computations:
  - Strong Liveness analysis
  - Points-to analysis

- Liveness and Points-to passes are interdependent.

- Both the computations are done alternatively until final value converges.
Conclusions: New Insights in Pointer Analysis

- Usable pointer information is very small and sparse
- Earlier approaches reported inefficiency and non-scalability because they computed far more information than the actual usable information
Conclusions: New Insights in Pointer Analysis

- Usable pointer information is very small and sparse
- Earlier approaches reported inefficiency and non-scalability because they computed far more information than the actual usable information
- Triumph of *The Genius of AND over the Tyranny of OR*
Conclusions: New Insights in Pointer Analysis

- Usable pointer information is very small and sparse
- Earlier approaches reported inefficiency and non-scalability because they computed far more information than the actual usable information
- Triumph of *The Genius of AND over the Tyranny of OR*
- Future work
  - Redesign data structures by hiding them behind APIs
    - Current version uses linked lists and linear search
  - Incremental version
  - Using precise pointer information in other passes in GCC
Precise Context Information is Small and Sparse

Our contributions: Value based termination, liveness

<table>
<thead>
<tr>
<th>Program</th>
<th>Total no. of functions</th>
<th>No. and percentage of functions for call-string counts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0 call strings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L-FCPA</td>
</tr>
<tr>
<td>lbm</td>
<td>22</td>
<td>16 (72.7%)</td>
</tr>
<tr>
<td>mcf</td>
<td>25</td>
<td>16 (64.0%)</td>
</tr>
<tr>
<td>bzip2</td>
<td>100</td>
<td>88 (88.0%)</td>
</tr>
<tr>
<td>libquantum</td>
<td>118</td>
<td>100 (84.7%)</td>
</tr>
<tr>
<td>sjeng</td>
<td>151</td>
<td>96 (63.6%)</td>
</tr>
<tr>
<td>hmmer</td>
<td>584</td>
<td>548 (93.8%)</td>
</tr>
<tr>
<td>parser</td>
<td>372</td>
<td>246 (66.1%)</td>
</tr>
<tr>
<td>h264ref</td>
<td>624</td>
<td>351 (56.2%)</td>
</tr>
</tbody>
</table>

9+ call strings in L-FCPA: Tot 4, Min 10, Max 52, Mean 32.5, Median 29, Mode 10

9+ call strings in L-FCPA: Tot 14, Min 9, Max 56, Mean 27.9, Median 24, Mode 9
Precise Usable Pointer Information is Small and Sparse

Our contribution: liveness

<table>
<thead>
<tr>
<th>Program</th>
<th>Total no. of BBs</th>
<th>L-FCPA 0 pt pairs</th>
<th>L-FCPA 1-4 pt pairs</th>
<th>L-FCPA 5-8 pt pairs</th>
<th>L-FCPA 9+ pt pairs</th>
<th>FCPA 0 pt pairs</th>
<th>FCPA 1-4 pt pairs</th>
<th>FCPA 5-8 pt pairs</th>
<th>FCPA 9+ pt pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>lbm</td>
<td>252</td>
<td>229 (90.9%)</td>
<td>61 (24.2%)</td>
<td>23 (9.1%)</td>
<td>82 (32.5%)</td>
<td>0</td>
<td>66 (26.2%)</td>
<td>0</td>
<td>43 (17.1%)</td>
</tr>
<tr>
<td>mcf</td>
<td>472</td>
<td>356 (75.4%)</td>
<td>160 (33.9%)</td>
<td>116 (24.6%)</td>
<td>2 (0.4%)</td>
<td>0</td>
<td>1 (0.2%)</td>
<td>0</td>
<td>309 (65.5%)</td>
</tr>
<tr>
<td>libquantum</td>
<td>1642</td>
<td>1520 (92.6%)</td>
<td>793 (48.3%)</td>
<td>119 (7.2%)</td>
<td>796 (48.5%)</td>
<td>3</td>
<td>46 (2.8%)</td>
<td>0</td>
<td>7 (0.4%)</td>
</tr>
<tr>
<td>bzip2</td>
<td>2746</td>
<td>2624 (95.6%)</td>
<td>1085 (39.5%)</td>
<td>118 (4.3%)</td>
<td>12 (0.4%)</td>
<td>3</td>
<td>12 (0.4%)</td>
<td>1</td>
<td>1637 (59.6%)</td>
</tr>
</tbody>
</table>

9+ pt pairs in L-FCPA: Tot 1, Min 12, Max 12, Mean 12.0, Median 12, Mode 12

| sjeng       | 6000             | 4571 (76.2%)      | 3239 (54.0%)        | 1208 (20.1%)        | 12 (0.2%)         | 221 (3.7%)     | 41 (0.7%)      | 0              | 2708 (45.1%)   |

9+ pt pairs in L-FCPA: Tot 6, Min 10, Max 16, Mean 13.3, Median 13, Mode 10

| hmmer       | 14418            | 13483 (93.5%)     | 8357 (58.0%)        | 896 (6.2%)          | 21 (0.1%)         | 24 (0.2%)      | 91 (0.6%)      | 15             | 5949 (41.3%)   |

9+ pt pairs in L-FCPA: Tot 6, Min 10, Max 16, Mean 13.3, Median 13, Mode 10

| parser      | 6875             | 4823 (70.2%)      | 1821 (26.5%)        | 1591 (23.1%)        | 25 (0.4%)         | 252 (3.7%)     | 154 (2.2%)     | 209            | 4875 (70.9%)   |

9+ pt pairs in L-FCPA: Tot 13, Min 9, Max 53, Mean 27.9, Median 18, Mode 9

| h264ref     | 21315            | 13729 (64.4%)     | ?                   | 4760 (22.3%)        | ?                 | 2035 (9.5%)    | ?              | 791 (3.7%)     |

9+ pt pairs in L-FCPA: Tot 44, Min 9, Max 98, Mean 36.3, Median 31, Mode 9