Workshop on Essential Abstractions in GCC

Graybox Probing for Machine Independent **Optimizations**

GCC Resource Center

(www.cse.iitb.ac.in/grc)

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



30 June 2011

30 June 2011

Machine Independent Optimizations: Outline

1/29

30 June 2011

Machine Independent Optimizations: Outline

1/29

Outline

Outline

• Example 1

- ► Constant Propagation
- ► Copy Propagation
- ► Dead Code Elimination
- ► Loop unrolling
- Example 2
 - ► Partial Redundancy Elimination
 - Copy Propagation
 - ► Dead Code Elimination



First Example Program

30 June 2011

Machine Independent Optimizations: First Example Program

2/29

Example Program 1

Example Program 1

int main() { int a, b, c, n; a = 1;b = 2;c = 3;n = c*2;while $(a \le n)$ a = a+1;if (a < 12)a = a+b+c;return a;

- What does this program return?
- 12
- We use this program to illustrate various shades of the following optimizations: Constant propagation, Copy propagation, Loop unrolling, Dead code elimination

30 June 2011

Compilation Command

Compilation Command

\$gcc -fdump-tree-all -02 ccp.c



4/29

Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bombay

30 June 2011

Machine Independent Optimizations: First Example Program

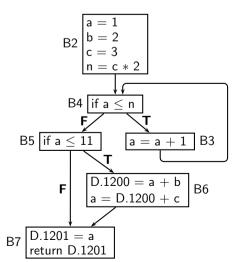
4/29

Example Program 1

Machine Independent Optimizations: First Example Program

```
Program ccp.c
int main()
{ int a, b, c, n;
  a = 1;
  b = 2;
  c = 3;
  n = c*2;
  while (a \le n)
    a = a+1;
  if (a < 12)
    a = a+b+c;
  return a;
```

Control flow graph



30 June 2011

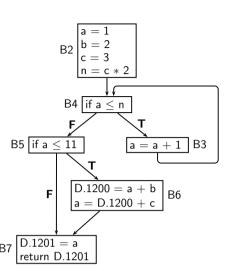
Essential Abstractions in GCC

Example Program 1



Control Flow Graph: Pictorial and Textual View

Control flow graph



Dump file ccp.c.013t.cfg

Essential Abstractions in GCC

GCC Resource Center, IIT Bomba



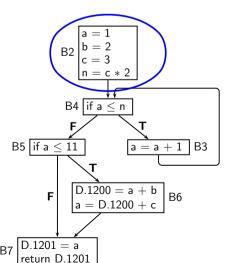
30 June 2011

Machine Independent Optimizations: First Example Program

5/29

Control Flow Graph: Pictorial and Textual View

Control flow graph



Dump file ccp.c.013t.cfg

```
<bb 2>:
a = 1;
c = 3;
n = c * 2;
goto <bb 4>;
```

Control Flow Graph: Pictorial and Textual View

30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bomba

5/29

30 June 2011 Machine Independent Optimizations: First Example Program

Control Flow Graph: Pictorial and Textual View

Control Flow Graph: Pictorial and Textual View

Control flow graph

Dump file ccp.c.013t.cfg

```
B2 \begin{bmatrix} a = 1 \\ b = 2 \\ c = 3 \\ n = c * 2 \end{bmatrix}

B4 if a \le n

F

D.1200 = a + b
a = D.1200 + c

B6
```

Essential Abstractions in GCC

return D.1201

GCC Resource Center, IIT Bombay



30 June 2011

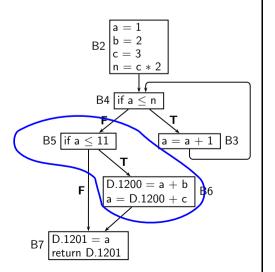
Machine Independent Optimizations: First Example Program

5/29

Control Flow Graph: Pictorial and Textual View

Control flow graph

Dump file ccp.c.013t.cfg



```
<bb 5>:
if (a <= 11)
    goto <bb 6>;
else
    goto <bb 7>;

<bb 6>:
D.1200 = a + b;
a = D.1200 + c;
```

waemine maepe

Control Flow Graph: Pictorial and Textual View

Notes

30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay

_ ()

30 June 2011

Machine Independent Optimizations: First Example Program

5/29

Control Flow Graph: Pictorial and Textual View



Control flow graph

Dump file ccp.c.013t.cfg

B4 if $a \le n$ B5 if a ≤ 11 |a = a + 1|B3D.1200 = a + ba = D.1200 + cD.1201 = areturn D.1201

<bb >7>: D.1201 = a;return D.1201;

Essential Abstractions in GCC

30 June 2011

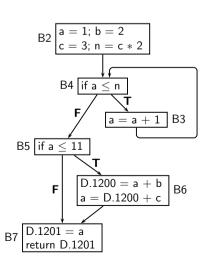
GCC Resource Center, IIT Bombay



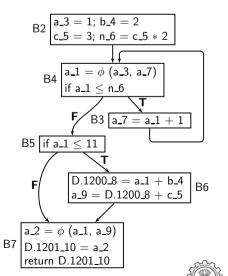
6/29

Machine Independent Optimizations: First Example Program Single Static Assignment (SSA) Form

Control flow graph



SSA Form



Control Flow Graph: Pictorial and Textual View

30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bomba

30 June 2011

Machine Independent Optimizations: First Example Program

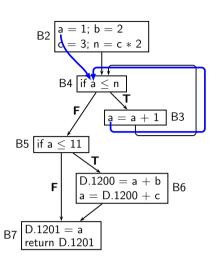
6/29

Single Static Assignment (SSA) Form

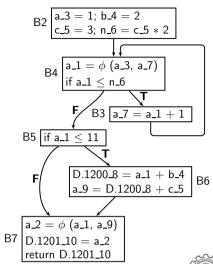


Single Static Assignment (SSA) Form

Control flow graph



SSA Form



Essential Abstractions in GCC

GCC Resource Center, IIT Bombay



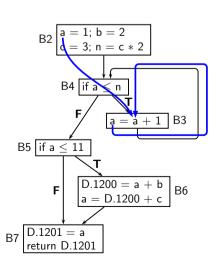
30 June 2011

Machine Independent Optimizations: First Example Program

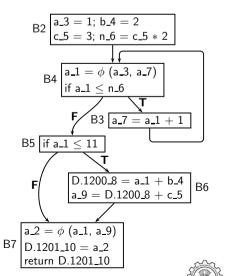
6/29

Single Static Assignment (SSA) Form

Control flow graph



SSA Form



Votes

30 June 2011

Essential Abstractions in GCC GCC Resource Center, IIT Bomba

ay (ii)

30 June 2011

Machine Independent Optimizations: First Example Program

6/29

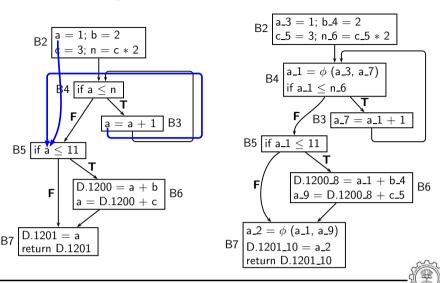
Single Static Assignment (SSA) Form



Single Static Assignment (SSA) Form

Control flow graph

SSA Form



Essential Abstractions in GCC

GCC Resource Center, IIT Bombay

Essential Abstractions in GCC

30 June 2011

Machine Independent Optimizations: First Example Program

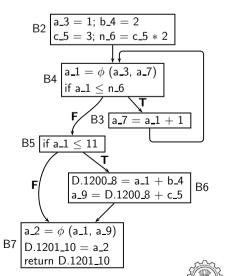
6/29

Single Static Assignment (SSA) Form

Control flow graph

B4 if $a \leq n$ + 1 B3 B5 if a ≤ 11 D.1200 = a + ba = D.1200 + c

SSA Form



Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bomba

6/29

Machine Independent Optimizations: First Example Program Single Static Assignment (SSA) Form

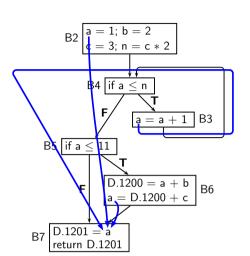
Single Static Assignment (SSA) Form

D.1201 = a

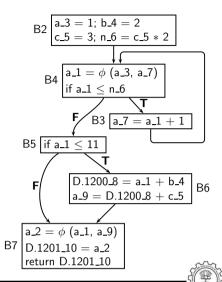
return D.1201

Single Static Assignment (SSA) Form

Control flow graph



SSA Form



Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bombay

Machine Independent Optimizations: First Example Program

7/29

Machine Independent Optimizations: First Example Program Properties of SSA Form

B2 a.3 = 1; b.4 = 2 c.5 = 3; n.6 = c.5 * 2B4 $a.1 = \phi$ (a.3, a.7) if $a.1 \le n.6$ F B3 a.7 = a.1 + 1B5 a.7 = a.1 + 1B7 a.9 = D.1200.8 + c.5B6 $a.2 = \phi$ (a.1, a.9) D.1201.10 = a.2 return D.1201.10

- A ϕ function is a multiplexer or a selection function
- Every use of a variable corresponds to a unique definition of the variable
- For every use, the definition is guaranteed to appear on every path leading to the use

SSA construction algorithm is expected to insert as few ϕ functions as possible to ensure the above properties



7/29

Single Static Assignment (SSA) Form

Notes

Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bombay

Properties of SSA Form

SSA Form: Pictorial and Textual View

CFG in SSA form

B2 a.3 = 1; b.4 = 2 c.5 = 3; n.6 = c.5 * 2B4 $a.1 = \phi$ (a.3, a.7)
if $a.1 \le n.6$ F B3 a.7 = a.1 + 1B5 if $a.1 \le 11$ T

D.1200_8 = a.1 + b.4 a.9 = D.1200.8 + c.5B6

B7 $a.2 = \phi$ (a.1, a.9) $a.2 = \phi$ (a.1, a.9) a.3 = a.2

Dump file ccp.c.017t.ssa



8/29

Essential Abstractions in GCC

30 June 2011

return D.1201_10

GCC Resource Center, IIT Bombay

Machine Independent Optimizations: First Example Program

- g. u...

SSA Form: Pictorial and Textual View

CFG in SSA form

B2 a.3 = 1; b.4 = 2 c.5 = 3; n.6 = c.5 * 2B4 $a.1 = \phi$ (a.3, a.7)
if $a.1 \le n.6$ B5 if $a.1 \le 11$ T

D.1200.8 = a.1 + b.4 a.9 = D.1200.8 + c.5B6

B7 $a.2 = \phi$ (a.1, a.9)
D.1201.10 = a.2
return D.1201.10

Dump file ccp.c.017t.ssa

SSA Form: Pictorial and Textual View

Notes

30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay

ay ()

8/29

30 June 2011 Machine Independent Optimizations: First Example Program

SSA Form: Pictorial and Textual View

CFG in SSA form $a_3 = 1$; $b_4 = 2$ $c_5 = 3$; $n_6 = c_5 * 2$ $a_1 = \phi$ (a_3, a_7) B3 $a_7 = a_1 + 1$ B5 if a_1 < 11 $D.1200_8 = a_1 + b_4$ $a_9 = D.1200_8 + c_5$ $a_2 = \phi$ (a_1, a_9) B7 $D.1201_10 = a_2$ return D.1201_10

Dump file ccp.c.017t.ssa

Essential Abstractions in GCC

GCC Resource Center, IIT Bomba



30 June 2011

30 June 2011

Machine Independent Optimizations: First Example Program

8/29

SSA Form: Pictorial and Textual View

CFG in SSA form $c_5 = 3$: $n_6 = c_5 * 2$ $a_1 = \phi$ (a_3, a_7) if a_1 \leq n_6 B3 $a_7 = a_1 + 1$ B5 if a_1 < 11 $D.1200_8 = a_1 + b_4$ $a_9 = D.1200_8 + c_5$ $= \phi$ (a_1, a_9) $D.1201_10 = a_2$ return D.1201_10

Dump file ccp.c.017t.ssa

SSA Form: Pictorial and Textual View

30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bomb



30 June 2011

Machine Independent Optimizations: First Example Program

8/29

SSA Form: Pictorial and Textual View



Machine Independent Optimizations: First Example Program A Comparison of CFG and SSA Dumps

Dump file ccp.c.013t.cfg

Dump file ccp.c.017t.ssa

```
<br/>
<br/>
a = 1;<br/>
b = 2;<br/>
c = 3;<br/>
n = c * 2;<br/>
goto <br/>
<br/>
bb 3>:<br/>
a = a + 1;<br/>
```

```
<bb 2>:
    a_3 = 1;
    b_4 = 2;
    c_5 = 3;
    n_6 = c_5 * 2;
    goto <bb 4>;

<bb 3>:
    a_7 = a_1 + 1;
```



Essential Abstractions in GCC

30 June 2011

GCC Resource Center, III Bombay

Machine Independent Optimizations: First Example Program

9/29

A Comparison of CFG and SSA Dumps

Dump file ccp.c.013t.cfg

if (a <= n)
 goto <bb 3>;
else
 goto <bb 5>;
<bb 5>:
 if (a <= 11)
 goto <bb 6>;
else
 goto <bb 7>;</br/>
else

Dump file ccp.c.017t.ssa

```
<bb 4>:
    # a_1 = PHI <a_3(2), a_7(3)>
    if (a_1 <= n_6)
        goto <bb 3>;
    else
        goto <bb 5>;

<bb 5>:
    if (a_1 <= 11)
        goto <bb 6>;
    else
        goto <bb 7>;
```

A Comparison of CFG and SSA Dumps

Notes

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay

30 June 2011

Machine Independent Optimizations: First Example Program

9/29

A Comparison of CFG and SSA Dumps



Machine Independent Optimizations: First Example Program A Comparison of CFG and SSA Dumps

Dump file ccp.c.013t.cfg

Dump file ccp.c.017t.ssa

```
<bb 6>:
D.1200 = a + b;
a = D.1200 + c;
<bb 7>:
D.1201 = a;
return D.1201;
```

```
<bb 6>:
 D.1200_8 = a_1 + b_4;
  a_9 = D.1200_8 + c_5;
<bb 7>:
  \# a_2 = PHI < a_1(5), a_9(6) >
 D.1201_10 = a_2;
  return D.1201_10;
```

Essential Abstractions in GCC



30 June 2011

Machine Independent Optimizations: First Example Program

10/29

Copy Renaming

Input dump: ccp.c.017t.ssa

<bb >7>: $\# a_2 = PHI < a_1(5), a_9(6) >$ $D.1201_10 = a_2;$ return D.1201_10;

Output dump: ccp.c.022t.copyrename1

```
<bb 7>:
 \# a_2 = PHI < a_1(5), a_9(6) >
 a_10 = a_2;
  return a_10;
```

A Comparison of CFG and SSA Dumps

30 June 2011

Essential Abstractions in GCC

10/29

30 June 2011

Machine Independent Optimizations: First Example Program

Copy Renaming





11/29

11/29

First Level Constant and Copy Propagation

```
Input dump: ccp.c.022t.copyrename1
                                      Output dump: ccp.c.023t.ccp1
 <bb 2>:
                                     <bb >>>:
   a_3 = 1;
                                       a_3 = 1;
   b_4 = 2;
                                       b_4 = 2:
   c_5 = 3;
   n_6 = c_5 * 2;
                                       n_6 = 6:
   goto <bb 4>;
                                       goto <bb 4>;
 <bb 3>:
                                     <bb 3>:
   a_7 = a_1 + 1;
                                       a_7 = a_1 + 1;
 <bb 4>:
                                     <bb 4>:
   \# a_1 = PHI < a_3(2), a_7(3) >
                                       \# a_1 = PHI < 1(2), a_7(3) >
   if (a_1 <= n_6)
                                       if (a_1 <= 6)
     goto <bb 3>;
                                         goto <bb 3>;
   else
                                       else
     goto <bb 5>;
                                         goto <bb 5>;
```

Essential Abstractions in GCC



30 June 2011

Machine Independent Optimizations: First Example Program

11/29

First Level Constant and Copy Propagation

Input dump: ccp.c.022t.copyrename1

Output dump: ccp.c.023t.ccp1

```
<bb 2>:
 a_3 = 1:
 b_4 = 2;
 c_5 = 3;
 n_6 = 6;
 goto <bb 4>;
<bb 6>:
 D.1200_8 = a_1 + b_4;
 a_9 = D.1200_8 + c_5;
```

```
<bb 2>:
  a_3 = 1:
  b_4 = 2:
  c_5 = 3;
  n_6 = 6;
  goto <bb 4>;
<bb 6>:
 D.1200_8 = a_1 + 2;
  a_9 = D.1200_8 + 3:
```

First Level Constant and Copy Propagation

Essential Abstractions in GCC



30 June 2011

Machine Independent Optimizations: First Example Program

11/29

First Level Constant and Copy Propagation



Second Level Copy Propagation

Second Level Copy Propagation

<bb 6>: $D.1200_8 = a_1 + 2;$ $a_9 = D.1200_8 + 3;$ <bb >7>: $\# a_2 = PHI < a_1(5), a_9(6) >$ $a_10 = a_2;$ return a_10;

Input dump: ccp.c.023t.ccp1

Output dump: ccp.c.027t.copyprop1

```
<bb 6>:
 a_9 = a_1 + 5;
 \# a_2 = PHI < a_1(5), a_9(6) >
 return a_2;
```



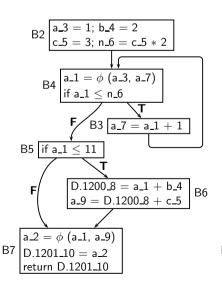
Essential Abstractions in GCC

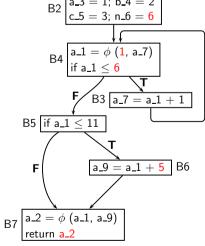
13/29

30 June 2011

Machine Independent Optimizations: First Example Program

The Result of Copy Propagation and Renaming







30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bomb

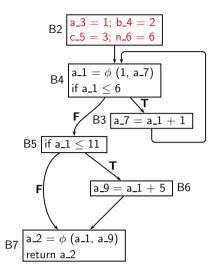
30 June 2011

Machine Independent Optimizations: First Example Program

13/29

The Result of Copy Propagation and Renaming

The Result of Copy Propagation and Renaming



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

Essential Abstractions in GCC

GCC Resource Center, IIT Bombar



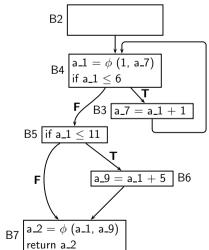
30 June 2011

Machine Independent Optimizations: First Example Program

15/29

Dead Code Elimination Using Control Dependence

Dump file ccp.c.029t.cddce1



```
<bb 2>:
  goto <bb 4>;
<bb 3>:
  a_7 = a_1 + 1;
<bb 4>:
  \# a_1 = PHI < 1(2), a_7(3) >
  if (a_1 <= 6) goto <bb 3>;
  else goto <bb 5>;
<bb 5>:
  if (a_1 <= 11) goto <bb 6>;
  else goto <bb 7>;
<bb 6>:
  a_9 = a_1 + 5;
<bb 7>:
 \# a_2 = PHI < a_1(5), a_9(6) >
  return a_2;
```

The Result of Copy Propagation and Renaming

30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bomb

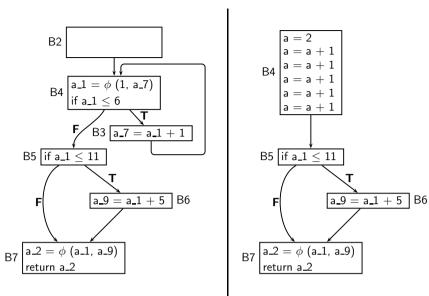
30 June 2011

Machine Independent Optimizations: First Example Program

15/29

Dead Code Elimination Using Control Dependence





Essential Abstractions in GCC

GCC Resource Center, IIT Bombay



30 June 2011

30 June 2011

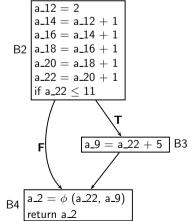
Machine Independent Optimizations: First Example Program

17/29

Complete Unrolling of Inner Loops

Dump file: ccp.c.058t.cunrolli

```
<bb 2>:
 a_12 = 2;
  a_14 = a_12 + 1;
  a_16 = a_14 + 1;
  a_18 = a_16 + 1;
  a_20 = a_18 + 1;
  a_22 = a_20 + 1;
  if (a_22 <= 11) goto <bb 3>;
  else goto <bb 4>;
<bb >3>:
 a_9 = a_22 + 5;
<bb 4>:
 \# a_2 = PHI < a_2(2), a_9(3) >
  return a_2;
```



30 June 2011

Essential Abstractions in GCC

GCC Resource Center, IIT Bomba

30 June 2011

Machine Independent Optimizations: First Example Program

17/29

Complete Unrolling of Inner Loops

Another Round of Constant Propagation

```
Input
      a_12 = 2
      a_14 = a_12 + 1
      a_16 = a_14 + 1
  B2 a_18 = a_16 + 1
      a_20 = a_18 + 1
```

```
a_22 = a_20 + 1
if a_22 ≤ 11
        a_9 = a_2 + 5 B3
```

```
Dump file: ccp.c.059t.ccp2
```

```
main ()
  <bb 2>:
    return 12;
```

Essential Abstractions in GCC



Second Example Program

Another Round of Constant Propagation

30 June 2011



Essential Abstractions in GCC

Example Program 2

int f(int b, int c, int n)
{ int a;

 do
 {
 a = b+c;
 }
 while (a <= n);

 return a;
}</pre>

We use this program to illustrate the following optimizations:

Partial Redundancy Elimination, Copy Propagation, Dead Code Elimination

Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bombay



30 June 2011 Machine Inde

Machine Independent Optimizations: Second Example Program

20/29

Compilation Command

\$gcc -fdump-tree-all -02 -S ccp.c



Notes

Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bombay `

20/29

ľ

Machine Independent Optimizations: Second Example Program

Compilation Command



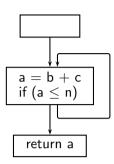




Example Program 2

loop.c int f(int b, int c, int n) { int a; do a = b+c;while $(a \le n)$; return a;

Control Flow Graph





Essential Abstractions in GCC

30 June 2011

22/29

Machine Independent Optimizations: Second Example Program **Dump of Input to PRE Pass**

```
B2
        a = b + c
  B3
        if (a < n)
           B5
B6
   return a
```

Control Flow Graph

```
loop.c.091t.crited
<bb 2>:
<bb 3>:
 a_3 = b_1(D) + c_2(D);
 if (a_3 \le n_4(D)) goto \le 5>;
 else goto <bb 6>;
<bb 5>:
 goto <bb 3>;
<bb 6>:
<bb 4>:
 \# a_6 = PHI < a_3(6) >
 return a_6;
```

Example Program 2

30 June 2011

Essential Abstractions in GCC

30 June 2011 Machine Independent Optimizations: Second Example Program

22/29

Dump of Input to PRE Pass



Machine Independent Optimizations: Second Example Program Input and Output of PRE Pass

```
loop.c.091t.crited
                                  loop.c.092t.pre
                                  <bb 2>:
<bb 2>:
                                    pretmp.2_7 = b_1(D) + c_2(D);
<bb 3>:
                                  <bb 3>:
  a_3 = b_1(D) + c_2(D);
                                    a_3 = pretmp.2_7;
                                    if (a_3 \le n_4(D))
 if (a_3 \le n_4(D))
       goto <bb 5>;
                                         goto <bb 5>;
  else goto <bb 6>;
                                    else goto <bb 6>;
<bb 5>:
                                  <bb 5>:
  goto <bb 3>;
                                    goto <bb 3>;
<bb 6>:
                                  <bb 6>:
<bb 4>:
                                  <bb 4>:
  \# a_6 = PHI < a_3(6) >
                                    \# a_6 = PHI < a_3(6) >
 return a_6;
                                    return a_6;
```

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay



30 June 2011 Machine Independent Optimizations: Second Example Program

24/29

Copy Propagation after PRE

```
loop.c.097t.copyprop4
loop.c.092t.pre
                                  <bb 2>:
<bb 2>:
                                    pretmp.2_7 = b_1(D) + c_2(D);
 pretmp.2_7 = b_1(D) + c_2(D);
                                   <bb 3>:
<bb 3>:
                                    a_3 = pretmp.2_7;
 a_3 = pretmp.2_7;
                                    if (n_4(D) >= pretmp.2_7)
 if (a_3 \le n_4(D))
                                      goto <bb 4>;
       goto <bb 5>;
                                    else
  else goto <bb 6>;
                                      goto <bb 5>;
<bb 5>:
                                   <bb 4>:
 goto <bb 3>;
                                    goto <bb 3>;
<bb 6>:
                                   <bb 5>:
<bb 4>:
                                    # a_8 = PHI oretmp.2_7(3)>
 \# a_6 = PHI < a_3(6) >
                                    a_6 = a_8;
                                    return a_8;
 return a_6;
```

Input and Output of PRE Pass

Notes

Essential Abstractions in GCC

30 June 2011

GCC Resource Center, IIT Bombay



Machine Independent Optimizations: Second Example Program

24/29

Copy Propagation after PRE



Dead Code Elimination

```
loop.c.098t.dceloop1
loop.c.097t.copyprop4
<bb 2>:
 pretmp.2_7 = b_1(D) + c_2(D);
                                  <bb 2>:
<bb 3>:
                                    pretmp.2_7 = b_1(D) + c_2(D);
 a_3 = pretmp.2_7;
                                  <bb 3>:
 if (n_4(D) >= pretmp.2_7)
    goto <bb 4>;
                                    if (n_4(D) \ge pretmp.2_7)
  else
                                      goto <bb 4>;
    goto <bb 5>;
                                    else
                                      goto <bb 5>;
<bb 4>:
  goto <bb 3>;
                                   <bb 4>:
                                    goto <bb 3>;
<bb 5>:
  \# a_8 = PHI < pretmp.2_7(3) >
                                  <bb 5>:
                                    # a_8 = PHI oretmp.2_7(3)>
  a_6 = a_8;
 return a_8;
                                    return a_8;
```

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay



30 June 2011

Machine Independent Optimizations: Second Example Program

26/29

Redundant ϕ Function Elimination and Copy Propagation

```
loop.c.098t.dceloop1
<bb 2>:
    pretmp.2_7 = b_1(D) + c_2(D);

<bb 3>:
    if (n_4(D) >= pretmp.2_7)
        goto <bb 4>;
    else
        goto <bb 5>;

<bb 4>:
        goto <bb 3>;

<bb 5>:
    # a_8 = PHI <pretmp.2_7(3)>
    return a_8;
```

```
loop.c.125t.phicprop2

<bb 2>:
    pretmp.2_7 = c_2(D) + b_1(D);
    if (n_4(D) >= pretmp.2_7)
        goto <bb 4>;
    else
        goto <bb 3>;

<bb 3>:
    return pretmp.2_7;

<bb 4>:
    goto <bb 4>;
```

Dead Code Elimination

Note

Essential Abstractions in GCC

GCC Resource Center, IIT Bombay



30 June 2011 Machine Independent Optimizations: Second Example Program

26/29

Redundant ϕ Function Elimination and Copy Propagation



Final Assembly Program

```
loop.c.125t.phicprop2
                                   loop.s
<bb 2>:
                                                8(%esp), %eax
                                        movl
                                                4(%esp), %eax
 pretmp.2_7 = c_2(D) + b_1(D);
                                        addl
 if (n_4(D) >= pretmp.2_7)
                                                %eax, 12(%esp)
                                        cmpl
    goto <bb 4>;
                                        jge
                                                .L2
  else
                                        rep
    goto <bb 3>;
                                        ret
                                   .L2:
<bb 3>:
                                   .L3:
 return pretmp.2_7;
                                               .L3
                                        jmp
<bb 4>:
 goto <bb 4>;
```

Why infinite loop?





30 June 2011

Machine Independent Optimizations: Second Example Program

28/29

Infinite Loop in Example Program 2

```
int f(int b, int c, int n)
{ int a;
   do
      a = b+c;
   while (a \le n);
   return a;
}
```

The program does not terminate unless a > n

Final Assembly Program

30 June 2011

Essential Abstractions in GCC

30 June 2011

Machine Independent Optimizations: Second Example Program

28/29

Infinite Loop in Example Program 2





Part 3

Conclusions

30 June 2011

Machine Independent Optimizations: Conclusions

29/29

30 June 2011

Machine Independent Optimizations: Conclusions

Conclusions

29/29

Conclusions

- GCC performs many machine independent optimizations
- The dumps of optimizations are easy to follow, particularly at the GIMPLE level
- It is easy to prepare interesting test cases and observe the effect of transformations
- One optimization often leads to another Hence GCC performs many optimizations repeatedly (eg. copy propagation, dead code elimination)