Adventures in Dataflow Analysis

CSE 401 Section 9-ish
Jack Eggleston, Aaron Johnston, & Nate Yazdani
Announcements

- Code Generation due
Announcements

- Code Generation due

- Compiler Additions due next Thursday, 12/6
  - Involves revisiting all parts of the compiler
Announcements

- Code Generation due

- Compiler Additions due next Thursday, 12/6
  - Involves revisiting all parts of the compiler

- Final Report due the following Saturday, 12/8
  - Ideally, also involves revisiting all parts of the compiler
Review of Optimizations

Source Code → Front End → IR → Back End → Target Code

- Scanner
- Parser
- Semantic Analysis
- Optimization
- Code Generation
Review of Optimizations

Peephole
Local
Intraprocedural / Global
Interprocedural
Review of Optimizations

Peephole  A few Instructions
Local
Intraprocedural / Global
Interprocedural
Review of Optimizations

Peephole      A few Instructions
Local         A Basic Block
Intraprocedural / Global
Interprocedural
Review of Optimizations

Peephole: A few Instructions
Local: A Basic Block
Intraprocedural / Global: A Function/Method
Interprocedural
Review of Optimizations

- **Peephole**: A few Instructions
- **Local**: A Basic Block
- **Intraprocedural / Global**: A Function/Method
- **Interprocedural**: A Program
Overview of Dataflow Analysis

- A framework for exposing properties about programs
- Operates using sets of “facts”
Overview of Dataflow Analysis

- A framework for exposing properties about programs
- Operates using sets of “facts”
- Just the initial discovery phase
  - Changes can then be made to optimize based on the analysis
Overview of Dataflow Analysis

- Basic Framework of Set Definitions (for a Basic Block $b$):
  - $\text{IN}(b)$: facts true on entry to $b$
  - $\text{OUT}(b)$: facts true on exit from $b$
  - $\text{GEN}(b)$: facts created (and not killed) in $b$
  - $\text{KILL}(b)$: facts killed in $b$
Reaching Definitions (A Dataflow Problem)

“What definitions of each variable might reach this point”

- Could be used for:
  - Constant Propagation
  - Uninitialized Variables

```java
int x;
if (y > 0) {
    x = y;
} else {
    x = 0;
}
System.out.println(x);```

“x=y”, “x=0”
Reaching Definitions (A Dataflow Problem)

“What definitions of each variable might reach this point”

- **Be careful**: Does not involve the *value* of the definition
  - The dataflow problem
    “Available Expressions”
    is designed for that

```java
int x;
if (y > 0) {
    x = y;
} else {
    x = 0;
}
y = -1;
System.out.println(x);
```

**still**: “x=y”, “x=0”
1 (a) & (b)
Equations for Reaching Definitions

- \( \text{IN}(b) \): the definitions reaching upon entering block \( b \)
- \( \text{OUT}(b) \): the definitions reaching upon exiting block \( b \)
- \( \text{GEN}(b) \): the definitions assigned and not killed in block \( b \)
- \( \text{KILL}(b) \): the definitions of variables overwritten in block \( b \)

\[
\begin{align*}
\text{IN}(b) &= \bigcup_{p \in \text{pred}(b)} \text{OUT}(p) \\
\text{OUT}(b) &= \text{GEN}(b) \cup (\text{IN}(b) - \text{KILL}(b))
\end{align*}
\]
Another *Equivalent* Set of Equations (from Lecture):

- **Sets:**
  - DEFOUT(b): set of definitions in b that reach the end of b (i.e., not subsequently redefined in b)
  - SURVIVED(b): set of all definitions not obscured by a definition in b
  - REACHES(b): set of definitions that reach b

- **Equations:**
  
  \[
  \text{REACHES}(b) = \bigcup_{p \in \text{preds}(b)} \text{DEFOUT}(p) \cup
  (\text{REACHES}(p) \cap \text{SURVIVED}(p))
  \]
1 (c) & (d)
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c

<table>
<thead>
<tr>
<th>Block</th>
<th>GEN</th>
<th>KILL</th>
<th>IN (1)</th>
<th>OUT (1)</th>
<th>IN (2)</th>
<th>OUT (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>L0</td>
<td>L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td>L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2</td>
<td>L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L3</td>
<td>L3</td>
<td>L0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
L0:  \( a = 0 \)
L1:  \( b = a + 1 \)
L2:  \( c = c + b \)
L3:  \( a = b \times 2 \)
L4:  if \( a < N \) goto L1
L5:  return \( c \)

<table>
<thead>
<tr>
<th>Block</th>
<th>GEN</th>
<th>KILL</th>
<th>IN (1)</th>
<th>OUT (1)</th>
<th>IN (2)</th>
<th>OUT (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>L0</td>
<td>L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td>L1</td>
<td></td>
<td></td>
<td>L0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2</td>
<td>L2</td>
<td></td>
<td></td>
<td>L0, L1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L3</td>
<td>L3</td>
<td>L0</td>
<td>L0, L1, L2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L4</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L5</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
L0: \( a = 0 \)  
L1: \( b = a + 1 \)  
L2: \( c = c + b \)  
L3: \( a = b \times 2 \)  
L4: if \( a < N \) goto L1  
L5: return c

<table>
<thead>
<tr>
<th>Block</th>
<th>GEN</th>
<th>KILL</th>
<th>IN (1)</th>
<th>OUT (1)</th>
<th>IN (2)</th>
<th>OUT (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>L0</td>
<td>L3</td>
<td></td>
<td>L0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td>L1</td>
<td></td>
<td>L0</td>
<td>L0, L1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2</td>
<td>L2</td>
<td></td>
<td>L0, L1</td>
<td>L0, L1, L2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L3</td>
<td>L3</td>
<td>L0</td>
<td>L0, L1, L2</td>
<td>L1, L2, L3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L4</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L5</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c

<table>
<thead>
<tr>
<th>Block</th>
<th>GEN</th>
<th>KILL</th>
<th>IN (1)</th>
<th>OUT (1)</th>
<th>IN (2)</th>
<th>OUT (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>L0</td>
<td>L3</td>
<td>L0</td>
<td>L0</td>
<td></td>
<td>L0</td>
</tr>
<tr>
<td>L1</td>
<td>L1</td>
<td></td>
<td>L0, L1</td>
<td>L0, L1</td>
<td>L0, L1, L2, L3</td>
<td>L0, L1, L2, L3</td>
</tr>
<tr>
<td>L2</td>
<td>L2</td>
<td></td>
<td>L0, L1</td>
<td>L0, L1, L2</td>
<td>L0, L1, L2, L3</td>
<td>L0, L1, L2, L3</td>
</tr>
<tr>
<td>L3</td>
<td>L3</td>
<td>L0</td>
<td>L0, L1, L2</td>
<td>L1, L2, L3</td>
<td>L0, L1, L2, L3</td>
<td>L1, L2, L3</td>
</tr>
<tr>
<td>L4</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
</tr>
<tr>
<td>L5</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
</tr>
</tbody>
</table>
L0:  a = 0
L1:  b = a + 1
L2:  c = c + b
L3:  a = b * 2
L4:  if a < N goto L1
L5:  return c

<table>
<thead>
<tr>
<th>Block</th>
<th>GEN</th>
<th>KILL</th>
<th>IN (1)</th>
<th>OUT (1)</th>
<th>IN (2)</th>
<th>OUT (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>L0</td>
<td>L3</td>
<td>L0</td>
<td>L0</td>
<td></td>
<td>L0</td>
</tr>
<tr>
<td>L1</td>
<td>L1</td>
<td></td>
<td>L0</td>
<td>L0, L1</td>
<td>L0, L1, L2, L3</td>
<td>L0, L1, L2, L3</td>
</tr>
<tr>
<td>L2</td>
<td>L2</td>
<td></td>
<td>L0, L1</td>
<td>L0, L1, L2</td>
<td>L0, L1, L2, L3</td>
<td>L0, L1, L2, L3</td>
</tr>
<tr>
<td>L3</td>
<td>L3</td>
<td>L0</td>
<td>L0, L1, L2</td>
<td>L1, L2, L3</td>
<td>L0, L1, L2, L3</td>
<td>L1, L2, L3</td>
</tr>
<tr>
<td>L4</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
</tr>
<tr>
<td>L5</td>
<td></td>
<td></td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
<td>L1, L2, L3</td>
</tr>
</tbody>
</table>
2 (a) & (b)
1. \[ Z = 4 \times B \]
   \[ Y = A + C \]

2. \[ Y = 5 \]
   \[ Z = Y + B \]

3. \[ X = A \times B \]
   \[ Z = Y + X \]

4. \[ X = A \times B \]
   \[ Z = Y + X \]

5. \[ Y = 3 \times B \]
   \[ Z = A + B \]

6. \[ Y = 3 \times B \]
   \[ X = A \times B \]

7. \[ Y = 2 \times B \]
1. 
   \[ Z = 4 \times B \]
   \[ Y = A + C \]

2. 
   \[ Y = 5 \]
   \[ Z = Y + B \]

3. 
   \[ X = A \times B \]
   \[ Z = Y + X \]
   \[ T_1 = 3 \times B \]

4. 
   \[ X = A \times B \]
   \[ Z = Y + X \]
   \[ T_2 = 2 \times B \]

5. 
   \[ Y = T_1 \]
   \[ Z = A + B \]

6. 
   \[ Y = T_1 \]
   \[ X = A \times B \]

7. 
   \[ Y = T_2 \]