Static Single Assignment Form

CS2210
Lecture 15

Reading
- Rest of chapter 8
- Chapter 9
**Static Single Assignment Form**

- **THE standard IR in optimizing compilers**
- **Properties**
  - Every use of a variable has at most one reaching definition
  - Separates values from the locations
    - Makes many optimizations more effective:
      - Constant propagation, value numbering, invariant code motion & removal, strength reduction, PRE
- **Mechanism:**
  - Create new target names for definitions by subscripting variables
  - Introduce $\phi$-functions at merge points as pseudo-definitions
  - Adjust uses
  - Want to minimize # $\phi$-functions
  - Minimizes translation overhead for code generation

---

**Dominance Frontier**

- $DF(x) = \{y \mid \exists z \in \text{pred}(y) \text{ so that } x \text{ dom}\ z \text{ and not } x \text{ sdom}\ y\}$
- Set of all CFG nodes which for which $x$ dominates a predecessor but not the node itself
- **Direct** computation is quadratic in number of CFG nodes
  - Use recursive equations and solve iteratively

CS2210 Compiler Design 2004/5
Dominance Frontier

- \( DF_{\text{local}}(x) = \{ y \in \text{succ}(x) | \text{idom}(y) \neq x \} \)
- \( DF_{\text{up}}(x, z) = \{ y \in DF(z) | \text{idom}(y) \neq x \} \)
- \( DF(x) = DF_{\text{local}}(x) \cup \{ z \in N, \text{idom}(z) = x \} DF_{\text{up}}(x, z) \)

Example

- \( \text{Dom}(5) = \{5, 6, 7, 8\} \)
- \( DF(5) = \{4, 5, 12, 13\} \)
Dominance Frontier Algorithm

Procedure ComputeDF(n)
begin
    S := {};
    for each node y in succ(n) do
        if idom(y) != n
            S := S U {y}; /* this loop computes Dflocal(n) */
    od
    for each child c of n in the dominator tree do
        compute DF[c]
        for each element w of DF[c] do
            if n does not dominate w or if n = w
                S := S U {w};
            fi
        od
    end
    DF[n] = S
end

Algorithm is
O(E + sizeof(DFs))
In practice linear in size
Of graph

Dominance Frontier Criterion

- Whenever node x contains a definition of some variable a, then any node in the dominance frontier of x needs a \( \Phi \)-function
  - Dominance frontier earliest points where definition is not guaranteed to be unique
  - Since \( \Phi \)-functions are definitions themselves have to iterate
Iterated Dominance Frontier

- Define DF for set of nodes:
  \[ DF(S) = \bigcup_{x \in S} DF(x) \]

- **Iterated Dominance Frontier:**
  \[ DF^+(S) = \lim_{i \to \infty} DF^i(S), \text{ where } \]
  \[ DF^{i+1}(S) = DF(S \cup DF^i(S)), \text{ and } DF^1(S) = DF(S) \]

- If \( S \) is set of nodes that assign to variable \( x \) (including the entry node) then \( DF^+(S) \) is the set of nodes that need \( \phi \)-functions for \( x \)

---

Example

- On board
CFG

Dominator Tree
## Dominance Frontiers

<table>
<thead>
<tr>
<th>node n</th>
<th>DF(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{}</td>
</tr>
<tr>
<td>2</td>
<td>{2}</td>
</tr>
<tr>
<td>3</td>
<td>{2}</td>
</tr>
<tr>
<td>4</td>
<td>{}</td>
</tr>
<tr>
<td>5</td>
<td>{7}</td>
</tr>
<tr>
<td>6</td>
<td>{7}</td>
</tr>
<tr>
<td>7</td>
<td>{2}</td>
</tr>
</tbody>
</table>

## Iterated Dominance Frontiers

- For i: $DF^1(\{1\}) = DF(\{1\}) = {}$
  - no phi functions for i
- For k,j:
  - $DF^1(\{1,5,6\}) = DF(\{1,5,6\}) = \{7\}$
  - $DF^2(\{1,5,6\}) = DF(\{1,5,6\} \cup DF^1(\{1,5,6\})) =$
    - $DF(\{1,5,6,7\}) = \{2,7\}$
  - $DF^3(\{1,5,6\}) = DF(\{1,5,6\} \cup DF^2(\{1,5,6\})) =$
    - $DF(\{1,5,6,7,2\}) = \{2,7\} = DF^+(\{1,5,6\})$
  - phi functions in blocks 2 and 7
Phi-Function Insertion

Variable Renaming