Error Detecting Instructions in Superscalar

Center for Reliable Computing

Error Detecting Instructions In Superscalar

Nahmsuk Oh
Center for Reliable Computing
Stanford University
RATS Seminar
April 5, 1999

Outline
- Previous work
- Error detecting instructions
- New work
  - EDI in Superscalar architecture
  - Experimental result
- Future Work

Previous Work
- Software error detection
- Signature Analysis by Instructions
- Error detecting Instructions

Error Detecting Instruction (EDI)
- Shadow instruction
- Duplicated instruction
- Comparison instruction
  - Compare master & shadow results
- Signature analysis instruction
  - Control flow checking between basic blocks

EDI Example
- Comparison right before store instruction

Example

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD R3, R1, R2</td>
<td>( R3 \leftarrow R1 + R2 )</td>
</tr>
<tr>
<td>MUL R4, R1, R3</td>
<td>( R4 \leftarrow R1 \times R3 )</td>
</tr>
<tr>
<td>ST 0(SP), R4</td>
<td>store R4 where SP points to.</td>
</tr>
<tr>
<td>ADD R3, R1, R2</td>
<td>an error corrupts addition here</td>
</tr>
<tr>
<td>MUL R4, R1, R3</td>
<td>( R3 ) contains incorrect value</td>
</tr>
<tr>
<td>MUL R24, R21, R23</td>
<td>( R4 = R24 )</td>
</tr>
<tr>
<td>BNE R4, R24, gotoError</td>
<td>branch to error handler</td>
</tr>
<tr>
<td>ST 0(SP), R4</td>
<td></td>
</tr>
<tr>
<td>ST offset(SP), R24</td>
<td></td>
</tr>
</tbody>
</table>

Drawback
- No optimization for EDI
- Overhead
  - Performance compared to fully optimized: 200 ~ 300%
Super-scalar Architecture
- Multi instructions issued in one cycle
- Multi computation units
  - R10000
    - Integer unit: ALU0, ALU1
    - Floating point unit: Adder, Multiplier
    - Memory access unit

New Work
- Resources NOT fully utilized in Superscalar
  - Data dependency
  - Control dependency
  - Limited resource in one cycle
- Idle resources for EDI

No Dependency

Dependency

EDI in Idle Resources (List Scheduling Algorithm)

EDI Scheduling Algorithm
- Integer Linear Programming
  - Exact & optimum solution
  - NP complete
- List Scheduling Algorithm
  - Heuristic Algorithm
  - Complexity = $O(n)$
Error Detecting Instructions in Superscalar

Experimental Result
- Insertion sort
  - Manual scheduling & optimization
- Processors
  - Superscalar: R10000
  - Non-superscalar: R4000

Performance Overhead

<table>
<thead>
<tr>
<th>R10000: 31%</th>
<th>R4000: 70%</th>
</tr>
</thead>
<tbody>
<tr>
<td>No EDI</td>
<td>3.602</td>
</tr>
<tr>
<td>With EDI</td>
<td>7.187</td>
</tr>
</tbody>
</table>

Future Work
- Enhance CAP (Post processor)
  - Support optimization
  - Efficient scheduling
- Experiment with more programs