Temporal Redundancy

Introduction

- Structural (Physical) Redundancy
  - Replicated hardware
  - Fast detection (duplication)
  - Fast localization (majority voting)
- Temporal Redundancy
  - Replicate in time
  - Provided by software

Temporal Redundancy

- Multiple Execution of Instructions
- High Latency
- Simple Method for Uniprocessor System
  - $2 \times$ execution time + comparison time
  - Does not improve coverage for permanent faults

Stutter Step Mode

- Definition
  - Repeat calculation and compare results
- Catch Transient Errors
  - Data transformations

Simple ALU Instructions

<table>
<thead>
<tr>
<th>$A \leftarrow B \text{ op. } C$</th>
<th>$A \leftarrow B \text{ op. } C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A' \leftarrow B \text{ op. } C$</td>
<td>$\text{cmp } A, A'$</td>
</tr>
<tr>
<td></td>
<td>$\text{bne err_handler}$</td>
</tr>
</tbody>
</table>

- Any Single Error
- Size of Pieces That Are Repeated
Temporal Redundancy

Reduce Overhead

<table>
<thead>
<tr>
<th>Operation</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>A op. B</td>
<td>A op. B</td>
</tr>
<tr>
<td>C op. D</td>
<td>C op. D</td>
</tr>
<tr>
<td>A op. B</td>
<td>C op. D</td>
</tr>
<tr>
<td>A op. B</td>
<td>C op. D</td>
</tr>
</tbody>
</table>

- A Series of Computations
  - A basic block
  - Compare the live variables at the end
  - An expression

More Error Detection

- When Added to Structural Redundancy
  - Detect Errors Not Observable at Pins
    - Collecting errors
    - Common-Mode Failure

Limits of Stutter Step Mode

- Errors in Data Transformation Instructions
- Limited Benefit for Memory Instructions
- Not Applicable to
  - Control transfer instructions
  - Control flow checking techniques
  - Special instruction types
  - Swap (atomic operation)

Alternatives

- Affine Transformations
  - Use checksums instead of code duplication
- Multithreading
  - Shadow data structures
  - Synchronize and compare

Multitasking on MP Systems

- Multiple Copies of Each Job
  - Synchronously
  - Asynchronously (with delay)
- Results Compared in Shared Memory
- Original and Copy Job on Different Processors

Example

```
   1
  /|
 / |
/  |
11
```

```
  2
 /|
/ |
/  |
12
```

```
  3
 /|
/ |
/  |
15
```

```
  4
 /|
/ |
/  |
14
```

Philip Shirvani - RATS  Page 2  5/5/97
Temporal Redundancy

Process Pools

Scheduling

Temporal Job Redundancy (1)
- With 3 or More Processors
- Lower Execution Time
  - Original jobs run with maximal parallelism
  - Dynamic occupation of free processors by copy jobs, less idle time
  - In normal operation and in presence of faults
- Disadvantages
  - Hard to Implement
  - Error detection may be delayed

Temporal Job Redundancy (2)
- With Triple Redundancy
  - No need to start 3rd if 1st and 2nd agree
- Inefficient Use of Resources
  - Important jobs duplicated
- Hybrid of Structural and Temporal Redundancy
  - For critical parts
  - Fast error detection

Process Pairs (1)
- Idea: Checkpointing and Restart
- Primary Process
  - Provides service
  - Sends checkpoints to the backup
- Backup Process
  - Process state; not executed
  - Takes over when primary fails
- Primary and Backup on Different Processors
- Separate Queues for Messages
  - Counter for sent messages

Process Pairs (2)
- Limited Error Detection
  - Hardware
    - Segment faults, illegal instruction
  - Software
    - Assertions, unrecoverable error
- Implementation Complexity
  - OS level, no change in applications
Temporal Redundancy

Software Reliability
- Improvement in Hardware Reliability
- Redundancy = Wasting Resources
- Software Bugs Becoming Major Problem
  - Source of 62% of outages in an example
  - Multiple execution of a buggy code?!?
- N-Version Programming
  - Design errors
- Process Pairs
  - Less complexity than N-version prog.

Tandem GUARDIAN System
- Message-based MP System
- On-line Transaction Processing
  - Availability, fast recovery
- Process Pairs for Critical Jobs
  - Intended for tolerating hardware faults
  - Little wasted redundancy

Handling Software Errors!
- Tolerates 70% of Faults in System Software That Cause Processor Failure
  - Processor state and sequence of events different; a software bug
  - Subtle faults (transient software faults) not exercised on restart (29%)
- Same Effect in Temporal Job Redundancy?
  - Used for error detection or recovery?

Conclusion
- Critical Outputs
  - Structural redundancy
- Temporal Redundancy Techniques
  - Error detection
    - Error in software
    - Error in hardware
  - Error detection latency
  - Error recovery