Motivation

Technology scaling ⇒ Increased in-the-field failures for commodity systems
- Wear-out, infant mortality, design defects, etc.
Need low-cost in-field techniques for detection, diagnosis, recovery, repair

SWAT - SoftWare Anomaly Treatment

Strategy

Watch for anomalous software behavior ⇒ Symptom
- Zero/low cost "always-on" monitors
Diagnose fault after detection
- Rarely invoked ⇒ may incur higher overheads

Previous results for SPEC
- 0.8% of faults result in SDCs
- 95% of faults detected in 10M instructions
⇒ Recovery needs checkpoint/buffer window of 10M

This work: Application-aware methods to improve SDC rate, recovery window

Using Application-Awareness for SDCs, Recovery Window

- Low-Cost Application-aware Address Out-of-Bounds detector
- Application-aware SDC and recovery window analysis
- Baseline SWAT on new I/O intensive client/server apps for I/O analysis

Results: Order of magnitude improvement in SDC rate, recovery window, output buffer size

Application aware Address Out-of-Bounds Detector

Amortize resiliency cost for HW/SW faults
- SW bug detection uses such detectors
Low-cost detector that monitors bounds
- HW faults ⇒ invalid/unallocated address
HW/SW coordination to identify legal bounds

Results
- 50% faults detected by new detector
- Dramatic reduction in recovery window
- Reduces system state corruption by half

Application-Aware SDC Analysis

Fault corrupts output produced by application traditionally ⇒ SDC
But some applications, even SPEC, tolerate errors in outputs!
Fault activation influences detection ⇒ Round-robin scheduling ↓ SDCs

Application-Aware SDC Rate of 8 SPEC Apps

<table>
<thead>
<tr>
<th>Output error tolerance</th>
<th>SW</th>
<th>App-Aware SW</th>
<th>SW</th>
<th>App-Aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permanent Faults</td>
<td></td>
<td>54</td>
<td>2</td>
<td>14</td>
</tr>
<tr>
<td>Transient Faults</td>
<td></td>
<td>7</td>
<td>14</td>
<td>0</td>
</tr>
</tbody>
</table>

Results:
- 50% faults detected by new detector
- Dramatic reduction in recovery window
- New 10K instruction window needs only 30 stores!
- Can be buffered using store buffer
- New techniques have dramatic effects on recovery

Implications of Recovery Window for I/O and Recoverability

Larger recovery window ⇒ Overhead for buffering I/O, user perception

Results:
- 10M instruction window needs 80KB buffer
- New 10K instruction window needs only 30 stores!
- Can be buffered using store buffer
- New techniques have dramatic effects on recovery

Conclusions and Future Work

Application Awareness ⇒ Much lower SDC rate, shorter recovery window, less I/O buffering

Future Work: App-aware SDC analysis of distributed client/server applications
- Low overhead recovery techniques for short latency