Modular Redundancy for Soft-Error Resilience in Large-Scale HPC Systems

Christian Engelmann

Computer Science and Mathematics Division
Oak Ridge National Laboratory
Trends in HPC System Reliability

- HPC systems continue to increase in size
  - Error rate increases due to higher component count
- HPC systems may increasingly contain accelerators
  - Soft error rate increases due to higher vulnerability
- Nanometer technology continues to decrease
  - Soft error rate increases further due to higher vulnerability
- HPC vendors continue to use mass-market components
  - Mass-market demands define HPC system reliability

Future HPC systems won’t be as reliable as today’s
Soft errors are a major concern for HPC resilience
Motivation for Modular Redundancy in HPC

- Redundancy on compute nodes is not entirely new
  - Diskless checkpointing (Plank et al.)
  - Algorithmic redundancy approaches (Dongarra et al.)

- Until now, the HPC community (researchers and vendors) stayed away from modular redundancy
  - “Big hammer” approach with fully redundant compute nodes

⇒ With increasing hard and (especially) soft error rates, compute-node redundancy needs to be considered as an alternative to checkpointing and preemptive migration

⇒ Respective research and development in modular redundancy for HPC environments is needed
Trends in HPC System Resilience

- Checkpoint/restart has limits
  - Efficiency decreases with higher error rate
  - Efficiency decreases further with larger aggregated memory
  - Incremental/compression approaches help in the short term
  - Preemptive migration helps further in the long term

- Preemptive migration has also limits
  - Error rate increases with lower prediction accuracy
  - Errors without precursor or pattern can’t be predicted
    - Can anyone predict a non-recoverable ECC memory error?

⇒ Future HPC systems won’t be as resilient as today’s
⇒ Resiliency strategy for high soft error rates is missing
System Availability Basics
(Terms, Concepts, Models and Metrics)

- A system’s availability can be between 0 and 1, or 0% and 100%
- A system’s availability in the long-run is based on its
  - Mean-time to failure (MTTF)
  - Mean-time to recover (MTTR)
- A system is rated by the number of nines in its availability metric
- Dependent system components are coupled serial
- Redundant system components are coupled parallel
- System components may have equal MTTF and MTTR

\[ A = \frac{MTTF}{MTTF + MTTR} = \frac{1}{1 + \frac{MTTR}{MTTF}} \]

<table>
<thead>
<tr>
<th>Availability</th>
<th>Annual Downtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>90%</td>
<td>36 days, 12 hours</td>
</tr>
<tr>
<td>99%</td>
<td>87 hours, 36 minutes</td>
</tr>
<tr>
<td>99.9%</td>
<td>8 hours, 45.6 minutes</td>
</tr>
<tr>
<td>99.99%</td>
<td>52 minutes, 33.6 seconds</td>
</tr>
<tr>
<td>99.999%</td>
<td>5 minutes, 15.4 seconds</td>
</tr>
<tr>
<td>99.9999%</td>
<td>31.5 seconds</td>
</tr>
</tbody>
</table>

\[
A_{\text{series}} = \prod_{i=1}^{n} A_i \\
A_{\text{parallel}} = 1 - \prod_{i=1}^{n} (1 - A_i) \\
A_{\text{equal-series}} = A_{\text{component}}^n \\
A_{\text{equal-parallel}} = 1 - (1 - A_{\text{component}})^n
\]
HPC System Availability at Scale
(5, 6 and 7 Nines Compute Node Availability)
Improving System Availability with Modular Redundancy

- Modular redundancy concepts have been around for a while
  - E.g. aerospace and command & control systems
- System availability is improved using redundant components
- Dual-modular redundancy (DMR) offers protection against hard errors and some soft errors
- Triple-modular redundancy (TMR) offers protection against hard and soft errors
- Dynamic dual- or triple-modular redundancy uses reboot or spare to reduce component MTTR

\[
A_{DMR} = 1 - (1 - A)^2 \\
A_{TMR} = 1 - (1 - A)^3
\]

\[
A_{DDMR} = 1 - (1 - A_1)(1 - A_2) \\
A_{DTMR} = 1 - (1 - A_1)(1 - A_2)^2
\]
Improving Compute Node Availability with Modular Redundancy

- Today’s large-scale HPC systems have tens-to-hundreds of thousands of diskless compute nodes consisting of
  - processor(s), memory module(s) and a network interface
- Deploying modular redundancy for these systems would require to double or triple the number of compute nodes
- However, the network infrastructure is able to recover soft errors by retransmitting messages
- We only need to double or triple the number of processors and memory modules within each compute node
- A modular redundancy mechanism is needed for replication, error detection and error recovery in a massively parallel HPC system
Compute Node Availability Improvement with Modular Redundancy

![Graph showing the relationship between component availability and system availability for simplex, duplex, and triplex redundancy systems. The graph illustrates that triplex redundancy provides the highest system availability for a given component availability.]
Compute Node Availability Improvement with Dynamic Modular Redundancy

\[ \frac{MTTR_1}{MTTR_2} = 60 \]
Improving HPC System Availability with Compute-Node Modular Redundancy

- The availability of a modular redundant compute node is based on $2\times/3\times$ parallel coupling
- The availability of a HPC system is based on $n\times$ serial coupling
- The availability of a compute-node modular redundant HPC system is based on $n\times$ serial of $2\times/3\times$ parallel components
- Dynamic modular redundancy additionally reduces the MTTR of 1 (DMR) or 2 (TMR) components

\[ A_{DMR} = [1 - (1 - A)^2]^n \]
\[ A_{TMR} = [1 - (1 - A)^3]^n \]

\[ A_{DDMR} = [1 - (1 - A_1)(1 - A_2)]^n \]
\[ A_{DTMR} = [1 - (1 - A_1)(1 - A_2)^2]^n \]
HPC System Availability Improvement with Modular Redundancy
(2, 3 and 4 Nines Compute Node Availability)
HPC System Availability Improvement with Dynamic Modular Redundancy (2, 3 and 4 Nines Compute Node Availability)
Observations

- DMR and TMR for compute nodes significantly increases compute node availability, which in turn dramatically increases HPC system availability
  - DMR: Compute node MTTF can be 100-1,000× less
  - TMR: Compute node MTTF can be 1,000-10,000× less
- DDMR and DTMR for compute nodes improve compute node availability even further, which in turn increases HPC system availability even more
  - DDMR: Compute node MTTF can be 1,000-10,000× less
  - DTMR: Compute node MTTF can be 10,000-100,000× less
# Financial Cost and Power Consumption
(Based on Current ORNL Jaguar Hardware and Market Prices)

<table>
<thead>
<tr>
<th>Solution</th>
<th>Processor</th>
<th>Memory</th>
<th>Price</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional checkpoint/restart</td>
<td>1x AMD Opteron 2356</td>
<td>2x4GB Micron DDR2-800 ECC</td>
<td>$ 500</td>
<td>75W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+$750</td>
<td>+ 2W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= $1250</td>
<td>= 77W</td>
</tr>
<tr>
<td>In-memory checkpoint caching</td>
<td>1x AMD Opteron 2356</td>
<td>4x4GB Micron DDR2-800 ECC</td>
<td>$ 500</td>
<td>75W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+$1500</td>
<td>+ 4W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= $2000</td>
<td>= 79W</td>
</tr>
<tr>
<td>In-memory checkpoint/restart with new boards</td>
<td>1x AMD Opteron 2356</td>
<td>2x4GB Micron DDR2-800 ECC</td>
<td>$ 500</td>
<td>75W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+$750</td>
<td>+ 2W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+$600</td>
<td>+ 4W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt;$1700</td>
<td>= 81W</td>
</tr>
<tr>
<td>DMR w/new boards &amp; more racks</td>
<td>2x AMD Opteron 2356</td>
<td>4x4GB Kingston DDR2-800 ECC</td>
<td>$ 1000</td>
<td>150W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+$600</td>
<td>+ 4W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt;$1600</td>
<td>=154W</td>
</tr>
<tr>
<td>TMR w/new boards &amp; more racks</td>
<td>3x AMD Opteron 2356</td>
<td>6x4GB Kingston DDR2-800 ECC</td>
<td>$ 1500</td>
<td>225W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+$900</td>
<td>+ 6W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt;$2400</td>
<td>&gt;231W</td>
</tr>
</tbody>
</table>
## Financial Cost and Power Consumption
(Based on Current ORNL Jaguar Hardware and Market Prices)

<table>
<thead>
<tr>
<th>Solution</th>
<th>Processor</th>
<th>Memory</th>
<th>Price</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional checkpoint/restart</td>
<td>1x AMD Opteron 2356</td>
<td>2x4GB Micron DDR2-800 ECC</td>
<td>=$1250</td>
<td>= 77W</td>
</tr>
<tr>
<td>In-memory checkpoint caching</td>
<td>1x AMD Opteron 2356</td>
<td>4x4GB Micron DDR2-800 ECC</td>
<td>=160%</td>
<td>=103%</td>
</tr>
<tr>
<td>In-memory checkpoint/restart with new boards</td>
<td>1x AMD Opteron 2356</td>
<td>2x4GB Micron DDR2-800 ECC 4x4GB Kingston DDR2-800 ECC</td>
<td>=136%</td>
<td>=105%</td>
</tr>
<tr>
<td>DMR w/new boards &amp; more racks</td>
<td>2x AMD Opteron 2356</td>
<td>4x4GB Kingston DDR2-800 ECC</td>
<td>&gt;128%</td>
<td>=200%</td>
</tr>
<tr>
<td>TMR w/new boards &amp; more racks</td>
<td>3x AMD Opteron 2356</td>
<td>6x4GB Kingston DDR2-800 ECC</td>
<td>&gt;192%</td>
<td>&gt;300%</td>
</tr>
</tbody>
</table>

*DMR w/new boards & more racks:* Not 2x/3x!
Conclusions

- DMR with 4-nine or TMR with 3-nine compute node rating provides enough system availability for HPC systems planned for the next 10 years with 1,000,000 compute nodes and beyond.
- DDMR with 3-nine or DTMR with 2-nine single component rating provides enough overall system availability for future HPC systems.
- The reduction of individual component reliability within a modular redundant system permits recovering the costs for using 2× or 3× the number of components.
- This tunable cost vs. reliability/availability trade-off is the counter argument to the traditional view that modular redundancy comes at 2× or 3× costs.
Conclusion and Future Work

- We have made the case for modular redundancy in large-scale HPC systems by
  - Explaining the limits for the current state of practice
  - Describing the significant increase in system availability modular redundancy offers
  - Demonstrating that modular redundancy in HPC systems allows for lowering compute node reliability and recovering the costs of using 2× or 3× the number of components

- Future work needs to focus on
  - Concepts and implementation-specific details for modular redundancy in massively parallel HPC systems
  - Mitigating the issue of increased power consumption