A Message-Logging Protocol for Multicore Systems

Esteban Meneses, Xiang Ni and Laxmikant V. Kalé

Parallel Programming Laboratory
University of Illinois at Urbana-Champaign

FTXS 2012
Message Logging
- Local rollback
- Less energy consumed
- Parallel recovery with migratable tasks

Multicore Systems
- Keep scaling FLOPS/s
- Almost Top 500 list entirely
- More cores per shared-memory node

7-pointStencil

Recovery Speedup
Number of Cores in Recovery

Recovery Speedup
Number of Cores in Recovery

7-pointStencil

Recovery Speedup
Number of Cores in Recovery

Number of Cores/Node
Year
BG/L BG/P BG/Q
XT3 XT4 XT5 XT6 XK6
Atlas T2K Trestles
Appro Cray IBM

Agenda

1. Unit of Failure
3. Experimental Results
4. Analysis of Reliability
5. Conclusions and Future Work
Failures in HPC Systems

- The right unit of failure
  - Core, subset of cores, node, subset of nodes

- System logs
  - The Computer Failure Data Repository (CFDR)
  - Collaborations
  - Failure databases

- Jaguar
  - Top 6 in the world
  - 537-day study (8/08-2/10): 1253 separable events
  - Errors: machine check exceptions (MCE), interconnect (CRC), software
One failure, one node

\( x \): number of nodes in a failure
Modeled through a random variable

Exponential decay
Geometric distribution

\[ f(x) = (1 - p)^{x-1} p \]

Heavy-tailed curve
Zipf’s distribution

\[ f(x) = \frac{1}{x^s} \sum_{i=1}^{n} \frac{1}{i^s} \]
Message Logging

- Messages stored at sender
- Non-deterministic decisions recorded (determinants)
- Message reception order
- Causal message logging
  - Determinants stored in their causal path
  - Piggybacking determinants
- Failure unit: Core → Node
- Intra-node messages *not* stored
- Only inter-node messages piggyback determinants
- Shared data structure for determinants
- Lockless determinant queue
Protocol

Core A → Node X
Core B → Node Y
Core C → Node Z
Core D → Node X

Checkpoint
Failure

m₁
m₂
m₃ ⊕ {d₁, d₂} ACK
m₄ ⊕ {d₁, d₂} ACK
m₁

{d₁, d₂}

Restart

Core A
Core B
Core C
Core D
Implementations

- **Charm++ runtime system**
- A heavyweight process per node
- One process = one communication thread + worker threads
- Two fault-tolerance strategies
  - Double in-memory checkpoint/restart
  - Causal message-logging for multicore systems
- Testbed: Steele (RCAC), Ranger (TACC) and Trestles (SDSC)
Low Execution Time Overhead

7-point Stencil

<table>
<thead>
<tr>
<th>Number of Nodes(Cores)</th>
<th>Checkpoint/Restart</th>
<th>Message Logging</th>
</tr>
</thead>
<tbody>
<tr>
<td>2(64)</td>
<td>0.92</td>
<td>1.04</td>
</tr>
<tr>
<td>4(128)</td>
<td>0.94</td>
<td>1.02</td>
</tr>
<tr>
<td>8(256)</td>
<td>0.96</td>
<td>1.00</td>
</tr>
<tr>
<td>16(512)</td>
<td>0.98</td>
<td>1.02</td>
</tr>
<tr>
<td>32(1,024)</td>
<td>1.00</td>
<td>1.04</td>
</tr>
</tbody>
</table>
Reduced Memory Overhead

Message Log (Fraction) vs. System (Cores/Node):

- Standard (1 core):
  - Memory overhead: 1.00

- Steele (8 cores):
  - Memory overhead: 0.42

- Ranger (16 cores):
  - Memory overhead: 0.33

- Trestles (32 cores):
  - Memory overhead: 0.17

Graph showing the reduced memory overhead for different systems.
Efficient single-node failure reliability

\(x\): number of nodes in a failure
Modeled through a random variable
\(n\): total number of nodes in the system
\(g\): average number of acquaintances per node

\[
\prod_{i=0}^{x-1} (n - 2i) \\
\prod_{i=0}^{x-1} (n - i)
\]

\[
\left[ \frac{(n-x)}{g} \frac{(n-1)}{g} \right]^x
\]
Survivability $S$ is the weighted average over all possible failures

$$S = \sum_{i=1}^{n} s(i)p(i)$$

<table>
<thead>
<tr>
<th></th>
<th>Geometric</th>
<th>Zipf’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checkpoint/Restart</td>
<td>0.9997</td>
<td>0.9992</td>
</tr>
<tr>
<td>Message Logging (g=2)</td>
<td>0.9988</td>
<td>0.9966</td>
</tr>
<tr>
<td>Message Logging (g=4)</td>
<td>0.9980</td>
<td>0.9945</td>
</tr>
<tr>
<td>Message Logging (g=8)</td>
<td>0.9964</td>
<td>0.9911</td>
</tr>
<tr>
<td>Message Logging (g=16)</td>
<td>0.9933</td>
<td>0.9854</td>
</tr>
</tbody>
</table>
Conclusions and Future Work

Conclusions:
- Most of failures in HPC systems involve one node
- A message-logging protocol for multicore systems can be efficiently implemented
- This protocol is almost as resilient as checkpoint/restart

Future Work:
- Explore more applications
- Understand scalability of message logging protocol
This work is partially supported by the US Department of Energy under grant DOE DE-SC0001845 and by a machine allocation on XSEDE under award ASC050039N.

Failure datasets:
- Jaguar: Terry Jones (ORNL).
- Mercury: Ana Gainaru (UIUC).
Thank You!

Q&A
Number of Determinants per Message

7-point Stencil

Determinants per Remote Message vs Core Number

Meneses, Ni and Kalé (UIUC)  Message-Logging for Multicore Systems  FTXS 2012  17 / 18
Communication Graph

NPB-CG

NPB-MG