Dynamic Load Balance for Optimized Message Logging in Fault Tolerant HPC Applications

Esteban Meneses, Greg Bronevetsky and Laxmikant V. Kalé
A key challenge is to adapt to the unavoidable variability in time and space (processes/processors) of future applications and systems.

International Exascale Software Project

Leverage Load Balancing Infrastructure for Fault Tolerance

Frequent Failures

Seguroia

1.6 million cores

2012

Exascale

100 million cores?

2018
Contents

• Rollback-Recovery
• Load Balancing
• Optimized Message Logging
• Experiments
• Conclusions and Future Work
Model

- Object-based over-decomposition.
- Asynchronous method invocation.
- Migratable-objects runtime system.
- Non-FIFO channels.
- Charm++ and Adaptive MPI.
- Fail-stop crashes, user-level checkpoint.
Rollback-Recovery

Checkpoint/Restart

Team-based Message Logging

Message Logging

Expensive Recovery

How do you form teams?

Memory Overhead

team = group of PEs

Checkpoint Line

Early Message

Late Message

Time

Checkpoint

Failure

Exensive Recovery

Expense Recovery

How do you form teams?

Memory Overhead

IEEE Cluster Computing 2011 • Austin, Texas

Thursday, September 29, 2011
Forming Teams

Random

Static

Dynamic

Communication Pattern (NPB CG.C.64)

Sender Rank

Receiver Rank

Number of Messages

IEEE Cluster Computing 2011 • Austin, Texas

Thursday, September 29, 2011
Load Balancing

- Runtime system collects statistics about objects:
  - Computational load.
  - Communication graph.

Option 1

Option 2
Load Balancer

Step 1

Step 2

Team X

PE A

PE B

Team Y

PE C

PE D
Optimized Message Logging

• Every non-deterministic event generates a determinant (message reception).

• Simple Causal Message Logging:
  • Messages stored at sender’s memory.
  • Determinants stored at receivers (at least once).

• Optimization: synchronized checkpoint.
Protocol

PE A

m₁

d₁

PE B

m₂

d₂

PE C

m₃ ⊕ \{d₁, d₂\}

remote messages (stored)

PE D

m₄ ⊕ \{d₁, d₂\}

local messages (not stored)

ACK

IEEE Cluster Computing 2011 • Austin, Texas

Thursday, September 29, 2011
Experiments

- Overhead of load balancer with team formation.
- Smart team formation benefits.
- Team size vs memory overhead.

<table>
<thead>
<tr>
<th>Application</th>
<th>Language</th>
<th>Load Imbalance</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPB-BT-MZ</td>
<td>MPI</td>
<td>Static</td>
</tr>
<tr>
<td>Mol3D</td>
<td>Charm++</td>
<td>Dynamic</td>
</tr>
</tbody>
</table>
Testbed

- Steele@RCAC
- 893 8-core nodes
- 16 GB RAM per node
- Gigabit Ethernet
Result: Load Balance
Result: Low Overhead

NPB-BT multi-zone (64 PEs, Steele)

<table>
<thead>
<tr>
<th>Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
</tr>
<tr>
<td>Objects</td>
</tr>
<tr>
<td>Average Size</td>
</tr>
<tr>
<td>St. Dev. Size</td>
</tr>
<tr>
<td>Coeff. Variation</td>
</tr>
<tr>
<td>Max Size</td>
</tr>
<tr>
<td>Min Size</td>
</tr>
</tbody>
</table>

edge-cut ratio = 0.26
Result: Strong Scaling

![Graph showing strong scaling results for Mol3D (Steele). The graph plots execution time (seconds) against the number of PEs. Two lines are shown: NoLB (red solid line) and TeamLB (blue dashed line). The execution time decreases as the number of PEs increases, indicating strong scaling. The x-axis represents the number of PEs (128, 256, 512, 1024), while the y-axis represents the execution time (64, 128, 256, 512 seconds).]
Result: Team Size

Mol3D (256 PEs, Steele)

- **NoLB**
- **TeamLB**

<table>
<thead>
<tr>
<th>Team Size (PEs)</th>
<th>Edge Cut (ratio)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0.8</td>
</tr>
<tr>
<td>16</td>
<td>0.6</td>
</tr>
<tr>
<td>32</td>
<td>0.4</td>
</tr>
<tr>
<td>64</td>
<td>0.2</td>
</tr>
</tbody>
</table>
Conclusions

- Migratable objects as a framework to provide fault tolerance and load balancing.
- Graph partitioning tools effective to drastically reduce memory overhead in message logging.
- Team size is a tradeoff parameter.
Contributions

- An extension to load balancing framework to dynamically form processor teams.
- A team-based algorithm for simple causal message logging.
Future Work

- More applications:
  - Adaptive mesh refinement.
  - Weather simulation.
- Incorporate team-based approach into SMP version.
- Test system with faster networks (Infiniband).
- Processor teams vs object teams.
Thank you!

Q&A
Object Teams

**Advantages:**

- Natural to migratable-objects model.
- Separation of concerns: load balance vs fault tolerance.
- More flexibility for cluster formation.

**Disadvantages:**

- In case of failure: unbounded number of teams to recover.
- Some local messages have to be stored.
- Load balancer involved in object distribution.
Moore’s Law?

- Top500
- Moore’s (24)
- Moore’s (18)

Years: 2002 to 2011

Cores:
- 0
- 137,500
- 275,000
- 412,500
- 550,000
Result: Dynamic Teams

![Graph showing time per iteration (milliseconds) vs iteration for LBTest (256 PEs, Steele). The graph compares NoLB and TeamLB methods. The x-axis represents iteration, and the y-axis represents time per iteration in milliseconds. The graph shows that TeamLB has a more consistent performance compared to NoLB.]