Making a DSM Consistency Protocol
Hierarchy-Aware: an Efficient Synchronization Scheme

Gabriel Antoniu, Luc Bougé, Sébastien Lacour
IRISA / INRIA & ENS Cachan, France

DSM2003, Tokyo, May 13\textsuperscript{th} 2003
Distributed Shared Memory

- Distributed compute nodes
- Shared virtual address space

Diagram:

- DSM
- Mem.
- Mem.
- Mem.
- CPU
- CPU
- CPU
- network
Hierarchical Network Architectures

- Flat architecture:
  - Expensive network
  - Technically difficult

- Hierarchical architecture:
  - Cheaper network
  - Technically easier

Latencies: FastEthernet: 50-100 µs / SCI/Myrinet: 5 µs
Motivation:
- high-performance computing / code coupling

Coupling with explicit data transfer:
- MPI, FTP, ...

Key factor:
- network latency
Latency Heterogeneity and Memory Consistency Protocol

Principle: avoid communications over high-latency links
Roadmap

- Design, implement, evaluate a hierarchical memory consistency protocol
- Same semantics as with a flat protocol
- Well-suited for clusters of clusters
- High-performance oriented
- Few related works:
  - Clusters of SMP nodes: Cashmere-2L (1997, Rochester, NY)
  - Clusters of clusters: Clustered Lazy Release Consistency (CLRC, 2000, LIP6, Paris) → cache data locally
DSM-PM2: the Implementation Platform

- Portability
- Multi-threading

**DSM-PM2**
- Consistency Protocols Library
- Basic Building Blocks
- Page Management
- Communication Management

**PM2**
- Threads (Marcel)
- Communications (Madeleine)
Starting Point: Home-Based Release Consistency

- Each page is attached to a Home-Node
- Multiple writers, eager version
- Home-Node:
  - holds up-to-date version of the page it hosts
  - gathers / applies page diffs

Sending page modifications at lock release time

Page request

Up-to-date version of the page

Home-node

write

diff

read
Flat Protocol & Hierarchical Architectures: Where Does Time Get Wasted?

1. At synchronization operations: lock acquisition and release

2. While waiting for message acknowledgements (consistency protocol)

- While retrieving a page from a remote node (data locality → CLRC, Paris)
1. Improving upon Synchronization Operations

Cluster A

Cluster B

Executing in critical section
Hierarchy-Unaware Lock Acquisitions

Cluster A

Cluster B

1 2

3 4

5

4 high-latency communications!
Priority to Local Threads

Cluster A

5
1

3

Cluster B

2

4

3 high-latency communications!

+ 1 low-latency communications!
Priority to Local Nodes

Cluster A

Cluster B

1 high-latency + 3 low-latency communications!
Diff Aggregation at Node Level

- Flat protocol

- Hierarchical protocol

aggregated diffs: 1 + 5
Avoiding Starvation

- Control the lack of fairness
- Bounds to limit the number of consecutive acquisitions of a lock:
  - By the threads of a node
  - By the nodes of a cluster
  - Can be tuned at run-time
Flat Protocol & Hierarchical Architectures: Where Does Time Get Wasted?

1. At synchronization operations: lock acquisition and release

2. While waiting for message acknowledgements (consistency protocol)

   - While retrieving a page from a remote node (data locality → CLRC, Paris)
Lock Release in a Flat Protocol

Cluster A

Cluster B

Node 0

Node 1

Home node

Node 2

diff

release

invalidate

ack

lock

Lock fully released

ack
Lock Release in a Hierarchical Architecture

Cluster A
Home node

Cluster B
Node 0
Node 1
Full release

Cluster C
Node 2

Time wasted waiting

lock

ack
diff

invalidate

release

ack
2. Partial Lock Release in a Hierarchical Architecture
Partial Lock Release in a Hierarchical Architecture

- Partially released locks can travel within a cluster
- Fully released locks (acks received from all clusters) can travel to remote clusters
Performance Evaluation: Thread Level

<table>
<thead>
<tr>
<th>max_tp</th>
<th>1</th>
<th>5</th>
<th>15</th>
<th>25</th>
<th>infinity</th>
</tr>
</thead>
<tbody>
<tr>
<td>speed-up</td>
<td>1</td>
<td>3.4</td>
<td>5.8</td>
<td>6.9</td>
<td>~60 (unfair)</td>
</tr>
</tbody>
</table>

Inter-node message \(\rightarrow\) Intra-node message

Modification Aggregation

<table>
<thead>
<tr>
<th>max_tp</th>
<th>1</th>
<th>5</th>
<th>15</th>
<th>25</th>
<th>infinity</th>
</tr>
</thead>
<tbody>
<tr>
<td>speed-up</td>
<td>1</td>
<td>2.1</td>
<td>4.7</td>
<td>7.3</td>
<td>unfair</td>
</tr>
</tbody>
</table>

Less inter-node messages

SCI Cluster (10,000 C.S. per thread)
Partial Lock Release (node level): Performance Gain

Inter-cluster message receipt overlapped with intra-cluster computation

(10,000 C.S. per thread)
## Conclusion

- Hierarchy-aware approach to distributed synchronization
  - Complementary with local data caches (CLRC)
- New concept of "Partial Lock Release":
  - Applicable to other synchronization objects (semaphores, monitors), except for barriers
  - Applicable to other eager release consistency protocols
- More hierarchy levels, greater latency ratios:

```plaintext
PING paraplapla.irisa.fr (131.254.12.8) from 131.254.12.68 : 56(84) bytes of data.
64 bytes from paraplapla.irisa.fr (131.254.12.8): icmp_seq=9 ttl=255 time=385 usec
PING ccgrid2003.apgrid.org (192.50.75.123) from 131.254.12.68 : 56(84) bytes of data.
64 bytes from sf280.hpcc.jp (192.50.75.123): icmp_seq=9 ttl=235 time=322.780 msec
```