COMP9244, Seminar 1

Shared memory multiprocessors

Leonid Ryzhyk
leonidr@cse.unsw.edu.au

Introduction

Why multiprocessors?

- Can not get more computational power out of ILP
- Multiple cores are more cost efficient than superscalars
- More efficient use of energy

Why shared memory multiprocessors?

- Efficient interprocess communication
- Natural programming model for many applications
- Compatibility with single-processor software
  - Difficult to scale
  - More expensive
**Memory hierarchy organisation**

Memory hierarchy organisation is selected based on:

→ The number of processors
→ Application domain
→ Scalability requirements

---

**Symmetric shared memory**

---

**SMP example: Intel Xeon**
SMP example: ARM11 MP

Distributed shared memory

DSM example: Sun WildFire

DSM example: SGI Altix
**DSM example: AMD Opteron**

**Interleaved shared cache**

**Interleaved cache example: Sun Niagara**

**Specialised multiprocessor example:**
**Cell BE**
Interconnection network design space

Interconnection network characteristics:
- Topology
- Routing algorithm
- Switching strategy
- Flow control

Interconnection network performance

Interconnection network performance characteristics:
- Latency
  \[ \text{Time}(n)_{S-D} = \text{Overhead} + \text{RoutingDelay} + \text{ChannelOccupancy} + \text{ContentionDelay} \]
- Bandwidth
  - Local
  - Aggregate
  - Bisection bandwidth

Interconnection network topology: Crossbar

- e.g., Niagara
Interconnection network topology: Fat tree

→ e.g., Altix

Interconnection topology: 2-D Hypercube, Torus

→ e.g., Cray XT3

Interconnection network topology: 4D Hypercube

Cache coherence protocols
Cache coherence definition

Cache coherence

→ Write serialisation
→ Write propagation

Snooping protocols for cache coherence

Bus snooping protocols:

→ Cache coherency controller connected to the L2 cache
→ L2 cache must be inclusive
→ 2 types of snooping protocols
  → update
  → invalidate
Directory-based cache coherence

Definition: A multiprocessor system is sequentially consistent if the result of any execution is the same as if the operations of all the processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program.

Memory consistency

Sequential consistency

Sequential consistency example

Initially Flag1 = Flag2 = 0

P1
Flag1 = 1
if (Flag2 == 0)
critical section
P2
Flag2 = 1
if (Flag1 == 0)
critical section
**Relaxed consistency models**

<table>
<thead>
<tr>
<th>Processor</th>
<th>Consistency model</th>
</tr>
</thead>
<tbody>
<tr>
<td>P4</td>
<td>processor order</td>
</tr>
<tr>
<td>PowerPC</td>
<td>weak</td>
</tr>
<tr>
<td>AMD64</td>
<td>processor order</td>
</tr>
<tr>
<td>IA64</td>
<td>weak</td>
</tr>
<tr>
<td>MIPS R10000</td>
<td>sequential</td>
</tr>
<tr>
<td>UltraSparc</td>
<td>TSO</td>
</tr>
</tbody>
</table>

**Safety nets**

Memory barriers:
- Coarse-grained (e.g., PowerPC `sync`)
- Fine-grained (e.g., IA-32 `lfence, sfence`)

**OS as a shared resource**

OS is a potential bottleneck in a multiprocessor system.

Sources of contention:
- Locks
- Shared data structure
- False sharing

**Operating systems for SMM**
**OS scalability**

Approaches to scalable OS design:
- Evolutionary
- Linux scalability project
- Design for scratch
  - Hurricane, Tornado, K42
- OS per CPU
  - Disco
  - Hive

**Other OS issues**

Other issues:
- Task placement and task migration
- Memory placement
- Synchronisation primitives

**Questions**