Announcements

- **Project #6 is available**
  - It’s due Tuesday at 9:00 AM

- **Office Hours Next Week**
  - Tu 11-12
  - W 10-12
  - Th 11-12
My Research Interests

- **Parallel Computing**
  - There are limits to how fast one processor can run
  - solution: use more than one processor

- **Issues in parallel computing design**
  - do the processors share memory?
    - is the memory “uniform”?
    - how do processors cache memory?
  - if not how do they communicate?
    - message passing
    - what is the latency of message passing
Parallel Processing

- What happens in parallel?
- Several different processing steps
  - pipeline
  - simple example: grep foo | sort > out
  - called: *multiple instruction multiple data* (MIMD)
- The same operation
  - every processor runs the same instruction (or no-instruction)
  - called: *single instruction multiple data* (SIMD)
  - good for image processing
- The same program
  - every processor runs the same program, but not “lock step”
  - called: *single program multiple data* (SPMD)
  - most common model
Issues in effective Parallel Computation

- **Load balancing**
  - every processor should have some work to do.

- **Latency hiding/avoidance**
  - getting data from other processors (or other disks) is slow
  - need to either:
    - hide the latency
      - processes can “pre-fetch” data before they need it
      - block and do something else while waiting
    - avoid the latency
      - use local memory (or cache)
      - use local disk (of file buffer cache)

- **Limit communication bandwidth**
  - use local data
  - use “near” data (i.e. neighbors)
My Research:

- Given a parallel program and a machine
- Try to answer performance related questions
  - Why is the programming running so slowly?
  - How do I fix it?
- Issues:
  - how to measure a program without changing it?
  - how do you find (and then present) the performance problem, not tons of statistics?
- Techniques:
  - dynamic data collection
  - automated search
  - analysis of process interactions
Introduction

- **Software today**
  - makes extensive use of libraries and re-usable components
  - Libraries used by an application may not be tuned to the application’s need

- **Fast software development/distribution with built-in (default) configurations**
  - Applications may not run well in all environments
  - There may be no single configuration good for all environments
Active Harmony

- Real-time performance optimization
- Automatic library selection (code)
  - Monitor library performance
  - Switch library if necessary
- Automatic performance tuning (parameter)
  - Monitor system performance
  - Adjust runtime parameters
System design

Application

Application Programming Interface

Library Specification Layer

Monitoring Component

Library 1 Parameter(s)

Library 2 Parameter(s)

Library n Parameter(s)

Harmony Server

Adaptation Controller

System (Execution Environment)
Cluster-based web service tuning

- Scalable, available, cost-effective architecture
Cluster-based web service tuning

<table>
<thead>
<tr>
<th>Performance (WIPS)</th>
<th>Browsing</th>
<th>Shopping</th>
<th>Ordering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best configuration for Browsing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best configuration for Shopping</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best configuration for Ordering</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original configuration</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Best configuration after 200 iterations**

<table>
<thead>
<tr>
<th>Workload Applied</th>
<th>Browsing</th>
<th>Shopping</th>
<th>Ordering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improvements compared to the default configuration</td>
<td>15%</td>
<td>16%</td>
<td>5%</td>
</tr>
</tbody>
</table>
Cluster-based web service tuning

- **External tuning – reconfiguration**
  - Initial: 3 proxy servers, 3 application servers

(a) One node:
proxy → application server
(browsing → ordering workload)

(b) One node:
application → proxy server
(Browsing workload)
Application tuning – Parallel Ocean Program (POP)

- The ocean component of CCSM (Community Climate System Model)
- Problem size – 3600x2400
- 480 processes (32 nodes, 15 processes/node using seaborg.nersc.gov)

![Graph showing performance vs. dimensions](image)

<table>
<thead>
<tr>
<th>Default block dimension</th>
<th>Suggested block dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>180x100</td>
<td>150x120</td>
</tr>
<tr>
<td></td>
<td>180x100</td>
</tr>
<tr>
<td></td>
<td>225x80</td>
</tr>
</tbody>
</table>
Application tuning – Parallel Ocean Program (POP)

- Performance tuning by parameters adjustment
- Numerous parameters
  - About 20 of them are performance related
  - Each with 2-4 possible values
- Without prior knowledge
- 32 processes (8 nodes, 4 processes/node running on hockney.nersc.gov)
- 12.1% improvement in execution time after trying 12 configurations
- 16.7% improvement after tuning (27 iterations)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>After tuning</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_iotasks</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>hmix_momentum_choice</td>
<td>anis</td>
<td>del2</td>
</tr>
<tr>
<td>hmix_tracer_choice</td>
<td>gent</td>
<td>del2</td>
</tr>
<tr>
<td>kappa_choice</td>
<td>constant</td>
<td>variable</td>
</tr>
<tr>
<td>slope_control_choice</td>
<td>notanh</td>
<td>clip</td>
</tr>
<tr>
<td>hmix_alignment_choice</td>
<td>east</td>
<td>grid</td>
</tr>
<tr>
<td>state_choice</td>
<td>jmcdf</td>
<td>linear</td>
</tr>
<tr>
<td>state_range_opt</td>
<td>ignore</td>
<td>enforce</td>
</tr>
<tr>
<td>ws_interp_type</td>
<td>nearest</td>
<td>4point</td>
</tr>
<tr>
<td>shf_interp_type</td>
<td>nearest</td>
<td>4point</td>
</tr>
<tr>
<td>sfwf_interp_type</td>
<td>nearest</td>
<td>4point</td>
</tr>
<tr>
<td>ap_interp_type</td>
<td>nearest</td>
<td>4point</td>
</tr>
</tbody>
</table>
BlueGene/L Nodes

32k/32k L1
440 CPU
“Double FPU”

32k/32k L1
440 CPU I/O proc
“Double FPU”

PLB (4:1)

128

L2

128

L2

snoop

256

256

128

256

256

4MB EDRAM
L3 Cache or Memory

1024+
144 ECC

SDRAM Control with ECC

144 bit wide DDR
256/512MB

4 global barriers or interrupts

Includes ECC

Shared L3 directory for EDRAM

Multiported Shared SRAM Buffer

DDR

includes ECC

L3 Cache

or Memory

Courtesy IBM
16 compute cards

2 optional IO cards

32-way (4x4x2) node card

Midplane (450 pins) torus, tree, barrier, clock, Ethernet service port

Ethernet-JTAG FPGA

dc-dc converters

IO Gb Ethernet connectors through tailstock

Latch ing and retention
BlueGene/L Networks

3 Dimensional Torus
- Point-to-point

Global Tree
- Global Operations

Global Barriers and Interrupts
- Low Latency Barriers and Interrupts

Gbit Ethernet
- File I/O and Host Interface

Control Network
- Boot, Monitoring and Diagnostics

Courtesy IBM