CIS 631
Parallel Processing

Lecture 1

Allen D. Malony
malony@cs.uoregon.edu

Department of Computer and Information Science
University of Oregon
Outline

- Course Overview
  - What is CIS 631 and what is expected of you?
  - What will you learn in CIS 631?
- Parallel Computing
  - Problem solving methodology
  - Parallelism models, architecture, and programming
  - Scalable parallel computing and performance
- Computational Science
  - Large-scale problems and high-performance
  - High-performance and scalable parallel computing
- Trends that shape the field
CIS 631 - Parallel Processing (Computing)

☐ Graduate course
  ☑ Prerequisite: CIS 629

☐ Lecture time
  ☑ Tuesday, Thursday: 12:00 - 1:20 pm, 116 ESL (Eslinger)

☐ Final schedule
  ☑ June 12, 8:00 - 10:00 am, 116 ESL

☐ Course webpage
  ☑ http://www.cs.uoregon.edu/classes/cis631

☐ Programming experience
  ☑ C, C++, Java, and/or Fortran (required)
  ☑ Linux or Unix background (desired)
Overview

- Broad field of computer science concerned with:
  - Architecture, HW/SW systems, languages, programming paradigms, algorithms, and theoretical models
  - Computing in parallel

- Performance is the *raison d’être* for parallelism
  - High-performance computing
  - Drives computational science revolution

- Topics of study
  - Parallel architectures
  - Parallel programming
  - Parallel algorithms
  - Parallel performance models and tools
  - Parallel applications
Course Book

- The Sourcebook of Parallel Computing
- Editors
  - Well known leaders in parallel computing
  - Chapters by well known researchers in the field
  - Builds on Center for Research in Parallel Computing

“The Sourcebook for Parallel Computing gives a thorough introduction to parallel applications, software technologies, enabling technologies, and algorithms. This is a great book that I highly recommend to anyone interested in a comprehensive and thoughtful treatment of the most important issues in parallel computing.”

Horst Simon, Director, Director, NERSC, Berkeley
Lectures

- Book provides a broad background and deep treatment
  - It is your main reference
  - Online tutorials and other materials are important sources

- Lectures should be more interactive
  - Not to be your main source of information
  - Covers topics I consider of interest

- Try to provide online access to lecture slides

- Some guest lectures throughout the quarter
  - Sameer Shende, University of Oregon
  - Holger Brunst, Technical University of Dresden, Germany
Assignments

☐ Programming exercises
☐ Programming project
☐ Term project
☐ Research summary paper
☐ Midterm exam
Programming Exercises

- Cluster benchmarks
  - Get you familiar with the cluster environment
- MPI programming
  - Get you familiar with message passing programming
  - Try out various MPI primitives
- Pthreads or OpenMP programming
  - Get you familiar with shared memory programming
  - Depends on OpenMP compiler availability
- Simple programs
- Build experience in parallel programming on cluster
- Individual work
Programming Project

- More extensive than programming exercises
  - Designed to improve and test your programming skills
- Probably MPI-based
- Individual work
Term Project

- Major programming project for the course
  - Non-trivial parallel application
  - Include performance analysis
- Project teams
  - 3 person teams, 3-4 teams
- Project proposal
  - Due May 15
- Project talk
  - Finals period
- Project report
  - Due Finals day
NeuroInformatics Center (NIC) Cluster

- Dell computational cluster
  - 16 dual-processor nodes
    - 2.8 MHz Pentium Xeon
    - 4 Gbyte memory
    - 36 Gbyte disk
    - Dual Gigabit ethernet adaptors
    - 2U form factor
  - Master node (same specs)
  - 2 Gigabit ethernet switches
- Target installation: first week in April
- To be named …
Term Paper

- Investigate parallel computing topic of interest
  - More in depth review
  - Individual choice
  - Summary of major points
- Requires minimum of ten references
  - Book has a large bibliography
  - NEC CiteSeer Scientific Literature Digital Library
    http://citeseer.nj.nec.com/cs
- Paper topic abstract and references due April 17
- Final term paper due last class meeting
- Individual work
Grading

☐ 5% Exercises
☐ 25% Term paper
☐ 20% Midterm exam
☐ 10% Project 1
☐ 40% Project 2
   10% Presentation
   30% Report
Schedule

☐ See course webpage
What will you get out of CIS 631?

- In-depth understanding of parallel computer design
- Knowledge of how to program parallel computer systems
- Exposure to different forms parallel algorithms
- Practical experience using a parallel cluster
- Background on parallel performance modeling
- Learn techniques for empirical performance analysis
- Have fun and make new friends
Parallel Processing – What is it?

□ A parallel computer is a computer system that uses multiple processing elements simultaneously in a cooperative manner to solve a computational problem.

□ Parallel processing includes techniques and technologies that make it possible to compute in parallel:
  - Hardware, networks, operating systems, parallel libraries, languages, compilers, algorithms, tools, …

□ Parallel computing is an evolution of serial computing:
  - Parallelism is natural
  - Computing problems differ in level / type of parallelism

□ Parallelism delivers performance
Why use parallel processing?

- Two primary reasons (both performance related)
  - Faster time to solution (response time)
  - Solve bigger computing problems in same time
- Other factors motivate parallel processing
  - Effective use of machine resources
  - Cost efficiencies
  - Overcoming memory constraints
- Serial machines have inherent limitations
  - Processor speed
  - Memory bottlenecks
- Parallelism is the future of computing
Perspectives on Parallelism

- Parallel computer architecture
  - Hardware needed for parallel execution?
  - Computer system design
- Operating system
- Parallel programming
  - Libraries (low-level, high-level)
  - Languages
  - Software development environments
- Parallel algorithms
- Parallel performance evaluation
- Parallel tools
Why study parallel computing today?

- Computing architecture: innovative computing structures often drive to novel programming models
- Technological convergence
  - The “killer micro” is ubiquitous
  - Laptops and supercomputers are fundamentally similar!
  - Trends cause diverse approaches to converge
- Technological trends make parallel computing inevitable
- Need to understand fundamental principles and design tradeoffs, not just taxonomies
  - Naming, Ordering, Replication, Communication
  - Performance
Inevitability of Parallel Computing

- Application demands
  - Insatiable need for computing cycles
- Technology Trends
  - Processor and memory
- Architecture Trends
- Economics
- Current trends:
  - Today’s microprocessors have multiprocessor support
  - Servers and workstations available as multiprocessors
  - Tomorrow’s microprocessors are multiprocessors
Application Characteristics

- Application performance demands hardware advances
- Hardware advances generate new applications
- New applications have greater performance requirements
  - Exponential increase in microprocessor performance
  - Innovations in parallel architecture and integration

- Range of performance demands
  - System performance must also improve as a whole
  - Performance requirements require computer engineering
  - Costs addressed through technology advancements
Computational Science

- Traditional scientific methodology
  - Theoretical science
    - Formal systems and theoretical models
    - Insight through abstraction, reasoning through proofs
  - Experimental science
    - Real system and empirical models
    - Insight from observation, reasoning from experiment design

- Computational science
  - Emerging as a principal means of scientific research
  - Use of computational methods to model scientific problems
    - Numerical analysis plus simulation methods
    - Computer science tools
  - Study and application of these solution techniques
“A revolution is underway in the practice of science and engineering, arising from advances in computational science and new models for scientific phenomena, and made possible by advances in computer science and technology.”

“.. fundamental problems in science or engineering with potentially broad social, scientific, and/or political impact, that could be advanced by applying high performance computing resources.”

- NSF Blue Ribbon Panel on High Performance Computing, August 1993
What is Computational Science?

Science Discipline
Physics, Chemistry, Biology, etc.

Computer Science
Hardware/Software

Applied Mathematics
Numerical Analysis, Modeling, Simulation

Computational Science
Multidisciplinary Teamwork and Collaboration
21st Century Science and Engineering Practice

- Three-fold science
  - theory
  - experiment
  - computational simulation

- Supported by
  - multimodal collaboration systems
  - distributed, multi-petabyte data archives
  - leading edge computing systems (*)
  - distributed experimental facilities
  - internationally distributed multidisciplinary teams

- Collectively defining a new future
  - creation of 21st century IT infrastructure
  - sustainable, multidisciplinary communities
Computational Challenges

- Computational science is thrives on computer power
  - Faster solutions
  - Finer resolution
  - Bigger problems
  - Improved interaction
  - BETTER SCIENCE!!!

- How to get more computer power?
  - Scalable parallel computing

- Computational science also thrives better integration
  - Couple computational resources
  - Grid computing
Broad Parallel Architecture Issues

- Resource Allocation
  - How many processing elements?
  - How powerful are the elements?
  - How much memory?

- Data access, communication, and synchronization
  - How do the elements cooperate and communicate?
  - How are data transmitted between processors?
  - What are the abstractions and primitives for cooperation?

- Performance and Scalability
  - How does it all translate into performance?
  - How does it scale?
What’s Driving Parallel Computing Architecture?

Processor-DRAM Memory Gap (latency)

- "Moore’s Law"
- Processor-Memory Performance Gap: (grows 50% / year)
- von Neumann bottleneck!!
Classifying Parallel Systems – Flynn’s Taxonomy

- Distinguishes multi-processor computer architectures along the two independent dimensions
  - Instruction and Data
    - Each dimension can have one state: Single or Multiple
- SISD: Single Instruction, Single Data
  - Serial (non-parallel) machine
- SIMD: Single Instruction, Multiple Data
  - Processor arrays and vector machines
- MISD: Multiple Instruction, Single Data (weird)
- MIMD: Multiple Instruction, Multiple Data
  - Most common parallel computer systems
Parallel Architecture Types

- Instruction-Level Parallelism
  - Parallelism captured in instruction processing
- Vector processors
  - Operations on multiple data stored in vector registers
- Shared-memory Multiprocessor (SMP)
  - Multiple processors sharing memory
  - Symmetric Multiprocessor (SMP)
- Multicomputer
  - Multiple computer connect via network
  - Distributed-memory cluster
- Massively Parallel Processor (MPP)
Supercomputing and Computational Science

- By definition, a supercomputer is of a class of computer systems that are the most powerful computing platforms at that time
- Computational science has always lived at the leading (and bleeding) edge of supercomputing technology
- “Most powerful” depends on performance criteria
  - Performance metrics related to computational algorithms
  - Benchmark “real” application codes
- Supercomputer architecture
  - 1980’s: vector processors, SMPs
  - 1990’s: MPP
  - 2000’s: clusters of SMP’s
Top 500 Linpack Benchmarking Methodology

- Listing of the 500 most powerful computers in the world
- Yardstick
  - $R_{\text{max}}$: maximal performance Linpack benchmark
    - Dense linear system of equations ($Ax = b$)
- Data listed
  - $R_{\text{peak}}$: theoretical peak performance
  - $N_{\text{max}}$: problem size needed to achieve $R_{\text{max}}$
  - $N_{1/2}$: problem size needed to achieve 1/2 of $R_{\text{max}}$
  - Manufacturer and computer type
  - Installation site, location, and year
Evolution of Supercomputing Architecture
History of the Most Powerful Supercomputer

All parallel computers (in one form or another)!
## Top 500 Linpack Benchmark List (June 2002)

<table>
<thead>
<tr>
<th>Computer (Full Precision)</th>
<th>Number of Processors</th>
<th>$R_{max}$ Gflop/s</th>
<th>$N_{max}$ order</th>
<th>$N_{1/2}$ order</th>
<th>$R_{peak}$ Gflop/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earth Simulator, NEC processors*</td>
<td>esc</td>
<td>5104</td>
<td>35610</td>
<td>1041216</td>
<td>265408</td>
</tr>
<tr>
<td>ASCI White-Pacific, IBM SP Power 3 (375 MHz)</td>
<td>ibnl</td>
<td>8000</td>
<td>7226</td>
<td>518096</td>
<td>179000</td>
</tr>
<tr>
<td>Compaq AlphaServer SC ES45/EV68 1GHz</td>
<td>psc</td>
<td>3016</td>
<td>4463</td>
<td>280000</td>
<td>85000</td>
</tr>
<tr>
<td>Compaq AlphaServer SC ES45/EV68 1GHz</td>
<td>psc</td>
<td>3024</td>
<td>4059</td>
<td>525000</td>
<td>105000</td>
</tr>
<tr>
<td>Compaq AlphaServer SC ES45/EV68 1GHz</td>
<td>cea</td>
<td>2560</td>
<td>3980</td>
<td>360000</td>
<td>85000</td>
</tr>
<tr>
<td>IBM SP Power3 208 nodes 375 MHz</td>
<td>ibnl</td>
<td>3328</td>
<td>3052.</td>
<td>371712</td>
<td>4992</td>
</tr>
<tr>
<td>Compaq Alphaserver SC ES45/EV68 1GHz</td>
<td>lanl</td>
<td>2048</td>
<td>2916</td>
<td>272000</td>
<td>4096</td>
</tr>
<tr>
<td>IBM SP Power3 158 nodes 375 MHz</td>
<td>ibnl</td>
<td>2528</td>
<td>2526</td>
<td>371712</td>
<td>102400</td>
</tr>
<tr>
<td>ASCI Red Intel Pentium II Xeon core 333MHz</td>
<td>snl</td>
<td>9632</td>
<td>2379.6</td>
<td>362880</td>
<td>75400</td>
</tr>
<tr>
<td>ASCI Blue-Pacific SST, IBM SP 604E (332 MHz)</td>
<td>ibnl</td>
<td>5808</td>
<td>2144.</td>
<td>431344</td>
<td>432344</td>
</tr>
<tr>
<td>ASCI Red Intel Pentium II Xeon core 333MHz</td>
<td>snl</td>
<td>9472</td>
<td>2121.3</td>
<td>251904</td>
<td>66000</td>
</tr>
<tr>
<td>Compaq Alphaserver SC ES45/EV68 1GHz</td>
<td>lanl</td>
<td>1520</td>
<td>2096</td>
<td>390000</td>
<td>71000</td>
</tr>
<tr>
<td>IBM SP 112 nodes (375 MHz POWER3 High)</td>
<td>ibm</td>
<td>1792</td>
<td>1791</td>
<td>275000</td>
<td>2688</td>
</tr>
<tr>
<td>HITACHI SR8000/MPP/1152 (450MHz)</td>
<td>u tokyo</td>
<td>1152</td>
<td>1709.1</td>
<td>141000</td>
<td>16000</td>
</tr>
<tr>
<td>HITACHI SR8000-F1/168 (375MHz)</td>
<td>leibniz</td>
<td>168</td>
<td>1653.</td>
<td>160000</td>
<td>19560</td>
</tr>
<tr>
<td>ASCI Red Intel Pentium II Xeon core 333MHz</td>
<td>snl</td>
<td>6720</td>
<td>1633.3</td>
<td>306720</td>
<td>52500</td>
</tr>
<tr>
<td>SGI ASCI Blue Mountain</td>
<td>lanl</td>
<td>5040</td>
<td>1608.</td>
<td>374400</td>
<td>138000</td>
</tr>
<tr>
<td>IBM SP 328 nodes (375 MHz POWER3 Thin)</td>
<td>noo</td>
<td>1312</td>
<td>1417.</td>
<td>374000</td>
<td>1968</td>
</tr>
<tr>
<td>Intel ASCI Option Red (200 MHz Pentium Pro)</td>
<td>snl</td>
<td>9152</td>
<td>1338.</td>
<td>235000</td>
<td>63000</td>
</tr>
<tr>
<td>NEC SX-5/128M8 (3.2ns)</td>
<td>osaka</td>
<td>128</td>
<td>1192.0</td>
<td>129536</td>
<td>10240</td>
</tr>
<tr>
<td>CRAY T3E-1200 (600 MHz)</td>
<td>us government</td>
<td>1488</td>
<td>1127.</td>
<td>148800</td>
<td>28272</td>
</tr>
<tr>
<td>HITACHI SR8000-F1/112 (375MHz)</td>
<td>leibniz</td>
<td>112</td>
<td>1035.0</td>
<td>120000</td>
<td>15160</td>
</tr>
</tbody>
</table>

Lecture 1

CIS 631 - Parallel Processing
Japanese Earth Simulator

- World’s fastest supercomputer!!
  - 640 NEC SX-6 nodes
    - 8 vector processors
  - 5104 total processors
  - Single stage crossbar
    - ~2900 meters of cables
  - 10 TB memory
  - 700 TB disk space
  - 1.6 PB mass storage
  - 40 Tflops peak performance
  - 35.6 Tflops Linpack performance
Top 10 Supercomputers (June 2002)

- Earth Simulator
- ASCI White
- AlphaServer SC ES45
- AlphaServer SC ES45/EV68
- SP Power3
- AlphaServer SC ES45
- ASCI Red
- ASCI Blue Pacific
- eServer pSeries p690 Turbo
- eServer pSeries p690 Turbo

TF/s: 40.83, 12.29, 6.03, 5.12, 4.99, 4.09, 3.21, 3.87, 3.99, 3.66
Research Centre Juelich Supercomputer

- Germany’s largest supercomputer!!!
  - 37 IBM p690 Turbo nodes
    - 32 Power 4 processors (1.3 GHz)
    - 64 GB RAM
  - 1184 total processors
  - IBM latest switch technology
  - 2.3 TByte RAM, 60 TByte disk
  - 1.2 PetaByte tape storage
  - 5.8 Tflops peak performance
    - 13 times present computing capability
Research Centre Juelich Supercomputer

- Installed at ZAM in summer 2003
- Building
  - Area: 1000 meter²
  - Energy: 1600 KVA
  - Air: 250,000 meter³/hour
- Availability
  - FZJ John von Neumann Institute of Computing
  - National German Grid
- Applications
  - Physics, chemistry, and life sciences
Amdahl’s Law

- $T_{seq}$: sequential execution time that cannot be parallelized
- $T_{par}$: sequential execution time that can be parallelized

$$T_1 = T_{seq} + T_{par} \quad T_{par} = T_1 - T_{seq}$$

$$T_p = T_{seq} + T_{par} / p \quad \text{(assume fully parallelized)}$$

- As $p \to \infty$, $T_p \to T_{seq}$

- Let $f_{seq}$ be the fraction $T_{seq} / T_1$ and $S_p = T_1 / T_p$

- Speedup $= S_p = T_1 / T_p = T_1 / (T_{seq} + T_{par} / p)$

  $$= 1 / (f_{seq} + T_{par} / pT_1) = 1 / (f_{seq} + (1 - f_{seq})/ p)$$

- As $p \to \infty$, $S_p = S_\infty \to 1 / f_{seq}$

- Speedup bound is determined by the degree of sequential execution time in the computation, not # processors!!!
Amdahl’s Law and Scaled Speedup

- Amdahl’s Law makes it hard to obtain good speedup

<table>
<thead>
<tr>
<th>$f_{seq} \times 100%$</th>
<th>10%</th>
<th>5%</th>
<th>2%</th>
<th>1%</th>
<th>.1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_\infty$</td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>100</td>
<td>1000</td>
</tr>
</tbody>
</table>

- Change perspective on the problem
- Consider scaling of problem size as # processors scale
- $T_{seq}$: sequential execution time (1 and $p$ processors)
- $T_{par}$: execution time in parallel mode on $p$ processors
- $T_p = T_{seq} + T_{par}$, $T_1 = T_{seq} + pT_{par}$
- Let $f_{par}$ be the fraction $T_{par} / T_p$
- Scaled speedup $= S_p = 1 + (p-1)f_{par}$
Scalable Parallel Computing

- Scalability in parallel architecture
  - Processor numbers
  - Memory architecture
  - Interconnection network
  - Avoid critical architecture bottlenecks
- Scalability in computational problem
  - Problem size
  - Computational algorithms
    - Computation to memory access ratio
    - Computation to communication ration
- Parallel programming models and tools
- Performance scalability
Next Lecture

- Parallel computer architectures
- Reading
  - Chapter 2
  - Culler lecture notes