

# High Performance Computing

COMP 633

Fall Semester 2007

<http://www.cs.unc.edu/~prins/Classes/633/>

## Meeting Times:

Tue, Thu 9:30 – 10:45 in SN 011 (Tue Aug 21 – Tue Dec 4)

## Instructor:

Jan Prins (SN 355, Tel: 962-1913, [prins@cs.unc.edu](mailto:prins@cs.unc.edu))

Office hours: by appt. or when door is open

## Secretary:

Madelyn Mann (SN 353, Tel: 962-1922, [mann@cs.unc.edu](mailto:mann@cs.unc.edu))

## **Overview**

This is an introductory graduate course covering several aspects of high-performance computing, primarily focused on parallel computing. Upon completion, you should

- (a) be able to design and analyze parallel algorithms for a variety of problems and computational models,
- (b) be familiar with the fundamentals of the architecture and systems software of high-performance parallel computing systems, and
- (c) have experience with the implementation of parallel applications on actual high-performance parallel computing systems, and be able to measure, tune, and report on their performance.

## **Course Announcements and Information**

The definitive source for course announcements, reading assignments, reference materials, and class handouts is the course web page at the top of this handout. Please consult it regularly!

## **Text and Readings**

There is no single text that adequately covers the material in this class. Course readings will be drawn from articles in the technical literature, textbook chapters, and instructor notes. The bibliography will be maintained on the course web page and readings will be distributed in class. A charge of \$25 for reproduction of materials must be paid by the end of the semester to receive a grade.

## Grading

Grades will be based on approximately three or four written assignments (~1/3 weight), three programming assignments performed individually or in groups of two (~1/3 weight), and two exams with combined weight ~1/3. All assignments may be freely discussed with your COMP 633 classmates, but your submissions must be written individually (or within your group, for programming assignments).

## Prerequisites

Undergraduate-level familiarity with the design and analysis of sequential algorithms (e.g. COMP 550), elementary operating systems concepts (e.g. COMP 530), and knowledge of basic computer organization (e.g. COMP 411) are expected.

## Related Courses

There are several related courses on parallel and distributed computing offered in this department.

**COMP 633 (this course)** is concerned with the design and implementation of scalable parallel computations, i.e. a single problem solved using multiple processors. Its focus is algorithms, architectures, and performance analysis.

**COMP 734 (Distributed Systems)** is concerned with the provision of ongoing reliable services to many users. Its focus is protocols, fault-tolerance, security, and scalability.

**COMP 735 (Distributed and Concurrent Algorithms)** is concerned with the specification and proof of safety and liveness properties of key algorithms used in concurrent systems such as mutual exclusion. Its focus is the application of formal techniques.

## Computer Usage

The selection of high-performance computing systems in the department and on campus is constantly changing. The equipment we will use this semester is not finalized, but most likely will include the campus cluster Topsail (1040 quad-core Intel Xeon processors and high speed interconnect), and the campus shared-memory multiprocessor Cedar/Cypress (128 Intel Itanium processors, 512 GB shared memory). Additional machines are available for projects targeting specific architectures, including GPU-based accelerators. We may also use the BIAS system scheduled for installation this fall in the department if it is available in time.

In all cases you will be making use of a shared resource. Please observe the usage guidelines and reservations for all these systems. Use common sense and monitor your program's consumption of resources when doing large-scale runs.

## Syllabus

The material in the course is organized around various models of parallel computing. With each model we will develop and analyze example algorithms, and study some practical issues such as hardware implementation, algorithm design, and performance evaluation.

Example algorithms include sorting, graph algorithms, linear algebra operations, and key algorithms from scientific computing (FFT, n-body simulation).

1. COURSE INTRODUCTION (1)
2. SHARED MEMORY MODELS (13)
  - PRAM and Work-Time Models: algorithm design and analysis techniques, relative power and limitations of PRAM models. (4)
  - Memory Models: memory-hierarchies and locality optimizations, UMA, NUMA and CC-NUMA shared memory architectures. (2)
  - Loop-Level Parallelism: iteration distribution and scheduling, performance measurement and tuning, OpenMP and CUDA. Nested parallelism and load balancing, Cilk. (3)
  - Thread-Level Parallelism: abstractions for exclusion and synchronization: locks, monitors and conditions, Java threads. (2)
  - Memory Consistency: coherence and consistency, implementation of synchronization and mutual exclusion operations in cache-coherent multiprocessors. (2)
3. DISTRIBUTED MEMORY MODELS (9)
  - Bulk Synchronous Processing Model: communication cost measures, algorithm design, performance prediction and measurement, cost parameters. (3)
  - Message Passing Model: SPMD programming, Message Passing Interface (MPI), collective communication, UPC. (3)
  - Interconnection Networks: topology and performance metrics, routing, congestion, and flow control; implementation of collective communication operations. (3)
4. DISTRIBUTED COMPUTING MODELS (1)
  - Client-server computing, peer-to-peer computing, and Grid computing: organization, capabilities, and limitations.