CIS*3090 Parallel Programming [0.50]

Course Outline for Fall 2016

INSTRUCTOR		COURSE LINKS		
Bill Gardner		Policies	Office Hours	Class Schedule
office	Reynolds 105	Resources	Assignments	Exams
phone	824-4120 x52696	Lecture Notes	Ombuddies	Grades
e-mail	gardnerw@uoguelph.ca			
web	www.uoguelph.ca/~gardnerw			

Students are responsible for monitoring e-mail to their university accounts concerning this course. Announcements will be posted on the CourseLink discussion groups.

Calendar course description

This course examines the current techniques for design and development of parallel programs targeted for platforms ranging from multicore computers to high-performance clusters, with and without shared memory. It includes theoretical models for, and hardware effects on, parallel computation, the definitions of speedup, scalability, and data- versus task-parallel approaches. The course will also examine strategies for achieving speedup based on controlling granularity, resource contention, idle time, threading overhead, work allocation, and data localization.

Prerequisites

(CIS*2030 or ENGG*3640), CIS*3110

Synopsis

Today's computer science students are entering a new era in parallel computing, featuring cheap multicores and high-performance clusters, but have received traditional largely-sequential training. This paradigm shift has been called "the end of the lazy programmer era." This course is aimed at helping soon-to-graduate students (1) move into jobs using current tools for parallel programming, and (2) acquire the theoretical background needed to keep abreast with rapid industry developments and to evolve with them. The textbook will provide foundational knowledge about modern parallel processor architectures and algorithms for organizing concurrent computations. Since parallel programming is all about speed, we will learn ways to measure execution performance and speedup through parallelization.

In terms of practical skills, high-performance (non-shared memory) cluster programming will be introduced via the University of Guelph **Pilot** library, based on MPI and utilizing message-passing. Programming for multicore shared memory processors will utilize the popular existing parallel programming technique of POSIX threads, and compiler-based OpenMP, supported by the latest suite of Intel tools, as well as Java threads. Heterogeneous architectures—GPUs (graphics processing units) and the Intel Xeon Phi—will be introduced.

Course topics

1. Historical perspective on concurrency in computing
2. Parallel computer system architectures

- Abstractions for modeling sequential and parallel computation
- Homogeneous vs. heterogeneous multiprocessors
- Shared vs. non-shared memory systems

3. Performance: speedup, Amdahl's Law, measurement, performance losses, difficulties in scaling
4. Abstractions and algorithms for parallel program design: task vs. data parallelism, scalable parallelism, reduce and scan, work assignment
5. Practical programming: message passing, Pilot library, POSIX threads, OpenMP, Java threads
6. Overview of CUDA and OpenACC for GPU programming
7. Overview of Intel Xeon Phi coprocessor
8. Future directions in hardware and software

Textbooks

Required text

Principles of Parallel Programming, by Calvin Lin and Larry Snyder, Addison-Wesley, 2009.

Avoid the first printing if possible. It has numerous small bugs affecting the code samples that you should carefully correct by hand: [errata]. With the second printing, you can skip this. There is only one edition.

Recommended texts (on reserve in library)

Patterns for Parallel Programming, by Mattson, Sanders, and Massingill, Addison-Wesley, 2005.

Structured Parallel Programming: Patterns for Efficient Computation, by McCool, Robison, and Reinders, Morgan Kaufmann, 2012.

The Art of Multiprocessor Programming, by Herlihy and Shavit, Morgan Kaufmann, 2008 [online].

Determination of final grade

Assignments	45%	Tests	55%
A1 due Oct. 4 & 12	10%	Quizzes	10%
A2 due Nov. 1 & 9	15%	Midterm Oct. 20	15%
A3 due Nov. 16	10%	Final exam Dec. 14	30%
A4 due Nov. 30	10%		

There will be three programming assignments using C and one using Java. Each assignment includes a written report with performance measurements. **Late assignments are not accepted.** These assignments are to be done individually. Therefore, appearances of unauthorized collaboration will be investigated for possible academic misconduct (see Policies link). Software tools may be used for this investigation.

Quizzes based on textbook chapters will be hosted on CourseLink and left open for about a week after each each chapter is discussed.