Introduction to Parallel Architectures and Programming
(An elective course for undergraduate and graduate studies)

G. Kedem, A. Lebeck, N. Pitsianis, and X. Sun
Aug. 25, 2008

Time, location and course numbers

Time : Tu. Th. 4:25pm – 5:40pm
Location : D-243 LSRC
Course numbers : CPS-196.1 and CPS-296.1

Course overview

In response to the paradigm shift in computer architectures that has taken place and its challenge on parallel algorithms and programming, this course introduces the basic knowledge and skills of parallel programming on modern architectures. Specific topics include single instruction multiple data (SIMD), multiple instruction multiple data (MIMD), multi-threading, distributed computation via shared memory and message passing, and their implementations on commodity multi-processors or multicores processors on desktops, laptops, as well as on Graphics Processor Units (GPUs) and the Cell processor in Playstation-3. Common tools for parallel programming will be introduced.

Prerequisites

The students are expected to have programming experience in C and basic knowledge of computer architecture.
Objectives

Upon completion of the course, the students will be able to program for more than one parallel architectures, test and debug parallel codes, and tune for performance, with certain existing tools.

Teaching and learning approach

Basic concepts, principles and approaches on parallel programming are introduced in lectures jointly given by Pitsianis, Sun, Kedem and Lebeck. The lectures are interleaved with programming practice, organized by a collection of parallel computation projects that vary in architectures and programming models. Independent and creative learning is combined and enhanced with discussion, individual practice and team collaboration.

Course materials

Lectures notes will be provided. Extra reading material will be also specified. The students will get access to parallel computing facility in hardware and software. Graduate students are expected to be able to map a computational task, preferably research related, with inherent concurrency into a parallel algorithm on a specific target architecture or parallel programming platform.

Assignments

There will be three programming assignments with incremental complexity for individual learning and implementation. There will be one final project, with or without collaborative efforts in small teams.

Evaluation

Final grade will depend 60% on individual programming assignments and 30% on the final group project, its poster presentation and performance, and 10% on intellectual participation in class.