Engineering Parallel Software

CS194: Engineering Parallel Software

Kurt Keutzer and Tim Mattson

Fall 2013

From cell phones to cloud computing, parallel processors are the computing platform of the future. This course will enable students to design, implement, optimize, and verify programs to run on parallel processors. Our approach to this course reflects our view that a well-designed *software architecture* is a key to designing parallel software, and a key to software architecture is design patterns and a pattern language. Our course will use this pattern language as the basis for describing how to design, implement, verify, and optimize parallel programs. Following this approach we will introduce each of the major patterns that are used in developing a high-level architecture of a program. These eight structural and thirteen computational patterns may be found at: http://parlab.eecs.berkeley.edu/wiki/patterns/patterns.

We also allow that writing efficient parallel programs requires insights into the hardware architecture of contemporary parallel processors as well as an understanding as to how to write efficient code in general. As a result a significant amount of time in the course will be spent on these topics as well.

Other lectures and laboratories of the course will focus on implementation using contemporary parallel programming languages, verification of parallel software using invariants and testing, and performance tuning and optimization.

Course Work and Grading

The course consists of twice-weekly lectures and once-weekly lab sessions. For the first two thirds of the course, there will be a series of programming assignments. There will be two examinations during the course.

Course Projects

The final third of the course will be an open-ended course project. Projects using quad-core cell phones will be among the acceptable platforms. Students will create their own projects in project teams of 4-6 students.

Course Staff

Professor: Kurt Keutzer

Guest Lecturer: Tim Mattson, Intel

TAs: Patrick Li, David Sheffield

Recommended Course Textbook

PDF of a revision of *Patterns for Parallel Programming*, T. Mattson, B. Sanders, B. Massingill, Addison Wesley, 2005. PDF will be provided.

Working List of Course Assignments

1. ***Computer Architecture -*** Measure L1/L2/L3 bandwidth and latency on our lab machines. Also, investigate measured ILP for a handful of different SGEMM implementations. Performance in MFlops/s increases, but ILP drops. Also serves as a warmup / refresher for the small subset of C++ we use for the lab assignments. Follows the material from lecture 3 (sequential processor performance)

2. ***Parallel Matrix Multiply (DGEMM) -*** Write naive parallel DGEMM using OMP for loops, OMP tasks, and pthreads.

Serves as a simple warm-up for the basic threading libraries. Advanced question on how GCC converts code with OpenMP pragmas into parallel code. Follows the material from lecture 2/4 (parallel programming on shared memory computers)

3. ***Optimize Matrix Multiply (DGEMM) -*** Optimize the naive parallel matrix multiply for both locality and data parallelism (using SSE2). Students get familiar with SSE2 intrinsics if they want to use them for their final projects. Follows the material from lecture 6/8 (memory subsystem performance)

4. ***Introduction to OpenCL -*** Students write both VVADD and SGEMM in OpenCL. They will write the kernels. Follows lecture 9 / 10. (Data parallelism and CUDA).

5. ***OpenCL + OpenGL -*** Students perform a handful of simple graphics operations on an image. Follows lecture 9 / 10. (Data parallelism and CUDA).

6. ***Advanced OpenCL -*** Students write a reduction routine using the ideas presented in class. They also write array compaction using scan. Follows lecture 9 / 10. (Data parallelism and CUDA).

Week	***Date***	***What***	***Topic***
Week 1	Tuesday 8/27	No class	
	Thursday 8/29	Lecture 1	First Lecture: Intro, Background, Course Objectives and Course Projects --Keutzer
Week 2	Tuesday 9/3	Lecture 2	A programers introduction to parallel computing: Amdahl's law, Concurrency vs. Parallelism, and the jargon of parallel computing. Getting started with OpenMP and Pthreads. --Mattson
	Thursday 9/5	Lecture 3	Sequential Processor Performance: Notions of performance: Insufficiency of Big-O, Example; Pipelining, Superscalar, etc.; Compiler Optimizations; Processor "Speed of Light" --Keutzer
	Monday 9/9	Discussion 1	Intro to the Lab Environment. Assignment 1 goes out.
	Tuesday 9/10	Lecture 4	C++ for Java/C Programmers; Working with OpenMP and Pthreads. Assignment 1 due. Assignment 2 goes out.

	Thursday 9/12	Lecture 5	--Keutzer
	Monday 9/16	Discussion 2	**Assignment 1 due**. Assignment 2 goes out.
	Tuesday 9/17	Lecture 6	Memory System Performance: Caches, Cache Hierarchies, benchmarking; Optimizing Matrix Multiplication --Keutzer
	Thursday 9/19	Lecture 7	Patterns – Another Way to Think About Parallel Programming - Keutzer
	Monday 9/23	Discussion 3	**Assignment 2 due.** Study for midterm.
	Tuesday 9/24	Lecture 8	Optimizing Matrix Multiply, Introduction to Parallel Processor Architectures: Multi-Core, Cache Coherence, Memory Consistency; SIMD / SIMT; Vectors; NUMA --Mattson
	Thursday 9/26	Lecture 9	Data Parallelism --Mattson
Week 6	Monday 9/30	Discussion 4	Assignment 3 goes out.
	Tuesday 10/1	Lecture 10	***Midterm 1***
	Thursday 10/3	Lecture 11	Introduction to CUDA and OpenCL --Sheffield
Week 7	Monday 10/7	Discussion 5	**Assignment 3 due.** Assignment 4 goes out.
	Tuesday 10/8	Lecture 12	CUDA and OpenCL continued. --Sheffield
	Thurs 10/10	Lecture 13	Distributed Memory Systems, Supercomputing, and MPI --Mattson

Week 8	Monday 10/14	Discussion 6	**Assignment 4 due.** Assignment 5 goes out.
	Tuesday 10/15	Lecture 14	Midterm review/Project proposals due.
	Thursday 10/17	Lecture 15	Design patterns, pattern languages, PLPP overview - Keutzer

Week 9	Monday 10/21	Discussion 7	**Assignment 5 due.** Assignment 6 goes out.
	Tuesday 10/22	Lecture 16	PLPP algorithm structure and supporting structures --Keutzer
	Thurs 10/24	Lecture 17	Structural patterns and software architecture --Keutzer

Week 10	Monday 10/28	Discussion 8	**Assignment 6 due.** Midterm 2 review and discussion.
	Tuesday 10/29	Lecture 18	Graph algorithms, dynamic programming, and speech recognition--Keutzer
	Thursday 10/31	Midterm 2	***Midterm 2***

Week 11	Monday 11/4	Discussion 9	Project meetings: show up with evidence of work!
	Tuesday 11/5	Lecture 19	Project meetings: show up with evidence of work! Speech - part2 --Keutzer, Sheffield
	Thursday 11/7	Lecture 20	Sparse linear algebra and image contour detection --Sheffield

Week 12	Monday 11/11	Discussion 10	Project meetings: show up with evidence of work!
	Tuesday 11/12	Lecture 21	Principle component analysis and 3D reconstruction --Sheffield
	Thursday 11/14	Lecture 22	Object recognition --Sheffield

Week 13	Monday 11/18	Discussion 11	Project meetings: show up with evidence of work!
	Tuesday 11/19	Lecture 23	Optimization patterns --Sheffield
	Thurs 11/21	Lecture 24	Future of parallel computing --Keutzer

Week 14	Tues 11/26	Lecture 25	Use class time to talk about projects

Week 15	Monday 12/2	Discussion 12	Final exam review
	Tuesday 12/3	Lecture 26	Project presentations
	Thurs 12/5	Lecture 26	Project presentations

Final Exam:
Final project write-up due date: December 6th, 2013
Grades:
35% Assignments
30% Midterm and final exam
35% Final project
10% attendance and class participation -- Bonus