CS C267 Syllabus

06/15/2022

Brief Description

Models for parallel programming. Overview of parallelism in scientific applications and study of parallel algorithms for linear algebra, particles, meshes, sorting, FFT, graphs, machine learning, etc. Survey of parallel machines and machine structures. Programming shared- and distributed-memory parallel computers, GPUs, and cloud platforms. Parallel programming languages, compilers, libraries, and toolboxes. Data partitioning techniques. Techniques for synchronization and load balancing. Detailed study and algorithm/program development of medium sized applications.

Background and Motivation

CS C267 was originally designed to teach students how to program parallel computers to efficiently solve challenging problems in science and engineering, where very fast computers are required either to perform complex simulations or to analyze enormous datasets. CS C267 is intended to be useful for students from many departments and with different backgrounds, although we will assume reasonable programming skills in a conventional (non-parallel) language, as well as enough mathematical skills to understand the problems and algorithmic solutions presented. CS C267 also satisfies part of the course requirements for the Designated Emphasis ("graduate minor") in Computational and Data Science and Engineering.

While this general outline remains, a large change in the computing world started in the mid 2000's: not only are the fastest computers parallel, but nearly *all* computers are becoming parallel, because the physics of semiconductor manufacturing will no longer let conventional sequential processors get faster year after year, as they have for so long (roughly doubling in speed every 18 months for many years). So, *all* programs that need to run faster will have to become parallel programs. (It is considered very unlikely that compilers will be able to automatically find enough parallelism in most sequential programs to solve this problem.) For background on this trend toward parallelism, click here.

Students in CS C267 will get an overview of the parallel architecture space, gain experience using some of the most popular parallel programming tools, and be exposed to a number of open research questions. The lectures will also cover a broad set of parallelization strategies for applications covering numerical simulation and data analysis to machine learning.

Cross-listed with

ENGIN C233

Pre-requisites

No formal pre-requisites. Prior programming experience with a low-level language such as C, C++, or Fortran is recommended but not required.

CS C267 is intended to be useful for students from many departments and with different backgrounds, although we will assume reasonable programming skills in a conventional (non-parallel) language, as well as enough mathematical skills to understand the problems and algorithmic solutions presented.

Format

This class is offered both in-person and online to a limited number of students.

- In-person format: 3 hours lecture and five 1 hour in-person labs

- Online format: 3 hours asynchronous lecture and five 1 hour on-line labs

Units

- 4 units for in-person
- 3 units for online

Grading

In-person	Online
Class survey: 1%	Class survey: 2%
HW1: 8%	HW1: 16%
HW2.1, HW2.2, HW2.3: 9% each	HW2.1, HW2.2, HW2.3: 18% each.
HW3: 9%	HW3: 18%
Weekly Quizzes: 10%	Weekly Quizzes: 10%.
Project 45%	NA

Late work policy:

2% of assignment worth is deducted every day past the due date. No credit is given after 10 days. This policy applies only to homework, the project pre-proposal, and the project proposal. This policy does not apply to quizzes, the pre-course survey, or the final project poster and report. This information is posted on the class webpage.

Details by week due

Week 1: Introduction, Memory Hierarchies and Matrix Multiplication

Class survey due

Week 2: Shared Memory Parallelism, Roofline and Performance Modeling

Week 3: Sources of Parallelism and Locality, Communication-Avoiding Matrix Multiplication

HW1 due: Optimizing Matrix Multiplication

Week 4: Data Parallel Algorithms, Introduction to GPUs

Week 5: Distributed Memory Machines, and Programming with MPI

HW2.1 due: Parallelizing particle simulation in shared memory

Week 6: Partitioned Global Address Space Languages (UPC++), Domain Specific Languages, Distributed Data Structures

HW2.2 due: Parallelizing particle simulation with GPUs

Week 7: Distributed-Memory Parallel Matrix Multiplication and Dense Linear Algebra

HW2.3 due: Parallelizing particle simulation with MPI in distributed memory

Week 8: Sparse Iterative Solvers (mostly SpMV) and Structured Grids

HW3 due: Parallelizing Genome Assembly using UPC++

Week 9: Parallel Machine Learning

Week 10: Graph Algorithms and Graph Partitioning

Week 11: Fast Fourier Transform and Cloud Computing

Week 12: Computational Cosmology and Dynamic Load Balancing

Week 13: Hierarchical Methods for the N-Body Problem, Sorting and Searching

Week 14: Computational Biology, Quantum Computing

Final projects due

Office Hours and Student Support

Faculty offer weekly office hours in person and/or via zoom. In addition, GSIs offer office hours via zoom. Faculty and GSI emails are displayed on the class home page as well as class-specific email that is monitored by all teaching staff to ensure a timely response. Finally, the class also offers a Piazza board.

Final Project Information

Proposal

- accounts for 4% of the grade
- page limit: 2 pages

Project & Poster Presentation

- Can be done alone or as a team
- final report length limit: 10 pages, single-column standard latex.
- Final project and poster presentation account for 40% of the grade.

Grading – The first two components are the most important

1. Practical content/creativity; implementation/tuning effort.
2. Experimental data: scaling/performance analysis/interesting inputs or outputs.
3. Theoretic content/creativity; design/analysis of algorithms.
4. Impact: difficulty and timeliness of the contribution.

Honor Code and Student Conduct:

Students are expected to conform to the UC Berkeley Honor Code which states: "As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others."

The Center for Student Conduct https://sa.berkeley.edu/conduct/policies has the full texts of campuswide policies and regulations regarding student rights, including: Privacy and Disclosure of Information from Student Records; Nondiscrimination Policy; Policy on Sexual Harassment and Sexual Violence; Civility and Respect in an Atmosphere of Academic Freedom

Reasonable Accommodation for Students' Religious Beliefs, Observations and Practices:

In compliance with Education code, Section 92640(a), it is the official policy of the University of California at Berkeley to permit any student to undergo a test or examination, without penalty, at a time when that activity would not violate the student's religious creed, unless administering the examination at an alternative time would impose an undue hardship which could not reasonably have been avoided.

Disabled Students' Services:

We tell the students that "UC Berkeley is committed to creating a learning environment that meets the needs of its diverse student body including students with disabilities. If you anticipate or experience any

barriers to learning in this course, please feel welcome to discuss your concerns with the me. If you have a disability, or think you may have a disability, you can work with the Disabled Students' Program (DSP) to request an official accommodation: dsp.berkeley.edu. If you have already been approved for accommodations through DSP, please meet with me so we can develop an implementation plan together."

Collaborative Policy

All the homework assignments are done in teams, where faculty assign teams of size 3 for HW1 to make sure that inexperienced programmers are teamed up with more experienced ones. Students are free to pick their own teams in later assignments. While faculty allow open collaboration within teams, each student is expected to write a few sentences about their personal contributions in what they turn in.

Ethics

Students are expected to identify any code found on-line and (re)used for their projects. While faculty expects that homework will be done "from scratch", some pre-identified/faculty-approved tools are permissible to use (eg automatic code-generation tools).

Online Engagement and Emergency Changes to Accessing Course Materials

Outside events (e.g. public health emergencies, campus safety directives, or temporary power outages) may require changes to the modes of engagement that will be available to students to complete the course requirements. If events occur at any point during the semester that require these changes, students will receive formal notification from both FPF administration and the instructor. The details of the specific changes or adaptations made to the course will be communicated via email, bCourses, and, when possible, in-class announcements. Students may receive an amended syllabus.

bCourses will continue to be students' main point of remote entry for class meetings, even if lectures, discussion sections, and office hours will be administered using third-party software, such as Zoom or Adobe Connect. Students will continue to use their CalNet ID login and berkeley.edu address to access content. In the event a student is required to download or update specific software for home use in order to access the course materials, or if students need specific technical assistance setting up their remote access, please use https://software.berkeley.edu/

Resources

See the previous class web page https://sites.google.com/lbl.gov/cs267-spr2022/ for details.
Regarding computing resources, we expect students to have access to laptops, but supply supercomputing facilities provided for free from both DOE (at NERSC/LBL) and NSF for homework and projects.