EE 599 Parallel Programming

Units: 4
Lecture: MW 3-450pm
Discussion: TBD on Fridays, 1 hr 50 mins

Note: This course has been approved by the CS department to count towards MS CS program requirements. No additional approval needed.

Location: Virtual

Instructor: Viktor K. Prasanna
Office: EEB 200C
Office Hours: TBD
Contact Info: prasanna@usc.edu

Teaching Assistant: TBD
Office: TBD
Office Hours:
Contact Info: TBD

Course Description

This graduate level course focuses on parallel and distributed computing using various programming models.

Topics covered in this course will include parallel computation models, message passing and shared memory paradigms, data parallel programing, performance modeling and optimization, memory system optimization techniques, fine grained computation models and High Level Design Tools for programming parallel platforms, communication primitives, stream programming models, emerging heterogeneous computing and programming models.

This course will study the abstractions for parallel programming as well as provide students with hands-on experience with state-of-the-art parallel computing platforms and tools including large scale clusters, edge devices and data center scale platforms. A course project will enable students to study various computing platforms and design efficient parallel algorithms on them, evaluate their performance and learn performance tuning.

Learning Objectives and Outcomes

- Understand the key parallel computational models
- Write parallel programs using message passing and shared memory paradigms
- Implement key algorithms using data parallel programming model
- Understand basic principles of performance modeling and optimization
- Understand memory system optimization techniques
- Implement key algorithms using fine grained computation models and High Level Design Tools
- Understand communication and coordination issues in parallel computing
- Implement key algorithms in a stream programming model
- Understand heterogeneous computing and programming models for accelerator enhanced parallel computation

Prerequisite(s): Background in high level programming (for ex. using C, C++) at the level of EE455x
Co-Requisite(s): None
Concurrent Enrollment: None
Recommended Preparation: EE 457 or EE 451

Course Notes

Copies of all course notes will be posted on Blackboard.

Required Readings and Supplementary Materials

Portion of the course will be based on the later chapters in the book - Introduction to Parallel Computing, Grama, Karypis, Kumar, Gupta; 2nd Edition (January 1, 2003); Publisher: Pearson College Div; ISBN-10: 0201648652; ISBN-13: 978-0201648652. Recent research publications and survey articles will be used to covered advanced materials after the 6th week of the semester. This book has lot of materials; advanced materials regarding programming models and examples will be covered from this book. Details will be provided in the lectures as well as in the discussion sessions. A sample of relevant literature is included in the appendix.

Description and Assessment of Assignments

This course will have a midterm and class project. There will also be 6-8 homework assignments as well as programming homeworks. Student accounts will be created on USC HPC GPU Cluster (https://carc.usc.edu/user-information/user-guides/high-performance-computing/using-gpus). For Cloud access, Amazon Web Services (AWS) and other educational services offered by cloud service providers (for

example, AWS Educate, https://aws.amazon.com/education/awseducate/) that offer cloud access at no cost to the students will be employed. The class project will begin after the midterm and be broken into three phases:

Phase 1: entails problem definition and project proposal submission. Students will submit a written report (6-8 pages, single spaced) detailing the context, related work, problem definition, project hypothesis, tools to be used and evaluation methodology.
Phase 2: The students individually or in groups of upto 3 will design and implement the project using various software and tools covered in the discussion sessions, evaluate the performance of their design and fine tune the software if needed.
Phase 3: will include project presentation and reporting. It will include a required report (10-12 pages, single spaced) and class presentation (about 20 mins and 5 mins for Q and A using power point or other tools) that will be graded for both content and clarity. All students are expected to participate in the class presentation. If needed due to time constraints some presentations will be held outside of normal class times. Templates for presentation and the final report will be provided.

Project Grading Rubric

The course project will be graded as follows:

Project Proposal: 30%
The project proposal will be assessed for context (10%), problem definition (20%), hypothesis (20%), software and tools to be used (50%).
Project Presentation: 20%
Presentation will be judged for both content (75%) and delivery (25%).
Final Report: 50%
The project final report will be assessed for description of the approach and design methodology (50%), results obtained (25%) and description and comparison with state of the art (25%).

Sample Projects

1. Ray tracing is one of the cores algorithms used in the film and graphics industry to create realistic computer generated imagery. However, obtaining high quality images using ray tracing is computationally intensive.
   The objective of this project is to identify opportunities for parallelization in the ray tracing algorithms and implement a parallel algorithm using programming tools and platforms studied in the course. The success of the project will be evaluated by comparing the performance of the parallel implementation against a baseline code using various performance metrics studied in the course. Implementations on CPU, GPU using heterogenous programming models and interfacing data parallel and task parallel approaches will also be studied.
2. Recently graph embedding techniques have been proposed for many machine learning applications including personalized recommendation systems. Throughput as well as latency are important metrics in implementing applications based on graph convolutional networks (GCN) used in graph embedding. Many techniques have been proposed for GCN training. GCN models can be deep and the input graphs can be very large. This project will explore techniques for parallelization using task and data parallel paradigms for both full batch and mini batch inference (embedding) and implement a complete application using accelerated GCN implementation.

Grading Breakdown

Assignment	% of Grade
Homework	20%
Programming Homework	20%
Midterm	30%
Project	30%
TOTAL	100%

Assignment Submission Policy

Assignments will be submitted electronically on Blackboard. The file format will be C/C++ for CPU based programs and CUDA for GPU based programs. Late assignments will be accepted with penalty, 5% per day, unless otherwise announced.

Grading Timeline

Homework and midterms will be graded and returned within 2 weeks.

Additional Policies

None.

Course Schedule: A Weekly Breakown*

Week	Topics/Daily Activities	Readings / Homework	Discussion Section	Deliverable/ Due Dates
1	Introduction, Parallel Computation Models (1): Parallel random access models and variants, examples, programming abstractions, simulations	Chapters 2.4.1 (Book suggested in required reading)	Account setup and lab overview	
2	Parallel Computation Models (2): Synchronous and asynchronous models, network models, performance analysis	HW 1 out Chapters 2.4.3-2.4.5, 2.5-2.7	PRAM examples and analysis	
3	Shared Memory and Message Passing programming models, illustrative examples, OpenMP, MPI	HW 2 out Chapters 6, 7	OpenMP Programming HW 1 out	HW 1 due
4	Data Parallel Programming: SIMT models, programming abstractions and examples, CUDA and related models	HW 3 out Chapter 2.3.1	MPI Programming HW 2 out	HW 2 due
5	Performance Modeling and Optimization of parallel programs, roof line model, external memory and Logp models.	Chapter5, Appendix, item 3	CUDA programming (1)	HW 3 due/ PHW 1 due
6	Memory System Optimization (Data Reuse, Data Layout, Replication for Conflict Free Memory access)	HW 4 out Appendix, item 18	CUDA programming (2)	PHW 1 due
7	Fine Grained Computation Models – use of HLS for application acceleration, Systolic Arrays, space time computation	HW 5 out Appendix, item 11 – chapter 10.2.3, Appendix, items 4, 6	HLS Programming HW 3 out	HW 4 due/ PHW 2 due
8	Communication bounds, trade offs and parallel and distributed communication avoiding algorithms	HW 6 out Appendix, item 20	Midterm Review	HW 5 due
9	Midterm / Introduction to Course Project			HW 6 due
10	Stream Programming Models, Cloud programming, MapReduce and high level models	HW 7 out Appendix, items 12, 13	Streaming Programming, Programming HW 4 out	PHW 3 due Project Proposal due
11	Heterogeneous Computing and Programming Models (1): Accelerators, interface mechanisms and performance modeling.	Appendix, item 13	Heterogeneous Computing	HW 7 due
12	Heterogeneous Computing and Programming Models. Extensions to OpenMP for accelerated computing. OpenCL, OPAE, Vitis, etc.	HW 8 out Appendix, item 9, 19,	Heterogeneous Computing, Programming HW 5 out	PHW 4 due
13	Advanced Topics (DSL, real-time programming, PGAS, SYCL)	Appendix, items 15,16, 17	Advanced programming topics	HW 8 due
14	Additional Advanced Lecture Materials (Parallel shortest path).	Appendix item 21	Advanced programming topics	PHW 5 due

15	Course Project Presentations or Possible guest lecture from industry (Microsoft, Intel, Apple, Google, etc.) and national labs.		Course Project Presentations based on student availability and schedule or Advanced programming topics	
FINAL Exam Period	Course Project Presentations based on student schedule			Final report due according to University's final exam schedule

*Student project presentations will be conducted as much as possible during the final exam period but as there is not enough time, we will allow to also schedule presentations during the last week. The actual schedule will depend on the number of projects and availability of the students during the final exam period. Based on this, additional topics in the reading list will be covered in the last week of the semester.

Statement on Academic Conduct and Support Systems

Academic Conduct:

Plagiarism – presenting someone else's ideas as your own, either verbatim or recast in your own words – is a serious academic offense with serious consequences. Please familiarize yourself with the discussion of plagiarism in SCampus in Part B, Section 11, "Behavior Violating University Standards" policy.usc.edu/scampus-part-b. Other forms of academic dishonesty are equally unacceptable. See additional information in SCampus and university policies on scientific misconduct, policy.usc.edu/scientific-misconduct.

Support Systems:

Student Health Counseling Services - (213) 740-7711 – 24/7 on call
engemannshc.usc.edu/counseling
Free and confidential mental health treatment for students, including short-term psychotherapy, group counseling, stress fitness workshops, and crisis intervention.

National Suicide Prevention Lifeline - 1 (800) 273-8255 – 24/7 on call
suicidepreventionlifeline.org
Free and confidential emotional support to people in suicidal crisis or emotional distress 24 hours a day, 7 days a week.

Relationship and Sexual Violence Prevention Services (RSVP) - (213) 740-4900 – 24/7 on call
engemannshc.usc.edu/rsvp
Free and confidential therapy services, workshops, and training for situations related to gender-based harm.

Office of Equity and Diversity (OED) | Title IX - (213) 740-5086
equity.usc.edu, titleix.usc.edu
Information about how to get help or help a survivor of harassment or discrimination, rights of protected classes, reporting options, and additional resources for students, faculty, staff, visitors, and applicants. The university prohibits discrimination or harassment based on the following protected characteristics: race, color, national origin, ancestry, religion, sex, gender, gender identity, gender expression, sexual orientation, age, physical disability, medical condition, mental disability, marital status, pregnancy, veteran status, genetic information, and any other characteristic which may be specified in applicable laws and governmental regulations.

Bias Assessment Response and Support - (213) 740-2421
studentaffairs.usc.edu/bias-assessment-response-support
Avenue to report incidents of bias, hate crimes, and microaggressions for appropriate investigation and response.

The Office of Disability Services and Programs - (213) 740-0776
dsp.usc.edu
Support and accommodations for students with disabilities. Services include assistance in providing readers/notetakers/interpreters, special accommodations for test taking needs, assistance with architectural barriers, assistive technology, and support for individual needs.

USC Support and Advocacy - (213) 821-4710
studentaffairs.usc.edu/ssa

Assists students and families in resolving complex personal, financial, and academic issues adversely affecting their success as a student.

Diversity at USC - (213) 740-2101

diversity.usc.edu

Information on events, programs and training, the Provost's Diversity and Inclusion Council, Diversity Liaisons for each academic school, chronology, participation, and various resources for students.

USC Emergency - UPC: (213) 740-4321, HSC: (323) 442-1000 – 24/7 on call

dps.usc.edu, emergency.usc.edu

Emergency assistance and avenue to report a crime. Latest updates regarding safety, including ways in which instruction will be continued if an officially declared emergency makes travel to campus infeasible.

USC Department of Public Safety - UPC: (213) 740-6000, HSC: (323) 442-120 – 24/7 on call

dps.usc.edu

Non-emergency assistance or information.

Appendix – Sample materials from Relevant Literature

1. Hai Jin, Wenchao Wu, Xuanhua Shi, Ligang He, and Bing B. Zhou. "TurboDL: Improving CNN Training on GPU with Fine-grained Multi-streaming Scheduling." *IEEE Transactions on Computers* (2020).
2. Ta-yang Wang, Ajitesh Srivastava and Viktor K. Prasanna, "A Framework for Task Mapping onto Heterogeneous Platforms." IEEE HPEC 2020.
3. Yu Jung Lo, Samuel Williams, Brian Van Straalen, Terry J. Ligocki, Matthew J. Cordery, Nicholas J. Wright, Mary W. Hall, and Leonid Oliker. "Roofline model toolkit: A practical tool for architectural and program analysis." In International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems, pp. 129-148. Springer, Cham, 2014.
4. Zhi-Gang Liu, Paul N. Whatmough, and Matthew Mattina. "Systolic Tensor Array: An Efficient Structured-Sparse GEMM Accelerator for Mobile CNN Inference." *IEEE Computer Architecture Letters* 19, no. 1 (2020): 34-37.
5. Wonchan Lee, Manolis Papadakis, Elliott Slaughter, and Alex Aiken. "A constraint-based approach to automatic data partitioning for distributed memory execution." In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-24. 2019.
6. Chen, Peng, Mohamed Wahib, Shinichiro Takizawa, Ryousei Takano, and Satoshi Matsuoka. "A versatile software systolic execution model for GPU memory-bound kernels." In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-81. 2019.
7. Abdelrahman, Tarek S. "Cooperative Software-hardware Acceleration of K-means on a Tightly Coupled CPU-FPGA System." *ACM Transactions on Architecture and Code Optimization (TACO)* 17.3 (2020): 1-24.
8. Kangas, Niko. "A Comparison of High-Level Synthesis and Traditional RTL in Software and FPGA Design." (2020).
9. Kathail, Vinod. "Xilinx Vitis Unified Software Platform." *The 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. 2020.
10. De Matteis, Tiziano, Johannes de Fine Licht, Jakub Beránek, and Torsten Hoefler. "Streaming Message Interface: High-performance distributed memory programming on reconfigurable hardware." In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-33. 2019.
11. Kastner, Ryan, Janarbek Matai, and Stephen Neuendorffer. "Parallel programming for FPGAs." arXiv preprint arXiv:1805.03648 (2018).
12. de Assuncao MD, da Silva Veith A, Buyya R. Distributed data stream processing and edge computing: A survey on resource elasticity and future directions. Journal of Network and Computer Applications. 2018 Feb 1;103:1-7.
13. Alam, Md Imran, Manjusha Pandey, and Siddharth S. Rautaray. "A comprehensive survey on cloud computing." International Journal of Information Technology and Computer Science 2.2 (2015): 68-79.
14. Mittal, Sparsh, and Jeffrey S. Vetter. "A survey of CPU-GPU heterogeneous computing techniques." ACM Computing Surveys (CSUR) 47.4 (2015): 1-35.
15. Portugal, Ivens, Paulo Alencar, and Donald Cowan. "A preliminary survey on domain-specific languages for machine learning in big data." 2016 IEEE International Conference on Software Science, Technology and Engineering (SWSTE). IEEE, 2016.
16. Kirsch, Christoph M., and Raja Sengupta. "The evolution of real-time programming." Handbook of Real-Time and Embedded Systems (2006): 11-1.
17. Alpay, Aksel, and Vincent Heuveline. "SYCL beyond OpenCL: The architecture, current state and future direction of hipSYCL." Proceedings of the International Workshop on OpenCL. 2020.
18. Panda, Preeti Ranjan, et al. "Data and memory optimization techniques for embedded systems." ACM Transactions on Design Automation of Electronic Systems (TODAES) 6.2 (2001): 149-206.
19. Cross-Platform, F. P. G. A., and Application Developers. "Simplify Software Integration for FPGA Accelerators with OPAE."

20. Demmel, Jim. "Communication avoiding algorithms." 2012 SC Companion: High Performance Computing, Networking Storage and Analysis. IEEE, 2012.
21. Andoni, Alexandr, Clifford Stein, and Peilin Zhong. "Parallel approximate undirected shortest paths via low hop emulators." Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing. 2020.