CME 213, ME 339 Spring 2023
Introduction to parallel computing using MPI, openMP, and CUDA

We are meeting in

200-034

Stanford University

CME 213, ME 339 Spring 2023
Introduction to parallel computing using MPI, openMP, and CUDA

Eric Darve, ICME, Stanford

"The city's central computer told you? R2D2, you know better than to trust a strange computer!" (C3PO)

Stanford University

Instructor

- Eric Darve, ME, ICME, darve@stanford.edu
- Research interests: numerical linear algebra, machine learning for mechanics and engineering, parallel computing

Teaching assistants

Pranil Joshi, pranil@stanford.edu

Fun on the quad

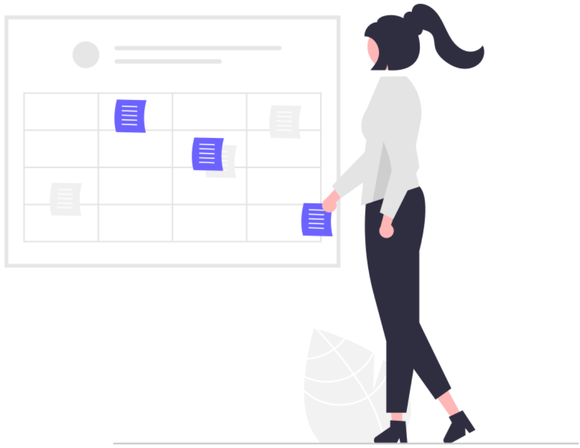
Teaching assistants

Quinn Hollister, bh9vw@stanford.edu

Guest lectures

- **NVIDIA engineers:** Jonathan Wong, Xavier Simmons, Akshay Subramaniam.
- Topics: CUDA optimizations, CUDA profiling, ML, etc.
- **Elliott Slaughter from SLAC.**
- Topics: Legion and Regent, task-based parallel runtimes.

Course Schedule

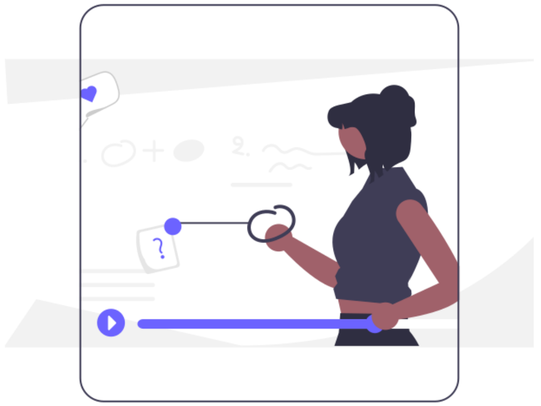


- **Lectures will be in ~~Hewlett Teaching Center 102~~ 200-034 MF and 260-113 on W.**
- We will have two regular lectures every week from 1:30 PM to 2:50 PM.
- On some weeks, we may move the Monday lecture to Friday to accommodate scheduling constraints.

Enrollment

- Combined Section Capacity: 60
- Wait List Capacity: 20
- Enrollment Total: 60
- Wait List Total: 5

C++ tutorial this week

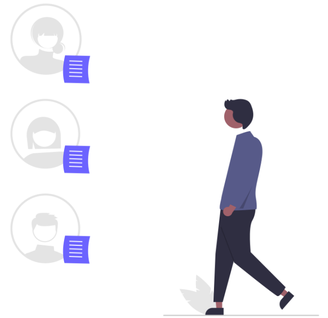


- Quinn will run a C++ tutorial this week.
- It will take place Friday during our regular class time.
- It will cover the basics of C++ and Makefiles.

Online resources

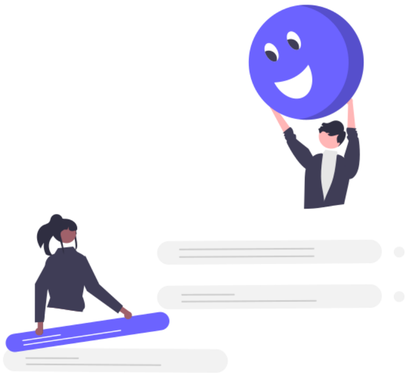
- Canvas: grades, class information, class and reading material, office hours.
- Discussion forum: Ed Discussion. See Canvas tab.
- Homework will be graded using Gradescope. See Canvas for the entry code.

How to provide feedback



- Come to office hours and after lectures.
- Send a private message using Ed Discussion. It will be visible to the entire teaching staff.
- Email Eric Darve, darve@stanford.edu.
- Use the anonymous Google form (see Canvas). Because it is anonymous, we won’t be able to reply to you.
- Private meetings with the instructor can be arranged. Office: bldg 520, room 125.

Rules of conduct on forum



- Be civil, considerate, and courteous to everyone. The forum is meant to be a safe and welcoming space to get help. If your message is not useful to other students, yourself, or the instructors, you should probably just delete it.

- Access to the various forums will be revoked without warning if you post an inappropriate, disrespectful, demeaning, or abusive message.

Grading

Homework assignments:	60%
Final project:	40%

More on this later

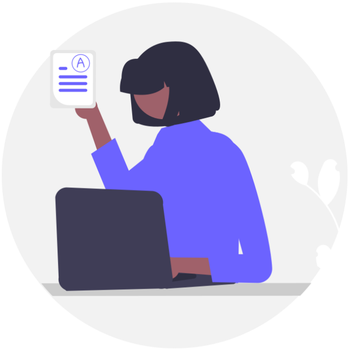
How to turn in your homework assignments

Assignments will typically be in two parts:

1. **Computer code** in C++. **A Python script** will be provided to send your files to the teaching staff. This requires you to copy the files to a private folder on *cardinal.stanford.edu*.

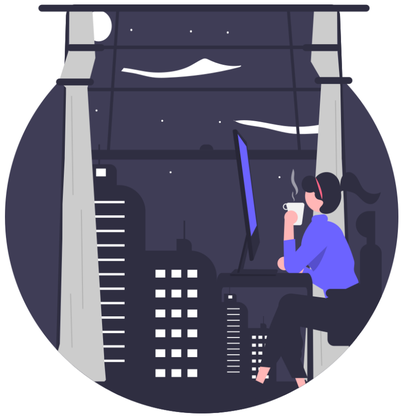
2. Written responses. These can be hand-written or typed. They will contain code outputs, plots, tabular data, written answers, etc. **A PDF file** must be turned in using **Gradescope**.

Regrades



- After receiving your grade on Gradescope, you are welcome to request a regrade using the Gradescope interface.
- No one is perfect. We strive to grade accurately, fairly, consistently and provide useful feedback to help you, but mistakes do happen. We will be happy to address any concern you have.
- To help with the logistics, we prefer that you **submit your regrade request at most 1 week after** the grade has been released.
- Email darve@stanford.edu for private concerns regarding grading.

Late submission policy



- You will be able to submit your code and paper up to **48 hours late for a 10% deduction.**
- Exceptions can be granted, but they must be requested at least **24 hours in advance** using the form on Canvas.
- A valid reason such as sickness and travel must be provided.

Overview of the content of the class

- Multicore programming; C++ threads, OpenMP, shared memory computers. Example: laptop and desktop computers.
- NVIDIA GPU programming; general purpose computing on GPUs, CUDA, openACC, performance, profiling, architecture of NVIDIA GPUs.
- Supercomputers, computer clusters, cloud computing, and distributed memory programming; MPI (Message Passing Interface). This is a library to program clusters comprised of computing nodes connected by a network.
- Concepts will be illustrated using examples that include sorting algorithms, linear algebra, finite-difference, neural networks, parallel primitives.

Pre-requisites

- Basic knowledge of UNIX (ssh, compilers, makefile)
- Knowledge of C and C++ (including pointers, memory, templates, polymorphism, standard library)
- Proficiency in scientific programming, including **testing, verification, and debugging**
- Pre-requisite classes: CME 211, CME 212, CS 106B, or CS 106L, or sufficient programming experience in C/C++.

C++ references

- cppreference.com: https://en.cppreference.com/w/cpp/language
- cplusplus.com: https://cplusplus.com/doc/tutorial/
- Many other online tutorials

Computing resources

For this class, you can expect to use:

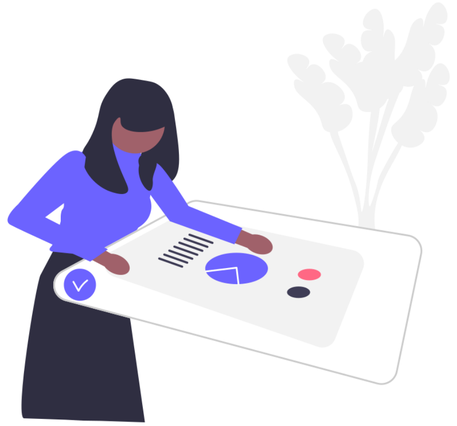
- Your own **laptop** is suitable for many assignments, including OpenMP and MPI. Some software needs to be installed, but this is a straightforward process.

- **Google Colab.** We will show you how to use Colab notebooks to run parallel codes in the Cloud. No computer set up is required! Google Colab can even run GPU code.

- **FarmShare.** You can use the `rice` cluster for some of the assignments (except assignments requiring GPU processors).

- **ICME GPU cluster;** accounts will be created for you on the cluster. **This cluster must be used for all performance benchmarks and for all assignments requiring GPU processors.** You can still use Google Colab for testing and debugging.

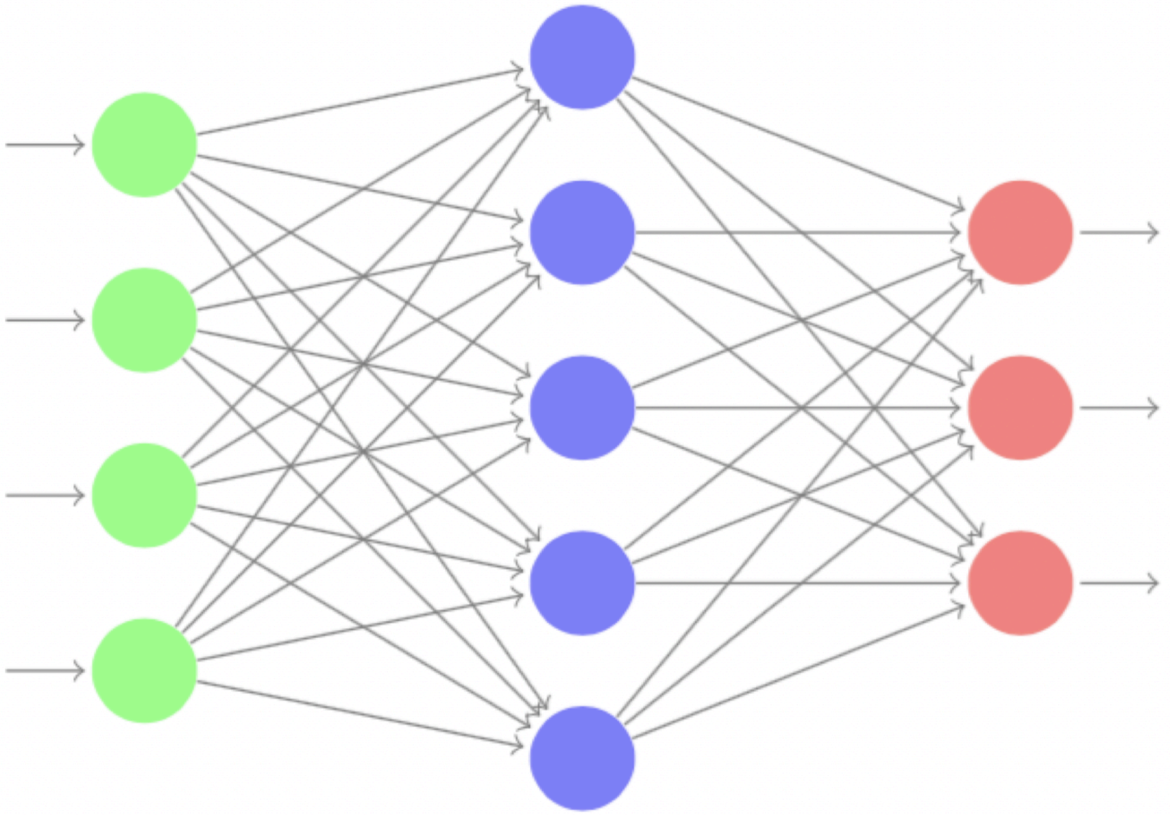
Final project



- We will get several homework assignments.
- After that, at the end of the quarter, there will be a final project.
- It is approximately equivalent to two homework assignments.

Goal of final project

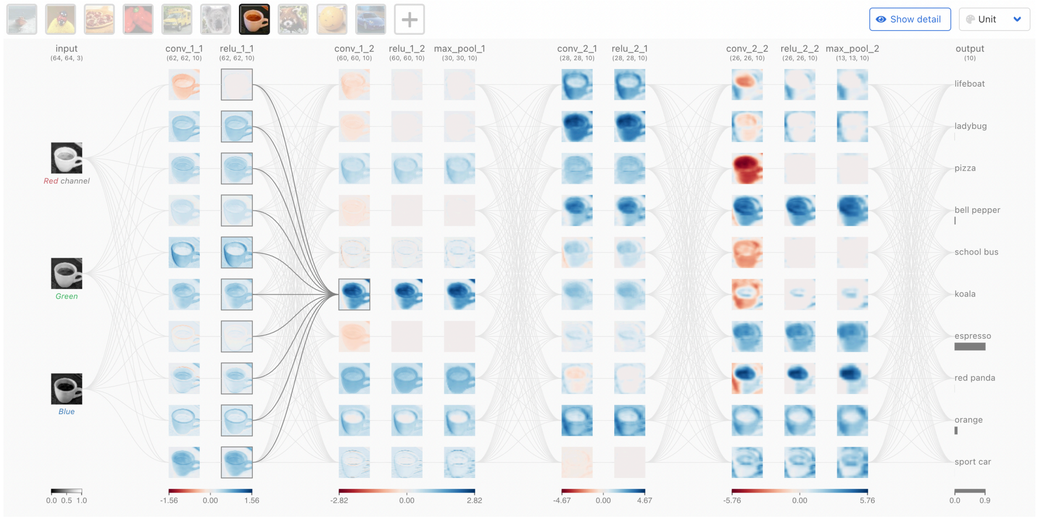
Implement a deep neural network to recognize MNIST digits.

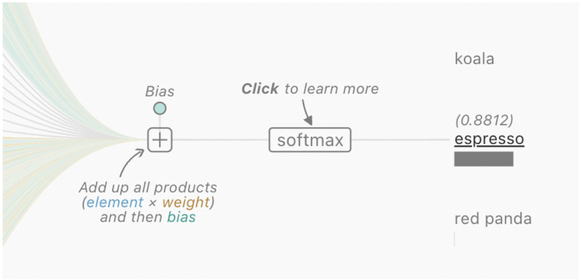


P(0)
P(1)
P(2)
P(3)
P(4)
P(5)
P(6)
P(7)
P(8)
P(9)

CNN explainer

- https://poloclub.github.io/cnn-explainer/
- See softmax layer at the end; it assigns probabilities to each class or label.
- Convolution layers will be replaced by dense layers = matrix-vector products with a dense matrix.

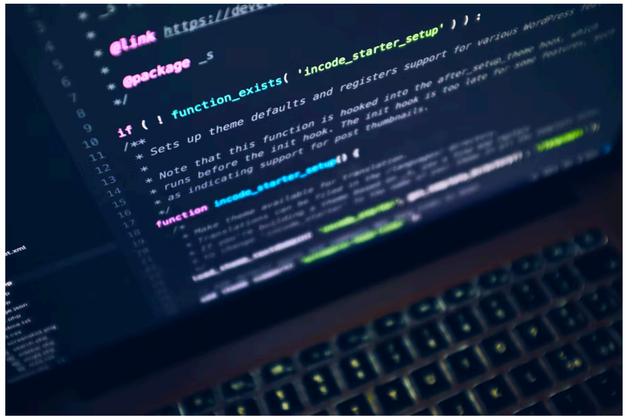




Tasks for final project

- More details will be provided later.
- Project consists primarily in:
  - Writing MPI code to be able to use 4 GPUs in parallel.
  - Writing CUDA code to implement a matrix-matrix product, and to calculate the non-linear activation functions.
  - You will need to implement the forward and back-propagation steps in the DNN (if you are familiar with these concepts already).

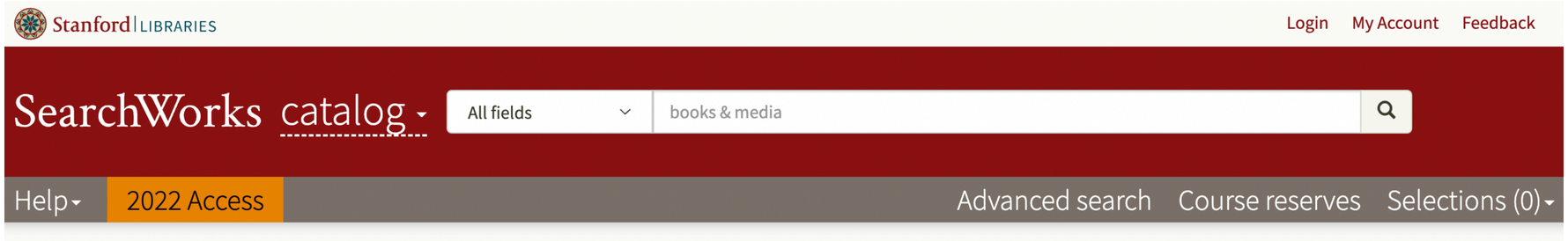
Starter code



- The final project is challenging. The scope of the work is greater than the homework assignments.
- A starter code will be provided. It contains a reference sequential CPU implementation, and all the code needed for testing and debugging.
- Benchmarking will require the ICME GPU cluster.

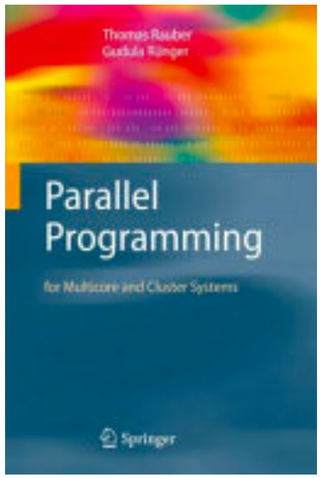
Books and reading material

- A lot of material is available online through searchworks or can be found on the internet.
- See http://searchworks.stanford.edu

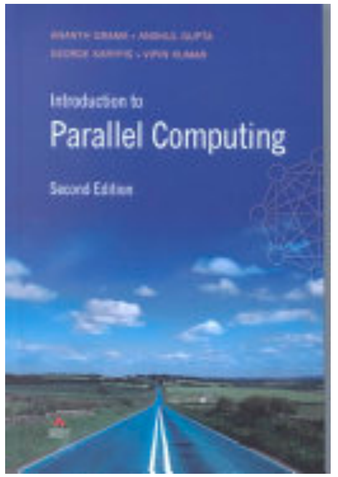


Recommended books
General parallel computing books

- Parallel programming: for multicore and cluster systems; Thomas Rauber, Gudula Rünger
- Applications focus mostly on linear algebra
- https://searchworks.stanford.edu/view/9113380



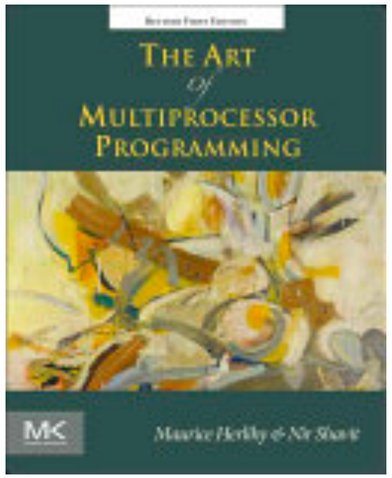
- Introduction to parallel computing
- Grama, Gupta, Karypis, Kumar
- Wide range of applications from sort to FFT, linear algebra and tree search
- https://searchworks.stanford.edu/view/5375111



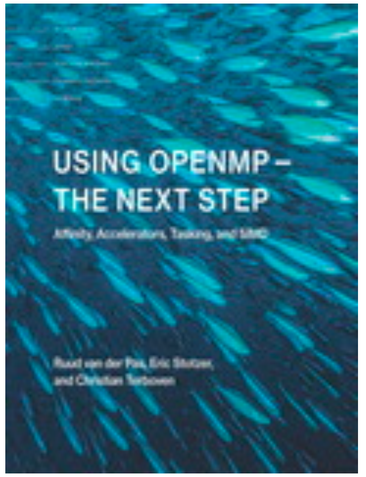
Multicore programming

- Using OpenMP: portable shared memory parallel programming
- Chapman, Jost, van der Pas
- In-depth coverage of OpenMP
- https://searchworks.stanford.edu/view/7113042

- The art of multiprocessor programming
- Maurice Herlihy, Nir Shavit
- Specializes on advanced multicore programming
- https://searchworks.stanford.edu/view/13197729



- Using OpenMP—the next step: affinity, accelerators, tasking, and SIMD
- Ruud van der Pas, Eric Stotzer, and Christian Terboven
- Covers recent extensions to OpenMP and some advanced usage
- https://searchworks.stanford.edu/view/13320198



CUDA—online documentation (preferred)

- Programming guides and API references

  http://docs.nvidia.com/cuda/index.html

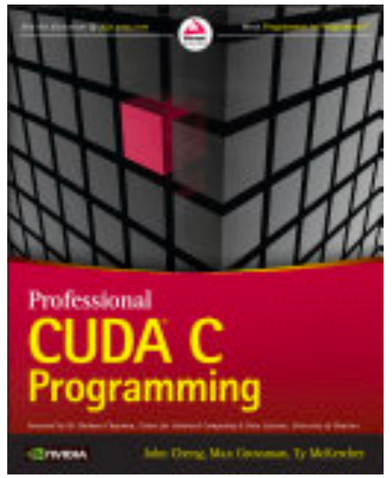
- Teaching and learning resources from NVIDIA

  https://developer.nvidia.com/cuda-education-training

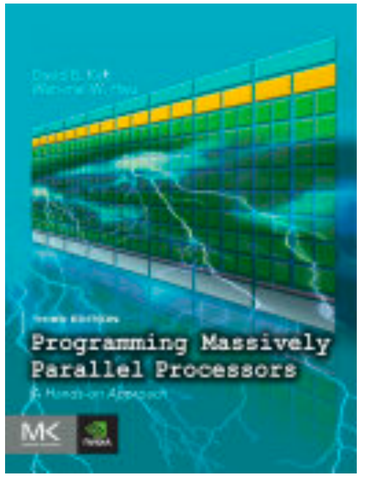
- Recommended reading:
  - CUDA_C_Best_Practices_Guide.pdf
  - CUDA_C_Programming_Guide.pdf

CUDA programming books

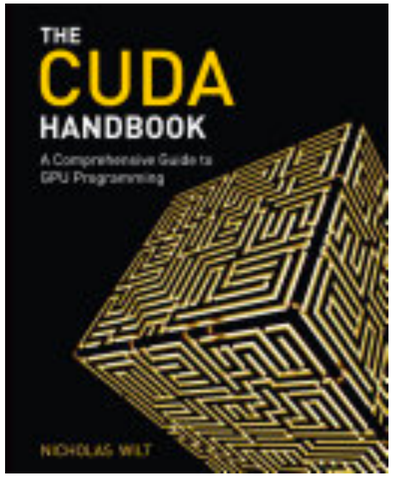
- Professional CUDA C programming
- John Cheng, Max Grossman, Ty McKercher
- Recommended for this class; has more advanced usage like multi-GPU programming
- https://searchworks.stanford.edu/view/13206511



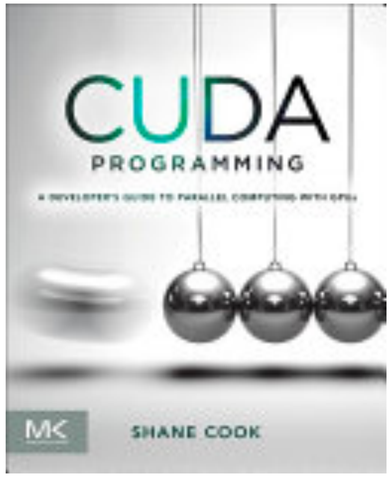
- Programming massively parallel processors: a hands-on approach
- David B. Kirk and Wen-Mei W. Hwu
- In its 3rd edition now; covers a wide range of topics including numerical linear algebra, applications, parallel programming patterns
- https://searchworks.stanford.edu/view/13246176



- The CUDA handbook: a comprehensive guide to GPU programming
- Nicholas Wilt
- Lots of advanced technical details on memory, streaming, the CUDA compiler, examples of CUDA optimizations
- https://searchworks.stanford.edu/view/10313336

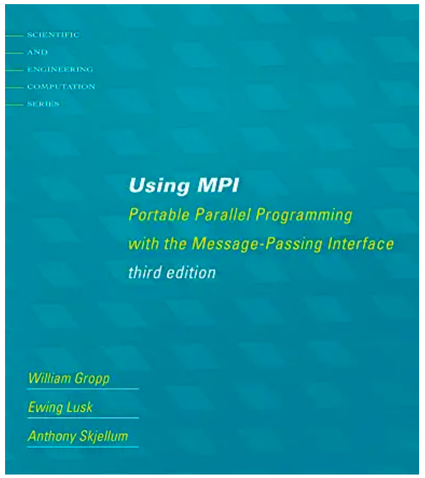


- CUDA programming: a developer's guide to parallel computing with GPUs
- Shane Cook
- Extensive CUDA optimization guide; practical tips for debugging, memory leaks
- https://searchworks.stanford.edu/view/12955187

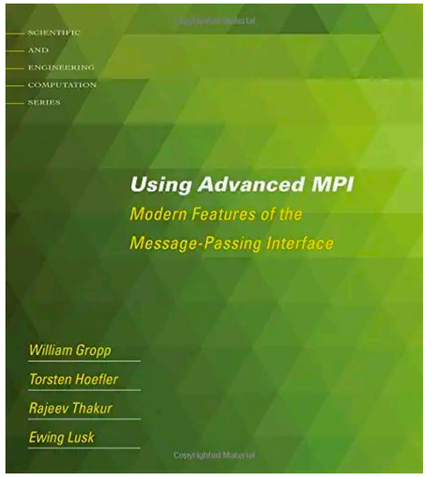


MPI books

- Using MPI: portable parallel programming with the Message-Passing-Interface
- William Gropp, Ewing Lusk, and Anthony Skjellum
- https://searchworks.stanford.edu/view/10945472



- Using Advanced MPI: Modern Features of the Message-Passing Interface
- William Gropp, Torsten Hoefler, Rajeev Thakur, and Ewing Lusk
- Same authors as previous entry; discusses recent and more advanced features of MPI
- https://searchworks.stanford.edu/view/13344892



Other related classes at Stanford

CS 149: *Parallel Computing*

This course is an introduction to parallelism and parallel programming. Most new computer architectures are parallel; programming these machines requires knowledge of the basic issues of and techniques for writing parallel software. Topics: varieties of parallelism in current hardware (e.g., fast networks, multicore, accelerators such as GPUs, vector instruction sets), importance of locality, implicit vs. explicit parallelism, shared vs. non-shared memory, synchronization mechanisms (locking, atomicity, transactions, barriers), and parallel programming models (threads, data parallel/streaming, MapReduce, Apache Spark, SPMD, message passing, SIMT, transactions, and nested parallelism). Significant parallel programming assignments will be given as homework. The course is open to students who have completed the introductory CS course sequence through 111.

EE 382A: *Parallel Processors Beyond Multicore Processing*

Formerly EE392Q. The current parallel computing research emphasizes multi-cores, but there are alternative array processors with significant potential. This hands-on course focuses on SIMD (Single-Instruction, Multiple-Data) massively parallel processors. Topics: Flynn's Taxonomy, parallel architectures, Kestrel architecture and simulator, principles of SIMD programming, parallel sorting with sorting networks, string comparison with dynamic programming (edit distance, Smith-Waterman), arbitrary-precision operations with fixed-point numbers, reductions, vector and matrix multiplication, image processing algorithms, asynchronous algorithms on SIMD ("SIMD Phase Programming Model"), Mandelbrot set, analysis of parallel performance.

What you can expect from me

- I am here to guide your learning and will challenge you to actively engage in the learning process through class activities, assignments, and more.

- I will strive for an inclusive and collaborative classroom—one that is respectful of gender, disability, age, race, and all other dimensions of diversity and identity, as well as each person's unique circumstances at this time.

- I will do my best to give you the tools, feedback, and support to succeed, and will always welcome suggestions for improvement.

- I highly encourage everyone to visit me in office hours or to set up a meeting, even if you don't feel that you have questions. I want to get to know you and support you in this learning experience! The best way to reach me is by email or through the forum; I typically respond within 24 hours (Monday–Friday).

Respect for diversity

- It is my intent that students from all diverse backgrounds, perspectives, and situations be well served by this course, that students' learning needs be addressed both in and out of class, and that the diversity that students bring to this class be viewed as a **resource, strength and benefit.**

- Please let me know ways to **improve the effectiveness** of the course for you personally or for other students or student groups.

- In addition, if any of our class meetings conflict with your **religious events,** please let me know so that we can make arrangements for you.

Mental Health

- The COVID-19 pandemic is a stressful time for us all.
- In addition, you may experience a range of other challenges that can create challenges to learning, such as strained relationships, increased anxiety, alcohol/drug problems, feeling down, difficulty concentrating and/or lack of motivation.

- If you or someone you know is feeling overwhelmed, depressed, and/or in need of support, services are available.
- Vaden web site: https://caps.stanford.edu/

Students with documented disabilities

- Stanford is committed to providing equal educational opportunities for disabled students. Disabled students are a valued and essential part of the Stanford community. **We welcome you to our class.**

- If you experience disability, please register with the **Office of Accessible Education** (OAE). Professional staff will evaluate your needs, support appropriate and reasonable accommodations, and prepare an Academic Accommodation Letter for faculty. To get started or to re-initiate services, please visit oae.stanford.edu.

Honor Code

File a concern

The Honor Code is an undertaking of the students, individually and collectively
(a) that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
(b) that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

- **In particular, you cannot copy: any material from another student, previous year solution sets, solutions found using Google, solutions generated using AI (e.g., chatGPT, Copilot, AlphaCode), or solutions found on the internet.**

- We welcome collaboration between students, but everything you submit as graded assignments should be written by yourself and **not copied from anyone else.**

Don’t forget the survey on Canvas

See Canvas for the link.

Survey

CME 213, Stanford University. This is an optional non-graded survey where you can share information about yourself and ask questions about the class. This survey is accessible by Eric Darve only.