Spring 2-1-2019

CSCI 580.01: Applied Parallel Computing Techniques

Travis J. Wheeler
University of Montana, travis.wheeler@umontana.edu

CSCI 480/580: Parallel Computing

Course information
Spring 2019
Meeting time: T/Th 9:30-10:50AM
Location: Social Science 362
Final Exam: none

Course material/submissions/grades are in Moodle (http://umonline.umt.edu)

Instructor information
Instructor: Travis Wheeler
Office: Social Science 420
E-mail: travis.wheeler@umontana.edu
Phone: 406-243-6219
Office Hours: TBD
or by appointment (see http://wheelerlab.org/schedule)

Course Overview
This course is an introduction to parallelism and parallel programming. Topics include the various forms of parallelism on modern computer hardware (e.g. multiple cores, SIMD vector instructions, GPUs, FPGAs, and networked clusters), with coverage of locality and latency, shared vs non-shared memory, and synchronization mechanisms (locking, atomicity, etc). We will address computer architecture at a high level, sufficient to understand the relative costs of operations like arithmetic and data transfer. We will discuss patterns that appear in essentially all programs that need to run fast. The emphasis will be on implementing actual parallel applications, rather than mere theory.

Course Prerequisites
CSCI 205 and 232, or instructor consent.

Regarding instructor consent: this course may prove useful for students from many departments and with different backgrounds. I will assume mature programming skills in a conventional (non-parallel) language, as well as enough mathematical skills to understand the problems and algorithmic solutions presented. Course projects will be implemented in C – I strongly recommend having familiarity with C prior to the beginning of class, for example using this online tutorial:
http://www.cprogramming.com/tutorial/c-tutorial.html.

Course Learning outcomes
- Able to recognize and analyze patterns/applications that benefit from parallelism
- Able to implement basic parallel computing in multithreaded environment (OpenMP, Pthreads)
- Able to implement basic parallel computing in GPU environment (CUDA)
- Able to implement basic SIMD computing approaches

- Able to implement basic parallel computing in a cluster environment (MPI)
- Able to analyze and measure performance of parallel computing systems, and to debug those systems
- Able to analyze the impact of latency and resource contention on throughput

Additionally, for graduate students:
- Experience presenting complex computer science (algorithm) methods and results to an audience

Required textbook

Parallel Computing for Science and Engineering, by Victor Eijkhout
https://moodle.umt.edu/pluginfile.php/1579508/mod_resource/content/1/EijkhoutParComp.pdf

Other resources

An open source textbook by Norm Matloff:
http://heather.cs.ucdavis.edu/~matloff/158/PLN/ParProcBook.pdf

OpenMP
https://www.youtube.com/playlist?list=PLLX-Q6B8xqZ8n8bwjGdzBJ25X2utwnoEG
http://openmp.org/mp-documents/Intro_To_OpenMP_Mattson.pdf
http://openmp.org/mp-documents/Mattson_OMP_exercises.zip

CUDA
https://www.udacity.com/course/intro-to-parallel-programming--cs344
https://github.com/udacity/cs344

Grading

Undergraduate	
Homework:	40%
Exams:	30%
In-class activities	30%

Graduate	
Homework:	30%
Exams:	20%
In-class activities	25%
Work/Presentations	25%

Grade cutoffs: curved, based on my opinion of the work of students at the boundary.

Approximate schedule (find updated reading and assignments in moodle)

Week	Content	Reading / Videos
1	Intro OpenMP	OpenMP videos 1-6
2	OpenMP	OpenMP videos 7-17
3	OpenMP	OpenMP videos 18-27
4	GPU	CUDA videos lesson 1
5	GPU/CUDA	CUDA videos lesson 2,3
6	GPU/CUDA	CUDA videos lesson 4
7	GPU/CUDA	CUDA videos lesson 5
8	GPU/CUDA	CUDA videos lesson 6
9	SIMD / SSE	
10	MPI	
11	MPI	
12	FPGA	
13	FPGA	
15	Presentations	

Working in groups (homework)

The best way to learn the material is by solving problems. You are encouraged to work together - the best way to understand the subtleties of the homework problems is to argue about the answers. Each of you should look at all the problems independently, and not just divide the list in two parts each time. After discussing problems and coming up with solutions, you will each write up a separate submission. Though the ideas behind your solutions may be quite similar, the text should not be identical – demonstrate your command of the problem with a personalized solution. I retain the right to question you about the material turned in. If it is evident that you don't understand what you turned in, your grade will be lowered.

(Don't be a leech and let your partner do all the work. Unless you learn how to solve problems, you will get burned on the exams and thus for your final grade.)

Working in groups (programming assignments)

I encourage discussion with others regarding programming assignments, as well. As with homework, these should be high-level discussions. Code should be written independently. If I suspect copying or plagiarism, I will ask you to explain each piece of the code to me, possibly resulting in a reduced grade or removal from class.

Late policy

Submissions for programming and homework assignments are due at the beginning of class. Late submissions will not be accepted. Every student will get one free extension on an assignment (programming or homework) for up to a week. You do not have to ask for this – just write that you are using your free extension when you turn it in. Don't waste this extension or feel obligated to use it; another extension will be given only in exceptional circumstances.

Cheating

It should go without saying that academic dishonesty (including plagiarism and cheating) will not be tolerated. Consult the university's student conduct code for more details. I will follow the guidelines given there. I will seek out the maximum allowable penalty for any academic dishonesty that occurs in this course. If you have questions about which behaviors are acceptable, please ask me.

Disabilities

Students with disabilities are encouraged to meet with me to discuss *any* accommodations they require.

Electronic devices

Turn off your cellphone, or set it to vibrate during class. Take calls outside the classroom. Students texting during class will be asked to leave.

Personal contact

I like to establish personal contact in my classes. Please feel welcome to stop by my office (during office hours or otherwise) to ask questions or say hello.