CS 484
Introduction to Parallel Programming
Fall 2018
Course Organization

Instructor: Laxmikant (Sanjay) Kale
Office Hours: Tuesday 10:00-11:00 am @ 4212 SC

TAs:
- Steven Qiou (sqiou2)
  - Office hours: Monday, Friday 3-4pm @ 0207 SC
- Bryan Lunt (bjlunt2)
  - Office hours: Tuesday 3-4pm, Thursday 1-2pm @ SC 2219

Course Description
This course is about writing effective programs to harness the unprecedented power provided by modern parallel computers so that the programs attain the highest possible levels of performance the machines are capable of. The parallel computers we focus on include multicore processors as well as clusters and supercomputers made from them. The programming systems and methodologies we learn will include OpenMP, Pthreads, MPI and Charm++. However, the focus of the course is not so much on the mechanics of these programming systems as on how to use them to attain and improve high performance. This performance orientation pervades throughout the course, and is enhanced by several case studies, small enough to understanding the lecture format yet complex enough to illustrate performance issues and trade-offs. The course also provides or covers an adequate analytical framework for understanding performance, including performance models, scalability analysis, and iso-efficiency.

Course Goals
By the end of the course, you will be able to: Write efficient parallel programs for multicore processors and distributed memory machines. Specifically:

- List architectural elements of modern processors and explain their impact on performance
- Analyze cache performance including counting cache misses and their impact on performance
- Sequential Performance Optimization: Analyzing performance, calculating arithmetic intensity of a given code, optimizing cache performance by restructuring code
- Basic OpenMP constructs: understand and explain meanings of “parallel for” and related simple constructs
- Identify if a loop is parallel and converting loops to parallel when possible, by restructuring code and use of privatize/shared/reduction annotations on variables, etc. using OpenMP; express loops using OPENMP pragmas
- Optimizing parallel performance using schedule clauses and explicit parallelism, non-loop parallelism and task parallelism
• Ability to explain and use in code, various synchronization constructs
• PThreads: basic constructs for creating threads and ensuring mutual exclusion; understanding code using them, and writing code using them
• Explain and use MPI primitives for point-to-point communication, one-sided communication, collective communication, creation and use of sub-communicators
• Analyze performance of parallel programs including scaling analysis, performance models, asymptotic performance, etc.
• Parallel Algorithms, case studies and their applications
  o Explain performance issues in a given parallel algorithm, and write expressions for performance in terms of machine and problem parameters.
  o Ability to compare alternative algorithmic formulations for the same problem using such analysis
  o Covered algorithms/problems will include matrix multiplication, sorting, stencil codes, etc.
• Articulate and quantify performance issues in using modern interconnection topologies, and fault tolerance

Classroom experience and lecture videos
We will be using a flipped classroom format this semester. Recordings of each week’s material will be made available at the beginning of the week (typically, on Sunday). The Tuesday classroom slot is kept unused. You can watch the recordings for that week during that slot or from your home. The Thursday class will include a short quiz on that weeks material, but it will be mostly graded for participation (although there may be some grade based on understanding of the material so please do prepare for it). In addition, we will have a discussion of your questions about the material and possibly some examples that will be worked out by course staff and possibly students during this class.

The assessments will consists of:
1. Weekly online quizzes
2. Short in-class quizzes, every Thursday
3. Homework
4. Machine Problems
5. Project (grad students only)
6. Exams

Grading (tentative)
Your final grade will be calculated based on the activities listed in the table below. Your official final course grade will be listed in Enterprise.

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graded weekly quizzes (online) - lowest 1 will be dropped</td>
<td>14%</td>
</tr>
<tr>
<td>Short in-class quizzes - lowest 3 will be dropped</td>
<td>3%</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Uploaded homework assignments</td>
<td>14%</td>
</tr>
<tr>
<td>Machine Problems</td>
<td>14%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>15%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Project (Grad students only)</td>
<td>15%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

For undergrad students, your grade will be calculated with a 0 grade for the project component, and will subsequently be scaled to 100% (i.e. getting 85/85 will result in 100%).

**Grading Scale:**
Grading will be “on a curve”, and on separate curves for grads (all must be 4 hours) and undergrads. What that means, is: students in each group (grads, undergrads) will be ranked by the weighted grade, with the weights defined as above. Then, we will use a relatively stable historic data to decide how many students are to get A, B, C grades (The D and below grades are exceptions). I believe that for a large class like this, it is a better way of assigning grades, which is less affected by relative difficulty of (say) exam questions from semester to semester. Grad cut-offs will be stricter (in weighted-grade terms) than undergrad cutoffs.

**Late Work Policy:** 1 day late submission will be accepted at the cost of 10% of the assignment's grade.

**Collaboration:** No collaboration is allowed on homework assignments or machine problems. General topics relating to the course may be discussed, but you should not work together on homework or machine problems.

**Student Code and Policies**
A student at the University of Illinois at the Urbana-Champaign campus is a member of a University community of which all members have at least the rights and responsibilities common to all citizens, free from institutional censorship; affiliation with the University as a student does not diminish the rights or responsibilities held by a student or any other community member as a citizen of larger communities of the state, the nation, and the world. See the [University of Illinois Student Code](#) for more information.

**Academic Integrity**
All students are expected to abide by [the campus regulations on academic integrity found in the Student Code of Conduct](#). These standards will be enforced and infractions of these rules will not be tolerated in this course. Sharing, copying, or providing any part of a homework solution or code is an infraction of the University's rules on academic integrity. We will be actively looking for violations of this policy in homework and project submissions. Any violation will be punished as severely as possible with sanctions and penalties typically ranging from a failing grade on this assignment up to a failing grade in the course, including a letter of the offending infraction kept in the student’s permanent university record.
Again, a good rule of thumb: *Keep every typed word and piece of code your own.* If you think you are operating in a gray area, you probably are. If you would like clarification on specifics, please contact the course staff.

**Disability Accommodations**

Students with learning, physical, or other disabilities requiring assistance should contact the instructor as soon as possible. If you’re unsure if this applies to you or think it may, please contact the instructor and [Disability Resources and Educational Services (DRES)](http://www.dres.uiuc.edu) as soon as possible. You can contact DRES at 1207 S. Oak Street, Champaign, via phone at (217) 333-1970, or via email at disability@illinois.edu.