Course Information

This handout explains how the course is organized and administered. It describes how you will be graded and what the course staff expects of you.

1 Immediate action items

- By 8 p.m. on Thursday, September 4, do the first part of Homework 1 to get a CSAIL account and fill out an information form.
- Attend any of the three recitations on Friday, September 5.

2 General information

Modern computing platforms provide unprecedented amounts of raw computational power. But significant complexity comes along with this power, to the point that making useful computations exploit even a fraction of the computing platform’s potential a substantial challenge. Indeed, obtaining good performance requires a comprehensive understanding of all layers of the underlying platform, deep insight into the computation at hand, and the ingenuity and creativity required to obtain an effective mapping of the computation onto the machine. The reward for mastering these sophisticated and challenging topics is the ability to make computations that can process large amount of data orders of magnitude more quickly and efficiently and to obtain results that are unavailable with standard practice.

This is an 18-unit class that provides a hands-on, project-based introduction to building scalable and high-performance software systems. Topics include performance analysis, algorithmic techniques for high performance, instruction-level optimizations, caching optimizations, parallel programming, and building scalable systems.

The prerequisites for this class are 6.004, 6.005, and 6.006. If you do not satisfy all of these prerequisites, you should not be taking this class. The course programming language is C. We will be offering a C-language tutorial on Monday, September 8, at 7 p.m., location to be announced.
3 Staff

<table>
<thead>
<tr>
<th>Position</th>
<th>Name</th>
<th>Contact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lecturer</td>
<td>Prof. Saman P. Amarasinghe</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>Lecturer</td>
<td>Prof. Charles E. Leiserson</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>Lecturer</td>
<td>Dr. Aparna Chandramowlishwaran</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>TA</td>
<td>William M. Leiserson</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>TA</td>
<td>Tao Benjamin Schardl</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>TA</td>
<td>Cong Yan</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>UTA</td>
<td>Damon A. Doucet</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>UTA</td>
<td>Michael J. Xu</td>
<td>Via Piazza</td>
</tr>
<tr>
<td>Support Staff</td>
<td>Cree Bruins</td>
<td><a href="mailto:cbruins@csail.mit.edu">cbruins@csail.mit.edu</a></td>
</tr>
<tr>
<td>Support Staff</td>
<td>Mary McDavitt</td>
<td><a href="mailto:mmcdavitt@mit.edu">mmcdavitt@mit.edu</a></td>
</tr>
</tbody>
</table>

4 Web

<table>
<thead>
<tr>
<th>Information</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td><a href="https://stellar.mit.edu/S/course/6/fa14/6.172/">https://stellar.mit.edu/S/course/6/fa14/6.172/</a></td>
</tr>
<tr>
<td>Course calendar</td>
<td><a href="http://goo.gl/3kxZ25">http://goo.gl/3kxZ25</a></td>
</tr>
<tr>
<td>Forum</td>
<td><a href="http://www.piazza.com">http://www.piazza.com</a></td>
</tr>
<tr>
<td>Piazza sign-up</td>
<td><a href="https://piazza.com/mit/fall2014/6172">https://piazza.com/mit/fall2014/6172</a></td>
</tr>
</tbody>
</table>

We will be using the Stellar course-management system. Please make sure that you are registered with Stellar as a member of 6.172 for the Fall 2014 semester. We will use Stellar to post assignments and accept written submissions.

The Piazza website provides a wiki-like service for organizing questions and answers regarding course content. The course staff will use Piazza, rather than email, for communications with students. Piazza allows students to submit questions to the course staff, which other students can see. Questions can be submitted with the author identified, but anonymous questions can also be submitted. The course staff can respond to questions and provide answers, but students can also respond and provide answers. You should have received an email inviting you to register on the 6.172 Piazza website. If not, you may invite yourself by using the Piazza sign-up link above. All questions about course content and administration should be posed via Piazza, not via email. If you are shy, please post your question to Piazza anonymously.

To communicate directly and privately with course staff, please write a question or note on Piazza and send it to “Instructors” only. Even if you want to communicate with a specific member of the course staff, please post to “Instructors,” but put a greeting in the body to the staff member with whom you want to communicate. Your question or note will not be viewable by the rest of the class unless you later choose to make it public. Please do not use email unless absolutely necessary, as that will delay a response, perhaps by several days.
5 Lectures

Lectures will be held on Tuesdays and Thursdays from 2:30 p.m. to 4:00 p.m. in 34-101. Please plan to attend regularly. All material covered in the lectures will be fair game for homeworks, projects, and quizzes. We will post lecture slides and notes on Stellar, but they often will not contain all of the information presented in a particular lecture and should not be considered a substitute for attendance.

6 Recitations

We will hold regularly scheduled recitations in 4-265 on Fridays. There are three sections: 10 a.m. to noon, 1 p.m. to 3 p.m., and 3 p.m. to 5 p.m. These two-hour recitations are designed to be hands-on tutorials covering tools and other practical topics. Recitations are mandatory. Missing more than one without an approved excuse will result in failing the course. If you must miss a recitation, notify your TA as soon as possible. You will still need to complete the recitation assignment and have a TA check it off.

Recitations will be assigned based on your preferences (see Homework 1). For the first recitation on Friday, September 5, please attend any section.

7 Office hours

Office hours will generally be held Sunday through Wednesday from 4:00 p.m. to 7:00 p.m. in a location to be announced. The course calendar will provide details when holidays and other events interfere with this schedule.

8 Grading

Each assignment will describe how you will be graded. The scores you receive on each assignment will be combined to produce your final grade after being weighted approximately as follows:

<table>
<thead>
<tr>
<th>Assignment</th>
<th>#</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekly homeworks</td>
<td>12</td>
<td>12%</td>
</tr>
<tr>
<td>Quizzes</td>
<td>2</td>
<td>30%</td>
</tr>
<tr>
<td>Projects 1–3</td>
<td>3</td>
<td>36%</td>
</tr>
<tr>
<td>Project 4</td>
<td>1</td>
<td>22%</td>
</tr>
</tbody>
</table>

If you receive insubstantial credit on any project, then you risk failing the class. Missing more than one recitation checkoff or more than one homework assignment may also result in a failing grade.
9 Late policy

Since this class has many tight deadlines and code will be released immediately after the deadline, there is no room for late submissions. You should submit whatever you have by the deadline, and we will award partial credit as appropriate. Specifically, no late work will be accepted without a note from Student Support Services. Please speak with your TA immediately if you believe you will have trouble completing an assignment on time. If you do receive insubstantial credit for an assignment, or if you have not completed a recitation checkoff, please talk with your TA immediately.

10 Homeworks

There will be 12 homework assignments throughout the semester which will help you prepare for your projects and evaluate your understanding of the course material.

11 Quizzes

There will be two in-class quizzes on Tuesday, October 9, and Thursday, November 20. The quizzes will be closed book and closed notes, but you will be permitted a crib sheet. All material covered by the lectures, projects, homeworks, and prerequisite courses, as well as any other material indicated by course staff, is fair game for the quizzes. More details will be forthcoming as the quizzes approaches.

There will be no final exam.

12 Projects

The bulk of your out-of-class time will be spent completing four projects of increasing scope and complexity. Each project involves a beta submission (two betas for Project 4), design and code review, and a final submission. The general structure of each assignment is described below.

*Project 1*

You will do the first project on your own for the beta submission, but pair up with another student for the final submission. Your partner must belong to the same recitation as you. The project will allow you to learn how to improve performance by using a performance monitoring tool and to experiment with word-level parallelism.

*Project 2*

The second project will be completed in pairs for both the beta and the final submissions, but you will not be allowed to pair with the same person you did in Project 1. As in Project 1, your partner must belong to the same recitation as you. You will begin by optimizing a single-threaded
physical simulation. Then, you will be introduced to Cilk Plus, a language, compiler, library, and tool chain for developing multithreaded applications, and you will parallelize your project. This project will expose you to many of the issues associated with correctness and performance in a multithreaded application.

**Project 3**

The third project will also be completed in pairs, but you will not be allowed to pair with either of the people you paired with in Projects 1 and 2. As before, your partner must be in the same recitation as you. Project 3 will require you to examine the complex real-world problem of high-performance memory management. You will implement a serial library which provides the `malloc`, `free`, and `realloc` functions (that is, the C memory-management API) as efficiently as possible for a number of different plausible workloads. You will also use auto-tuning techniques to find configurations with the highest performance.

**Project 4**

The fourth and final project will be completed in teams of four students, but for this project, your team *can* include former partners from Projects 1–3, and they need not be from your same recitation. Unlike the previous projects, Project 4 involves two beta submissions, instead of just one, as well as a final submission. You will work to optimize a game engine, which plays a two-player board game. Unlike for the previous projects, we will provide relatively little direction. You will need to use everything you have learned during the course to identify performance bottlenecks and eliminate them. Higher performance programs should play better. Although we will grade your performance on an absolute scale independently from how others’ programs in the class perform, we will run exhibition tournaments among the programs for fun.

**13 MITPOSSE**

The course staff has recruited senior software engineers from industry to share with you their invaluable experience and give you concrete advice on your design and code. These Masters in the Practice of Software Systems Engineering (MITPOSSE) will review your designs and code. They will not have any involvement in the grading process, but your participation in code and design reviews will be evaluated by the course staff to determine the MITPOSSE portion of your project grade. Furthermore, your Master’s expertise can help you produce projects with a higher grade.

Your assigned Master has agreed to volunteer a significant amount of their time to help you learn. We’ve hand-picked these volunteers from among the very best software engineers. They know what they are talking about. All have extensive experience with real-world, on-the-job design reviews.

Please accord these Masters your greatest respect, since they are volunteering their time. You can learn a lot from them. Be punctual. Be prepared. Be gracious. Thank them.
14 Project structure

Each project will consist of a beta submission, code and design review, and a final submission. You must also submit a write-up which describes your implementation for each beta and final submission. The exact grading scheme for the projects will vary. The handout for each project will include a section describing exactly how your implementation will be evaluated.

The teams with the fastest correct implementations for the beta and final submissions will receive bonus points, as well as the adoration of their peers. After all, being fast is cool!

Beta submission

For the beta, you should focus on creating a correct implementation and developing a good set of tests. You should also try to make it fast! We will generally give more weight to the performance of your final submission, however.

After the beta submission deadline, all submissions will be anonymized and posted online so that you can learn from what others have done. Please do not put personal details that could be used to identify your submission, such as names of team members, into your code or comments. In addition, beta results will be posted, so that you can see how your submission compared to the rest of the class. (You are not competing against each other, but it is helpful to see how others have done.) Please look at others’ submissions to learn from them. You are allowed to borrow ideas, but you may not steal code outright for your final submission. The work you submit must be your own, but you may seek inspiration from your classmates’ submissions after they are posted. A good strategy to avoid copying code inadvertently is to leave a significant interval of time between when you look at someone else’s code and when you write your own version.

Code and design reviews (MITPOSSE)

After the beta submission, your Master will comment on your code using the Caesar code-review tool. You will coordinate with your Master to schedule a face-to-face design review. Please ensure that you read and reply to your Master’s code-review comments before your design review. Although your Master will not be grading you, your level of engagement with the code-review process will be evaluated by course staff to determine the MITPOSSE portion of your project grade. This portion of the grade will be assigned individually to each member of the team. If you fail to attend design reviews with your Master, you risk failing the class.

Final submission

The final submission will generally be due a week or more after the beta. We will announce a performance goal for the final project shortly after the beta submission deadline. The performance goal will be set to be challenging, but achievable with effort. Projects which do not meet or exceed the performance goal will receive partial credit based on how close they get to the performance goal. We also will expect that you have incorporated feedback from your Master and have improved your implementation’s code quality and documentation. Your final submission will be anonymized and posted online.
Write-ups

A couple of days after each beta or final submission, you must submit a write-up describing your code. Describe the structure of your code, how you diagnosed the bottlenecks, the optimizations you implemented, and what kinds of speedup the various optimizations produced. Feel free also to describe things that didn't work and what you learned from the experiments.

Team dynamics

With the exception of the code-review part of the project grade, team members will generally receive the same grade on their joint project. You are required to make a substantial contribution to each project. If a team member does not make a substantial contribution to the project, the course staff will adjust the student’s individual grade accordingly.

To help ensure that all team members contribute, we will be reviewing the git commit logs to assess the dynamics of your team. You should ensure that commits are well balanced among your team members’ user names. We understand that with pair programming, commits from one team member may represent work by others, but it is up to the pair to ensure that commits are balanced.

If you feel that a team member is not pulling his or her weight, please contact the course staff as soon as possible.

15 Academic honesty

Institute guidelines relating to academic honesty require that we inform you of our expectations regarding permissible academic conduct. It is your responsibility to satisfy both the letter and the spirit of these rules. If any part of this policy is unclear, or if you have any questions or concerns, please ask a member of the course staff for clarification.

If you violate this policy, you will be referred to the Committee on Discipline to face the possibility of expulsion from MIT and other punitive actions. We take academic dishonesty extremely seriously. Please do not put us in a position where we must deal with it.

The course staff will use technological and other means to detect cheating. If one party shares material with another, we treat both the giver and receiver as equally guilty of academic dishonesty.

You may not share ideas, algorithms, approaches to solutions, or answers to written questions with anyone who is not a member of your group, whether or not they are in the class.

When working in a group, you may (of course) share ideas, code, and anything else that may be appropriate within the group, but be sure that you are making a fair contribution to your group. We ask you to briefly describe the contributions of each group member in the written material you submit with each project. If you have not made a fair contribution to that work, putting your name on your group’s work is considered academically dishonest.

You may not permit anyone besides the staff, the members of your group, and your Master (see Section 13) to view your source code, your compiled binaries, or your written documentation until the course staff posts them, typically shortly after the submission deadline. You may not
view anyone else’s solutions or materials except those that the course staff posts. You may not copy or transcribe a solution from any source. The work you submit must be your own.

After the beta submissions are published, you are free to review them and be inspired by your classmates’ approaches to the project. You may not directly copy code, however. A good way to ensure that you do not copy is to leave a period of time between when you look at someone else’s code and when you write your own code.

You may use general conceptual material, such as what you might obtain from a textbook, regardless of its source. For instance, Introduction to Algorithms by Cormen, Leiserson, Rivest, and Stein is an excellent resource for looking up algorithms. If you do use any material from an external source, please cite it briefly and clearly in your documentation.

You may share tools, compiler settings, utility code, relevant websites, and other generally useful information via Piazza. Doing so contributes to cooperative learning. (Remember, you are not competing with one another for grades.) You may not choose with whom you share, however. If you choose to share, it must be with the entire class. Also, do not share solution ideas or code specific to the project. Students who contribute meaningfully to the class’s knowledge pool will receive extra credit towards their final grade.

In summary, make sure that you are turning in your own work! If you have any questions or concerns, talk to the course staff. If you feel that you may have violated this policy, it will go far better for you if you report your possible transgression to us than if we find out by other means.

16 If you need help …

Please get started on your assignments early, since programming assignments often take longer than you expect, even if you take account of the fact that programming assignments often take longer than you expect. If you do need help, feel free to communicate to the course staff via a private Piazza post. Earlier is better, since if you wait until the night before the assignment is due, you may not receive help in time for it to be useful, but first, please make a reasonable attempt to resolve the issue on your own.

![Speed is fun!](image)

Enjoy the class!