6.895 Course Description

Basic information:
Time: TR 9:30-11:00AM
Room: 36-155
Prof: Nancy Lynch, NE43-365, 3-7225, lynch@theory.lcs.mit.edu
Course assistant: Joanne Talbot Hanley, NE43-366, 3-6054

Enrollment and prereqisites:
The course is intended for graduate students interested in doing research in the area of distributed algorithms. It assumes knowledge of the material in 6.852, the basic MIT distributed algorithms course. Formally, this means that you've either taken 6.852 or gotten special permission from Prof. Lynch to take 6.895 without 6.852. Students should have a strong background in mathematics, and should be familiar with basic algorithms (e.g., 6.046) and computer systems (e.g., 6.033).

Course goals and summary:
This course will explore the recent theoretical literature on distributed algorithms, emphasizing algorithms for highly "dynamic" settings such as the Internet or mobile computing systems. In such environments, processes and channels may fail and recover, and may exhibit variations in their timing behavior. Participants in the algorithms may join and leave the system (as well as fail) during the course of computation. Algorithms must adapt to such changes, for example, reconfiguring on-the-fly.

The traditional theory of distributed algorithms deals mostly with fairly "static" settings, e.g., fixed networks or shared memory systems with a known set of participants. This theory must be "stretched" to cover the newer, more dynamic settings. New algorithms are needed to cope with the new possibilities. New lower bounds and other impossibility results will arise, based on new uncertainties. All of this work will require new models, and new metrics for evaluating performance and fault-tolerance.

The papers to be covered in the course represent the current state-of-the-art in the development of such a theory. Clearly, much more remains to be done. When we read the papers, we will consider what role their ideas can play in this theory, and what new work they suggest. This should help interested students to find particular questions that they can pursue for independent research.

Problems of special interest will include: (1) Communication, including various kinds of multicast (e.g., causal, totally-ordered, atomic), gossiping protocols, and view-oriented group communication. (2) Distributed data management, including weak and strong consistency guarantees, replicated data management, and quorum-based computing. Other problems we will consider will include failure detection, reaching consensus, resource allocation, performing distributed work, clock synchronization, and namespace management.

Since this will be a theory course, we will focus on algorithms for which it is possible to make definite, precise claims about what they do. These claims will include correctness, performance, and fault-tolerance results. We will also consider impossibility results, especially those that express inherent limitations on what can be accomplished in dynamic settings. Some not-so-theoretical papers with interesting systems ideas will also be covered; for these, we will discuss possibilities for future related theoretical work.

More details: The course will be based on a long list (see handout 2) of readings, including readings about algorithms in a tightly-coupled shared-memory setting, and algorithms for less-tightly-coupled client-server settings, network settings, and mobile computing settings. In each setting, we will consider cases where the set of participants is known and where it is unknown. Also, we will consider a variety of timing and failure assumptions. Algorithms and impossibility results may be sensitive to any or all of these different model parameters and assumptions. In the course of studying these many results, we will attempt to sort out which results apply in which settings.

The order in which we will cover the papers is based generally on the types of models that are discussed in the papers, ranging from the best-behaved, most predictable, most tightly coupled models to the worst-behaved, least predictable, least tightly coupled models:

1. Shared memory, with a known set or an unknown set of participants.
2. Client-server computing, known or unknown participants.
3. Networks, known participants.
4. Networks, unknown participants.
5. Mobile computing.

Each day, we will discuss a particular topic, based on reading several papers. In general, we will try to select about two papers per class as "required reading", and use the rest for background. All students are expected to read at least the required papers ahead of time. Someone will lead the discussion each day: about half the time, this will be the instructor, and the other half, the students in the class. Whoever leads the class discussion will presumably want to read the background papers as well as the required papers. We will also bring in some outside experts to participate in (or lead) some of the class meetings.

In class, we will not cover every detail of the papers. Rather, we will focus on answering questions such as:

- What is the technical contribution of the paper? What are the key algorithmic ideas? Why does the algorithm work? What is its performance? How does the impossibility argument go?
- What is the significance of this work? What models/assumptions does the algorithm or impossibility result apply to? Is this significant in practice? What does this have to do with dynamic distributed settings?
- What new work (relevant to dynamic distributed computing settings) does this suggest? What would make the results more practical?

We expect that everyone in the class will participate actively in the class discussions.

What students will actually do (for a grade):
This course will have no examinations or formal problem sets. Students will be expected to:

1. Read the papers assigned for each class meeting. This will be quite a few papers, though not as many as actually appear on the very long reading list. We will designate which of these papers are "required reading" for each class as we go along.

2. Participate in class discussions about those papers. Help everyone to understand the technical results, their significance, and any future work they might suggest.

3. "Take charge" of at least one of the topics and lead the discussion for at least one class meeting. Produce some kind of draft notes for that class.

4. Help produce a set of scribe notes for the course, by producing the final versions of the notes for some of the class meetings.

Each student's grade will be based mainly on effort and participation, and in particular, on how well he/she helps others in the course to learn the material.

Collaboration policy: Anyone may collaborate with anyone on anything; in fact, collaboration is strongly encouraged.