CMSC 23310/33310
Advanced Distributed Systems

Last updated: April 2, 2012

Department of Computer Science
University of Chicago

Spring 2012 Quarter

Dates: March 27 – May 29, 2011
Lectures: Thursdays 10:30-11:50 in Ryerson 276
Discussion: Tuesdays 10:30-11:50 in Cobb 304
Website: `http://www.classes.cs.uchicago.edu/archive/2012/spring/23310-1/`

Lecturer: Borja Sotomayor
E-mail: borja@cs.uchicago.edu
Office: Searle 209-A
Office hours: Open door policy (see page 8)

Contents of this Document

Course Description and Learning Goals

In recent years, large distributed systems have taken a prominent role not just in scientific inquiry, but also in our daily lives. When we perform a search on Google, stream content from Netflix, place an order on Amazon, or catch up on the latest comings-and-goings on Facebook, our seemingly minute requests are processed by complex systems that sometimes include hundreds of thousands of computers, connected by both local and wide area networks.

Recent papers in the field of Distributed Systems have described several solutions (such as MapReduce, BigTable, Dynamo, Cassandra, etc.) for managing large-scale data and computation. However, building and using these systems poses a number of more fundamental challenges: How do we keep the system operating correctly even when individual machines fail? How do we ensure that all the machines have a consistent view of the system's state? (and how do we ensure this in the presence of failures?) How can we determine the order of events in a system where we can't assume a single global clock?

Many of these fundamental problems were identified and solved over the course of several decades, starting in the 1970's. In this course, we will engage in reading and discussing seminal work in Distributed Systems from the last 35 years to (1) identify the fundamental issues raised in this earlier work, (2) relate those issues to current research problems, and (3) evaluate and compare the solutions proposed in both early and recent work. During this course, students will also implement a distributed system that requires them to manage distribute resources and evaluate whether the resulting system has certain properties, such as reliability, scalability, etc.

At the end of the quarter, students will be able to:

1. Identify the research questions posed in a scholarly paper and the solutions proposed in that paper.

2. Identify the main contributions and conclusions in a scholarly paper, and determine whether they are well supported by evaluating and criticizing the arguments, proofs, or experimental results in that paper.

3. Compare and contrast different distributed systems for managing large-scale data and computation.

4. Evaluate whether an implementation of a distributed system is reliable, fault-tolerant, scalable, and/or highly available.

Students interested in taking this course should take into account that CMSC 23300 (Networks and Distributed Systems) is a prerequisite for this course. Students can petition to have this requirement waived, as long as they have taken at least one other 200-level CS systems course.

Course Organization

This course is divided into two components:

Reading and Discussion of Primary Sources: One or two papers will be assigned each week, to be discussed on Tuesdays (on Thursdays, we will present background material that will aid in understanding the paper/s assigned for the following Tuesday). Students will have to submit brief response papers each week, as well as a final paper.

Distributed Systems Programming Project: Throughout the quarter, students will have to implement a distributed system. Unlike projects that students may have done in previous 200-level CS systems courses, this project is not designed to directly complement all the material covered in the lectures. Instead, the project will be open-ended: students must propose a project themselves (subject to instructor approval), choose the languages and frameworks to develop the project, and meet certain reporting milestones during the quarter. Developing this project will expose students to the practical challenges of implementing a distributed system, and will prepare them to evaluate, compare, and contrast implementations of distributed systems in the field.

These two components are described in more detail below. The course calendar, including the reading assignments for each week and the project deadlines, is shown in Table 1.

Paper discussion

Every week, we will discuss 1–2 papers in class. The actual discussion section will take place on Tuesdays, while Thursdays will be used to wrap up the discussion from Tuesday, and to present background material necessary for the following week's paper/s. Figure 1 summarizes the timeline surrounding the discussion of a paper or papers in a given week.

At the beginning of the quarter, students will be divided into three groups: A, B, and C. Although the composition of the groups will remain fixed throughout the quarter, the *role* that each group will take during a discussion section will rotate every week. There are three roles:

The Questioners: Before a discussion section, this group is responsible for preparing a list of 4–5 discussion questions about the paper to be discussed in class.

The Answerers: During a discussion section, this group is responsible for leading a discussion conducent to answering the questions posed by The Questioners. To allow enough time for The Answerers to prepare, The Questioners must submit their questions by Sunday night.

The Critics: For a given discussion, this group must criticize both the questions posed by The Questioners and the answers provided by The Answerers. Although they can begin to do this during the discussion section, time will also be allocated on Thursday for The Critics to speak (to give them enough time to process the results of the Tuesday discussion).

In each discussion, each group will have a designated spokesperson, which will also rotate every week. The spokesperson is not only the voice of his/her group, but also responsible for engaging

Thursday (in class)	Presentation of background material.
Sunday, 6pm	The Questioners must send their discussion questions to The Answerers
Monday, 6pm	Individual response papers due.
Tuesday (in class)	**Discussion session**
Thursday (in class)	The Critics criticize the questions and answers from Tuesday.

Figure 1: Important dates before and after the discussion of a paper in a Tuesday class

his/her group in discussion of questions, answers, or critiques (depending on the role of group). Every student in the class will be a spokesperson at least once.

All students will submit a brief (300-500 word) response paper for each week's paper by Monday at 6pm. The exact topic of the reponse paper will vary for each paper but, in general, will require students to identify the strengths and weaknesses of the paper.

Students must also submit a final paper (˜2,000 words) by Tuesday, June 5th at 8pm (Graduating students must submit their final paper by Tuesday, May 29th at 8pm). The topic of the paper will be set no later than 5th week, and will involve reading at least two papers that propose solutions to similar research problems, which the students must be able to explain, compare, and contrast.

Project

Throughout the quarter, students must implement a distributed system. This will be an open-ended project: students will be allowed to propose their own project (subject to instructor approval), use any language and platform, and can work individually or in pairs (groups of three will be allowed only with the instructor's consent). However, the project must involve writing code that will either run on multiple machines, or will affect the behaviour of multiple machines. To give students an idea of what an acceptable project would look like, we will be presenting a list of possible projects that can be done in collaboration with Distributed Systems researchers at the Computation Institute.

The project will have four milestones:

- By the end of 2nd Week, must meet with the instructor to discuss a project proposal. After the meeting, students must submit a half to full page summary of the project to be posted on the course wiki. This summary must include a description of the main deliverables the student/s expect to produce by the end of the quarter.
- Two progress reports, including a meeting with the instructor, must take place during fourth week and seventh week.

- In the time allocated by the Registrar for this course's final exam, students must present their projects to the entire class. A separate date will be set for graduating students.

Deadlines

Deadlines in this course will be lax, as long as delays in submissions are (1) sporadic and, (2) in the order of minutes or, at most, 1–2 hours. The instructor will contact students who are considered to be abusing this policy.

Papers

We will be discussing the following papers in this course:

Getting Started

[Lam85] Leslie Lamport. Solved problems, unsolved problems and non-problems in concurrency. *SIGOPS Oper. Syst. Rev.*, 19(4):34–44, October 1985

[Dij74] Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, November 1974

Fault Tolerance

[PSL80] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, April 1980

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982

[CL02] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.*, 20(4):398–461, November 2002

Distributed Consensus

[FLP85] Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, April 1985

[Lam98] Leslie Lamport. The part-time parliament. *ACM Trans. Comput. Syst.*, 16(2):133–169, May 1998

[Lam01] Leslie Lamport. Paxos made simple. *ACM SIGACT News*, 32(4):18–25, December 2001

Distributed Time

[Lam78] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, July 1978

[Fid88] C. J. Fidge. Timestamps in message-passing systems that preserve the partial ordering. *Proceedings of the 11th Australian Computer Science Conference*, 10(1):5666, 1988

Table 1: CMSC 23310/33310 Spring 2012 Calendar

Week	Date	Lecture	Papers	Project
1	Tu 27 March	Introduction		
	Th 29 March	Fault Tolerance	[Lam85] and [Dij74]	
2	Tu 3 April	Discussion	[LSP82]	Project Proposal
	Th 5 April	Distributed Time I		
3	Tu 10 April	Discussion	[Lam78]	
	Th 5 April	Distributed Consensus I		
4	Tu 17 April	Discussion	[FLP85]	1st Progress Report
	Th 19 April	Distributed Consensus II		
5	Tu 24 April	Discussion	[Lam98] and [Lam01]	
	Th 26 April	Distributed Time II		
6	Tu 1 May	Discussion	[Fid88] and [Mat89]	
	Th 3 May	Limits of Distributed Systems		
7	Tu 8 May	Discussion	[Lyn89] and [GL02]	2nd Progress Report
	Th 10 May	Recent Work: Distributed Locking		
8	Tu 15 May	Discussion	[Bur06] and [CGR07]	
	Th 17 May	Recent Work: Distributed Data		
9	Tu 22 May	Discussion	[CDG+06], [DHJ+07] and [LM10]	
	Th 24 May	Recent Work: Distributed Computing		
10	Tu 29 May	Discussion	[DG08]	

[Mat89] Friedemann Mattern. Virtual time and global states of distributed systems. In *Parallel and Distributed Algorithms*, pages 215–226. North-Holland, 1989

Other topics and surveys

[Lyn89] N. Lynch. A hundred impossibility proofs for distributed computing. In *Proceedings of the eighth annual ACM Symposium on Principles of distributed computing*, PODC '89, pages 1–28, New York, NY, USA, 1989. ACM

[GL02] Seth Gilbert and Nancy Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *SIGACT News*, 33(2):51–59, June 2002

Recent papers

[Bur06] Mike Burrows. The chubby lock service for loosely-coupled distributed systems. In *Proceedings of the 7th symposium on Operating systems design and implementation*, OSDI '06, pages 335–350, Berkeley, CA, USA, 2006. USENIX Association

[CDG+06] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: a distributed storage system for structured data. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7*, OSDI '06, pages 15–15, Berkeley, CA, USA, 2006. USENIX Association

[DHJ+07] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. Dynamo: amazon's highly available key-value store. In *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, SOSP '07, pages 205–220, New York, NY, USA, 2007. ACM

[CGR07] Tushar D. Chandra, Robert Griesemer, and Joshua Redstone. Paxos made live: an engineering perspective. In *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, PODC '07, pages 398–407, New York, NY, USA, 2007. ACM

[DG08] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008

[LM10] Avinash Lakshman and Prashant Malik. Cassandra: a decentralized structured storage system. *SIGOPS Oper. Syst. Rev.*, 44(2):35–40, April 2010

Grading

The final grade will be divided as follows: 20% response papers (each weighed equally), 20% final paper, 20% participation in discussions, 40% project. The project grade will be further divided thusly: 15% for the project proposal, 25% for each progress report, and 35% for the final presentation. There will be no midterms or final exam.

Types of grades

Students may take this course for a quality grade (a "letter" grade) or a pass/fail grade. Students may declare, before handing in their final paper, whether (depending on their final grade) they want to receive a letter grade, a pass/fail grade or withdraw from the course (a *W* grade). For example, students can declare "If my final grade is a C+ or lower, I will take a *P* (Pass) instead of a letter grade and, if my grade is an *F*, I wish to take a *W*". By default, all students are assumed to be taking the course for a quality grade.

> Note: *Students taking this course to meet general education or concentration requirements must take the course for a letter grade.*

Policy on Academic Honesty

The University of Chicago has a formal policy on academic honesty that you are expected to adhere to:

`http://studentmanual.uchicago.edu/academic/index.shtml#honesty`

In brief, academic dishonesty (handing in someone else's work as your own, taking existing code and not citing its origin, etc.) will *not* be tolerated in this course. Depending on the severity of the offense, you risk getting a hefty point penalty or being dismissed altogether from the course. All occurrences of academic dishonesty will furthermore be referred to the Dean of Students office, which may impose further penalties, including suspension and expulsion.

Even so, collaboration between students is certainly allowed (and encouraged) *as long as you don't hand someone else's work as your own.* If you have discussed parts of an assignment with someone else, then make sure to say so. If you consulted other sources, please make sure you cite these sources.

If you have any questions regarding what would or would not be considered academic dishonesty in this course, please don't hesitate to ask the instructor.

Asking Questions

This course has an *open door policy* for asking questions. Instead of setting fixed office hours, you are welcome to consult with the instructor at any time. Nonetheless, you should try to give the instructor, whenever possible, some advance warning of your visit (by e-mail) to make sure that he will be in the office at that time.

The preferred form of support for this course is though the *course mailing list*, which can be used to ask questions and share useful information with your classmates. You can subscribe to the mailing list in the following web page:

`http://mailman.cs.uchicago.edu/mailman/listinfo/cmsc23310`

All questions regarding the assigned readings and the projects must be sent to the mailing list, and not directly to the instructor, as this allows your classmates to join in the discussion and benefit

from the replies to your question. This rule will be applied strictly: if you send a message directly to the instructor, you will only get a reply telling you to send your question to the mailing list. Take into account that, given the open-ended nature of the projects, discussing the projects openly (even their internal details) is encouraged and is not considered a breach of academic honesty policies.

Acknowledgements

We gratefully acknowledge the suggestions and feedback provided by Jacob Matthews (Google) and Lars Bergstrom (University of Chicago) in preparing the reading list for this course.