CMSC 23310/33310
Advanced Distributed Systems

Last updated: April 17, 2014

Department of Computer Science
University of Chicago

Spring 2014 Quarter

Dates: March 31 – June 4, 2014
Time and Location: Mondays and Wednesdays 1:30-2:50 in Cobb 301
Website: `http://www.classes.cs.uchicago.edu/archive/2014/spring/23310-1/`

Lecturer: Borja Sotomayor
E-mail: borja@cs.uchicago.edu
Office: Ryerson 151
Office hours: By appointment

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 BigTable, MapReduce, Spanner, Raft, 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.

A B+ or higher in 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 three components:

Reading and Discussion of Primary Sources: Several papers will be assigned each week, to be discussed on both Monday and Wednesday.

Homeworks: Two short homework assignments will be given in the first half of the quarter.

Project and Paper: In the second half of the quarter, students will have to implement a distributed system drawing upon the seminal work covered in the first half of the quarter. Based on their projects, students will have to write a final paper evaluating the features and performance of their project.

The discussion component is described in more detail below.

Paper discussion

Every week, we will discuss several papers in class (the *Course Schedule* section below provides a week-by-week list of papers). The papers for a given week will have a common theme, but the papers will be split between the Monday and Wednesday classes.

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: This group is responsible for preparing a list of 4–5 discussion questions about the papers to be discussed in class. For a given week, The Questioners must prepare their questions during the preceding week, and send them to the rest of the class by 3pm on Friday (of the preceding week). This means that The Questioners must read all the papers for their assigned week several days in advance of the actual discussion sessions.

The Answerers: During a discussion, this group takes the lead in answering the questions posed by The Questioners. In practice, this means that, whenever there is silence in the discussion, everyone looks at The Answerers to keep the discussion moving.

The Observers: During a discussion, this group will take notes on a shared document. These notes are not meant to be a transcription of what is being said in the discussion; they should capture the major take-away points of the discussion, as well as any issues The Observers feel should be discussed in more depth. The Observers can also search for additional resources, or answers to unresolved questions, on the Internet during the discussion itself.

These roles do not preclude anyone in the class from participating in the discussion. A member of The Observers can jump in when a question is posed, and a member of The Answerers can pose a new question on the fly.

Course Schedule

Week 1

The Monday, March 30, class will *not* be a discussion session. It will be an introductory lecture, and there is no required reading. *There will be no class on Wednesday, April 2*

Week 2 — Distributed Time

Required reading for Monday, April 7

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

Required reading for Wednesday, April 9

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

Suggested reading

- Parameswaran Ramanathan, Kang G. Shin, and Ricky W. Butler. Fault-tolerant clock synchronization in distributed systems. *Computer*, 23(10):33–42, October 1990
- David L. Mills. Improved algorithms for synchronizing computer network clocks. *SIGCOMM Comput. Commun. Rev.*, 24(4):317–327, October 1994

Week 3 — Distributed Consensus I

Required reading for Monday, April 14

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

Required reading for Wednesday, April 16

- Butler W. Lampson and Howard E. Sturgis. Crash recovery in a distributed data storage system, 1979
- D. Skeen and M. Stonebraker. A formal model of crash recovery in a distributed system. *IEEE Trans. Softw. Eng.*, 9(3):219–228, May 1983

Suggested reading

- M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, April 1980
- Philip A. Bernstein, Vassco Hadzilacos, and Nathan Goodman. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1987
- Fred B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Comput. Surv.*, 22(4):299–319, December 1990
- Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.*, 20(4):398–461, November 2002

Week 4 — Limits of Distributed Systems

Required reading for Monday, April 21

- 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
- Danny Dolev, Cynthia Dwork, and Larry Stockmeyer. On the minimal synchronism needed for distributed consensus. *J. ACM*, 34(1):77–97, January 1987

Required reading for Wednesday, April 23

- 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

Suggested reading

- 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

Week 5 — Paxos

Required reading for Monday, April 28 and Wednesday April 30

- Leslie Lamport. The part-time parliament. *ACM Trans. Comput. Syst.*, 16(2):133–169, May 1998
- Leslie Lamport. Paxos made simple. *ACM SIGACT News*, 32(4):18–25, December 2001

Week 6 — Distributed Consensus II

Required reading for Monday, May 5

- 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

- 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

Required reading for Wednesday, May 7

- Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm, 2014

Week 7 — Distributed Hash Tables

Required reading for Monday, May 12

- Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM Comput. Commun. Rev.*, 31(4):149–160, August 2001

- Antony I. T. Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, Middleware '01, pages 329–350, London, UK, UK, 2001. Springer-Verlag

Required reading for Wednesday, May 14

- 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

Week 8 — Distributed Data

Required reading for Monday, May 19

- Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. *SIGOPS Oper. Syst. Rev.*, 37(5):29–43, October 2003

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

Required reading for Wednesday, May 21

- James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Rajesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, and Dale Woodford. Spanner: Google's globally-distributed database. In *Proceedings of the 10th USENIX Conference on Operating Systems*

Design and Implementation, OSDI'12, pages 251–264, Berkeley, CA, USA, 2012. USENIX Association

Suggested reading

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

Week 9 — Distributed Currency

NOTE: No class on Monday, May 26 – Memorial Day

Required reading for Wednesday, May 28

- Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008

Week 10 — Review

Required reading for Monday, June 2

- Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, November 1974
- Leslie Lamport. Solved problems, unsolved problems and non-problems in concurrency. *SIGOPS Oper. Syst. Rev.*, 19(4):34–44, October 1985

No required reading for Wednesday, June 4

Grading

The final grade will be divided as follows:

- 15% homeworks (each weighed equally)
- 45% participation in discussions, broken down into:
  - 15%: In-class participation
  - 15%: Piazza participation
  - 15%: Participation in The Observers

  See below for a detailed rubric for each of these components.
- 20% project

- 20% final paper

There will be no midterms or final exam.
The "in-class participation" grade is an individual grade, scored out of 10:

- 10: Student participates consistently in all or most class discussions, even when part of The Observers.
- 9: Student participates consistently in all or most class discussions.
- 8: Student has actively participated in class discussions, but participation has not been consistent (e.g., very active in one discussion, completely silent in another)
- 7: Student has participated in class discussions, but falls below expectations.
- 0: Student has not participated in any class discussions.

The "Piazza participation" grade is an individual grade, mostly based on participation when the student is part of The Questioners. It is scored out of 10:

- 10: Student is consistently active on Piazza, not just contributing good questions when the student's group is The Questioners but also writing/answering posts outside his/her group.
- 9: Student consistently contributes good questions, but is only active on Piazza when his/her group is The Questioners.
- 8: Student has contributed questions or written/answered posts, but participation has not been consistent (e.g., very active one week, completely silent in another)
- 7: Student has contributed questions or written/answered posts, but falls below expectations.
- 0: Student has not written any meaningful posts or comments on Piazza.

The "Participation in The Observers" grade is a *group* grade. A grade is assigned to the entire group whenever their role is The Observers, and the final grade is the average of those grades. Each student in the group gets the same grade. It is scored out of 10:

- 10: Discussion log is detailed and well-written, and the group has supplemented it with external references (not limited to the suggested reading for that week) and/or provided answers to questions that were left unanswered during the discussion.
- 9: Discussion log is detailed, divided into concrete sections, and well-written. The group has gone beyond just presenting their raw notes from the discussion, and has put some effort into polishing up the notes. Someone who has not attended the discussion or even read the paper would get the gist of what was discussed that week.
- 8: Discussion log accurately reflects the structure and content of the discussion, but it is closer to a collection of notes than a polished account of the discussion. Someone who attended the discussion would find it useful to recall what was discussed, but someone who did not could find it hard to parse.

- 7: The discussion log reflects some, but not all, of the discussion. It lacks structure and is composed of a collection of unpolished notes.
- Any discussion logs worse than a 7 will receive a 0.

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 or a pass/fail 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". By default, all students are assumed to be taking the course for a quality grade. Requests for withdrawals must be made before the final paper is handed in.

> 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://college.uchicago.edu/policies-regulations/academic-integrity-student-conduct`

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, discussing the concepts necessary to complete the homeworks and project is certainly allowed (and encouraged). Under *no circumstances* should you show (or email) another student your code or post your solution to a web-page or social media site. If you have discussed parts of an assignment with someone else, then make sure to say so in your submission (e.g., in a README file or as a comment at the top of your source code file). 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

The preferred form of support for this course is though *Piazza* (`http://www.piazza.com/`), an on-line discussion service which can be used to ask questions and share useful information with your classmates. Students will be enrolled in Piazza at the start of the quarter.

All questions regarding assignments or material covered in class must be sent to Piazza, and not directly to the instructors or TAs, as this allows your classmates to join in the discussion and benefit from the replies to your question. If you send a message directly to the instructor, you will get a reply politely asking you to send your question to Piazza.

Piazza has a mechanism that allows you to ask a private question, which will be seen only by the instructors and teaching assistants. This mechanism should be used *only* for questions that require revealing part of your solution to a homework or project.

Piazza also allows students to post anonymously. *Anonymous posts will be ignored.* This is a majors-level course: you are expected to feel comfortable sharing your questions and thoughts with your classmates without hiding behind a veil of anonymity.

Finally, all course announcements will be made through Piazza. It is your responsibility to check Piazza often to see if there are any announcements. Please note that you can configure your Piazza account to send you e-mail notifications every time there is a new post on Piazza. Just go to your "Account/Email Settings", and click on "Edit Email Notifications" under CMSC 23310. We encourage you to select either the "Real Time" option (get a notification as soon as there are new posts) or the "Smart Digest" option (get a summary of all the posts sent over the last 1-6 hours – you can select the frequency).

Acknowledgements

We gratefully acknowledge the suggestions and feedback provided by Anthony Nicholson, Jacob Matthews, Will Robinson, and Matthew Steffen (all at Google) and Lars Bergstrom (Mozilla) in preparing the reading list for this course.