Course Syllabus
COMP 735 – Distributed and Concurrent Algorithms
Spring 2023

Meeting Place: FB 007
Meeting Time: 2:00 - 3:15, TuTh
Course Web Page: `http://www.cs.unc.edu/~anderson/teach/comp735`. The powerpoint slides and other things can be found here.

Instructor: Prof. Jim Anderson
Telephone: 590-6057
Office: FB 316
E-mail: anderson@cs.unc.edu
Office Hours: 12:30 - 1:30, TuTh

Goals of the course: To present fundamental algorithms and impossibility results from the concurrent-programming literature, and to cover techniques for formally specifying and verifying concurrent systems. Both message-passing and shared-memory models of concurrency will be considered. At the end of the course, students will have a general knowledge of the concurrent-programming literature, and will be able to develop new concurrent algorithms and verify their correctness. Perhaps the most important skill to be developed is the ability to intuitively "see" how or why a concurrent program works (a skill most students probably take for granted when it comes to sequential programs). In other words, this class will teach you how to "think" concurrently.

Who should take this class: Anyone who is likely to engage in work or research involving concurrent or distributed systems (with the prevelance of multicore machines today, this is basically everyone).

Prerequisites: Comp 550 (undergrad algorithms) and Comp 530 (undergrad operating systems). If you haven't had one or both prerequisites and would like to take the class anyway, please see me.

Grading:	Take-Home Assignments	65%
	Survey or Research Paper	30%
	Class Participation	5%

Take-home assignments will be given approximately once every three weeks. **These assignments are required to be individual efforts** (in contrast to, say, Comp 750). Any violations of this policy will be considered cheating and will be referred to the Student Attorney General. Any questions regarding the assignments should be referred to the instructor. You should not discuss the assignments with each other *at all*. Any attempts to find solutions on the web, in the files of other students who have taken this or similar courses, etc., will be considered an honor code violation. Any attempts to post my solutions on the web will be considered an honor code violation. (Sorry if this sounds harsh—it's better to be clear about these things up-front than to have misunderstandings later.)

Note: I do re-use old assignments. As noted above, any attempts to access old assignments from the files of students who have taken this course previously, or from other sources, will be considered an honor code violation.

Students will be required to investigate a research topic of their choosing in some depth, and to write a research paper or survey paper on that topic. Topics related to your research area are especially encouraged. Some example topics are given below. We will have a more in-depth discussion about the requirements for the paper later in the semester.

Class Etiquette: You are expected to maintain proper etiquette in class. This includes:

- not making a habit of arriving late, leaving in the midst of class, or skipping class,
- not talking, sleeping, reading newspapers, eating, etc. in class,
- keeping cellphones, pagers, etc. off in class,
- and not using your laptop to browse the web in class.

Class Participation: This class will be far more enjoyable for everyone if all students come to class ready and willing to discuss the material to be covered. I plan to reward those who participate in class by increasing their final grade by up to half a letter grade. I also reserve the right to add a similar negative "reward" for those who do not observe appropriate etiquette in class.

Text: The primary "text" will be research papers (see list below). We will also cover several chapters of the following books.

- *Distributed Systems*, second edition, Addison-Wesley, 1993, ISBN 0-201-62427-3, Sape Mullender, editor.
- P. Bernstein, V. Hadzilacos, and N. Goodman, *Concurrency Control and Recovery in Database Systems*, Addison-Wesley, 1987, ISBN 0-201-10715-5.

We cover just a few chapters from these books. You don't need to buy them.

If you really would like to have a textbook that covers (most of) the material from this course, the following is probably your best bet. (I have always listed this book as a potential resource, but I don't think any student has ever bought it, so I didn't order copies. If you want a copy, I suggest you use Amazon or some other similar on-line service.)

- N. Lynch, *Distributed Algorithms*, Morgan Kaufmann, 1996, ISBN 1-55860-348-4.

Possible Paper Topics: Clock synchronization, transactions, distributed simulation, distributed shared-memory systems, leader election algorithms, synchronization algorithms for real-time systems, frameworks for specifying/verifying concurrent systems (especially frameworks that emphasize program composition), automated verification tools, distributed search, distributed graph algorithms, distributed algorithms in multimedia systems, randomized concurrent algorithms, routing algorithms, lower bound results, distributed hashing, state maintenance in dynamic distributed systems, issues in peer-to-peer systems, synchronization in multicore systems, security issues in distributed systems, distributed algorithms for cloud-based computing. Students may also cover some additional papers on some of the topics to be covered in class by the instructor. For a topic to be approved, it must have a significant formal or algorithmic component.

Probable List of Topics: The following is a probable list of topics to be covered, along with a list of papers for each topic. I am willing to make slight changes to this list based on the interests of the class. Papers that are background or supplemental reading are denoted BR and SR, respectively. These papers will not be covered in class, but have been included because they contain information that may prove useful. (*You do not have to read these papers*. They are included for completeness, and also because they may contain things like definitions that are useful in understanding the papers that will be directly covered.)

- **Reasoning about distributed and concurrent programs (3 weeks).**

  Topics: state assertions, temporal assertions, safety properties, partial correctness, invariants, liveness properties, total correctness, well-founded sets, fairness, proof rules, example correctness proofs.

Papers:

A.U. Shankar, "An Introduction to Assertional Reasoning of Concurrent Systems," *ACM Computing Surveys*, Vol. 25, No. 3, September 1993, pp. 225-262.

BR (verification of sequential programs): D. Gries, *The Science of Programming*, Springer-Verlag, 1981.

SR: S. Owicki and D. Gries, "An Axiomatic Proof Technique for Parallel Programs," *Acta Informatica*, Vol. 6, 1976, pp. 319-340.

SR: Z. Manna and A. Pnueli, "Adequate Proof Principles for Invariance and Liveness Properties of Concurrent Programs," *Science of Computer Programming*, Vol. 4, 1984, pp. 257-289.

- **Synchronization algorithms for shared-memory systems (4.5 weeks).**

  Topics: mutual exclusion, barrier synchronization, readers/writers, local-spin algorithms, wait-free and lock-free synchronization.

  Papers — Mutual Exclusion with Read and Write Instructions:

  E. W. Dijkstra, "Solution of a Problem in Concurrent Programming Control," *Communications of the ACM*, Vol. 8, No. 9, September 1965, pp. 569.

  D. E. Knuth, "Additional Comments on a Problem in Concurrent Programming Control," *Communications of the ACM*, Vol. 9, No. 5, May 1966, pp. 321-322.

  L. Lamport, "A New Solution of Dijkstra's Concurrent Programming Problem," *Communications of the ACM*, Vol. 17, No. 8, August 1974, pp. 453-455.

  G. L. Peterson, "Myths About the Mutual Exclusion Problem," *Information Processing Letters*, Vol. 12, No. 3, June 1981, pp. 115-116.

  L. Lamport, "A Fast Mutual Exclusion Algorithm," *ACM Transactions on Computer Systems*, Vol. 5, No. 1, February 1987, pp. 1-11.

  J.-H. Yang and J. Anderson, "A Fast, Scalable Mutual Exclusion Algorithm," *Distributed Computing*, Vol. 9, No. 1, August 1995, pp. 51-60.

  SR: J. Anderson, Y.-J. Kim, and T. Herman, "Shared-Memory Mutual Exclusion: Major Research Trends Since 1986," *Distributed Computing*, Vol. 16, 2003, pp. 75-110, special issue celebrating the 20th anniversary of PODC.

  Papers — Mutual Exclusion with Read-Modify-Write Instructions:

  J. Mellor-Crummey and M. Scott, "Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors," *ACM Transactions on Computer Systems*, Vol. 9, No. 1, February 1991, pp. 21-65.

  SR: J. Anderson, Y.-J. Kim, and T. Herman, "Shared-Memory Mutual Exclusion: Major Research Trends Since 1986," *Distributed Computing*, Vol. 16, 2003, pp. 75-110, special issue celebrating the 20th anniversary of PODC.

  Papers — Barrier Synchronization Algorithms:

  J. Mellor-Crummey and M. Scott, "Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors," *ACM Transactions on Computer Systems*, Vol. 9, No. 1, February 1991, pp. 21-65.

  Papers — Readers/Writers:

  P.J. Courtois, F. Heymans, and D.L. Parnas, "Concurrent Control with Readers and Writers," *Communications of the ACM*, Vol. 14, No. 10, October 1971, pp. 667-668.

L. Lamport, "Concurrent Reading and Writing," *Communications of the ACM*, Vol. 20, No. 11, November 1977, pp. 806-811.

SR: G.L. Peterson, "Concurrent Reading While Writing," *ACM Transactions on Programming Languages and Systems*, Vol. 5, No. 1, 1983, pp. 46-55.

SR: J. Mellor-Crummey and M. Scott, "Scalable Reader-Writer Synchronization for Shared-Memory Multiprocessors," *Third ACM Symposium on Principles and Practice of Parallel Programming*, April 1991.

Papers — Wait-Free Synchronization using Read and Write Instructions:

Note: Most of the papers on this topic are rather long and quite formidable. I plan to merely hit the highlights when covering this material.

L. Lamport, "On Interprocess Communication, Parts I and II," *Distributed Computing*, Vol. 1, 1986, pp. 77-101.

Note: When reading the next two papers, just read the introductory parts and informal algorithm descriptions. Skip the formal proofs.

A. Singh, J. Anderson, and M. Gouda, "The Elusive Atomic Register," *Journal of the ACM*, Vol. 41, No. 2, March 1994, pp. 311-339.

Y. Afek, H. Attiya, D. Dolev, E. Gafni, M. Merritt, and N. Shavit, "Atomic Snapshots of Shared Memory," *Journal of the ACM*, Vol. 40, No. 4, 1993, pp. 873-890.

SR: J. Anderson, "Composite Registers," *Distributed Computing*, Vol. 6, 1993, pp. 141-154.

SR: J. Anderson, "Multi-Writer Composite Registers," *Distributed Computing*, Vol. 7, 1994, pp. 175-195.

Papers — Wait-Free and Lock-Free Synchronization using Stronger Primitives:

M. Herlihy, "Wait-Free Synchronization," *ACM Transactions on Programming Languages and Systems*, Vol. 13, No. 1, 1991, pp. 124-149.

M. Herlihy, "A Methodology for Implementing Highly Concurrent Data Objects," *ACM Transactions on Programming Languages and Systems*, Vol. 15, No. 5, 1993, pp. 745-770.

M. Herlihy and J.E. Moss, "Transactional Memory: Architectural Support for Lock-Free Data Structures," *Proceedings of the 20th Annual International Symposium on Computer Architecture*, 1993, pp. 289-300.

N. Shavit and D. Touitou, "Software Transactional Memory," *Distributed Computing*, Vol. 10, 1997, pp. 99-116.

SR: M. Herlihy and J. Wing, "Linearizability: A Correctness Condition for Concurrent Objects," *ACM Transactions on Programming Languages and Systems*, Vol. 12, No. 3, July 1990, pp. 463-492.

SR: J. Anderson and M. Moir, "Universal Constructions for Large Objects," *Proceedings of the Ninth International Workshop on Distributed Algorithms*, Lecture Notes in Computer Science 972, Springer-Verlag, September 1995, pp. 168-182.

SR: S. Ramamurthy, M. Moir, and J. Anderson, "Real-Time Object Sharing with Minimal System Support," *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing*, ACM, New York, May 1996, pp. 233-242.

SR: M. Michael and M. Scott, "Simple, Fast, and Practical Non-Blocking and Blocking Concurrent Queue Algorithms," *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing*, ACM, New York, August 1996, pp. 267-275.

- **Synchronization algorithms for message-passing systems (1 week).**

  Topics: mutual exclusion, logical clocks, fundamentals of resource allocation (dining philosophers, drinking philosophers).

  Papers:

  L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," *Communications of the ACM*, Vol. 21, No. 7, July 1978, pp. 558-565.

  G. Ricart and A.K. Agrawala, "An Optimal Algorithm for Mutual Exclusion in Computer Networks," *Communications of the ACM*, Vol. 24, No. 1, January 1981, pp. 9-17.

  M. Maekawa, "A $\sqrt{N}$ Algorithm for Mutual Exclusion in Decentralized Systems," *ACM Transactions on Computer Systems*, Vol. 3, No. 2, May 1985, pp. 145-159.

  E. W. Dijkstra, "Two Starvation Free Solutions to a General Exclusion Problem," EWD 625, Plataanstraat 5, 5671 Al Nuenen, The Netherlands.

  K. M. Chandy and J. Misra, "The Drinking Philosophers Problem," *ACM Transactions on Programming Languages and Systems*, Vol. 6, No. 4, October 1984, pp. 632-646.

- **Fault tolerance in message-passing systems (5 weeks).**

  Topics: Byzantine agreement (algorithms and impossibility results), distributed consensus (algorithms and impossibility results), atomic commit protocols (in particular, why they work in light of the impossibility results), synchrony (in particular, how "synchronous" does a system have to be to tolerate faults?), broadcast and multicast algorithms, active and passive replication, self-stabilizing systems.

  Papers — Byzantine agreement and consensus:

  L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Transactions on Programming Languages and Systems*, Vol. 4, No. 3, July 1982, pp. 382-401.

  M. Fischer, N. Lynch, and M. Patterson, "Impossibility of Distributed Consensus with One Faulty Process," *Journal of the ACM*, Vol. 32, No. 2, April 1985, pp. 374-382.

  SR: M. Pease, R. Shostak, and L. Lamport, "Reaching Agreement in the Presence of Faults," *Journal of the ACM*, Vol. 27, No. 2, April 1980, pp. 228-234.

  Papers — Atomic commit protocols:

  Material on atomic commit protocols taken from Chapter 7 of the book *Concurrency Control and Recovery in Database Systems*, P. Bernstein, V. Hadzilacos, and N. Goodman, Addison-Wesley, 1987.

  SR: L. Lamport, "Paxos Made Simple," *ACM SIGACT News (Distributed Computing Column)*, Vol. 32, No. 4, pp. 18-25, December 2001.

  SR: L. Lamport, "The Part-Time Parliament," *ACM Transactions on Computer Systems*, Vol. 16, No. 2, May 1998, pp. 133-169.

  Papers — Broadcast protocols:

  Chapter 5 (Fault-Tolerant Broadcasts and Related Problems) of *Distributed Systems*.

  Papers — Replication:

  Chapter 7 (Replication Management using the State-Machine Approach) of *Distributed Systems*.

  Chapter 8 (The Primary-Backup Approach) of *Distributed Systems*.

  Papers — Self-Stabilization:

E.W. Dijkstra, "Self-Stabilizing Systems in Spite of Distributed Control," EWD391, in *Selected Writings on Computing: A Personal Perspective*, Springer-Verlag, Berlin, 1982, pp. 41-46.

- **Algorithms for detecting stable properties in message-passing systems (1 week).**

  Topics: deadlock detection, termination detection, diffusing computations, distributed snapshots.

  Papers:

  E.W. Dijkstra, W.H.J. Feijn, A.J.M. van Gasteren, "Derivation of a Termination Detection Algorithm for Distributed Computations," *Information Processing Letters*, Vol. 16, 1983, pp. 217-219. Also EWD 840.

  E.W. Dijkstra, C.S. Scholten, "Termination Detection for Diffusing Computations," *Information Processing Letters*, Vol. 11, August 1980, pp. 1-4.

  K.M. Chandy and L. Lamport, "Distributed Snapshots: Determining Global States of Distributed Systems," *ACM Transactions on Computer Systems*, Vol. 3, No. 1, February 1985, pp. 63-75.

  K.M. Chandy, J. Misra, and L. Haas, "Distributed Deadlock Detection," *ACM Transactions on Computer Systems*, Vol. 1, No. 2, May 1983, pp. 144-156.

(This equates to 29 classes and we only have 28 this year, so we'll have to reduce something. I'll just figure this out on the fly.)

University Resources:

Accessibility Resources: The University of North Carolina at Chapel Hill facilitates the implementation of reasonable accommodations, including resources and services, for students with disabilities, chronic medical conditions, a temporary disability or pregnancy complications resulting in barriers to fully accessing University courses, programs and activities. Accommodations are determined through the Office of Accessibility Resources and Service (ARS) for individuals with documented qualifying disabilities in accordance with applicable state and federal laws. See the ARS Website for contact information: `https://ars.unc.edu` or email `ars@unc.edu`.

Counseling and Psychological Services (CAPS): CAPS is strongly committed to addressing the mental health needs of a diverse student body through timely access to consultation and connection to clinically appropriate services, whether for short or long-term needs. Go to their website: `https://caps.unc.edu/` or visit their facilities on the third floor of the Campus Health Services building for a walk-in evaluation to learn more.

Title IX Resources: Any student who is impacted by discrimination, harassment, interpersonal (relationship) violence, sexual violence, sexual exploitation, or stalking is encouraged to seek resources on campus or in the community. Please contact the Director of Title IX Compliance (Adrienne Allison Adrienne.allison@unc.edu), Report and Response Coordinators in the Equal Opportunity and Compliance Office (reportandresponse@unc.edu), Counseling and Psychological Services (confidential), or the Gender Violence Services Coordinators (gvsc@unc.edu; confidential) to discuss your specific needs. Additional resources are available at safe.unc.edu.