Advanced Topics in Concurrent Programming

Computer Science Seminar, 236832, Winter 2016/7

Instructor: Erez Petrank, erez@cs, office: Taub 528. Phone: 829-4942.

Announcements:

- There exists an errata file with known errors in the book. If your lecture is from the book, please take a look at it.
- About the schedule of the talks: the students who will register will pick about 20 of the topics below. Each selected topic will be given in a single class (50 minutes), meaning 2 topics a week. So the exact timing of a lecture will be determined only when all registered students will pick their topics, which will probably only happen at the first week on the semester. But the order between talks is given below.

Topics:

Introduction (all talks in this section are taken from the Art of Multiprocessor Programming by Herlihy & Shavit available at the library):

1. Introduction to Concurrency: hardware and software, see Appendix A and B of the book.
2. "Spin locks and contention in practice", chapter 7 in the book.
3. "Monitors and Blocking Synchronization", chapter 8 in the book.
4. Properties of concurrent computation: correctness and progress, chapter 3 in the book.

Concurrent Algorithms (all talks in this section, except 7, 14, and 15, are taken from the Art of Multiprocessor Programming by Herlihy & Shavit available at the library):

5. Linked Lists: the Role of Locking, chapter 9. *Assigned to Erez Petrank.* *Presentation*
6. Concurrent Queues and the ABA Problem, chapter 10. *Assigned to Erez Petrank.*
7. Wait-free Queues (paper).
8. Concurrent Stacks and Elimination, chapter 11.
9. Counting, Sorting, and Distributed Coordination, chapter 12.
10. Concurrent hashing and natural parallelism, chapter 13.
11. Skiplists and Balanced Search, chapter 14.
12. Priority Queues, chapter 15.
13. Futures, Scheduling, and Work Distribution, chapter 16.
14. Memroy management for lock-free data structures: hazard pointers (paper).
15. Memory management for lock-free data structures: optimistic access (paper).
16. Barriers, chapter 16.

Debugging:

17. Debugging tools: Microsoft's Chess (paper).
18. IBM’s synchronization coverage for testing (see paper).
19. A Randomized Scheduler for Bug Finding (see paper and a follow-up).
20. Deterministic parallelism (see paper).
21. Record-Replay parallel executions (see paper)

Systems Scalability:

22. An Analysis of Linux Scalability to Many Cores. (see paper).

Concurrent data structures on GPUs:

23. Performance Evaluation of Concurrent Lock-free Data Structures on GPUs. (see paper).
24. Mega-KV: a case for GPUs to maximize the throughput of in-memory key-value stores. (see paper).

Concurrent data structures on non-volatile memory:

25. Consistent and Durable Data Structures for Non-Volatile Byte-Addressable Memory. (see paper).
26. Persistent B+-Trees in Non-Volatile Main Memory. (see paper).

Server architectures:

27. Reliability: Amazon's Dynamo.
28. Facebook's (now Apache's) Cassandra: see this paper.
29. Google's Big-Table.
30. Spanner: Google's Globally-Distributed Database.

Languages and Runtimes for Concurrent and Distributed Programming:

31. Microsoft's Orleans runtime (see paper).
32. IBM's X10 programming language. (See webpage for researchers.).
33. Google's Map-Reduce. See also this paper.
34. Sun's Fortress (see paper, and slides from a tutorial).
35. The Cilk programming language and implementation (see paper).
36. Microsoft's Singularity project (see paper).

Grading

- Lecture (at least 85%),
- Participation (at most 15%),
- Reduction of 5 points for each (unjustified) absence.
- Each justified absence requires summary of missed lecture.
- The talk will be graded by: Knowledge of the material, Slides effectiveness, Communication of the ideas, and working within the time limit of 50 minutes.
- Main goal: make people understand!

Administrative

- Course syllabus (in Hebrew)
- Prerequisites: Algorithms 1 (234247) and Operating Systems (234123). Further knowledge in concurrent systems will help to understand the topics presented.
- Reception hours: TBD. (If this is not a good time for you, contact me by email to schedule an alternative.)

Material Covered in the Seminar:

The day that every (new) computer employs parallel processing has arrived. But writing concurrent programs is notoriously difficult and a large amount of traditional sequential code exists. A major challenge to the industry and (applied) research today is to find ways to make the programming for concurrent systems easier and accessible for all programmers. We will present concurrency programming setting, discuss several concurrent data structures, and present server architecture. Another major challenge is the debugging and more generally the reliability of concurrent programs. Some bugs only appear infrequently under specific scheduling scenarios. What are the right tools to discover such bugs and fix them? What are the tools to verify that a code is correct? More challenges relate to progress guarantees, specifically, lock-freedom. In this seminar, we will discuss some of these questions and check which answers exist today.

Assignments:

Each student will prepare a talk on one of the topics above.

Course Registration:

Registration is manual and we will attempt to match the registered students with the seminar topics. Please send an email to the lecturer (erez@cs) with your name, student id, accumulated average, umber of academic points accumulated by the end of the current semester, relevant courses studied, and relevant experience in industry or lab projects.

The final list of registered students will be determined after the first lecture in the beginning of the winter semester.