Distributed Consensus

CS 675: Distributed Systems (Spring 2020)

Midterm review

Yue Cheng

Some material taken/derived from:

- Princeton COS-418 materials created by Michael Freedman and Wyatt Lloyd.
- MIT 6.824 by Robert Morris, Frans Kaashoek, and Nickolai Zeldovich.
- Utah CS6450 by Ryan Stutsman.

Midterm

- Wednesday, March 4th, 4:30 – 6:30pm
  - 120 minutes
  - Open-book, open-notes (you may use class notes, papers, and lab materials; you may read them on your laptop, but you are not allowed to use any network)

- Covering topics from lec-1 to lec-5 (but not including Raft)
  - Go-specific debugging questions
  - High-level design questions

Concurrency in Go

- Labs that were completed
  - Possible race condition bugs in Go
  - Go channels
  - Go mutex locks

MapReduce

- Why MapReduce
  - Google workload characteristics
- How MapReduce works
  - Paper
- How data flows within a MapReduce job
  - Use of local file system and use of GFS
- Main shortcomings of using MapReduce

Time & Clocks

- Cristian's algorithm

- Logical Clock algorithm
  - Guarantees if a → b, then C(a) < C(b)
  - How to guarantee a total order of events
  - NYC+SF bank applications of LC – its assumptions: Why it works; under what assumptions it may not work

- Vector Clock algorithm
  - If V(a) < V(b), then a → b
  - If V(a) $\not<$ V(b) and V(b) $\not<$ V(a), then a || b
  - Can use to infer when an event b was aware of / influenced by a

Paxos

- The safety and liveness property of Paxos
- Basic Paxos algorithm
- Basic Paxos examples
- Only proposers knows which value has been chosen
  - If other peers want to know, they must execute Paxos with their own proposal