How to Read a Paper?

CS 475: Concurrent & Distributed Systems (Fall 2021)

Yue Cheng

How to read a paper: Raft Example

In Search of an Understandable Consensus Algorithm (Extended Version)

Diego Ongaro and John Ousterhout
Stanford University

Abstract

Raft is a consensus algorithm for managing a replicated log. It produces a result equivalent to (multi-)Paxos, and it is as efficient as Paxos, but its structure is different from Paxos; this makes Raft more understandable than Paxos and also provides a better foundation for building practical systems. In order to enhance understandability, Raft separates the key elements of consensus, such as leader election, log replication, and safety, and it enforces a stronger degree of coherency to reduce the number of states that must be considered. Results from a user study demonstrate that Raft is easier for students to learn than Paxos. Raft also includes a new mechanism for changing the cluster membership, which uses overlapping majorities to guarantee safety.

1 Introduction

Consensus algorithms allow a collection of machines to work as a coherent group that can survive the failures of some of its members. Because of this, they play a key role in building reliable large-scale software systems. Paxos [15, 16] has dominated the discussion of consensus algorithms over the last decade: most implementations of consensus are based on Paxos or influenced by it, and Paxos has become the primary vehicle used to teach students about consensus.

state space reduction (relative to Paxos, Raft reduces the degree of nondeterminism and the ways servers can be inconsistent with each other). A user study with 43 students at two universities shows that Raft is significantly easier to understand than Paxos: after learning both algorithms, 33 of these students were able to answer questions about Raft better than questions about Paxos.

Raft is similar in many ways to existing consensus algorithms (most notably, Oki and Liskov's Viewstamped Replication [29, 22]), but it has several novel features:

- **Strong leader:** Raft uses a stronger form of leadership than other consensus algorithms. For example, log entries only flow from the leader to other servers. This simplifies the management of the replicated log and makes Raft easier to understand.
- **Leader election:** Raft uses randomized timers to elect leaders. This adds only a small amount of mechanism to the heartbeats already required for any consensus algorithm, while resolving conflicts simply and rapidly.
- **Membership changes:** Raft's mechanism for changing the set of servers in the cluster uses a new *joint consensus* approach where the majorities of two different configurations overlap during transitions. This allows the cluster to continue operating

How to read the Raft paper: Summary

- Before 1st pass: go thru the Raft lecture notes & watch the YouTube video (by John Ousterhout)
  - Make notes about any confusing parts

How to read the Raft paper: Summary

- Before 1st pass: go thru the Raft lecture notes & watch the YouTube video (by John Ousterhout)
- 1st pass: Read abstract, Section 1+2, conclusion
- 2nd pass: Read Section 3-5 in detail, make notes
  - Use **Figure 2** as a guide (how one implements Raft)

How to read the Raft paper: Summary

- Before 1st pass: go thru the Raft lecture notes & watch the YouTube video (by John Ousterhout)
- 1st pass: Read abstract, Section 1+2, conclusion
- 2nd pass: Read Section 3-5 in detail, make notes
  - Use **Figure 2** as a guide (how one implements Raft)
- Iterate, on day 2, day 3… That'd give you more insights
  - While you are doing labs…

How to read the Raft paper: Summary

- Before 1st pass: go thru the Raft lecture notes & watch the YouTube video (by John Ousterhout)
- 1st pass: Read abstract, Section 1+2, conclusion
- 2nd pass: Read Section 3-5 in detail, make notes
  - Use **Figure 2** as a guide (how one implements Raft)
- Iterate, on day 2, day 3… That'd give you more insights

- Some key points:
  - What is state machine replication and why do we need it?
  - How does Raft handle normal operations?
    - Leader election (Lab 2A)
    - Log replication (Lab 2B)
    - Log persistence (Lab 2C)
  - How does Raft handle network failures & leader changes?
    - Log consistency (Lab 2B/2C)
    - Log persistence (Lab 2C)
  - …