Wasfi Momen

Paxos Algorithms Graduate Project Proposal

Paxos algorithms discussed frequently in academic papers are scarcely implemented in distributed systems. Unfortunately, Paxos holds a reputation of being complex and hard to understand. Nevertheless, the arrival of cloud computing has spurred the development of distributed algorithms and solutions with Paxos a strong candidate for implementation. In this project, we will attempt to summarize the research done on Paxos and implement our own version of Paxos to compare to other algorithms.

To summarize the discussion around Paxos, the focus will be on the same perspective that is taken in an undergraduate networks class.. While the high-level proofs will be mentioned, the main mode of explanation will be akin to how TCP is introduced through the various forms of RDT and the needs required to make a substantial protocol. Paxos consists of proposers, acceptors, learners, and leaders [1]. By drawing parallels to the client-server and P2P models, these proposers and acceptors will be explained as the basics of Paxos with leaders and learners mentioned as additional tools. Other research with Paxos consists of real-world applications and further improvements on the algorithm which will be mentioned to give a view of the development of Paxos in the distributed systems field [2]. By the end of the explanation, we should have a good idea on on a specification of Paxos similar to the RFCs on TCP.

For the practice implementation of Paxos, many papers and resources use Python code to create a real-world scenario using Paxos. We will use Tom Cocagne's Essential Paxos repository for the basic implementation [3]. The aim is to create a simple Paxos implementation that can track a file upload across the network with each unreliable node contributing, the algorithm account for failure of nodes, and the final consensus of the file. We hope to implement a real-world scenario similar to Internet censorship where bandwidth speeds are limited or DNS servers are disconnected where the end hosts are unreliable.

From the results of the project, we will have a compilation of the research done across Paxos at the complexity of an undergraduate networks class and a practical implementation example. By comparing Paxos to other well-known models like client-server and P2P, we can introduce Paxos as a simpler concept than current academic papers. After our research, we can create a discussion of how and why to use Paxos in applications or whether it should be used at all.

[1] Leslie Lamport, "Paxos Made Simple"*, ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121, December 2001)*, December 2001

[2] T. F. Rezende, P. Sutra, R. Q. Saramago, L. Camargos, "On Making Generalized Paxos Practical", *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA),* March 2017

[3] Tom Cocagne, Essential Paxos, January 2013, Github Repository, https://github.com/cocagne/paxos

[4] Chandra, Tushar D., Robert Griesemer, and Joshua Redstone. "Paxos made live: an engineering perspective." *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*. ACM, 2007.