A Review of ‘Paxos made simple’

Prashanth R Duggirala

Summary

Paper by LESLIE LAMPORT, 2001

The paper tries to re-explain the paxos algorithm for fault tolerant distributed systems in a simple and plain manner instead of a metaphorical explanation as the original paper by the same author which first mentioned this algorithm. In the presence of failures such as nodes crashing or messages getting delayed or lost, the paxos algorithm tries to ensure that the value of a data object/item stored across a distributed system be consistent based on a majority vote between the nodes. Once a value is chosen based on the vote, the new value can only be learned and updated by any other nodes that were not a part of the voting majority, eventually making the whole system consistent.

The algorithm presents a distributed system where there are three types of nodes: proposers, acceptors and learners and any single node can perform multiple roles. The paxos algorithm details an interaction protocol between these nodes in order to reach an agreement on the value of a data object/item. It contains two phases called prepare and accept phases, where in the first phase proposers send a prepare message to the rest of the nodes who act as acceptors. The acceptors respond to the prepare message if the proposal ID is greater than the proposal IDs of the previous proposal they have received so far. A proposal ID is a unique number which is used to distinguish between proposals and can be implemented in multiple ways (for example, as a timestamp). In the next phase, the proposer sends an accept message once he receives a majority of positive responses and then the acceptors can inform the learners of the new updated value. To guarantee progress, a distinguished proposer must be selected as the only one to try issuing proposals. If the distinguished proposer can communicate successfully with a majority of acceptors, and if it uses a proposal with number greater than any already used, then it will succeed in issuing a proposal that is accepted. By abandoning a proposal and trying again if it learns about some request with a higher proposal number, the distinguished proposer will eventually choose a high enough proposal number. The system can also have one or more nominated distinguished learners to which acceptors send their acceptance messages and then these broadcast to the rest of the learners.

Strong points

1. Paxos algorithm aims at having the same value of an object stored across a set of distributed nodes based on a majority consensus between the nodes, without the need of a centralized coordinator node.

2. The paper aims at solving the problem of choosing a single value from a set of proposed values in a distributed system. Its aim is to ensure safety and liveness (that all known nodes eventually get to know about the accepted value).

3. The paper states an idea of having a system with a single proposer called the distinguished proposer where in only the distinguished proposer can propose updates.

Weak points

1. This paper assumes a non-byzantine model. A couple of malicious proposers could disrupt progress by continuously proposing things with large values of the proposal number. The presumption that messages will not be corrupted makes the protocol less robust.

2. Performance of this consensus protocol is greatly dependent on rareness of failures. This is almost impossible to ensure in distributed systems of large scale. The prepare phase of the paxos run has to be restarted every time there is a failure and the frequency of these failures considerably impacts the general performance.

3. The basic paxos algorithm described in the paper can have a huge network overhead in terms of the number of messages exchanged between the proposers, acceptors and learners in each paxos run.

Comments expanding upon weak points

Paxos first defines a protocol capable of reaching an agreement on a single decision, such as a single replicated log entry. It is referred as single-decree Paxos. Paxos then combines multiple instances of this protocol to facilitate a series of decisions such as a log this is called Multi-Paxos algorithm. There is no widely agreed upon algorithm for multi-Paxos. Lamport's descriptions are mostly about single-decree Paxos; he sketched possible approaches to multi-Paxos, but many details are missing. The paper mentions a variant of paxos which involves a distinguished proposer, and requires re-election of new proposer upon failure of a current one. Having such a leader based protocol can cast doubts upon the degree of decentralization this algorithm achieves.

References:

https://medium.com/coinmonks/paxos-made-simple-3b83c05aac37

https://blog.acolyer.org/2015/03/04/paxos-made-simple/

Paxos Made Moderately Complex, Robbert van Renesse, Department of Computer Science, Cornell University

https://en.wikipedia.org/wiki/Paxos_(computer_science)#Cheap_Paxos

http://pages.cs.wisc.edu/~swift/classes/cs739-sp11/blog/2011/03/paxos_made_simple.html

In Search of an Understandable Consensus Algorithm (Extended Version), Diego Ongaro and John Ousterhout, Stanford University