Paxos Made Moderately Complex

Robbert Van Renesse and Deniz Altinbuken

Overview

Paxos Made Moderately Complex presents and explains the multidecree variant of the Paxos protocol first proposed by Leslie Lamport in his 1998 paper, ‘Part Time Parliament’. The paper first defines the terms that frequently occur after which it explains the environment in which State Machine Replication works. After this, the paper defines the problem of achieving consensus in an asynchronous environment and goes on to explain how Paxos is successful at helping the replicas achieve consensus on which client command they must execute.

The paper then goes into great depth on a step-by-step account of how Paxos works, including, once again defining important terms, and significantly, the states and invariants that are used by this protocol and later on how the invariants hold true as Paxos proceeds further. It then arrives at the heart of Paxos, the Synod, explaining the system of ballots, proposals, commanders, scouts, and the active and passive modes of the leaders as well as the associated invariants and how they hold. The optimal conditions Paxos are described along with the less optimal circumstances, which also introduces the concept of failure detection, and the assumptions that Paxos are built on that help it contradict the FLP impossibility result.

The paper then goes on to discuss tweaks that can be made to the protocol discussed earlier, that tailor it for real-world distributed systems. After this, some variants of Paxos are discussed. In the conclusion, the authors list a number of related papers and works. The appendix consists of a Python package that implements Paxos.

Thoughts on Paxos as a Concept

The single most defining feature of Paxos, in my opinion, is the simplicity of its invariants that is contrasted with the complexity of its applications in terms of hypothetical yet not impossible circumstances and obscure edge cases. The intuitiveness of the invariants makes it easy to identify cases in which they would or wouldn’t hold. Even though this paper introduced numerous invariants, they synergise with the demands of the environment so effectively that the difficulty of keeping track of the invariants is mitigated.

Failure detection is, in my opinion, another defining characteristic of Paxos, which is key in helping it contradict the FLP impossibility statement. I think that using timing assumptions in the scheme of adapting timeout intervals to the ballot numbers is extremely clever. For me, it also clearly illustrated Paxos’ applicability with regard to real-world systems, especially since the section before it was highly theoretical.

Paxos also seems to be defined by its configurability. I think that one of the best things about Paxos is that it can be modified greatly (as seen in the final two sections) without violating any of the safety or progress principles. I get the impression that with Paxos, there is a vast probability space and that there are many more variants of Paxos that can be put together.

However, this ‘wide probability space’ seems to be a double-edged sword. Asynchronous environments are inherently rife with uncertainty and Paxos has to work with that uncertainty. The downside to this seems to be that this leaves room for a high number of edge cases, some that the invariants will hold for, and some that it will not. Even if the invariant holds though, it could sometimes be difficult to work out exactly how the series of events would play out following Paxos.

Reading through this paper, I came away with the impression that Paxos exists at the intersection of real-world application and theory of computation, which is an interesting position. I think that a feature that highlights this situation is the use of assumptions that Paxos is based on. Without assuming clock drift, Paxos would not be able to guarantee termination. The clock drift assumption has its basis in real time. However, a more fundamental assumption, like assuming that messages that reach replicas are non-faulty seems more theoretical to me, i.e. it seems possible that messages could be corrupted in real-world systems.

Thoughts on this paper in particular

I appreciated the structure of this paper. It had the difficult task of presenting a great deal of information about the workings of a deceptively simple algorithm and was, in my opinion, successful in conveying the workings of Paxos in a cohesive and mostly comprehensive manner. I believe that it’s a point of favour to this paper that at no point did I feel like I needed to consult some supplementary material to understand what the paper was talking about. Although I was initially alarmed by the size of this paper, I gradually realized that the authors effectively used the space to explain Paxos thoroughly. Even though the paper didn’t go into depth about each and every edge case, the fundamentals were explained clearly enough that it would be effective as reference for dealing with edge cases.

I also appreciate the paper’s decision to not go into proofs of various statements, as seen in Part-Time Parliament but instead focus on invariants. I feel that invariants are easier to connect to real-world circumstances in systems, and are more intuitive. It was fairly easy to follow why an invariant would hold in a system.

I think that the addition of the Python package in the appendix was a good one, since personally it was easier to understand the Python code than the corresponding pseudocode for a given component.

However, I also have a few minor quibbles and nitpicks regarding this paper, but I recognize that these are very trivial and subjective. First, I have to wonder why the authors did not explain a single-decree Paxos. Given their commitment to thorough explanations, and the fact that they included variants later, they could have introduced multi-Paxos as a variant. Secondly, regarding the variants at the end, I felt that this section was a bit superfluous and didn’t really fit in with the rest of the paper. The earlier sections were very meticulous, but ‘Paxos Made Practical’ and the variants were a lot harder to understand, and I’m not entirely sure I’ve actually understood them all. Third, and this is the most minor, the reliance on shorthand for the invariants in a paper that relies on them so heavily meant that I had to keep flipping back and forth through the paper, to confirm what exactly ‘R4’ and ‘A3’ were. I wish they tried to incorporate the actual invariants in statements full of this shorthand. But I understand why they chose not to do this since the paper is long enough as it is!

Conclusion

Paxos Made Moderately Complex is a well-written paper that comprehensively and meticulously explains the multi-decree Paxos protocol, along much auxiliary information regarding definitions and explanations. This paper gave me an extremely thorough understanding of Paxos and I believe that this paper is a good source of reference for Paxos.