Paper Review: SpotLess: Concurrent Rotational Consensus Made Practical through Rapid View Synchronization

1 Introduction

This paper introduces SPOTLESS, a novel concurrent rotational consensus protocol that brings improvements and practical use. SPOTLESS stands apart with several main traits: chained rotational consensus for request replication with reduced communication overhead, a Rapid View Synchronization (RVS) protocol fd for easier view handling and recovery from faults and a strong concurrent system allowing multiple request handling for better data flow. The paper also highlights how SPOTLESS works in Apache ResilientDB, a reliable blockchain database.

2 Strong Points

- **Chained Rotational Consensus**: SPOTLESS uses a chained rotational consensus design that replicates requests with reduced message cost and offers low-cost failure recovery. This approach helps SPOTLESS achieve high performance by eliminating the complex view-change protocol used in traditional consensus protocols. It also contributes to resilience by simplifying the recovery process and minimizing the impact of malicious primaries.
- **Rapid View Synchronization (RVS)**: RVS is a core component of SPOTLESS that enables the protocol to work in a wider range of network conditions. RVS does not require a Global Synchronization Time to synchronize views and can recover valid earlier views using reliable replicas. This allows both performance improvement, by permitting ongoing low-cost primary rotation and durability, by supplying strong failure recovery even in shaky communication settings.
- **Concurrent Consensus Architecture**: SPOTLESS runs multiple instances of its chained consensus protocol concurrently. This design lets SPOTLESS reach high speed by handling requests together, avoiding slowdowns found in older rotational methods like HOTSTUFF. The concur-

rent architecture also contributes to resilience by distributing the workload across multiple instances, reducing the impact of any single instance failure.

3 Weaknesses

- The Rapid View Synchronization(RVS) protocol incorporates regular leader replacement which leaves it open to latency delays because of leaders intentionally slowing down the proposal to maximize the fees received [1].
- The paper does not specify how SPOTLESS works in order to mitigate tail forking attacks that can increase latency.
- SPOTLESS relies on RVS to handle view changes efficiently but requires periods of network reliability to ensure timely synchronization which is not always guaranteed in unreliable network conditions.

4 Detailed Feedback

- **Leader Slowness Impact**: Non-Faulty Leaders can also intentionally slow down the network in order to maximize their potential profits. These leaders can postpone block proposals to accumulate transactions with higher fees, increasing their rewards. This phenomenon can cause increased latency in processing the transaction from the clients thus impacting system performance.
- **Tail Forking Attacks**: In chained consensus mechanisms, multiple coordinated faulty leaders in sequence can significantly reduce system throughput[2]. The paper does not explicitly specify how Spotless works in avoiding such latencies caused due to faulty leaders.
- **Reliance on Synchronous Periods for Rapid View Synchronization (RVS)**: Spotless and other leader-based protocols rely on view changes for primary rotations, this works well with minor network issues. However in frequent network delays, view changes add further communication overhead. Leader-less protocols can be used to avoid this as every participant works independently and validates transactions without a central coordinator(leader) for each consensus round.

5 Conclusion

Overall , the paper presents a novel and potentially impactful approach to chained consensus protocols. The Rapid View Synchronisation(RVS) provides an efficient mechanism to deal with faulty leaders and replace them in a concurrent chain based consensus mechanism. Through comprehensive evaluations, the paper also demonstrates that SPOTLESS surpasses existing consensus protocols like PBFT, HOTSTUFF, NARWHAL-HS, and RCC in terms of throughput, latency, and resilience.

References

[1] Kang, D., Gupta, S., Malkhi, D., & Sadoghi, M. (2024). HotStuff-1: Linear Consensus with One-Phase Speculation. arXiv preprint arXiv:2408.04728

[2] Neil Giridharan, Florian Suri-Payer, Matthew Ding, Heidi Howard, Ittai Abraham,and Natacha Crooks. 2023. BeeGees: Stayin' Alive in Chained BFT. In Proceedings of the 2023 ACM Symposium on Principles of Distributed Computing (Orlando, FL,USA) (PODC '23). Association for Computing Machinery, New York, NY, USA, 233–243. https://doi.org/10.1145/3583668.3594572