Building High Throughput Permissioned Blockchain Fabrics: Challenges and Opportunities

Suyash Gupta, Jelle Hellings, Sajjad Rahnama, Mohammad Sadoghi
Exploratory Systems Lab, Department of Computer Science
University of California, Davis, CA 95616-8562, USA

ABSTRACT

Since the introduction of Bitcoin—the first widespread application driven by blockchains—the interest in the design of blockchain-based applications has increased tremendously. At the core of these applications are consensus protocols that securely replicate client requests among all replicas, even if some replicas are Byzantine faulty. Unfortunately, these consensus protocols typically have low throughput, and this lack of performance is often cited as the reason for the slow wider adoption of blockchain technology. Consequently, many works focus on designing more efficient consensus protocols to increase throughput of consensus.

We believe that this focus on consensus protocols only explains part of the story. To investigate this belief, we raise a simple question: *Can a well-crafted system using a classical consensus protocol outperform systems using modern protocols?* In this tutorial, we answer this question by diving deep into the design of blockchain systems. Further, we take an in-depth look at the theory behind consensus, which can help users select the protocol that best-fits their requirements. Finally, we share our vision of high-throughput blockchain systems that operate at large scales.

PVLDB Reference Format:
Suyash Gupta, Jelle Hellings, Sajjad Rahnama and Mohammad Sadoghi. Building High Throughput Permissioned Blockchain Fabrics: Challenges and Opportunities. *PVLDB*, 13(12): 3441-3444, 2020.
DOI: https://doi.org/10.14778/3415478.3415565

1. INTRODUCTION

Since the introduction of *Bitcoin*—the first wide-spread application driven by *blockchain*—the interest in the design of blockchain-based applications has increased tremendously. This interest has resulted in several blockchain-inspired fabrics and database systems [2, 3, 17, 27, 39, 40]. Further, blockchain-based systems have been employed to address challenges in various other fields such as food production, managing land property rights, energy trading, and managing identities.

Proceedings of the VLDB Endowment, Vol. 13, No. 12
ISSN 2150-8097.
DOI: https://doi.org/10.14778/3415478.3415565

At the core of these blockchain applications are Byzantine-Fault Tolerant (BFT) consensus protocols that ensures all replicas of this blockchain application reach *consensus* on the ordering of incoming client request, this even if some of the replicas are Byzantine [8, 24, 27, 34, 45].

A decade after the introduction of blockchains in crypto-currencies and after several prominent research projects, we see that crypto-currencies are still the major *known use-cases* of blockchains. This raises a key question: *Why have blockchain applications seen such a slow wider adoption?* The low throughput and high latency of BFT consensus are cited as key reasons for this. Prior works have shown that traditional distributed databases can achieve throughputs of the order 100 K transactions per second [29, 30], while initial *permissionless blockchain applications* such as Bitcoin [38] and Ethereum [44] have throughputs of only a few transactions per second. In these crypto-currency applications, low throughputs are seen as acceptable, as these costly techniques enable a fully decentralized currency that is not controlled by a single government or corporation. Indeed, crypto-currency blockchains typically have *open-membership*, as anyone can join these blockchains.

Although permissionless blockchains highlight the notions of decentralization and resilience, their open-membership is often unnecessary for fault-tolerant transaction processing. This led to the design of industry-grade *permissioned* blockchains, where only a select group of users, some of which may be untrusted, can participate [3]. These permissioned designs employ traditional BFT consensus to provide throughputs of up-to 10 K transactions per second [2, 3], which is still short of the performance expected of modern systems. Several prior works [9, 11, 34, 45] blame the low throughput and scalability of permissioned blockchains on the underlying BFT consensus. Although these claims are not false, they only explain part of the story. From our perspective, permissioned blockchain applications can achieve wider adoption by improving in three vital directions.

First, it is well-known that an efficient protocol may not always lead to a high-throughput implementation. The same principle applies to consensus protocols used in existing permissioned blockchain fabrics. Although these fabrics provide platforms to employ blockchains in various use-cases, these fabrics fall short in their architectural details [2, 3, 6, 21]. We claim that the low throughput of these fabrics is due to missed opportunities during their design and implementation. Indeed, a well-crafted blockchain fabric can have an order-of-magnitude increase in its throughput, e.g., by exploiting parallelization and pipelining opportunities.

Second, the full replication employed by current blockchain applications stands in their way of achieving ever-higher performance. To make blockchains more usable, their design needs to evolve to incorporate sharding and specialization. To support such designs, new *resilient techniques* besides consensus need to be developed.

Finally, permissioned blockchain systems need tuning to specific settings. E.g., blockchains can be used for *federated data management*, the collective management of a single database among various stakeholders. Federated data management is in itself a major step towards dealing with *data quality issues* arising from the non-federated interchange of information between various stakeholders and, as such, can reduce the huge negative economic impact of bad data [16, 33, 42]. In federated data management, resilience is less of a priority, and the focus of such blockchain systems is on fast data update and retrieval, efficient query processing, and modular data analysis.

2. OUTLINE OF THE TUTORIAL

In this tutorial, we will provide a deep dive into consensus protocols with a focus on data management. To do so, we take an in-depth look at Byzantine fault-tolerant consensus protocols, the main technique powering permissioned blockchains. The tutorial is intended for an audience that has prior knowledge of databases and will be of interest to both theoreticians and practitioners who want to employ blockchain concepts to their work.

This tutorial starts with a general-purpose introduction to blockchains from the perspective of data management. Following the introduction, the tutorial will focus on three avenues. First, we look at the theoretical framework in which permissioned blockchains operate. Then, we look at practical high-performance consensus protocols and at current developments. This theoretical framework provides users of a permissioned blockchain system with the right tools to select the consensus protocols that *best-fits* their requirements. Second, we look at the architectural challenges in the design of high-performance permissioned blockchain systems, in which we show how existing principles of thread parallelization and task pipelining can be applied to blockchain fabrics. Further, we illustrate ways to optimize data access and query processing in permissioned blockchain applications. Finally, we look at the design challenges for high-performance permissioned blockchain systems of the future that can deal with huge amounts of data. We conclude by presenting our vision on future developments. Next, we explain these three avenues in detail.

BFT Consensus Protocols. Blockchains are, at their basis, fully replicated distributed systems that aim to maintain data consistency. The well-known CAP Theorem puts restrictions on the types of failures these blockchains can deal with while guaranteeing continued services [7, 20]. The CAP Theorem puts rather general limitations on the design of blockchains, however. More specific limitations are also known, as the Byzantine consensus problem and other related problems, such as the Byzantine agreement problem and the interactive consistency problem, have received considerable attention.

It is well-known that the Byzantine agreement problem can only be solved when using synchronous communication [19, 37, 43]. In a synchronous environment with $\mathbf{n}$ replicas of which $\mathbf{f}$ are Byzantine (e.g., malicious), Byzantine agreement requires that $\mathbf{n} > 3\mathbf{f}$ [12, 13]. When strong cryptographic primitives are available, this can be improved to $\mathbf{n} > \mathbf{f}$ [15, 35, 41] (although practical systems will still require $\mathbf{n} > 2\mathbf{f}$). Additionally, bounds on the amount of communication and the quality of the network are known [10, 12, 13, 14, 15, 18].

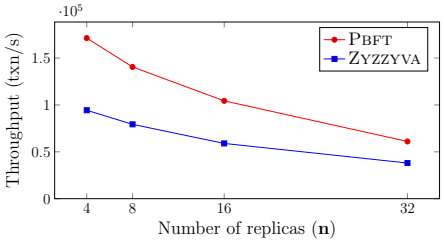

Figure 1: Two permissioned applications employing distinct BFT protocols.

Having provided a theoretical background, we take a step towards detailing practical consensus protocols. We do so by a full coverage of the *Practical Byzantine Fault Tolerance* consensus protocol (PBFT) of Castro et al. [8]. Next, we also look at the lineage of consensus protocols that refine and improve PBFT. This detailed overview will cover many of the practical consensus protocols currently in use and, simultaneously, also covers recent developments. Our coverage will include protocols such as PBFT, HotStuff [45], Zyzzyva [4, 34], FaB [36], SynBFT [1], RBFT [5], PoE [23], and Multi-BFT [24, 25]. All of these protocols make some tradeoffs and, hence, achieve optimal throughput only under specific conditions. In our tutorial, we will discuss these tradeoffs and conditions in detail, as this enables one to select the protocol that *best-fits* the requirements of their particular blockchain application.

Architectural Paradigm. Although an efficient consensus protocol can help increase the throughput of the associated permissioned blockchain application, the design of the system and its implementation matters equally. In our tutorial, we will show that classical BFT protocols that are perceived to be slow such as PBFT [8] can *always* outperform niche-case optimized BFT protocols such as Zyzzyva [34] if implemented in a well-designed and skillfully-optimized blockchain fabric. We use Figure 1 to illustrate such a possibility. In this figure, we measure the throughput of RESILIENTDB, our permissioned blockchain system [27, 28], and intentionally make it employ the "slow" PBFT protocol. Next, we compare the throughput of RESILIENTDB against a permissioned blockchain system that adopts practices suggested in BFTSmart [6], and employs the fast Zyzzyva protocol. We observe that the *system-centric* design of RESILIENTDB can use the costly three-phase PBFT protocol (of which two phases require quadratic communication among replicas), while still outperforming systems utilizing Zyzzyva (a single-phase protocol with linear communication among replicas).

Decades of academic research [29, 30] have helped the community in designing efficient distributed databases and applications. In this tutorial, we study existing practices to design an optimal permissioned blockchain fabric. These practices include the use of speculation [34, 36], execute-order or order-execute paradigms [3], component modular-

ity [6], transaction batching or streaming [2, 11], and out-of-order message processing [28]. Further, we will discuss how data storage, data maintenance, data retrieval, and NoSQL and relational database support is provided by our state-of-the-art permissioned blockchain fabrics [28]. We will demonstrate this functionality via a user interface that can ease query processing and data analysis.

Challenges and our Vision. As outlined above, the key component of any permissionless and permissioned blockchain remains the underlying BFT consensus protocol that provides *reliable* replication. Unfortunately, these protocols are *challenged* by the scalability and performance required by many modern big-data-driven applications. In specific, we see that there is no obvious way to scale up BFT consensus: adding more replicas will only increase the cost of replication and decrease the throughput of the system, even when using the most efficient consensus protocols.

We will close our tutorial by discussing recent steps toward the design of new fault-tolerant architectures that step away from the full-replicated nature of blockchains, this to increase scalability and the ability to serve big-data-driven applications. To put our vision in practice, we will first look at two low-level techniques, *cluster-sending* [31] and *delayed-replication* [32]. Next, we look at high-level designs enabled by these techniques to provide high-performance parallelized consensus [24, 25] and to provide high-performance consensus in sharded and geo-scale aware architectures [27].

3. BIOGRAPHICAL SKETCHES

Suyash Gupta is a Ph.D. Candidate at the Computer Science Department at University of California, Davis. At UC Davis, he is a senior member of *Exploratory Systems Lab* and works under the supervision of Prof. Sadoghi. He also works as the *Lead Architect* at the blockchain company *Moka Blox* and *ResilientDB*. He also holds a Master of Science degree from Purdue University and a Master of Science (Research) degree from Indian Institute of Technology Madras. His current research focuses on attaining safe and efficient fault-tolerant consensus in distributed and blockchain systems. He also has published works that present efficient compiler optimizations and designs for parallel and distributed algorithms.

Jelle Hellings finished his graduate studies at the Eindhoven University of Technology, Netherlands in 2011, with a final research project focused on external memory algorithms for indexing trees and directed acyclic graphs. He then moved to Hasselt University, Belgium, where he did his doctoral research in the Databases and Theoretical Computer Science research group. He finished his doctoral research on the expressive power of query languages in 2018. Since the summer of 2018, Jelle is a Postdoc Fellow at UC Davis in the *Exploratory Systems Lab* led by Prof. Sadoghi. His current research focus is on the theoretical bounds of consensus protocols in malicious environments and, more general, on exploring new directions for replicated systems in malicious environment.

Sajjad Rahnama is a Ph.D. student at the Computer Science Department of the University of California Davis supervised by Prof. Sadoghi. He is also a member of Exploratory Systems Lab. He also works as the *System Designer* at the blockchain company called Moka Blox and is the main developer for ResilientDB. His current research focuses on secure transaction processing and designing global-scale fault-tolerant protocols, distributed systems, and their applications in blockchain. He holds a B.Sc. in Computer Science from Amirkabir University of Technology, Tehran, Iran. Prior to starting his Ph.D., he worked in several tech companies as a infrastructure engineer and developer.

Mohammad Sadoghi is an Assistant Professor in the Computer Science Department at UC Davis. He leads the ExpoLab research group with the aim to pioneer a distributed ledger that unifies secure transactional and real-time analytical processing (L-Store), all centered around a democratic and decentralized computational model (ResilientDB). He has co-founded a blockchain company called Moka Blox LLC, a ResilientDB spin-off. He has over 80 publications in leading database conferences/journals and 34 filed U.S. patents. He has co-authored a book entitled "Transaction Processing on Modern Hardware", Morgan & Claypool Synthesis Lectures on Data Management, and currently co-authoring a book entitled "Fault-tolerant Distributed Transactions on Blockchain", Morgan & Claypool series.

Acknowledgments

This tutorial is based on the outline of our upcoming book on fault-tolerant transaction processing on blockchains [26]. Furthermore, this tutorial is an evolution of a tutorial presented at Middleware 2019 [22]. We have updated that tutorial with new and upcoming techniques and insights.

This work is partially supported by the U.S. Department of Energy, Office of Science, Office of Small Business Innovation Research, under Award Number DE-SC0020455.

4. REFERENCES

[1] I. Abraham, S. Devadas, D. Dolev, K. Nayak, and L. Ren. Synchronous byzantine agreement with expected $\mathcal{O}(1)$ rounds, expected $\mathcal{O}(n^2)$ communication, and optimal resilience, 2018.

[2] M. J. Amiri, D. Agrawal, and A. E. Abbadi. CAPER: A cross-application permissioned blockchain. *PVLDB*, 12(11):1385–1398, 2019.

[3] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick. Hyperledger Fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, pages 30:1–30:15. ACM, 2018.

[4] P.-L. Aublin, R. Guerraoui, N. Knežević, V. Quéma, and M. Vukolić. The next 700 bft protocols. *ACM Trans. Comput. Syst.*, 32(4):12:1–12:45, 2015.

[5] P.-L. Aublin, S. B. Mokhtar, and V. Quéma. RBFT: Redundant byzantine fault tolerance. In *IEEE 33rd International Conference on Distributed Computing Systems*, pages 297–306. IEEE, 2013.

[6] A. Bessani, J. Sousa, and E. E. Alchieri. State machine replication for the masses with BFT-SMART. In *44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 355–362. IEEE, 2014.

[7] E. Brewer. CAP twelve years later: How the "rules" have changed. *Computer*, 45(2):23–29, 2012.

[8] M. Castro and B. Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.*, 20(4):398–461, 2002.

[9] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi. Towards scaling blockchain systems via sharding. In *Proceedings of the 2019 International Conference on Management of Data*, pages 123–140. ACM, 2019.

[10] R. A. DeMillo, N. A. Lynch, and M. J. Merritt. Cryptographic protocols. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pages 383–400. ACM, 1982.

[11] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan. BLOCKBENCH: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1085–1100. ACM, 2017.

[12] D. Dolev. Unanimity in an unknown and unreliable environment. In *22nd Annual Symposium on Foundations of Computer Science*, pages 159–168. IEEE, 1981.

[13] D. Dolev. The byzantine generals strike again. *J. Algorithms*, 3(1):14–30, 1982.

[14] D. Dolev and R. Reischuk. Bounds on information exchange for byzantine agreement. *J. ACM*, 32(1):191–204, 1985.

[15] D. Dolev and H. Strong. Authenticated algorithms for byzantine agreement. *SIAM J. Comput.*, 12(4):656–666, 1983.

[16] W. W. Eckerson. Data quality and the bottom line: Achieving business success through a commitment to high quality data. Technical report, The Data Warehousing Institute, 101communications LLC., 2002.

[17] M. El-Hindi, C. Binnig, A. Arasu, D. Kossmann, and R. Ramamurthy. BlockchainDB: A shared database on blockchains. *PVLDB*, 12(11):1597–1609, 2019.

[18] M. J. Fischer and N. A. Lynch. A lower bound for the time to assure interactive consistency. *Inform. Process. Lett.*, 14(4):183–186, 1982.

[19] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, 1985.

[20] S. Gilbert and N. Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News*, 33(2):51–59, 2002.

[21] G. Greenspan. Multichain private blockchain, 2015.

[22] S. Gupta, J. Hellings, S. Rahnama, and M. Sadoghi. An in-depth look of BFT consensus in blockchain: Challenges and opportunities. In *Proceedings of the 20th International Middleware Conference Tutorials*, pages 6–10. ACM, 2019.

[23] S. Gupta, J. Hellings, S. Rahnama, and M. Sadoghi. Proof-of-execution: Reaching consensus through fault-tolerant speculation, 2019.

[24] S. Gupta, J. Hellings, and M. Sadoghi. Brief announcement: Revisiting consensus protocols through wait-free parallelization. In *33rd International Symposium on Distributed Computing (DISC 2019)*, volume 146, pages 44:1–44:3. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019.

[25] S. Gupta, J. Hellings, and M. Sadoghi. Scaling blockchain databases through parallel resilient consensus paradigm, 2019.

[26] S. Gupta, J. Hellings, and M. Sadoghi. *Fault-Tolerant Distributed Transactions on Blockchains*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2020. (to appear).

[27] S. Gupta, S. Rahnama, J. Hellings, and M. Sadoghi. ResilientDB: Global scale resilient blockchain fabric. *PVLDB*, 13(6):868–883, 2020.

[28] S. Gupta, S. Rahnama, and M. Sadoghi. Permissioned blockchain through the looking glass: Architectural and implementation lessons learned. In *40th International Conference on Distributed Computing Systems*. IEEE, 2020.

[29] R. Harding, D. Van Aken, A. Pavlo, and M. Stonebraker. An evaluation of distributed concurrency control. *PVLDB*, 10(5):553–564, 2017.

[30] S. Harizopoulos, D. J. Abadi, S. Madden, and M. Stonebraker. OLTP through the looking glass, and what we found there. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 981–992. ACM, 2008.

[31] J. Hellings and M. Sadoghi. Brief announcement: The fault-tolerant cluster-sending problem. In *33rd International Symposium on Distributed Computing (DISC 2019)*, volume 146, pages 45:1–45:3. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019.

[32] J. Hellings and M. Sadoghi. Coordination-free byzantine replication with minimal communication costs. In *23rd International Conference on Database Theory*, volume 155, pages 17:1–17:20. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2020.

[33] T. N. Herzog, F. J. Scheuren, and W. E. Winkler. *Data Quality and Record Linkage Techniques*. Springer New York, 2007.

[34] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong. Zyzzyva: Speculative byzantine fault tolerance. In *Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles*, pages 45–58. ACM, 2007.

[35] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.

[36] J.-P. Martin and L. Alvisi. Fast byzantine consensus. *IEEE Trans. Depend. Secure Comput.*, 3(3):202–215, 2006.

[37] S. Moran and Y. Wolfstahl. Extended impossibility results for asynchronous complete networks. *Inform. Process. Lett.*, 26(3):145–151, 1987.

[38] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009.

[39] S. Nathan, C. Govindarajan, A. Saraf, M. Sethi, and P. Jayachandran. Blockchain meets database: Design and implementation of a blockchain relational database. *PVLDB*, 12(11):1539–1552, 2019.

[40] F. Nawab and M. Sadoghi. Blockplane: A global-scale byzantizing middleware. In *35th International Conference on Data Engineering (ICDE)*, pages 124–135. IEEE, 2019.

[41] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.

[42] T. C. Redman. The impact of poor data quality on the typical enterprise. *Commun. ACM*, 41(2):79–82, 1998.

[43] G. Taubenfeld and S. Moran. Possibility and impossibility results in a shared memory environment. *Acta Inform.*, 33(1):1–20, 1996.

[44] G. Wood. Ethereum: a secure decentralised generalised transaction ledger, 2016. EIP-150 revision.

[45] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham. HotStuff: BFT consensus with linearity and responsiveness. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 347–356. ACM, 2019.