Performance of Byzantine Fault Tolerant Blockchains

GARY SHAPIRO

BSc, MEng

Supervisor: Professor Vincent Gramoli
Associate Supervisor: Professor Alan Fekete

A thesis submitted in fulfilment of
the requirements for the degree of
Master of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

4 June 2025

Statement of Originality

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes. I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Gary Shapiro

Statement of Attribution

The research presented in this thesis was conducted under the supervision of Professor Vincent Gramoli at The University of Sydney, Australia. The primary results of this research were first introduced in the following publication:

(1) Shapiro, G., Natoli, C., & Gramoli, V. (2020, November). *The performance of Byzantine fault tolerant blockchains*. In *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)* (pp. 1-8). IEEE.

As the supervisor for the candidature upon which this thesis is based, I confirm that the authorship and attribution statements provided above are accurate.

Professor Vincent Gramoli

Statement of Acknowledgment

The research reported in this thesis was supported by the award of a Research Training Program scholarship to the Master of Philosophy Candidate.

Abstract

Blockchains have garnered significant attention, leading to a proliferation of systems available for adoption. However, choosing the right blockchain for a specific application remains challenging due to a lack of comparative analyses of core metrics such as throughput, latency, and scalability. While several studies have focused on performance evaluation, few address blockchains that are both efficient—avoiding complex Proof-of-Work cryptographic puzzles—and secure, achieving deterministic consensus despite Byzantine failures. This thesis evaluates the performance of three blockchains designed to handle such adversarial behaviors: Burrow, Quorum, and Red Belly Blockchain. For this purpose, we modified the Hyperledger Caliper benchmarking tool by addressing three key limitations: unnecessary overheads, online cryptographic signatures, and centralized clients. Our findings reveal the maximum send rate that Burrow and Quorum can sustain, and demonstrate that Red Belly Blockchain achieves up to 8 times higher throughput than the other blockchains.

Acknowledgements

"If we knew what we were doing, it would not be called research, would it?" - Albert Einstein.

This research journey has come to an end. While the final outcome differs from what I initially set out to achieve, such is the nature of research—it rarely unfolds exactly as planned. I am proud of what has been achieved.

I have much to be grateful for and many to thank. First and foremost, I am deeply grateful to Hashem for granting me the strength and ability to undertake this journey.

I would like to thank Prof Vincent Gramoli. Having undertaken this postgraduate research many years after completing my university studies and with a family and business to run, I know that I was not the ideal student. His academic excellence is exemplary and I really appreciate his guidance and patience throughout the process.

Speaking of ideal students, I would like extend my heartfelt thanks to my wonderful wife. She is my role model in all areas of life and academic research is no exception. Her love and support gets me through every day. My beautiful children also deserve thanks for filling my life with purpose and insipiring me to be the best I can be.

Thank you to my parents. They have been investing in me since the day I was born and have always encouraged me to push as hard as I can. Without their love and attention I would not have even taken my first step and certainly not taken this one.

At the university, I am especially grateful to Dr. Chris Natoli. His input and mentorship were instrumental in achieving what I have. I also extend my gratitude to all the members of the research group and to Prof. Irena Koprinska for her sagely advice.

Lastly, to those beyond the university, I would like to thank my business partner, colleagues and friends as well as my family for all their support and interest they have shown in my research journey.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Background

The plethora of blockchain proposals and their evaluation in isolation of one another raised questions on the suitability of a particular proposal for a targeted application. As of November 2024, there are approximately $9,916$ blockchains available[1] but very little information on their benefits in terms of throughput, latency and scalability. When this information is provided [1, 2, 3], it often results from tests run by the developers of the blockchain, in isolation of other blockchains, and within a very specific experimental environment.

As a result, the open source community [4] as well as the database community [5] have started designing benchmarks that would allow the performance comparison of various blockchains on the same grounds. The former attempt targets only blockchains offering smart contracts and prevents comparison against transaction-based blockchains (e.g. Bitcoin [6]). The latter attempt compares blockchains that are made inherently slow to cope with potentially malicious permissionless users (e.g. Ethereum [7]) to blockchains that cannot tolerate malicious participants (e.g., Hyperledger Fabric [8]).

Despite these preliminary efforts, to our knowledge there is no thorough evaluation of secure blockchain systems. In this thesis, we tackle this problem by thoroughly evaluating the performance of three blockchain systems that are secure in that they tolerate Byzantine failures, namely Burrow [9], Quorum [10] and Red Belly Blockchain [11]. These blockchain systems were precisely designed to either support financial workloads, offer scalability or

[1] Number of coins in existence according to https://coinmarketcap.com/coins/views/all/.

be efficient. Interestingly, evaluating the performance of blockchains led us to identify and remedy three major drawbacks that have plagued previous evaluations of blockchain systems.

1.2 Limitations of Existing Evaluations

Our analysis revealed that prior benchmarking efforts often suffer from methodological limitations that lead to inaccurate or biased performance measurements. These issues stem from centralized workload generation, unnecessary overheads introduced by deployment tools, and skewed metrics caused by cryptographic preprocessing. This section outlines these three key limitations and explains how they can significantly distort the perceived performance of blockchain systems.

1.2.1 Distributed Workload Generation

Some benchmarks have been designed in a centralized fashion, having a single client machine, potentially running multiple concurrent client software, to send requests to the blockchain nodes [12, 5]. As blockchains are distributed systems that aim at scaling to large numbers of participating nodes, it is crucial to generate a workload sufficiently high to stress test these numerous blockchain nodes. Without provisioning enough physical client resources, the risk is that an experiment may report performance that is misleadingly capped by the client resources rather than representing the true capacity of the blockchain itself.

1.2.2 Limiting Misleading Overheads

For the sake of simplifying deployment and monitoring, several blockchain systems come with packaging options that limit their access to physical resources. Caliper [4] requires a docker container for ease of deployment[2], yet containers are known to induce unnecessary overheads [13]. Distributed synchronization services help monitoring blockchains, like Zookeeper, but these services are known to bottleneck at the leader network interface, hence

[2] https://github.com/hyperledger/caliper/blob/master/packages/caliper-tests-integration/ethereum_tests/networkconfig.json.

offering results that could be misleading [14]. It is thus crucial to limit the effects of these levels of indirections on the observed performance.

1.2.3 Offline Cryptographic Preprocessing

The cryptographic functions to guarantee the authentication in a blockchain system were found particularly CPU-intensive at various occasions [11]. Part of these CPU-intensive tasks are produced on the client side: the benchmark should sign sufficiently many transactions to generate a workload. Benchmarks that measure the time needed to produce this CPU-intensive workload, like [4], return a biased throughput that does not represent the capacity of the blockchain system but that is limited by the time it took for the benchmark to sign the transactions to be sent.

1.3 Outline

In this thesis we address the aforementioned issues. We found that deploying up to 32 Amazon Web Service (AWS) c4.xlarge virtual machines (VMs) was sufficient to outline important differences between Byzantine Fault Tolerant (BFT) blockchains. Interestingly, we identified that Burrow and Quorum would fail at high workloads and, when not failing, would offer performance almost one order of magnitude lower than Red Belly Blockchain.

We do not pretend that our proposal aims at evaluating all blockchains. First, proof-of-work blockchains can be made arbitrarily slow in order to increase their security. Second, although proof-of-authority blockchains can be compared to BFT ones, they often require parameterizing the block period that dictates their performance as well. Third, we observed that many BFT blockchains are surprisingly too unstable to be evaluated. Finally, the performance evaluation of smart contracts is out of the scope of this thesis and left to future work.

The rest of this thesis is structured as follows. Chapter 2 provides background on blockchain systems and distributed architectures. Chapter 3 evaluates the performance of blockchain

systems, focusing on throughput, latency, and scalability across different configurations and workloads. Chapter 4 details the benchmarking methodologies, including adjustments to Caliper and the impact of distributed clients on performance. Chapter 5 compares the results across three blockchain systems: Burrow, Red Belly, and Quorum, highlighting their performance capabilities and limitations. Finally, Chapter 6 concludes and outlines future research directions.

CHAPTER 2

Literature review

Blockchain benchmarks are often developed and tailored specifically for one blockchain. It thus makes it hard to compare the performance of blockchains as the performance is usually obtained using different benchmarks. To solve this issue, a number of benchmarking frameworks have been developed to compare different blockchains or its consensus component on the same ground. In this section we first present a number of benchmarking frameworks. We then discuss the selection of distributed ledgers and finally analyse related works. Some of the works presented build upon our research while others form the basis upon which our research is built.

2.1 Benchmarking Frameworks

To enable fair and reproducible comparisons between blockchain systems, several benchmarking frameworks have been proposed. These frameworks aim to abstract away implementation-specific details while providing standardised metrics across different platforms. In this section, we present the most relevant benchmarking tools, highlight their supported features, and discuss their applicability to our research.

2.1.1 Caliper

Hyperledger Caliper [4] is a benchmarking framework aimed at measuring blockchain performance through defined use-cases and can be used in multiple systems. Caliper aims at providing a generic interface which can be implemented to extend the framework to other blockchain systems. This provides a foundation to achieve results that can be accurately

and fairly compared. Caliper supports Hyperledger's blockchains such as Fabric, Sawtooth, Iroha, Burrow and Besu. Caliper's pre-defined workloads specify which contract is called, which functions are used and the rate at which the transactions are sent. Caliper's architecture utilises a master-worker system, where worker clients submit RPC transactions to the blockchain nodes. The transactions are submitted asynchronously and upon reception increment a count and obtain the latency of the response. It outputs success rate, transaction and read throughput, transaction and read latency and resource consumption. At the time of implementation, Caliper has minimal support for distributed worker client machines.

2.1.2 Blockbench

Blockbench [5] provides a number of workloads ranging from blockchain specific use-cases to more traditional database system benchmarks such as YCSB [15]. Blockbench compares a crash fault tolerant blockchain (Hyperledger Fabric) to a BFT blockchain (Quorum) and to a proof-of-work blockchain (Ethereum). Our focus is on comparing exclusively BFT blockchains as they are appealing to critical applications.

2.1.3 Chainhammer

Chainhammer [12] provides a benchmark framework supporting only Ethereum-compatible blockchains, based on different consensus algorithms (geth clique [16], parity aura [17], and Quorum's [10] RAFT and IBFT). This benchmark provides metrics of transactions per second, block information. Chainhammer was used to experiment with blockchains under varying machine sizes, but not to test scalability.

2.1.4 BFT-Bench

BFT-Bench [18] measures the performance of various BFT State Machine Replications (SMRs) executing a sequence of consensus instances for distributed replicas to agree on a common state, similar to blockchain nodes agreeing on a block. However, SMRs do not support signature and verification of transaction requests that are necessary in blockchain

systems. These verifications are known to significantly impact blockchain performance [11], this is why we take these into account in our evaluation.

2.1.5 Diablo

Diablo [19] presents a comprehensive benchmark suite for evaluating blockchain performance. It addresses the challenge of selecting the right blockchain for specific applications due to the proliferation of blockchain proposals. The authors present an extensive evaluation of 6 state-of-the-art blockchains, demonstrating how experimental settings impact performance. Diablo provides a unifying framework for fair blockchain comparison, including 5 realistic Decentralized Applications (DApps) for testing. The study reveals that while certain optimizations can help handle demanding workloads, none of the tested blockchains can fully support the load of these realistic DApps.

2.1.6 BBSF

Blockchain Benchmark Standardized Format (BBSF) [20] introduces a framework designed to provide standardized, transparent, and fair benchmarks for blockchain platforms. BBSF allows users and developers to compare different blockchain platforms using metrics derived from realistic workloads. The authors discuss the challenges in developing such a comprehensive blockchain benchmark and present an implementation called Blockbench v3, which focuses on Web3 applications and workloads for layer-1 blockchains. The paper demonstrates that BBSF generates more verifiable results compared to previous benchmarking approaches and serves as an ongoing effort to characterize blockchain performance.

2.1.7 Simulators

BFTSim [21] simulates the execution of BFT consensus algorithms and runs on top of an `ns-2` network simulator. It allows researchers to rapidly test a BFT algorithm after writing it in a declarative specification language and simulates its cryptographic operations. BlockSim [22] aims at simulating different blockchains using a discrete-event model to scale

to a large number of nodes. BlockSim was used to simulate Ethereum and Bitcoin and demonstrated that the cost of encrypting communications is costly. Our evaluation does not rely on a specific model and assesses the performance a BFT blockchain's user can experience in a real environment.

2.2 Distributed Ledger Selection

As part of our research we considered and subsequently excluded a number of distributed ledgers.

2.2.1 Crash fault tolerant distributed ledgers

Hyperledger Fabric [8] and Corda [23] are two permissioned distributed ledger technologies. Unfortunately, there is no full fledged solution of these distributed ledgers offering Byzantine fault tolerance.

2.2.1.1 Hyperledger

Although a research prototype of the orderer of Hyperledger Fabric builds upon BFT-SMaRt and tolerates Byzantine behaviors, the full system of Fabric v1.x has not been designed towards this goal.

2.2.1.2 Corda

An initial version of Corda was originally designed to build upon BFT-SMaRt, however, this version is not stable and the developers recommend to use the version based on Raft that cannot tolerate intrusions [24].

2.2.2 Blockchains requiring synchrony

Other blockchain systems do not tolerate unexpected message delays. They assume synchrony [25] and an attacker may double spend if some messages get unexpectedly delayed [26]. These blockchains are thus vulnerable in large networks to attacks, natural disasters or human misconfigurations. Solida [27] assumes synchrony even though it builds upon PBFT [28]. Dfinity [26] is proved under synchrony but executes in asynchronous rounds. Avalanche [29] is analyzed under synchrony but is conjectured to work under partially synchrony. OmniLedger [30] needs synchrony to assign participants to shards.

2.2.3 Ethereum for permissioned settings

Even though Ethereum [7] is originally a proof-of-work blockchain, it has been deployed in permissioned environments, like the environment we consider. A key difference between Ethereum and the BFT blockchains we study here, is that Ethereum assumes synchrony [31] whereas the blockchains we consider here simply assume partial synchrony [25], hence the latter blockchains do not need to know the time it will take to deliver a message. As a result, the performance of Ethereum is driven by a difficulty parameter that dictates the expected period between two consecutive block creations, as was observed in previous evaluations of Ethereum in a permissioned context [32]. Proof-of-authority has been proposed in Ethereum as an alternative to avoid the difficulty parameter of proof-of-work. It was shown to offer a throughput of up to 84 transactions per second [33], however, it requires selecting a block period and it has proven vulnerable in partially synchronous settings [34].

2.3 Related Works

There have been a number of other works performing analysis on permissioned blockchains. Pongnumkul et al. [35] perform an analysis of Ethereum against Hyperledger Fabric in a permissioned context. The benchmark results utilize smart contract invocations to compare throughput and latency, highlighting that Hyperledger outperforms Ethereum with throughput

and latency. Leppelsak [36] presents a thesis performing analysis of private blockchains utilizing Hyperledger Caliper. The paper highlights Hyperledger Fabric's performance with varying levels of transaction requests and interacts with smart contracts. Similarly, Thakkar et al. [37] presents experimental analysis of Hyperledger Fabric and optimizations made to increase overall throughput.

At the time of implementation, Baliga et al [38] presented the most comparable performance analysis to ours, but utilizes Caliper to evaluate exclusively the Quorum blockchain. This analysis provides insights into how Quorum's configurations effect the overall throughput and latency, focusing directly on the impact of changes to Quorum. More recently, research published following our work has focused on developing comprehensive benchmark frameworks for evaluating blockchain systems. Diablo, a benchmark suite for blockchains, offers an extensive evaluation of six state-of-the-art blockchains and includes five realistic decentralized applications (DApps) for testing [19]. Other frameworks include BBSF, which aims for standardized, transparent, and fair benchmarks [20] and Gromit, which conducted a large-scale study of seven representative systems [39].

In this paper, we utilize the Quorum extension of Caliper to provide comparisons against other blockchains; namely Burrow and Red Belly Blockchain. We analyze how the performance of Quorum with default configurations compares against others with similar environmental setups.

CHAPTER 3

Methods and Results

In this section, we present the BFT blockchain systems that we evaluate and the ones we could not evaluate. A summary of the blockchains we evaluate can be seen in Table 3.1 where n is the number of nodes and f is the number of Byzantine failures they tolerate.

TABLE 3.1. Comparison of selected BFT blockchains

Blockchain	Consensus	Fault tolerance	Developer	Goals
Burrow [9]	Tendermint [40]	$n > 3f$	Hyperledger	Simplicity and speed[1]
Quorum [10]	IBFT [41]	$n > 3f$	J.P. Morgan	Financial applications[2]
Red Belly [11]	DBFT [42]	$n > 3f$	U. Sydney and CSIRO	Security and scalability[3]

3.1 Selected Distributed Ledgers

In this section, we provide a brief overview of the three selected blockchains—Burrow, Quorum, and Red Belly Blockchain—highlighting their key architectural features and the motivations for including them in our study.

3.1.1 Burrow

Burrow [9] is a blockchain optimized for the proof-of-stake setting. It does not run proof-of-work, which allows for higher performance and can run as a consortium or private blockchain that tolerates Byzantine failures. It supports both the Ethereum Virtual Machine (EVM)

[1] https://wiki.hyperledger.org/display/burrow/Hyperledger+Burrow.
[2] https://www.goquorum.com.
[3] https://gramoli.redbellyblockchain.io/web/doc/talks/facebook.pdf.

and the Web Assembly (WASM) programming language. Its builds upon the Tendermint algorithm [40] to reach consensus.

3.1.2 Quorum

Quorum [10] is a permissioned blockchain system that builds upon the Istanbul Byzantine Fault Tolerant (IBFT) consensus protocol [41]. Although the initial version of IBFT suffered from a liveness issue, some fixes were proposed [41]. Quorum targets financial applications and can treat EVM-based smart contracts.

3.1.3 Red Belly Blockchain

Red Belly Blockchain [11] builds upon the leaderless DBFT consensus protocol [42] that was designed for blockchains. Instead of having a leader imposing its proposal, it combines multiple proposals into a superblock to scale to large numbers of nodes. It spawns multiple instances of a binary consensus algorithm that was recently proved correct using model checking [43]. In addition to avoiding the leader, Red Belly Blockchain supports a UTXO model and exploits verification sharding to avoid having all nodes verifying the exact same sets of transactions in order to scale to hundreds of geo-distributed nodes. We interpret scalability as the ability for performance not to deteriorate as we increase the system size.

3.2 Unsupported blockchains

Most BFT blockchains are either no longer supported or their code was made proprietary, and offer at best a non-stable version.

3.2.1 Ethermint

Ethermint [44] is the combination of Tendermint, a BFT consensus algorithm [40], with the Ethereum Virtual Machine (EVM). At the time of implementation, it was in the pre-alpha development stage[4] and despite our efforts we could not deploy it on a distributed system.

3.2.2 Concord

Concord[5] is a permissioned blockchain that builds upon the Byzantine fault tolerant consensus algorithm SBFT [45] and is compatible with the EVM. The consensus algorithm was evaluated within a key-value store application and a blockchain application, however, the developers could not help us make the publicly available version of SBFT run on multiple machines. While it might be possible to deploy a proprietary version, its code is not available so we could not compare it to other blockchains.

3.2.3 SMaRtChain

SMaRtChain [46] presents a permissioned blockchain utilizing BFT-SMaRt with reconfigurable consensus to maintain state machine replication, showing promising results. The blockchain layer provides an example execution by supporting value transfer transactions and the consensus layer is compatible with state execution models such as the EVM. Although SMaRtChain provides a permissioned BFT blockchain, it does not provide methods to interact with the chain that can be utilized within Caliper's benchmark model.

3.3 Evaluation on a Distributed System

In this section we present the experimental results achieved using Burrow, Quorum and Red Belly Blockchain on a distributed system. To this end, we modified the Hyperledger Caliper benchmark to interact with Red Belly Blockchain and to overcome three limitations

[4] https://github.com/ChainSafe/ethermint.
[5] https://github.com/vmware/concord-bft

by (1) removing unnecessary overheads, (2) offloading cryptographic signatures and (3) multiplying clients. Section 3.3.1 describes the experimental settings, Section 3.3.2 presents the initial experiments run with Burrow and Red Belly utilising a single Caliper client machine, Section 3.3.3 introduces multiple Caliper client machines while Section 3.3.4 shows the comparative results achieved for Burrow, Red Belly and Quorum.

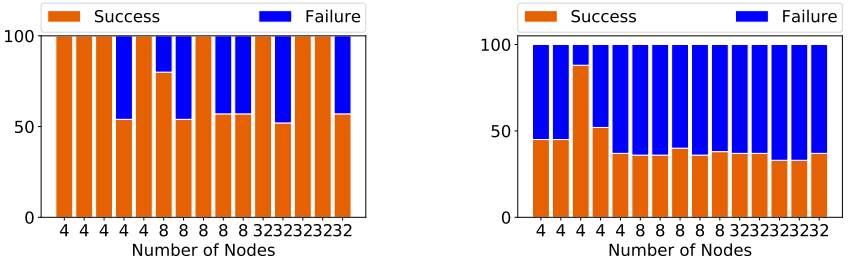
3.3.1 Experimental settings with low overheads

In the remainder, we refer to *client* as the benchmark runtime sending the transaction requests, and to *server* as the blockchain runtime receiving and treating the requests. We deployed each of the servers and clients on separate Amazon Web Services (AWS) EC2 virtual machines (VMs) physically located in the same AWS region. Each VM is a c4.xlarge instance with 4 Intel Xeon E5-2666v3 vCPUs running at 2.9 GHz with 7.5 GiB of memory, provided with "High" network performance as defined by Amazon, and equipped with Ubuntu 18.04 LTS and NodeJS v10.13.0. Each experiment is run at least 5 times for a duration of 20 seconds with a minimum send rate of 200 transactions per second (TPS). As a result each individually plotted data point in our graphs corresponds to the performance computed over at least 20,000 transactions. The error bars, when present, indicate the standard deviations.

In order to minimize the cryptographic overheads, we pre-generate the signatures of the transactions offline before collecting the statistics on the performance: once the transactions are pre-generated and correctly signed, the clients start sending them with specific *send rate* and measuring the performance of the blockchain servers. This decoupling between signing and evaluating the performance is necessary in order to avoid client bottlenecks induced by the CPU-intensive signing process at the client side. In order to minimize unnecessary overheads further, we do not make use of additional middleware or virtualization layer like Zookeeper or Docker containers.

(A) Throughput (B) Latency

FIGURE 3.1. 4, 8, and 32 Burrow nodes - 1 Caliper client machine



(A) Burrow - send rate 2,000 TPS (B) Burrow - send rate 3,000 TPS

FIGURE 3.2. Failed transactions for Burrow

3.3.2 A workload that does not overload the blockchain

We start our series of experiments with a simple configuration where we dedicate one VM located in Oregon to the benchmark client and between 4 and 32 VMs to the blockchain servers. This client VM sends all its requests to the same blockchain server during the entire experiment. In order to achieve fault tolerance, this server must communicate with other servers before the requests can be fully treated by the service.

3.3.2.1 The limited capacity of Hyperledger Burrow

Figure 3.1a depicts the performance of the Burrow blockchain with $n \in \{4, 8, 32\}$ blockchain servers or Burrow nodes. To this end, we measure the throughput—expressed as the number of transactions treated by the blockchain service per second (TPS)—as we increase the client send rate from 100 to 3000 TPS. We observe that the peak throughput of Burrow is 311 TPS,

which is achieved with a send rate of 500 TPS. When the send rate increases further, the throughput decreases as the server struggles to keep up with the send rate increase. We further observe that once the send rate reaches 2,000 TPS, Burrow begins to fail. Interestingly, we observed that a benchmark containing numerous rounds resulted in more frequent system crashes than one-round benchmarks.

Figure 3.2 depicts the corresponding failure rate as the percentage of failed transactions for each benchmark run with a send rate of 2,000 TPS or 3,000 TPS. At 2,000 TPS the blockchain handles the load most of the time at 4 server nodes but begins to fail most of the time at 8 and 32 nodes. Once the send rate reaches 3,000 TPS, the blockchain fails transactions on every run of the benchmark regardless of the number of nodes. Throughout the execution of the benchmark, we observed that a benchmark containing numerous rounds resulted in system crashes more frequently than one-round benchmarks.

3.3.2.2 Red Belly performance increases with the workload

We evaluated the Red Belly blockchain in a similar manner to the previous Burrow experiment, with one client sending requests to the same blockchain server. Figure 3.3 depicts the throughput of Red Belly as we increase the send rate with $n = 4$ server nodes. Unlike Burrow, Red Belly's highest throughput reaches 1,395 TPS and a corresponding average latency of 7.2s with a send rate of 2,000 TPS. As the throughput keeps increasing with the send rate, it is probably not the peak capacity of Red Belly. Moreover, the fact that no transactions fail confirms a previous observation that Red Belly Blockchain offers starvation freedom: every correctly submitted transaction gets eventually committed.

3.3.2.3 Adjusting the workloads

The previous results illustrated the problem of stress testing a single server node with a single client node. First, as the performance of Red Belly keeps increasing with the send rate, one can provision more client resources to identify faster the Red Belly peak capacity. Second, the instability of Burrow indicates that each benchmark should only be run with a single test

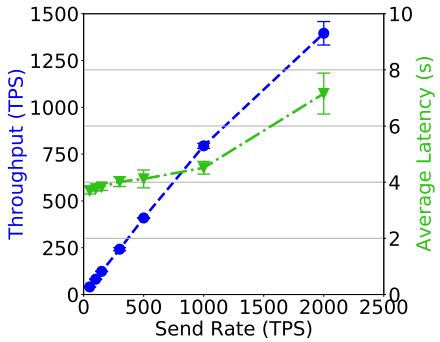
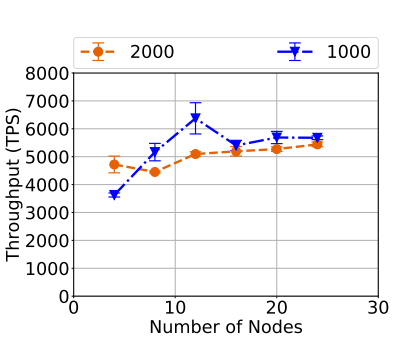
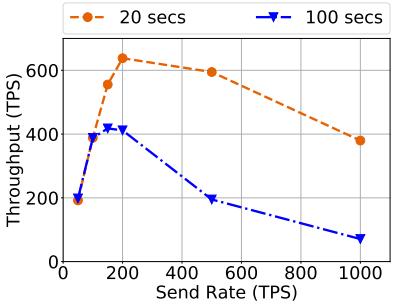


FIGURE 3.3. 4 Red Belly nodes - 1 Caliper client machine



(A) Red Belly - Throughput with varying send rate of 1000 and 2000 TPS



(B) 4 Burrow nodes - Throughput for the same send rate for varying duration of 20s and 100s

FIGURE 3.4. Equal number of blockchain nodes and Caliper client machines

round to minimize failures. This is the reason why we multiply the number of clients and balance their send rate to multiple servers in Section 3.3.3.

3.3.3 On the need for multiple benchmarking clients

We now explore whether higher throughput can be achieved by multiplying the client machines. As before, we experiment with Burrow and Red Belly. At the time of implementation, Caliper had minimal support for distributed worker client machines, so we first modified Caliper.

3.3.3.1 Caliper modifications

We wrote scripts to configure and launch multiple client instances on separate VMs and orchestrate the benchmark. One of the scripts launches the specified number of client and server nodes through AWS CLI. It then configures Caliper with network parameters and launches simultaneously the benchmark on multiple VMs. The benchmark was configured to run one test round with the simple benchmark configuration. The throughput is now computed as the sum of the throughputs reported by each client node.

3.3.3.2 The more clients, the higher the load

Figure 3.4a shows the performance of Red Belly Blockchain as the number of nodes increases for send rates of 1,000 and 2,000 TPS. The peak throughput of 6,375 TPS is achieved with 12 clients each sending 1,000 TPS to 12 Red Belly servers. Figure 3.4a shows that when the send rate of each client is 2,000 TPS the throughput is not as high, indicating that there is an optimal send rate before the client overloads the server.

Figure 3.4b shows the throughput of Burrow in two different length experiments with 4 servers and 4 clients as the send rate increases. The result indicates that for a sending rate above 100 TPS, Burrow achieves lower throughput in an experiment of 100s than in an experiment of 20s. This is mainly due to the blockchain system not being able to process transactions fast enough, which results in an increased number of failed transactions and hence a lower measured throughput.

3.3.3.3 Observation

Improvements during these experiments highlighted that better results are observed when using distributed Caliper clients, each sending simultaneous transactions to the blockchain servers. We observed that there exists an optimal send rate and optimal duration for a benchmark to obtain peak performance, as blockchain nodes may become overloaded with high send rates, and longer benchmarks may stack unprocessed transactions leading to memory issues or failed transactions.

These observations led to the experiment duration being standardized to 20 seconds, as well as utilizing multiple configurations to determine the optimal send rate prior to final metric gathering.

3.3.4 Comparing blockchain performance

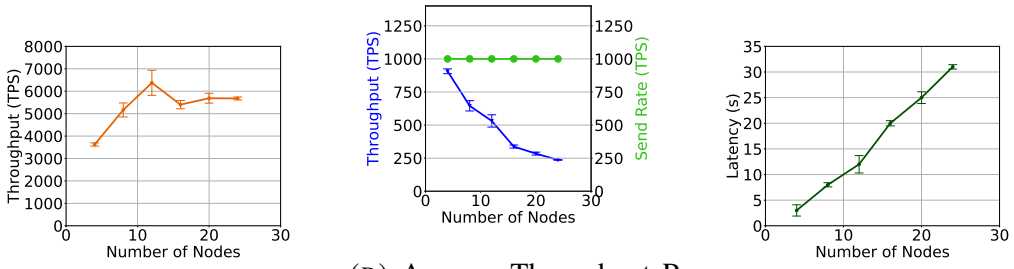
The next set of experiments compares the three blockchain systems namely, Burrow, Red Belly and Quorum. To this end, we deployed multiple Caliper client machines, each sending transactions simultaneously to the same number of blockchain servers at an optimal send rate with an increasing number of client and server nodes.

3.3.4.1 Large performance variations per blockchain

We compare the performance of the three blockchains, Burrow, Red Belly Blockchain and Quorum, by measuring their throughput, latency and scalability as the system enlarges. We first present isolated results indicating the total throughput, the throughput per server as we change the send rate and the latency for each blockchain (Figure 3.5, 3.6 and 3.7) before plotting the performance of the three blockchains on the same graphs (Figure 3.8).

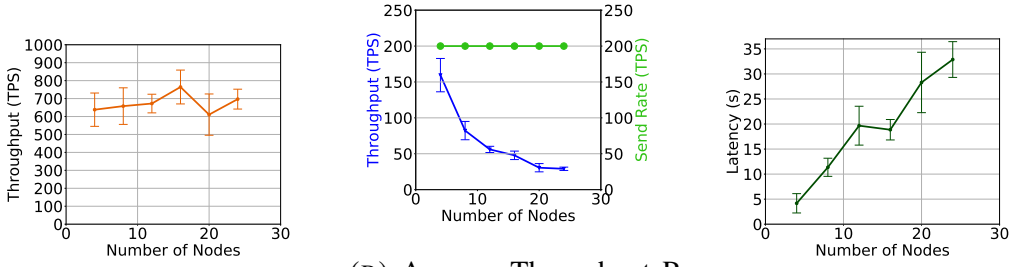
Figure 3.5 depicts the Red Belly performance with a send rate of 1,000 TPS. Figure 3.5a shows that the throughput increases as the number of nodes increases and eventually plateaus to a constant throughput. The peak throughput of 6,375 TPS is achieved with 12 Red Belly nodes and a corresponding latency of 12s. Although the total throughput increases, the average throughput per node decreases from 906 TPS down to 237 TPS as the number of nodes increases, as can be seen from Figure 3.5b. This is due to the superblock optimization of Red Belly that combines multiple proposals [11].

Figure 3.6 depicts the Burrow performance with a send rate of 200 TPS. Recall that we chose this send rate to maximize Burrow's throughput as Burrow fails under high loads as explained in Section 3.3.3. Figure 3.6a shows that the throughput remains relatively constant as the number of nodes increases. The peak throughput achieved at 16 nodes is 765 TPS with a



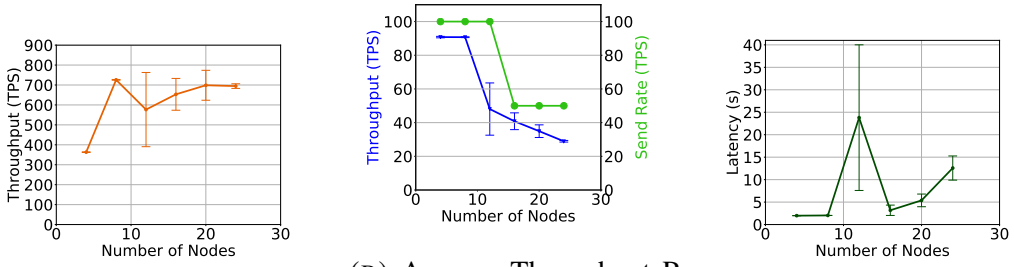
(A) Total Throughput (B) Average Throughput Per Node (C) Latency

FIGURE 3.5. Red Belly - send rate 1,000 TPS per Caliper client machine



(A) Total Throughput (B) Average Throughput Per Node (C) Latency

FIGURE 3.6. Burrow - send rate 200 TPS per Caliper client machine

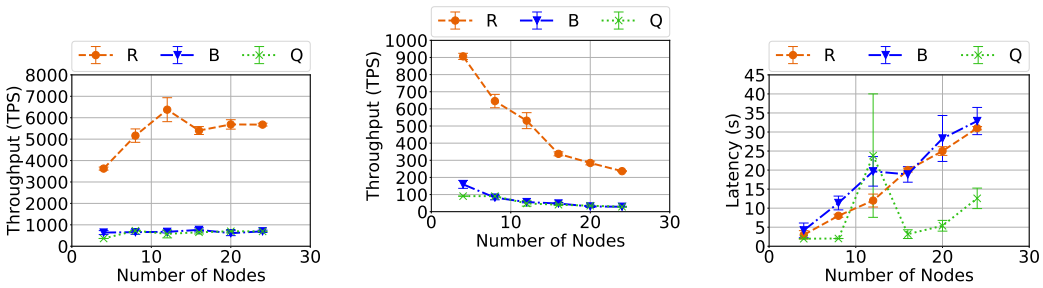


(A) Total Throughput (B) Average Throughput Per Node (C) Latency

FIGURE 3.7. Quorum - send rate 100 TPS per Caliper client machine

corresponding latency of 18.86s. Figure 3.6b shows that the average throughput per node decreases down to 29 TPS as the number of nodes increases.

Figure 3.7 depicts the Quorum performance. Figure 3.7b shows that we set the send rate to 100 TPS for 4, 8 and 12 nodes, however, we had to reduce it to 50 TPS as the blockchain system failed when the send rate was higher at more than 12 nodes. As can be seen from Figure 3.7c the latency peaked to 24s with 12 nodes. Figure 3.7a shows that the throughput



(A) Total Throughput (B) Average Throughput Per Node (C) Latency

FIGURE 3.8. Comparison of Red Belly (R), Burrow (B) and Quorum (Q)

TABLE 3.2. Comparison summary of the blockchains, when deployed on c4.xlarge AWS instances

Blockchain	Consensus	Peak throughput
Burrow [9]	Tendermint [40]	765 TPS
Quorum [10]	IBFT [41]	726 TPS
Red Belly [11]	DBFT [42]	6375 TPS

peaks at 725 TPS with 8 nodes before dipping to 576 TPS and then coming back up to a maximum of 694 TPS with the lower send rate of 50 TPS. Figure 3.7b shows the average throughput per node starts at 91 TPS with 4 nodes but drops sharply to 29 TPS with 24 nodes.

Figure 3.8 shows a comparison of all three blockchains side by side. Figure 3.8a and Figure 3.8b show that Red Belly achieves a peak total throughput of 6,375 TPS and an average throughput per node of 906 TPS. This is almost one order of magnitude greater than both Burrow and Quorum. Figure 3.8c shows that the latency for Red Belly and Burrow are relatively similar. This is due to only Red Belly exploiting superblocks. The latency achieved by Quorum is relatively lower, however there is a spike at 12 nodes before the send rate decreases. The results achieved in terms of throughput are relatively similar for Burrow and Quorum, as can be seen in Figure 3.8a and Figure 3.8b, although Quorum has a lower latency as can be seen in Figure 3.8c.

3.3.4.2 Red Belly Blockchain scalability

Although testing showed the optimum send rate for Quorum was 100 TPS, once the number of nodes increased to 16, the Quorum blockchain system failed and Caliper was unable to measure results. As such the send rate was altered to 50 TPS per Caliper machine to accommodate for the higher node numbers in the network. We observed that both Quorum and Burrow only managed to achieve a throughput of under 800 TPS while Red Belly was able to achieve a peak throughput of 6,375 TPS as depicted in Table 3.2.

CHAPTER 4

Conclusion

In this section we summarise the work undertaken and suggest direction for future work.

4.1 Summary

The research presented in this thesis provides a comprehensive evaluation of three secure blockchain systems that are Byzantine Fault Tolerant, namely Burrow, Quorum, and Red Belly Blockchain. By accounting for the limitations of existing benchmarking frameworks we were able to undertake a comparitive evaluation of these systems and identify significant differences in performance in terms of throughput, latency, and scalability. Red Belly Blockchain outperformed Burrow and Quorum, achieving peak throughput nearly an order of magnitude higher. These contributions pave the way for more accurate and fair comparisons of blockchain systems in real-world scenarios.

4.2 Future Direction

While our research has laid a foundation for understanding the performance of Byzantine Fault Tolerant blockchain systems, several avenues still remain for future exploration. A proposed next step would be to expand the evaluation to include a broader array of state-of-the-art blockchain systems. Such an analysis would offer a more comprehensive comparison across the field and help elucidate the trade-offs between performance and security inherent in different designs.

Future research could also focus on the evaluation of blockchain systems in simulated real-world workloads and practical application scenarios. Testing these systems in environments that mimic their intended use cases would provide valuable insight into their suitability for specific applications and help organizations select the most appropriate blockchain for their needs. In addition, long-term performance and stability analysis could provide critical information on how these systems perform over extended periods, which is vital for real-world deployments requiring sustained reliability.

By pursuing these directions, future work can build on our findings to advance our understanding of blockchain performance and scalability. This knowledge will be instrumental for both researchers and application developers, ultimately fostering the broader adoption of efficient and robust blockchain technologies.

Bibliography

[1] Marco Cavicchioli. *EoS in the lead for TPS thanks to the new testnet*. 2020. URL: https://en.cryptonomist.ch/2020/02/12/eos-tps-testnet/.

[2] Stephen O'Neal. *Who Scales Best? Inside Blockchains' Ongoing Transactions-Per-Second Race*. 2019. URL: https://cointelegraph.com/news/who-scales-it-best-inside-blockchains-ongoing-transactions-per-second-race.

[3] Unita. *QtumX Reaches 10,000 TPS in Benchmark Tests*. 2019. URL: https://blog.qtum.org/qtumx-reaches-10-000-tps-in-benchmark-tests-cee6452166fd.

[4] Hyperledger Foundation. *Hyperledger Caliper*. 2019. URL: https://hyperledger.github.io/caliper/.

[5] Tien Tuan Anh Dinh et al. 'BLOCKBENCH: A Framework for Analyzing Private Blockchains'. In: *SIGMOD*. 2017, pp. 1085–1100. ISBN: 9781450341974.

[6] Satoshi Nakamoto. *Bitcoin: a peer-to-peer electronic cash system*. 2008. URL: http://www.bitcoin.org.

[7] Gavin Wood. *ETHEREUM: A Secure Decentralised Generalised Transaction Ledger*. Yellow paper. 2015.

[8] Elli Androulaki et al. 'Hyperledger fabric: a distributed operating system for permissioned blockchains'. In: *EuroSys*. 2018, pp. 1–15.

[9] Casey Kuhlman et al. *Hyperledger Burrow (formerly eris-db)*. Mar. 2017. URL: https://www.hyperledger.org/wp-content/uploads/2017/06/HIP_Burrowv2.pdf.

[10] JP Morgan Chase. *Quorum whitepaper*. Aug. 2018. URL: https://github.com/jpmorganchase/quorum/blob/master/docs/Quorum%20Whitepaper%20v0.2.pdf.

[11] Tyler Crain et al. *Evaluating the Red Belly Blockchain*. Tech. rep. 1812.11747. arXiv, 2018.

[12] Andreas Krüger. *Chainhammer: Ethereum Benchmarking*. 2017. URL: https://github.com/drandreaskrueger/chainhammer.

[13] Christian Gorenflo et al. 'FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second'. In: *IEEE ICBC*. 2019, pp. 455–463.

[14] Patrick Hunt et al. 'ZooKeeper: Wait-Free Coordination for Internet-Scale Systems'. In: *USENIX ATC*. 2010, p. 11.

[15] Brian F. Cooper et al. 'Benchmarking cloud serving systems with YCSB'. In: *ACM SOCC*. Ed. by Joseph M. Hellerstein et al. 2010, pp. 143–154.

[16] Peter Szilagyi. *Clique Proof-of-Authority Consensus Protocol*. 2017. URL: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-225.md.

[17] Parity. *Parity Aura: Authority Round*. URL: https://openethereum.github.io/wiki/Aura.

[18] Divya Gupta et al. 'BFT-Bench: A Framework to Evaluate BFT Protocols'. In: *ACM/SPEC ICPE*. 2016, pp. 109–112.

[19] Vincent Gramoli et al. 'Diablo: A benchmark suite for blockchains'. In: *Proceedings of the Eighteenth European Conference on Computer Systems*. 2023, pp. 540–556.

[20] Kunpeng Ren et al. 'BBSF: blockchain benchmarking standardized framework'. In: *Proceedings of the 1st Workshop on Verifiable Database Systems*. 2023, pp. 10–18.

[21] Atul Singh et al. 'BFT Protocols Under Fire.' In: *NSDI*. Vol. 8. 2008, pp. 189–204.

[22] C. Faria and M. Correia. 'BlockSim: Blockchain Simulator'. In: *IEEE Blockchain*. 2019, pp. 439–446. DOI: 10.1109/Blockchain.2019.00067.

[23] Richard Gendal Brown et al. 'Corda: An Introduction'. In: *R3 CEV, August* (2016).

[24] Runchao Han et al. 'On the Performance of Distributed Ledgers for Internet of Things'. In: *Internet of Things* (Aug. 2019).

[25] Cynthia Dwork et al. 'Consensus in the Presence of Partial Synchrony'. In: *J. ACM* 35.2 (Apr. 1988).

[26] Timo Hanke et al. *DFINITY Technology Overview Series, Consensus System*. Tech. rep. 1805.04548. arXiv, May 2018.

[27] Ittai Abraham et al. 'Solida: A Blockchain Protocol Based on Reconfigurable Byzantine Consensus'. In: *OPODIS*. 2017, 25:1–25:19.

[28] Miguel Castro and Barbara Liskov. 'Practical Byzantine Fault Tolerance and Proactive Recovery'. In: *ACM Trans. Comput. Syst.* 20.4 (Nov. 2002), pp. 398–461. ISSN: 0734-2071.

[29] Team Rocket. *Snowflake to Avalanche: A Novel Metastable Consensus Protocol Family for Cryptocurrencies*. Unpublished manuscript. 2018.

[30] Eleftherios Kokoris-Kogias et al. 'OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding'. In: *IEEE S&P*. 2018, pp. 583–598.

[31] Christopher Natoli and Vincent Gramoli. 'The Balance Attack or Why Forkable Blockchains Are Ill-Suited for Consortium'. In: *IEEE/IFIP DSN*. 2017, pp. 579–590.

[32] Schäffer M. *Performance and Scalability of Smart Contracts in Private Ethereum Blockchains*. 2019.

[33] Fátima Leal et al. 'Performance Evaluation of Private Ethereum Networks'. In: *SN Computer Science* 1.285 (2020). DOI: https://doi.org/10.1007/s42979-020-00289-7.

[34] Parinya Ekparinya et al. 'The Attack of the Clones against Proof-of-Authority'. In: *NDSS*. Internet Society, Feb. 2020.

[35] S. Pongnumkul et al. 'Performance Analysis of Private Blockchain Platforms in Varying Workloads'. In: *Proceedings of the 26th International Conference on Computer Communication and Networks (ICCCN)*. 2017, pp. 1–6.

[36] Hendrik Folke Leppelsack. *Experimental Performance Evaluation of Private Distributed Ledger Implementations*. 2018.

[37] P. Thakkar et al. 'Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform'. In: *IEEE MASCOTS*. 2018, pp. 264–276.

[38] Arati Baliga et al. *Performance Evaluation of the Quorum Blockchain Platform*. Tech. rep. 1809.03421. arXiv, 2018.

[39] Bulat Nasrulin et al. 'Gromit: Benchmarking the performance and scalability of blockchain systems'. In: *2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. IEEE. 2022, pp. 56–63.

[40] Ethan Buchman et al. *The latest gossip on BFT consensus*. Tech. rep. 1807.04938. arXiv, Nov. 2019. URL: http://arxiv.org/abs/1807.04938.

[41] Roberto Saltini and David Hyland-Wood. *Correctness Analysis of IBFT*. Tech. rep. 1901.07160v2. arXiv, Aug. 2019.

[42] Tyler Crain et al. 'DBFT: Efficient Leaderless Byzantine Consensus and its Applications to Blockchains'. In: *IEEE NCA*. 2018.

[43] Pierre Tholoniat and Vincent Gramoli. 'Formal Verification of Blockchain Byzantine Fault Tolerance'. In: *6th Workshop on Formal Reasoning in Distributed Algorithms (FRIDA'19)*. Oct. 2019. URL: https://gramoli.redbellyblockchain.io/web/doc/pubs/frida19.pdf.

[44] Interchain foundation. *A Beginner's Guide to Ethermint*. 2017. URL: https://blog.cosmos.network/a-beginners-guide-to-ethermint-38ee15f8a6f4.

[45] Guy Golan-Gueta et al. 'SBFT: A Scalable and Decentralized Trust Infrastructure'. In: *IEEE/IFIP DSN*. 2019, pp. 568–580.

[46] Alysson Bessani et al. 'From Byzantine Replication to Blockchain: Consensus is Only the Beginning'. In: *IEEE/IFIP DSN*. 2020, pp. 424–436.