Impact of Network Topologies on Blockchain Performance

Vincenzo P. Di Perna
University of Urbino
Urbino, Italy
vincenzo.diperna@unicam.it

Marco Bernardo
University of Urbino
Urbino, Italy
marco.bernardo@uniurb.it

Francesco Fabris
University of Trieste
Trieste, Italy
ffabris@units.it

Sebastiao Amaro
IST U. Lisboa and INESC-ID
Lisbon, Portugal
sebastiao.amaro@tecnico.ulisboa.pt

Miguel Matos
IST U. Lisboa and INESC-ID
Lisbon, Portugal
miguel.marques.matos@tecnico.ulisboa.pt

Valerio Schiavoni
University of Neuchâtel
Neuchâtel, Switzerland
valerio.schiavoni@unine.ch

Abstract

Since blockchains are increasingly adopted in real-world applications, it is of paramount importance to evaluate their performance across diverse scenarios. Although the network infrastructure plays a fundamental role, its impact on performance remains largely unexplored. Some studies evaluate blockchain in cloud environments, but this approach is costly and difficult to reproduce. We propose a cost-effective and reproducible environment that supports both cluster-based setups and emulation capabilities and allows the underlying network topology to be easily modified. We evaluate five industry-grade blockchains – Algorand, Diem, Ethereum, Quorum, and Solana – across five network topologies – fat-tree, full mesh, hypercube, scale-free, and torus – and different realistic workloads – smart contract requests and transfer transactions. Our benchmark framework, Lilith, shows that full mesh, hypercube, and torus topologies improve blockchain performance under heavy workloads. Algorand and Diem perform consistently across the considered topologies, while Ethereum remains robust but slower.

CCS Concepts

• **Networks** → **Network performance analysis**; **Topology analysis and generation**; • **General and reference** → **Measurement**; **Validation**.

ACM Reference Format:
Vincenzo P. Di Perna, Marco Bernardo, Francesco Fabris, Sebastiao Amaro, Miguel Matos, and Valerio Schiavoni. 2025. Impact of Network Topologies on Blockchain Performance. In *The 19th ACM International Conference on Distributed and Event-based Systems (DEBS '25), June 10–13, 2025, Gothenburg, Sweden.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3701717.3730540

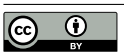
1 Introduction

A blockchain is a distributed, tamper-proof ledger system that permanently records transactions across a network of possibly untrusted nodes. With the growing adoption of the blockchain technology in various domains – such as digital currencies, finance, supply chains, and healthcare – an increasing number of studies have been conducted to explore its diverse aspects, including the properties of consensus protocols [41, 55, 59, 66, 70] and the system transaction processing capabilities in terms of transactions per second (TPS) [38]. Alongside these studies, various benchmark tools [17, 30, 36, 40, 46, 49, 60, 65] are available to validate the performance and behavior of blockchain systems.

Despite ongoing efforts, systematic and reproducible benchmarking of blockchains remains an open problem. On the one hand, large-scale peer-to-peer systems like blockchain networks are inherently complex and dynamic, with vast, constantly shifting, and unpredictable topologies. For instance, the Bitcoin network alone consists of over 20,000 full nodes [24]. On the other hand, when conducting large-scale, long-term experiments on cloud platforms, one can incur substantial costs. We estimate an average expense ranging from 2,000 to 14,000 USD for maintaining up to 200 high-performance nodes continuously operating 24/7 across the four major cloud providers: *Amazon Web Services* (AWS), *Google Cloud Platform* (GPC), *Azure*, and *Alibaba* (see Fig. 1).

Moreover, cloud platforms obscure the details of the underlying network topology and offer limited flexibility in adjusting network properties to accommodate variations in experimental configurations (*e.g.,* specifying packet drop rates, ad-hoc latencies, bandwidth capacities, *etc.*), while the inherent network variability – such as increased bandwidth, reduced latencies, dynamic node removal – further compromises the reproducibility of experiments, both within the same experimental session and over longer timeframes. For instance, Fig. 2 shows significant variations of performance among AWS regions across different continents with reference to 2023 and 2024 (see the caption for AWS region specifics). Our analysis in Fig. 3 for 10 AWS regions in April, July, and November 2023, and March 2024, shows notable daily fluctuations in network links. Compared with dataset from 2023 [42], we observed a 3% decrease in average latency (from 194 to 188 ms) and an 85.5% improvement in average throughput (from 74.6 to 138.3 Mbps).

Validating the performance and behavior of a blockchain by using benchmark tools is fundamental, especially given the significant discrepancies between the performance claims made by blockchain providers and the observed results. For instance, Solana declares a throughput of 200,000 TPS and a sub-second time for block finality [76], but independent researchers observed only 8,845 TPS with an average latency above 12 seconds [42]. These discrepancies highlight the importance of developing accurate benchmark tools for validating and testing such claims.



DEBS '25, Gothenburg, Sweden

ACM ISBN 979-8-4007-1332-3/25/06
https://doi.org/10.1145/3701717.3730540

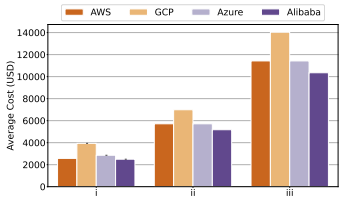


Figure 1: Cost comparison of four cloud platforms and three instance types: (i) high, (ii) low, (iii) medium performance.

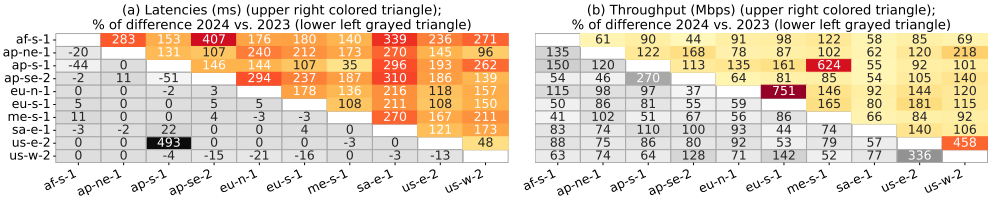


Figure 2: Latencies (ms) and throughput (Mbps) heatmaps of our 2024 measurements (upper colored triangles); % of difference 2024 vs. 2023 (lower grayed triangles). The 10 AWS regions are: *af-s-1* (Cape Town), *ap-ne-1* (Tokyo), *ap-s-1* (Mumbai), *ap-se-2* (Sydney), *eu-n-1* (Stockholm), *eu-s-1* (Milan), *me-s-1* (Bahrain), *sa-e-1* (Sao Paulo), *us-e-2* (Ohio), and *us-w-2* (Oregon).

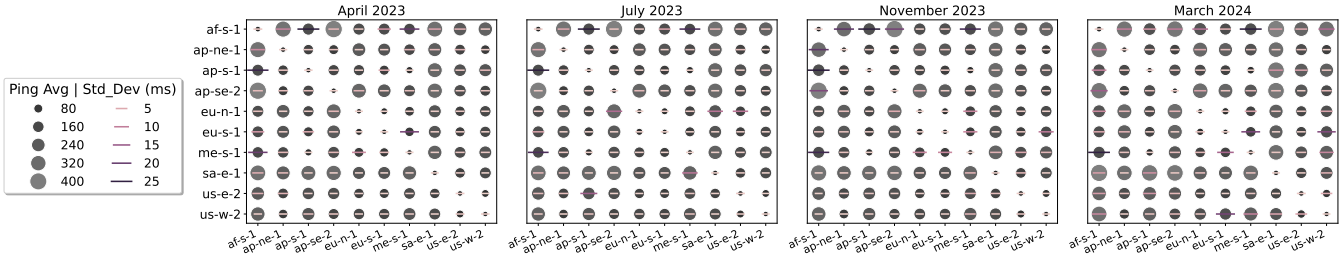


Figure 3: Four snapshots of monthly latency variations and standard deviation across the 10 AWS regions from Fig. 2.

An essential aspect of benchmarking blockchains is to control the underlying network topology [18, 48], which should be a best practice in blockchain evaluation to study and improve performance. Despite the significant impact of network dynamics (*e.g.,* changes in link properties, node failures, *etc.*), few studies have examined their influence on blockchain performance, with notable examples including the effect of network latency on key parameters (*e.g.,* block size, frequency, or propagation [29, 58]) as well as node failures and network contraction (*e.g.,* China's 2021 ban on Bitcoin mining [63]) and their consequences on the network [64].

Our research aims to validate cluster as a cost-effective, reproducible environment for the study of the impact of network topologies on blockchain performance. Aside from prior work on hypercube topology benefits for Bitcoin [71], to the best of our knowledge our work is the first in-depth experimental study across multiple real-world blockchains.

In this paper we measure the performance of five industry-grade blockchain systems – *Algorand* [41], *Diem* [22], *Ethereum* [74], *Quorum* [31], and *Solana* [76] – across five network topologies – *fat-tree*, *full mesh*, *hypercube*, *scale-free*, and *torus* – under different workloads – *smart contract* requests and transfer transactions. Although blockchains often adopt topologies suited to their design, such as fat-tree or hypercube for private ones like Diem and scale-free for public systems like Ethereum, this study explores blockchain topology reconfigurations to improve efficiency without altering their public or private nature. Here are our key contributions:

- We show, via a 12-month-long network trace of cloud performance monitoring (released at https://zenodo.org/records/11457020), that cloud-based deployments must account for varying and challenging network conditions.
- We release several ready-to-use network topologies and traces, enabling practitioners to experiment with distributed systems beyond our blockchain measurement study.
- The main conclusions are: (i) Algorand and Diem show consistent performance across various network topologies. (ii) Ethereum has the lowest TPS rate but remains resilient to network issues, *i.e.,* packet loss, link congestion, node crash, and increasing latency. (iii) Full mesh and hypercube topologies improve performance across all tested blockchains. (iv) Torus topology excels under heavy workloads. (v) Increasing the number of nodes reduces commit rate and raises block latency, while higher network bandwidth has no effect on latency. (vi) Quorum is the blockchain more affected by network dynamics.

For our measurements we built Lilith, a framework integrating *Diablo* [42], a benchmark tool for blockchain, and *Kollaps* [18], a distributed network topology emulator. The goal is to achieve reproducibility by having control over the network topology.

Roadmap. §2 describes the state of the art of blockchain benchmark tools. §3 outlines the architecture of Lilith. §4 explains our evaluation setup. §5 describes our results. §6 discusses the results and limitations of our approach. Finally, §7 concludes the paper.

2 Background and Related Work

A blockchain is a distributed ledger that maintains a cryptographically linked list of blocks. Each block contains transactions and references to the previous block, forming an immutable record chain. Consensus mechanisms achieve agreement on the block order without a centralized trusted party. There are several consensus mechanisms, *e.g., Proof of Work* (PoW) [55], *Proof of Stake* (PoS) [59], *Proof of Authority* (PoA) [70], as well as consensus based on classical *Byzantine Fault Tolerance* (BFT) [25] and its variants [41, 66, 78, 80]. Blockchains can execute smart contracts [74], which are specialized programs designed to automatically execute predefined actions when certain conditions are met. There are many types of blockchains which can be categorized as: (i) *public*, where any node can join, transact, and validate transactions; (ii) *private*, for

authorized entities and governed by a central authority; and (iii) *consortium*, managed by collaborating organizations. Our measurements compare the impact of network topology on five blockchains: three public ones (Algorand [41], Ethereum [74], and Solana [76]), a private one (Diem [22]), and a consortium one (Quorum [31]).

Table 1 compares existing blockchain benchmark tools across several dimensions, including network experiment capabilities, centralized vs. distributed orchestration, and simulation vs. emulation working mode. We also include the benchmark criteria from [65]: reproducibility (consistent results with the same test setup), versatility (identical experiments across diverse infrastructures), observability (insights into performance metrics and system impact), portability (to facilitate comparisons between distinct blockchain systems), ease to run (sharing configurations and deployment details in a widely adopted format like YAML across different testbeds), and realistic workload (evaluating scenarios that mimic real-world conditions). We use ● and ○ for the presence/absence of a feature, respectively, on the basis of the literature, ⁘ when no valid reference is found, and ◑ for tools that address limited network features.

Evaluating large networks is challenging due to limited computational resources, which often restrict evaluations to consensus processes, failing to capture full system behavior [73, 75]. Tools like *Minichain* [75] and *BlockLite* [73] emulate only PoW. To achieve more realistic results, other tools [16, 34, 35] focus on adjusting transaction/block structure, algorithms for wallet creation, and message signing. Some tools allow for dynamic events during the execution of the simulation [16]. Other tools [35, 62] do not run the real code and hence fail to precisely capture the load on the different resources, such as the transaction processing data structures. *SimBlock* [20] employs an event-driven model, distinguishing itself from real-world blockchains by calculating block creation time based on the success probability of block generation. Unlike traditional blockchains, where each node independently determines block generation difficulty, SimBlock assigns the same mining difficulty level to all nodes.

We stress that none of the listed tools simulate the entire set of features of a blockchain. For instance, BlockSim [16] assumes that miners include as many transactions in each block as possible, whereas in actual systems miners may generate a block with only a few transactions or even none at all. Moreover we observe a lack of standardization, as each tool reports different system aspects. SimBlock [20] treats block generation and message transmission as separate events, removing the need to replicate mining power. BlockLite [73] performs the actual PoW workload by solving a cryptographic puzzle. *HIVE* [14], an Ethereum test environment, cannot inspect the client internal state. Hyperledger *Caliper* [45] does not provide functionalities for resource reservation.

Some blockchain benchmark tools offer more flexible and feature-rich options. *BlockBench* [30] and *Diablo* [42] leverage ad-hoc scripts to manage the deployment of a blockchain but lack abstractions over their targeted testbed. However, despite recents efforts [43, 47], such an approach does not capture the underlying behavior of a wide-area network. *BCTMark* [65] ensures cross-infrastructure portability. It defines the network structure with a YAML topology file, streamlining node initialization. *COCONUT* [40] let users define and adapt different blockchain parameters, including consensus

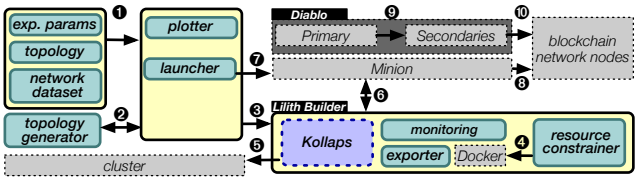


Figure 4: Architecture and execution flow of a Lilith experiment (dashed contour lines represent existing components).

algorithms, transaction structures, distribution of network components, *etc.*. *Core-Bitcoin Network Simulator* [17] can automatically tune mining difficulty, leveraging the *Pumba* network emulator [2], with built-in result analysis tools (*e.g.,* automatic plotting). *Boston Blockchain Benchmarking* (BBB) [60], *JABS* [77], and *DLPS* [36] address limitations in earlier approaches.

Our tool Lilith aims to provide a precise and feature-rich benchmarking standard, enabling testing across multiple blockchain systems while allowing for variations in network configurations. Its decentralized experiment orchestration minimizes interference between nodes, enhancing the accuracy and consistency of evaluations while ensuring reliable performance insights.

3 Lilith Overview

Lilith is a blockchain benchmark framework that we developed to support multiple deployment environments including clusters. It integrates, extends, and coordinates Diablo's workflow (see §3.1) and Kollaps's network emulation capabilities (see §3.2). It features a topology generator that creates *ad-hoc* network topologies, which are then fed into the Kollaps module for precise emulation. Additionally, Lilith can enforce arbitrary resource constraints, similar to selecting a specific instance in the cloud. Fig. 4 illustrates the execution flow of Lilith, which we detail in the following sections.

3.1 The Diablo Benchmark Suite

We selected Diablo [42] as a building block since it is the one matching most of the features in Table 1 and supports the different blockchain types discussed in §2. An experiment coordinator called *primary* manages a distributed workload generation mechanism between *secondary* nodes interacting with the specific blockchain via a dedicated client interface, ensuring synchronized evaluations. Unlike many other tools that require an existing blockchain infrastructure, along with the addresses of its nodes, the latest version of Diablo [42] introduces a set of Perl scripts called *Minion* [7], which automate the process of building the infrastructure. Diablo can inject realistic workloads, *e.g.,* smart contracts and transfer transactions, with varying volumes and complexities, and supports multi-region AWS deployments.

3.2 The Kollaps Network Emulator

To assess the impact of network properties on blockchain systems we leverage Kollaps [18], a decentralized network emulator for large-scale applications that models end-to-end properties (*e.g.,* latency, bandwidth, and packet loss). Its fully distributed model

Table 1: Comparison of several blockchain benchmark tools (◌ = not found; ○ = no match; ● = match; ◑ = partial match; Centr. = Centralized; Distr. = Distributed; Simul. = Simulation; Emul. = Emulation; SC = Smart Contract; TT = Transfer Transactions).

Tool	Year	Network settings	Orchestration	Mode	Reproducibility	Versatility	Observability	Portability	Ease to run	Workload
Shadow-Bitcoin [54]	2015	◌	Centr.	Simul.	○	○	○	○– Bitcoin	○	TT
HIVE [14]	2016	◌	Distr.	Simul.	○	○	○	○– Ethereum	○	◌
Simcoin [3]	2016	◌	Centr.	Simul.	○	○	●	○– Bitcoin	○	TT
Bitcoin-Simulator [39]	2016	◑	Centr.	Simul.	○	○	○	●– Bitcoin-like	○	TT
BlockBench [30]	2017	◌	◌	Simul.	○	●	○	●– Private	○	SC
Caliper [45]	2018	◑	Distr.	Emul.	○	○	●	●– Ethereum,Hyperledger	○	SC
Minichain [75]	2019	◑	Centr.	Emul.	○	○	○	●	○	◌
BlockLite [73]	2019	◌	Centr.	Emul.	○	○	○	●– PoW-public	○	◌
BlockSim-f [34]	2019	◑	Centr.	Simul.	○	○	○	●	○	◌
SimBlock [20]	2019	◑	Centr.	Simul.	○	○	○	●	○	TT
BlockSim-m [16]	2020	◑	Centr.	Simul.	○	○	○	●	○	TT
Core-Bit-Netw-Simul. [17]	2020	◑	Centr.	Simul.	○	○	○	●	○	◌
BBB [60]	2020	◑	Centr.	Emul.	○	○	○	●– Private	○	◌
SIMBA [35]	2020	◑	Centr.	Simul.	○	○	○	●	○	◌
BCTMark [65]	2020	◑	Distr.	Emul.	○	○	●	●	○	TT
BlockPerf [62]	2021	○	Distr.	Both	○	○	○	○– Bitcoin	○	TT
DLPS [36]	2021	◑	Distr.	Emul.	○	○	●	●	○	TT
Gromit [56]	2022	◑	Distr.	Emul.	○	○	●	●	○	SC
Diablo v2 [42]	2022	○	Distr.	Emul.	●	●	●	●	●	TT,SC
JABS [77]	2023	◑	Centr.	Simul.	○	○	○	●	○	TT
COCONUT [40]	2023	◌	Distr.	Emul.	●	●	○	●	●	TT,SC
GFBE [50]	2024	○	Distr.	Emul.	●	●	●	●	●	SC
STABL [43]	2024	◑	Distr.	Emul.	○	●	●	●	●	TT,SC
Lilith	2024	●	Distr.	Emul.	●	●	●	●	●	TT,SC

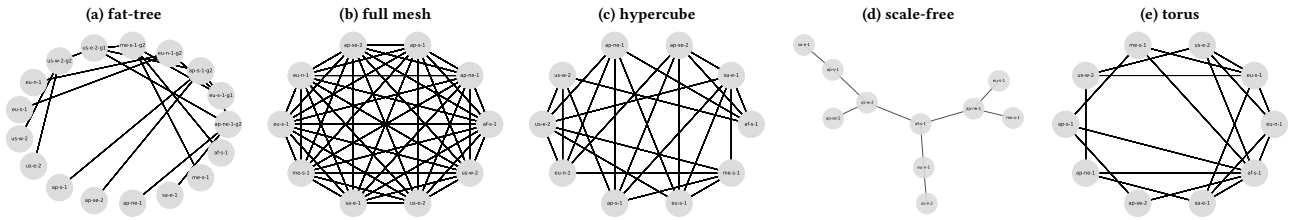


Figure 5: Real-world network topologies served by Lilith.

ensures scalability while supporting dynamic changes to the topology, *e.g.,* link or node removals, service disruptions, *etc.*. The routing paths of user-level data flows on the emulated topologies can be dynamically determined – as in the link-state protocol Open Shortest Path First (OSPF) that calculates routes based on the shortest path tree – based on criteria such as latency or hop count. Kollaps supports native processes, virtual machines, and can directly integrate with container orchestrators like *Docker Swarm* and *Kubernetes*. Its network modeling features make it ideal for, though not limited to, cluster environments.

3.3 Network Topology Generation

The first step (Fig. 4-❶) is to define the experiment parameters, including (i) the network dataset, (ii) the target topology (Fig. 4-❷), and (iii) additional experiment parameters, *e.g.,* the number of blockchain nodes or the characteristics of the workload.

The network dataset consists of experimental measurements of the network properties related to nodes and regions. By specifying the network dataset structure (a CSV file with columns `src-region`, `dst-region`, `latency`, `throughput`, `hops`), users can conduct experiments and replicate network scenarios captured from various sources. This provides a method to replicate existing deployments such as a cloud environment. Additionally, users should provide the target topology shape that defines how nodes are connected. In our experiments, Lilith supports five topologies (Fig. 5), with nodes as validators, end users as Diablo entities, and gateways as cloud regions connected by the defined topology. Additionally, it utilizes the 10 AWS regions listed in Fig. 2.

A *fat-tree* topology, often used in data centers, consists of multiple gateways organized in layers. For our emulated network of 10 AWS regions, we allocate 20% (2 gateways per region) at the first level and 50% (5 regions) at the second level. Gateways for the first and second levels are selected based on the lowest latencies in the dataset, while links between gateways at these levels are assigned randomly to limit dynamic routing paths and promote information flow throughout the topology. Blockchain nodes connect to the gateways at the second level.

Full mesh is a topology where each node is directly connected to every other node, reflecting the connectivity of cloud-based deployments. In such topologies, high-latency links simulate long intercontinental connections. This topology is easy to construct, with all gateways interconnected, simplifying the scan of the network dataset to capture region/gateway pairs and latency information.

A *hypercube* topology connects nodes via binary-based adjacency, forming a multidimensional structure common in parallel and IoT systems. We encode each of the 10 AWS regions as 4-bit binary vectors [0,9] and generate links by toggling one bit at a time, accepting only valid region indices.

In a *scale-free* topology few nodes have significantly more connections than others [26]. We build topologies of this kind by using the preferential attachment algorithm [26], so that nodes with higher degrees are more likely to be selected. This topology resembles Internet-like (WAN) networks [26].

A *torus* topology connects nodes in a wrap-around grid, enabling efficient data transfer, and is often used in supercomputing [72]. We construct a 2D torus (2 rows and 5 columns) for 10 AWS regions, placing one gateway per slot. Starting with the lowest-latency region pair in the first column, we iteratively add columns by selecting new region pairs linked to the most recently added gateways, prioritizing low latency. Each gateway connects to its previous column and adjacent row gateway.

By incorporating latency and throughput into the structure of the network dataset, evaluators can construct customized topologies that prioritize specific performance goals, such as low latency over high throughput. Organizing the dataset accordingly and selecting regions based on these metrics enable more targeted and meaningful performance assessments. Furthermore, with Kollaps's routing selection mechanism, we can determine whether to optimize for the best latency or the fewest hops in the link selection.

3.4 Benchmark Execution

Lilith manages the installation of Docker daemons and Kollaps on the cluster nodes, deploys the necessary experiment images, and sets up a network to distribute the experiment across the machines (Fig. 4-❸). The initialization phases can also enforce resource limits (Fig. 4-❹), to mimic specific computing or networking constraints (leveraging `cgroup`'s container properties). The architecture integrates with Docker containers, though a similar approach can be used with other execution units (e.g., virtual machines, native processes) with additional engineering efforts. Once all steps are completed, the services can be initiated (Fig. 4-❺).

The Lilith builder interacts (Fig. 4-❻) with Diablo's Minion so that the launcher (Fig. 4-❼) triggers Diablo's mechanics to start its benchmark suite. Diablo's Minion [7] component is in charge of installing and bootstrapping the blockchain network (Fig. 4-❽). This process includes installing both Diablo and the blockchains dependencies on the designated machines. The primary node coordinates the experiment (Fig. 4-❾): it orchestrates the execution of the various secondary nodes. To ensure a ready blockchain environment where validators are aware of the current blockchain state and clients can actively send transaction requests, the primary coordinates the blockchain configuration by creating accounts and the genesis block, which is then distributed to all blockchain nodes. A given workload effectively starts when the secondaries inject the workload transactions into the blockchain network (Fig. 4-❿). Once finished, they collect and transmit the results back to the primary. The Lilith architecture includes a modular monitoring component, which is in charge of collecting network usage with the `vnstat` tool [1]. Also, an exporter module enables further refinements of both benchmark results and execution trace during post-processing. The exporter gathers network monitoring data, validator logs from the blockchain nodes, and benchmark execution records from the primary node. A script plots the results as shown in §5, allowing performance and network metrics to be checked.

4 Evaluation Setup

This section presents our experimental setup by detailing the experiment configurations, the blockchains under test, the used workloads, and the properties of the considered topologies. Specifically, we analyze the impact of five distinct topologies on five blockchains running six different workloads. We focus on four key performance metrics: (i) *commit ratio*, *i.e.,* the ratio of submitted to committed transactions; (ii) *throughput*, measured in terms of committed transactions per second (CTPS); (iii) *block latency*, *i.e.,* the average time required to finalize transactions, also referred to as *block finality* [42]; and (iv) *network load*, measured in Mbps.

Our evaluation intends to provide insights into the following specific research questions:

RQ1 How do topologies affect blockchain performance and which is the optimal topology for each blockchain?
RQ1 What is the impact on performance of network perturbations, such as packet loss, congestion, node failures, and increased latency, and which blockchain is most affected?

4.1 Topologies

In our experiments, we examine the five topologies described in §3.3. Table 2a summarizes their structural properties, as well as aggregated statistics over the imposed throughput and latency properties. Specifically, we report: *degree*, the average number of neighbors per gateway; *links*, the total number of links; and *average latency* and *average throughput*, computed from the respective network datasets used to define each topology.

4.2 Workloads

We consider a variety of workloads, ranging from modest to very high TPS peaks over a short time interval, modeling transfer transactions and smart contracts, as detailed next (see Table 2b).

DDoS is a constant workload of 10,000 injected TPS staging for 2 minutes used to stress the robustness of the specific blockchain under sustained high-demanding scenarios, *e.g.,* a DDoS attack.

FIFA models the FIFA website workload during the 1998 soccer world cup [21]. It implements a simple (yet highly contended) counter smart contract with an *add* function. We use a very high sending rate (45,000 injected TPS) over a 100 seconds experiment.

GAFAM simulates a financial market smart contract, allowing users to buy and check stock availability for *Google, Apple, Facebook, Amazon*, and *Microsoft*. Using data derived from [5], the system runs for 3 minutes, reaching an initial shortly lived peak of 19,800 injected TPS before stabilizing between 25 and 140 injected TPS. For simplicity, we round the peak to 20,000 injected TPS.

Gaming executes the trace of an online battle arena video game (*Dota2*) [69]. The trace lasts 276 seconds, invoking at an almost constant high update rate of 13,000 injected TPS.

Table 2: Topologies (a) and workloads (b) for tested blockchains (c).

Topology	Degree	Links	Avg. Lat. (ms)	Avg. TPut (Mbps)
fat-tree (k ports=4, l level=2)	k	$k^l/2$	102	712
full mesh (N nodes)	$N-1$	$N \times (N-1)/2$	194	600
hypercube (n dimensions=4)	n	$n \cdot 2^{(n-1)}$	208	560
scale-free (N nodes)	(Based on Power Law)		218	432
torus (N nodes/row=5, n dim.=2)	$2n$	$n \cdot N^n$	197	728

(a) Topologies used in our experiments.

Workload	Type	Scenario	Duration (s)	Injected TPS
DDoS	Transfer Tx	Constant rate	120	10,000
FIFA	Smart contract	High sending rate	100	45,000
GAFAM	Smart contract	Burst	180	20,000 down to 100
Gaming	Smart contract	Intensive	276	13,000
PayPal	Transfer Tx	Constant rate	300	200
VISA	Transfer Tx	Constant rate	300	1,800

(b) Workloads used in our experiments.

Blockchain	Consensus	VM	DApp	Block Finality (s)	Claimed TPS
Algorand	BA [41]	AVM	PyTeal [8]	3.3 [9]	7.5K [9]
Diem	HotStuff [80]	MoveVM	Move	100 [61]	60–1K [81]
Ethereum	Clique [70]	geth	Solidity	10–20 [4]	10–15 [67]
Quorum	IBFT [66]	geth	Solidity	2–15 [52]	0.7K–2.5K [10]
Solana	TowerBFT [78]	Sealevel	Solang	12 [42]	65K [11]

(c) Blockchains used in our experiments.

The **PayPal** payment system typically processes an average of 193 TPS [53]. For simplicity, in our experiments we modeled it as a constant workload of 200 injected TPS over a 5-minute period.

Similarly, we approximate the **VISA** system, reported at 1,770 TPS in [42], with a constant workload of 1,800 injected TPS.

4.3 Blockchains under Test

Next, we detail the blockchains under study, summarized in Table 2c. They are representative of different approaches (*e.g.,* public, private, and consortium blockchains), smart contract languages (*e.g.,* Solidity, PyTeal, Move), execution virtual machines (*e.g.,* AVM, MoveVM, geth, Sealevel), and consensus mechanisms (*e.g.,* BA, HotStuff, Clique, IBFT, TowerBFT). To ensure consistency, we used the blockchain configurations from [42], with each blockchain version specified by the respective repository commits.

Algorand [41] uses a pure PoS consensus algorithm, selecting nodes via sortition to propose blocks. Transactions are finalized immediately upon inclusion, minimizing fork risks. The platform provides a blocking API for transaction confirmation and uses WebSockets over HTTP (TCP) for node communication, leveraging a gossip protocol [28]. Nodes validate messages to avoid duplicates, with default settings allowing up to 15 connections per IP and 2,400 incoming connections per port. During our experiments, we focused on detecting transaction commits by polling the blockchain only after blocks were added, which significantly boosted performance. Tests were conducted using Algorand at commit `116c06e`.

Diem [22] uses an adapted version of HotStuff [80] for deterministic finality with low communication overhead. Its nodes limit memory pools to 100 transactions per signer, addressed in tests by submitting transactions from 2,000 accounts. Built on the *libp2p* project, it employs RPC and DirectSend for efficient message delivery [19]. Our testing was based on Diem's testnet branch at commit `4b3bd1e`. Though no longer active, Diem provides valuable insights into permissioned infrastructures.

Ethereum [74] is a public blockchain platform for decentralized applications (DApps) and smart contracts, using *devp2p* protocols [37] like UDP-based Node Discovery and TCP-based RLPx [32]. It limits protocol messages to 10 MB to maintain efficiency, disconnecting oversized messages [33]. In our experiments, the Clique PoA variant [70] was used, with blocks added at 1-second intervals by validators in a round-robin manner. Dynamic fee adjustments were configured to handle gas variability introduced by the London update (August 2021). Our benchmarks employ the Go implementation of the Ethereum protocol, specifically at commit `72c2c0a`.

Quorum [31] is an enterprise-focused, permissioned Ethereum variant. It uses two modules from the Constellation p2p system: the Transaction Manager, which handles private transactions and encrypted payload exchange, and the Enclave, which protects cryptographic operations and private keys. Following previous studies [42], we configured Quorum with the IBFT consensus algorithm [66] at commit `919800f` to address message delays and vulnerabilities in PoA systems [70].

Solana [76] uses the TowerBFT consensus mechanism [78], combining features of BFT and PoS [59] for scalability and throughput. It also employs *Proof of History* (PoH) [68] for timestamping. Like Ethereum, Solana faces forks and requires 30 confirmations to finalize transactions [6]. Blocks are added every 400 milliseconds, with a simplified data structure and EdDSA signature replacing ECDSA. The network uses a custom UDP-based protocol for transaction transmission and block propagation, supported by the Turbine mechanism for scalability [27]. Validators communicate in peer groups, allowing scalability beyond 1,000 validators [79]. Solana's API supports commitment levels and block monitoring, with our evaluation periodically fetching block hashes within typical DApp time constraints. We used commit `0d36961`.

4.4 Testing Configurations

Our cluster is composed of seven Dell PowerEdge R630 server machines, each equipped with two 16-core/32-thread Intel Xeon E5-2683v4 clocked at 2.10 GHz CPU and 128 GB of RAM, connected by a Dell S6010-ON 40 GbE switch. The nodes run Ubuntu Linux 22.04 LTS, kernel v5.15.0-107-generic.

We evaluate the impact of topologies by using different configurations, taking into account (i) the number of nodes per region, (ii) varying link latencies, and (iii) the bandwidth capacity for each blockchain node. We also study how the blockchains are affected by faults happening in the network, *i.e.,* emulating packet drop, congestion, node failures, and increased latency, to observe failure in the reception/transmission of messages between nodes during the workload execution. We present our results in the next section.

5 Measurement Results

We study the impact of topologies on the various blockchains by focusing on the three key performance metrics established at the beginning of §4. Given the duration of the experiments and the limited time constraints, we report the average across 3 runs.

5.1 Small Deployment

We begin by deploying one node per region, for a total of 10 nodes. The results are depicted, for each topology, on the left side of Fig. 6. For less demanding workloads (GAFAM and PayPal), Algorand and Diem achieve a commit ratio greater than 80%, which is close to 100% for PayPal, with Diem slightly outperforming the former. Although the results are consistent across different topologies, we observe that Algorand experiences a 50% drop in CTPS (to 86 CTPS) on a scale-free topology with the GAFAM workload.

Quorum achieves a commit rate above 85% for the PayPal workload in hypercube (Fig.6e) and torus (Fig.6i) topologies, due to low congestion in the torus and the high connectivity of hypercube, which ease transaction retrieval. Solana also exceeds 85% in most cases, except for the fat-tree (Fig.6a) and scale-free (Fig.6g) topologies, where congestion drops the rate to below 60%.

Under heavy workloads Algorand performs better with a rate below 8% for transfer transactions (DDoS, see Fig. 6c) and below 3% for smart contract transactions (Gaming and FIFA, see Fig. 6a). This is explained by Algorand's high capacity transaction pool, which can handle up to 75,000 transactions [15].

As noted in prior works [23, 42], Quorum fails to show commits for the most demanding workloads. We ascribe this to the leader bottleneck in the BFT consensus used in Quorum, which can saturate memory pools or network queues under high workloads.

Ethereum maintains a commit rate below 10% (see Fig. 6a). It achieves higher block latency (≥150 ms) for PayPal and VISA (less demanding transfer workloads) compared to latency below 88 ms for high-demanding workloads (DDoS, FIFA, Gaming). This is mainly due to the period between its consecutive blocks regardless of the network bandwidth.

Algorand and Diem obtain the best results with the PayPal workload independent from the topology. Interestingly, VISA is too demanding for all blockchains, with a commit rate always below 30%. Algorand achieves the best results with the DDos workload (up to 8% commit rate over 10,000 injected TPS). Regarding smart contract executions, we observe the best results with Algorand and Diem (achieving 100 CTPS for FIFA, up to 150 CTPS for GAFAM, and around 200 CTPS for Gaming). Quorum and Solana obtain between 30 and 120 CTPS for non-intensive smart contracts. Finally, benchmarks show poor results for all intensive smart contract workloads (FIFA and Gaming).

5.2 Scaling the Network

We now focus on larger networks, deploying four nodes per region for a total of 40 nodes, limited by our cluster resources to avoid overload. The results for each topology are shown on the right side of Fig. 6. As expected, a higher number of nodes per region results in more links contending for the region gateway capacity. Even with a larger set of available nodes, Quorum fails to handle high workloads. This is expected since the consensus protocol used by Quorum has a quadratic cost with the number of nodes. Solana commits only for the GAFAM workload for all the topologies, while PayPal and VISA just with torus (see Fig. 6i), failing for all others. We explain this behavior with network congestion: Solana may drop newly submitted transactions if the outstanding rebroadcast queue of an RPC node exceeds 10,000 transactions [12], preventing them from being forwarded to the leader. Additionally, validators can become unresponsive during periods of high load [13, 51].

Ethereum achieves a low CTPS regardless of the number of nodes in the network. Algorand is the least affected by network size or by the additional network congestion. Notably, credential messages (*e.g.,* used to authenticate participants within Algorand) are significantly smaller in size (between 100 and 200 bytes) compared to block proposals (5 MB). Their smaller size allows them to propagate more swiftly through the network. This faster propagation helps peer nodes prioritize block proposals more effectively, thereby alleviating network congestion.

5.3 Network Throughput

We study how the topology affects network traffic among the blockchain nodes. Fig. 7 depicts these results. We notice that workloads with transfer transactions are lighter compared to smart-contract-based ones. For instance, by looking at benchmarks with one node per region (see Fig. 7 left side), the PayPal workload consumes up to 34 Mbps on hypercube topology, while FIFA consumes up to 583 Mbps on the same topology. Moving towards four nodes per region (see Fig. 7 right side), we notice a general increase of network data along the GAFAM workload. In this case, the PayPal workload consumes up to 59 Mbps across all the tested topologies, while FIFA consumes up to 896 Mbps on every topology. With a configuration of one node per region Solana shows the highest network throughput, followed by Algorand and then Diem. But with four nodes per region Algorand shows a higher network throughput, followed by Solana and Quorum. On the other hand, Ethereum typically has the lowest bandwidth utilization, followed by Quorum with one node per region and Diem with four nodes per region.

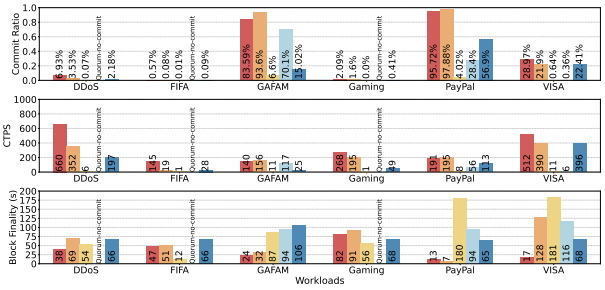
5.4 Network Dynamics

We now examine the effects of network degradation – packet loss, congestion, node failures, and increasing latency – on blockchain performance. These experiments used a full mesh topology and one node per region. For the packet loss, congestion, and node failure experiments, we employ the PayPal workload (200 injected TPS for 300 seconds) to test on a simple and sustained workload. Dynamic events were triggered 60 seconds into the workload, with normal conditions restored after another 60 seconds.

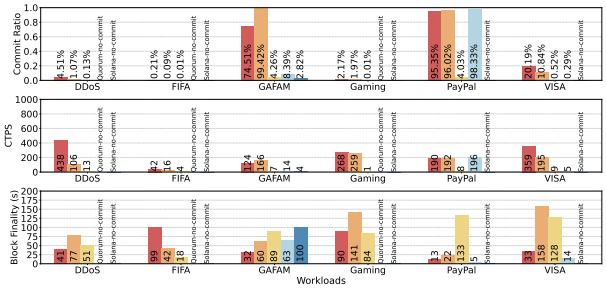
In the packet loss experiments (see Fig. 8a), we applied discrete packet drop percentages (0%, 10%, 20%, 30%) to 1/3 of the links, selected at random according to a uniform distribution. As expected, the performance of all systems degraded substantially with higher packet loss. Algorand and Solana remained robust at 10% loss. Interestingly, Quorum's throughput sometimes improved, likely due to random link selection allowing more efficient transaction distribution or isolating non-congested nodes.

During the bandwidth congestion experiments (see Fig. 8b), we select 10%, 20%, and 30% of the links at random according to a uniform distribution and reduce their bandwidth to 20% of their original capacity. Blockchains showed consistent performance under increased congestion, with Quorum being the most resilient.

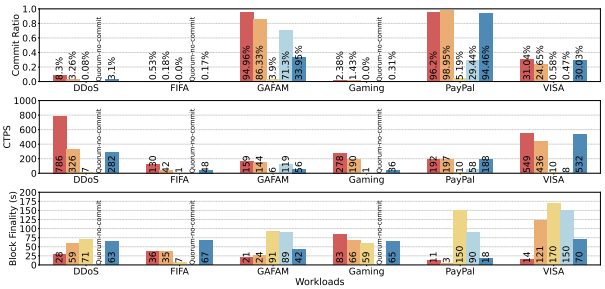
Node crash experiments (with 10%, 20%, and 30% of total nodes crashed) revealed that Algorand struggled significantly with 30% crashes (see Fig. 8c), while Ethereum, despite lower performance,



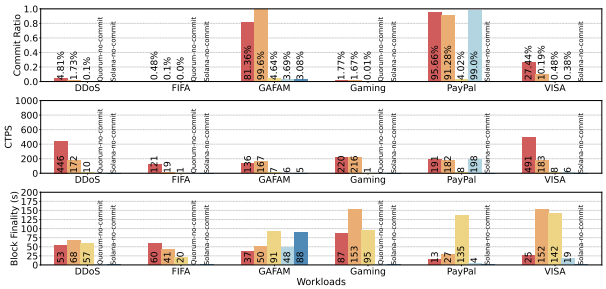
(a) Fat-tree, one node/region.



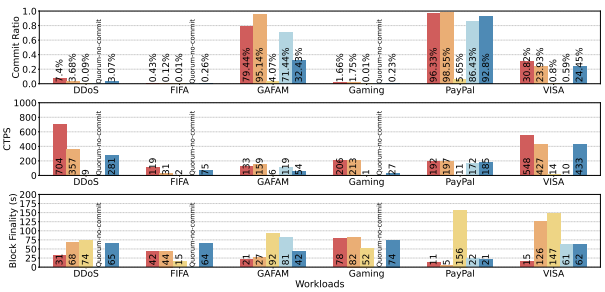
(b) Fat-tree, four nodes/region.



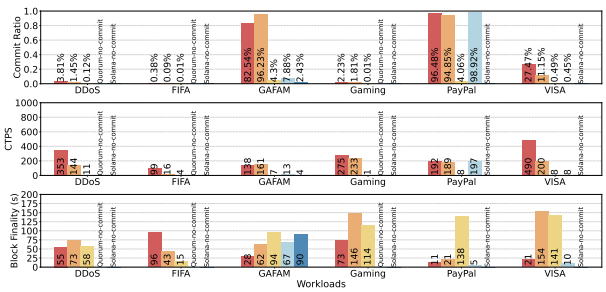
(c) Full mesh, one node/region.



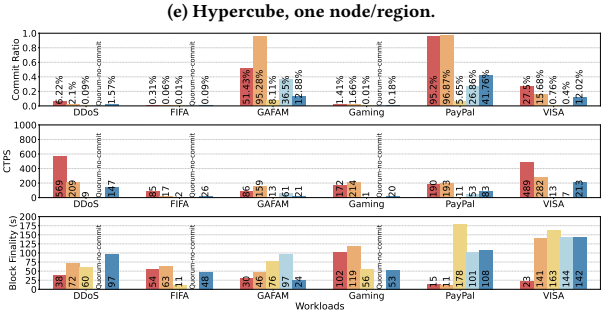
(d) Full mesh, four nodes/region.



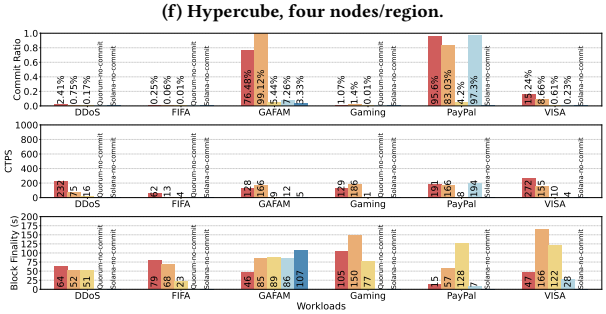
(e) Hypercube, one node/region.



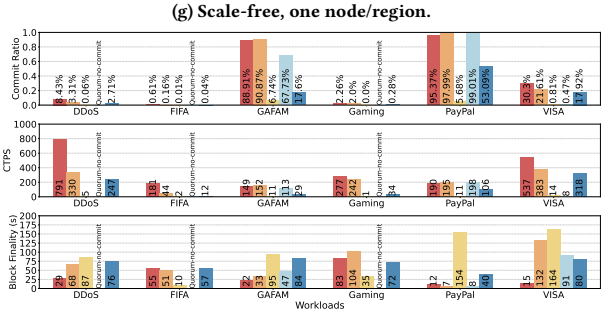
(f) Hypercube, four nodes/region.



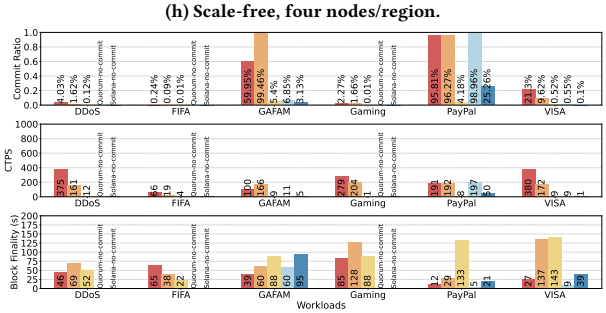
(g) Scale-free, one node/region.



(h) Scale-free, four nodes/region.



(i) Torus, one node/region.



(j) Torus, four nodes/region.

Figure 6: Blockchain performance across various workloads using the 2023 AWS dataset (Algorand, Diem, Ethereum, Quorum, Solana).

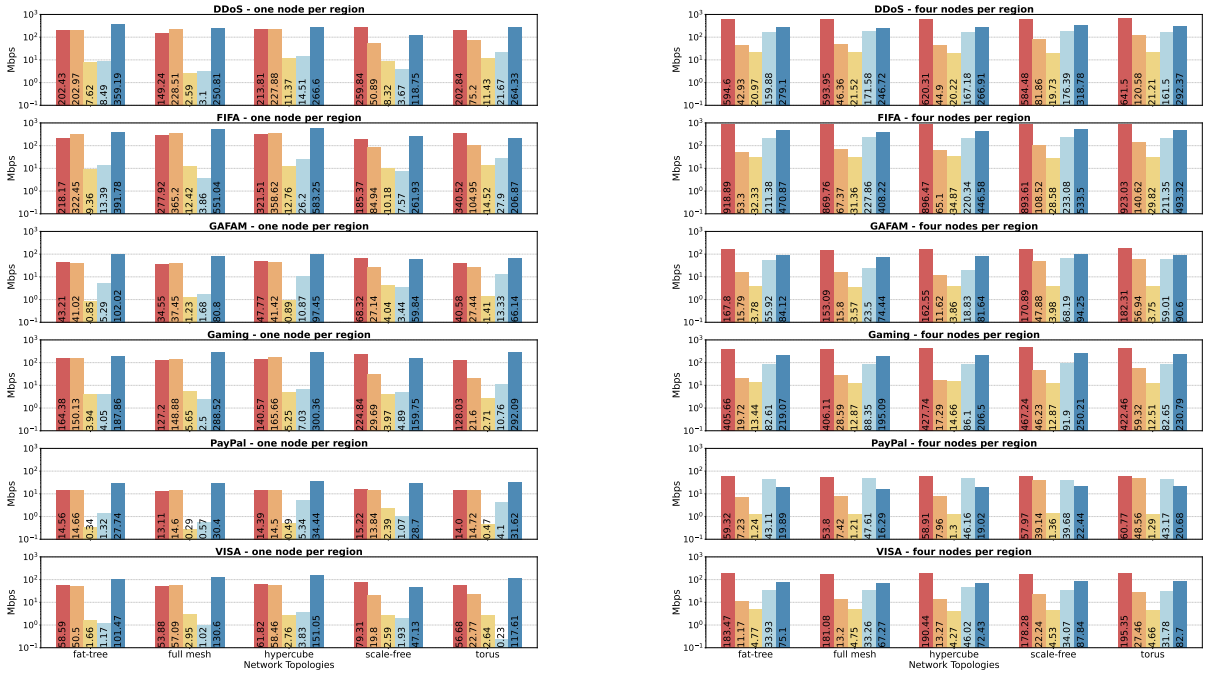


Figure 7: Network load (Mbps) during workload execution (Algorand, Diem, Ethereum, Quorum, Solana).

was the most resilient. Diem, Quorum, and Solana exhibited predictable, gradual performance drops as crashes increased.

For the increased latency experiments (see Fig. 9 and Fig. 10), we extracted minimum and maximum latency values for region pairs from the network dataset, ranging approximately from 30 ms to 500 ms. We employed adapted and extended workloads: for the PayPal workload, we conducted a 20-minute experiment where latencies began to increase at the 10-minute mark (N1), gradually peaking over 200 seconds (N2) and maintaining that level for an additional 60 seconds (N3) before returning to baseline for the last 6 minutes; in the GAFAM workload, we performed a 5-minute experiment where latencies rose after the first minute (S1), peaked after another minute (S2), and remained at the maximum until the third minute (S3), gradually reverting to initial levels thereafter. These scenarios were repeated starting with a baseline factor of 1×, followed by incremental factors of 10×, simulating real-world conditions such as peak transaction periods or network congestion during significant market changes [44]. We observed (see Fig. 9 and Fig. 10) that blockchains utilizing traditional consensus mechanisms, like Quorum, exhibit longer recovery times as latencies increase. Algorand also demonstrates repercussions with the PayPal workload due to rising latency, while Diem, Ethereum, and Solana show more robust behavior as latencies increase.

6 Discussion and Limitations

Among the blockchains presented in §4 and used in our experiments of §5, Algorand and Diem are the most performant ones, exhibiting more consistent results (*e.g.,* higher CTPS) across different topologies. Note that the results from Algorand are remarkable, as it is a public blockchain with a decentralized consensus mechanism.

As the number of nodes per region/gateway in the network increases, the blockchains exhibit a reduced commit rate while their average block latency augments. This is expected since most blockchains (and in particular the ones under test) are known to scale poorly with the number of nodes due to the inefficiencies of the underlying consensus protocol [57].

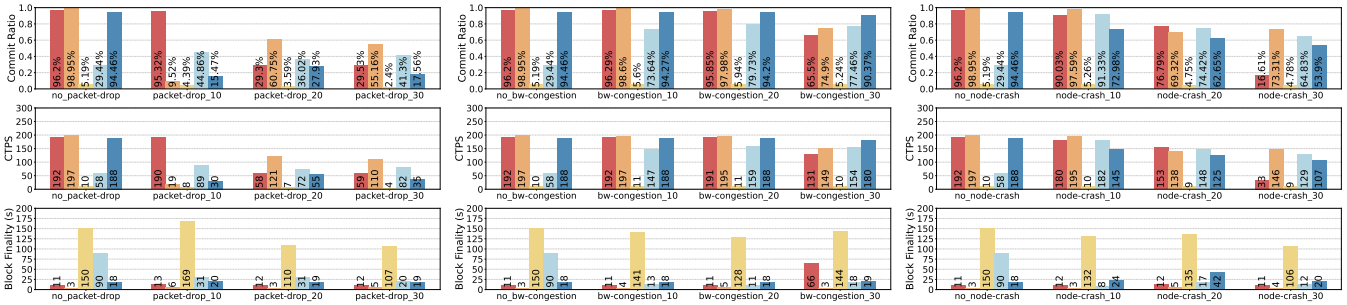
In the light of our experimental results, we are now able to answer the research questions posed in §4:

RQ1 Torus generally allows for better results with more demanding workloads. We achieve the best overall results, across the five tested blockchains, atop full mesh and hypercube, due to their high degree and link capacity.

RQ2 Our results demonstrate that Diem and Solana are the most affected by packet loss, followed by Quorum and Algorand. Classical consensus mechanism, as in Quorum, are less reactive to increasing latencies. Despite consistently exhibiting lower CTPS compared to the others, Ethereum is the least affected by these dynamic network events.

While public blockchains like Ethereum use scale-free topologies and private ones like Diem adopt fat-tree or hypercube structures, our work shows that reorganizing node layouts can improve efficiency without altering the fundamental public/private distinction.

Limitations. Lilith turns out to be resource-intensive for large-scale blockchain simulations, especially with thousands of nodes. For instance, Solana requires 8GB of RAM and 16 threads per node, limiting scalability. Our experiments were limited to a 40-node network, sufficient to capture behaviors representative of real-world blockchains. While larger networks could offer deeper insights, some studies show that performance indices like consensus efficiency, throughput, and latency stabilize with relatively few



(a) Packet drop. (b) Link bandwidth congestion. (c) Node crash.

Figure 8: Network dynamics, one node per region, full mesh topology (Algorand, Diem, Ethereum, Quorum, Solana).

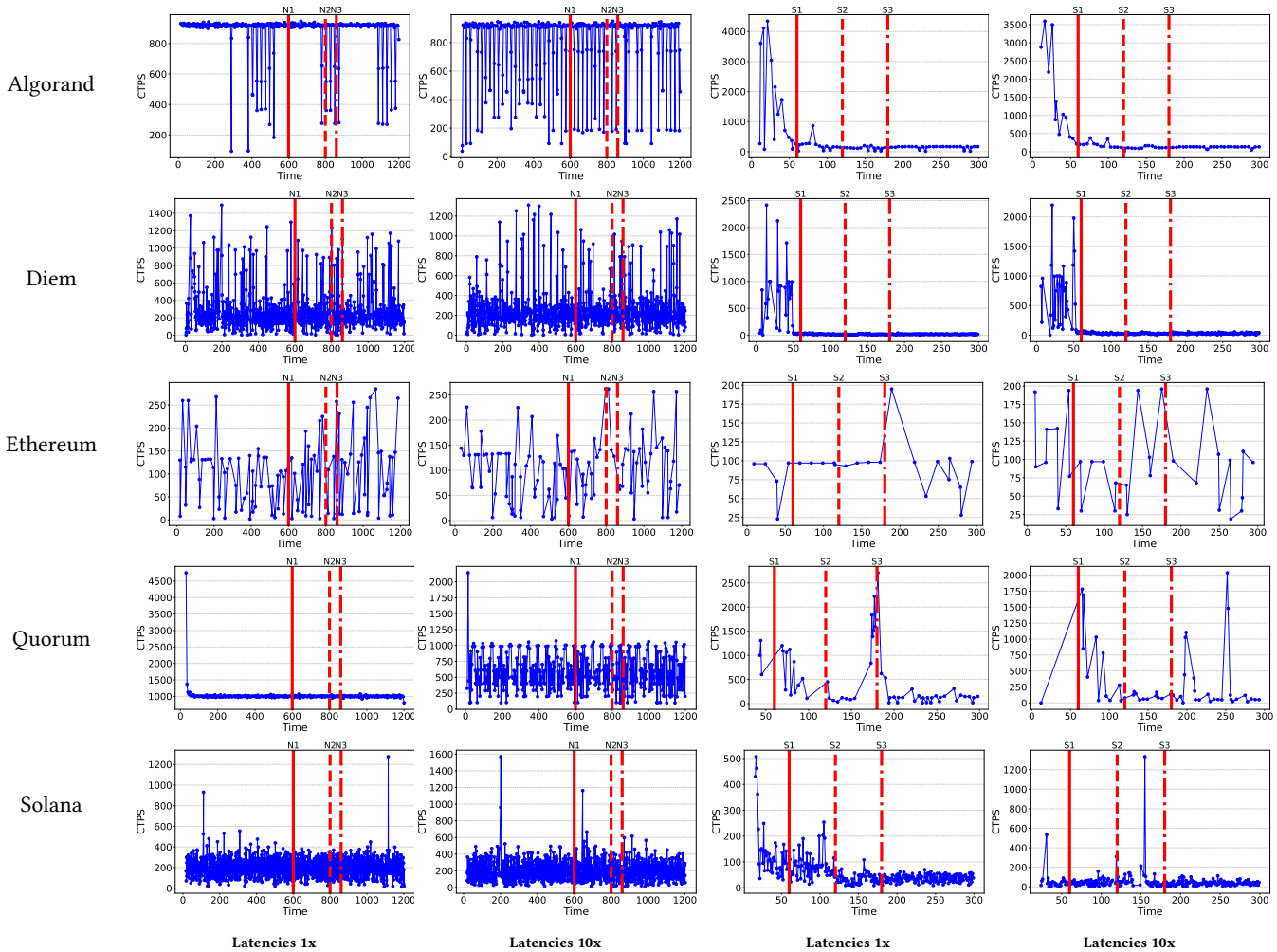


Figure 9: Benchmark with increasing latencies using one node per region in a full mesh topology with the PayPal workload. Red lines represent latency variation events (see §5.4).

Figure 10: Benchmark with increasing latencies using one node per region in a full mesh topology with the GAFAM workload. Red lines represent latency variation events (see §5.4).

nodes [47]. Our setup, though smaller than public blockchains, enables controlled experiments on performance trends. Manual adjustments addressed deployment variations, limiting Lilith flexibility. A full mesh topology ensured comparability; decentralized topologies will be explored using Lilith. Beyond networking, factors like consensus and block size also affect performance and will be studied. We release our dataset (https://doi.org/10.5281/zenodo.11409100), with variance data showing Algorand and Diem as stable, and Quorum and Solana more sensitive. Findings inform resilient designs for permissioned systems.

7 Conclusions and Future Work

We evaluated blockchain performance under various network topologies by using Lilith, answering research questions **RQ1-2** (posed in §4) in a controlled, reproducible on-premise environment. By incorporating AWS network properties, we emulated realistic geo-distributed setups. Lilith enabled efficient management of network infrastructures, leveraging research-grade resources to replicate cloud-like configurations and dynamic network setups. Future work includes scaling across multiple clusters, analyzing real fault traces, and adding blockchains like Avalanche.

Reproducibility. We release the full Lilith code at https://doi.org/10.5281/zenodo.11409100 including the topologies and instructions to reproduce all the experiments. The 16-month AWS cloud probing dataset (§1) is available at https://zenodo.org/records/11457020 and contains 10,142 hourly snapshots across 34 AWS regions in JSON format (schema detailed on Zenodo). The data was collected via periodic queries to https://cloudping.co.

Acknowledgments

This research has been supported by the PRIN 2020 project NiRvAna – Noninterference and Reversibility Analysis in Private Blockchains. The scholarship of the first author at the Italian PhD Program in Blockchain and Distributed Ledger Technology is funded by PNRR – Piano Nazionale di Ripresa e Resilienza according to D.M. 351/2022. This work was partially executed within the context of the REDONDA project (CHIST-ERA-22-SPiDDS-05).

References

[1] 2014. https://humdi.net/vnstat/ Accessed: 2025-05-02.
[2] 2016. Alexei-led/PUMBA: Chaos Testing, network emulation, and stress testing tool for containers. https://github.com/alexei-led/pumba Accessed: 2025-05-02.
[3] 2017. GitHub - sbaresearch/simcoin: Blockchain simulation framework with Docker and Python. — github.com. https://github.com/sbaresearch/simcoin. Accessed: 2025-05-02.
[4] 2019. Geth-Poa-tutorial/readme.md at master · Ibrahimmashaly/Geth-Poa-tutorial. https://github.com/ibrahimmashaly/geth-poa-tutorial/blob/master/README.md#block-time Accessed: 2025-05-02.
[5] 2022. https://www.nasdaq.com/ Accessed: 2025-05-02.
[6] 2022. https://github.com/solana-labs/solana/blob/master/programs/vote/src/vote_state/mod.rs#L34 Accessed: 2025-05-02.
[7] 2023. Diablo Blockchain Benchmark Suite — diablobench.github.io. https://diablobench.github.io/. Accessed: 2025-05-02.
[8] 2024. https://pyteal.readthedocs.io/en/latest/ Accessed: 2025-05-02.
[9] 2024. https://developer.algorand.org/docs/get-started/basics/why_algorand/ Accessed: 2025-05-02.
[10] 2024. https://cryptoexchange.com/learning/what-is-quorum Accessed: 2025-05-02.
[11] 2024. https://coincu.com/240765-solana-tps-how-many-tps-can-solana-handle/ Accessed: 2025-05-02.
[12] 2024. https://solana.com/docs/advanced/retry Accessed: 2025-05-02.
[13] 2024. https://solana.com/docs/advanced/confirmation Accessed: 2025-05-02.
[14] 2024. Ethereum/Hive: Ethereum end-to-end Test Harness. https://github.com/ethereum/hive. Accessed: 2025-05-02.
[15] Algorand. 2024. Node configuration settings. Algorand Developer Portal. https://developer.algorand.org/docs/run-a-node/reference/config/ Accessed: 2025-05-02.
[16] M. Alharby and A. van Moorsel. 2020. BlockSim: An Extensible Simulation Tool for Blockchain Systems. *Frontiers in Blockchain* 3 (2020). https://doi.org/10.3389/fbloc.2020.00028
[17] L. Alsahan, N. Lasla, and M. Abdallah. 2020. Local Bitcoin Network Simulator for Performance Evaluation using Lightweight Virtualization. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*. 355–360. https://doi.org/10.1109/ICIoT48696.2020.9089630
[18] S. Amaro, M. Matos, and V. Schiavoni. 5555. KOLLAPS: Decentralized and Efficient Network Emulation for Large-Scale Systems . *IEEE/ACM Transactions on Networking* 01 (5555), 1–16. https://doi.org/10.1109/TNET.2024.3478050
[19] Z. Amsden, R. Arora, S. Bano, M. Baudet, S. Blackshear, and A. Bothra. 2019. The Libra Blockchain. https://mitsloan.mit.edu/shared/ods/documents?PublicationDocumentID=5859 Accessed: 2025-05-02.
[20] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo. 2019. SimBlock: A Blockchain Network Simulator. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 325–329. https://doi.org/10.1109/INFCOMW.2019.8845253
[21] M. Arlitt and T. Jin. 2000. A workload characterization study of the 1998 world cup web site. *IEEE network* 14, 3 (2000), 30–37.
[22] M. Baudet, A. Ching, A. Chursin, G. Danezis, F. Garillot, Z. Li, D. Malkhi, O. Naor, D. Perelman, and A. Sonnino. 2019. State machine replication in the libra blockchain. *The Libra Assn., Tech. Rep* 7 (2019).
[23] H. Benoit, V. Gramoli, R. Guerraoui, and C. Natoli. 2021. Diablo: A Distributed Analytical Blockchain Benchmark Framework Focusing on Real-World Workloads. (2021), 13. http://infoscience.epfl.ch/record/285731
[24] Bitnodes. 2025. https://bitnodes.io/ Accessed: 2025-05-02.
[25] M. Castro and B. Liskov. 1999. Practical Byzantine Fault Tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation* (New Orleans, Louisiana, USA) *(OSDI '99)*. USENIX Association, USA, 173–186.
[26] Q. Chen and D. Shi. 2004. The modeling of scale-free networks. *Physica A: Statistical Mechanics and its Applications* 335, 1 (2004), 240–248. https://doi.org/10.1016/j.physa.2003.12.014
[27] R. Chern. 2024. Turbine: Block propagation on Solana. https://www.helius.dev/blog/turbine-block-propagation-on-solana Accessed: 2025-05-02.
[28] M. Conti, A. Gangwal, and M. Todero. 2019. Blockchain Trilemma Solver Algorand has Dilemma over Undecidable Messages. In *Proceedings of the 14th International Conference on Availability, Reliability and Security* (Canterbury, CA, United Kingdom) *(ARES '19)*. Association for Computing Machinery, New York, NY, USA, Article 16, 8 pages. https://doi.org/10.1145/3339252.3339255
[29] C. Decker and R. Wattenhofer. 2013. Information propagation in the Bitcoin network. In *IEEE P2P 2013 Proceedings*. 1–10. https://doi.org/10.1109/P2P.2013.6688704
[30] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.L. Tan. 2017. BLOCKBENCH: A Framework for Analyzing Private Blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data* (Chicago, Illinois, USA) *(SIGMOD '17)*. Association for Computing Machinery, New York, NY, USA, 1085–1100. https://doi.org/10.1145/3035918.3064033
[31] T. Espel, L. Katz, and G. Robin. 2017. Proposal for protocol on a quorum blockchain with zero knowledge. *Cryptology ePrint Archive* 2017 (2017), 1093.
[32] Ethereum. 2014. Devp2p/rlpx.md at master - Ethereum/DEVP2P. GitHub. https://github.com/ethereum/devp2p/blob/master/rlpx.md Accessed: 2025-05-02.
[33] Ethereum. 2024. Devp2p/caps/eth.md at master - Ethereum/DEVP2P. GitHub. https://github.com/ethereum/devp2p/blob/master/caps/eth.md Accessed: 2025-05-02.
[34] C. Faria and M. Correia. 2019. BlockSim: Blockchain Simulator. In *2019 IEEE International Conference on Blockchain (Blockchain)*. 439–446. https://doi.org/10.1109/Blockchain.2019.00067
[35] S. M. Fattahi, A. Makanju, and A. Milani Fard. 2020. SIMBA: An Efficient Simulator for Blockchain Applications. In *2020 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*. 51–52. https://doi.org/10.1109/DSN-S50200.2020.00028
[36] G. Fridgen, J. Sedlmeir, P. Ross, A. Luckow, J. Lockl, and D. Miehle. 2021. The DLPS: A New Framework for Benchmarking Blockchains. https://doi.org/10.24251/HICSS.2021.822
[37] Y. Gao, J. Shi, X. Wang, Q. Tan, C. Zhao, and Z. Yin. 2019. Topology Measurement and Analysis on Ethereum P2P Network. In *2019 IEEE Symposium on Computers and Communications (ISCC)*. 1–7. https://doi.org/10.1109/ISCC47284.2019.8969695
[38] E. Georgiadis. 2019. How many transactions per second can bitcoin really handle? Theoretically. *IACR Cryptol. ePrint Arch.* (2019), 416. https://eprint.iacr.org/2019/416
[39] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun. 2016. On the Security and Performance of Proof of Work Blockchains. In *Proceedings*

of the 2016 ACM SIGSAC Conference on Computer and Communications Security (Vienna, Austria) *(CCS '16)*. Association for Computing Machinery, New York, NY, USA, 3–16. https://doi.org/10.1145/2976749.2978341

[40] F. C. Geyer, H.A. Jacobsen, R. Mayer, and P. Mandl. 2023. An End-to-End Performance Comparison of Seven Permissioned Blockchain Systems. In *Proceedings of the 24th International Middleware Conference* (Bologna, Italy) *(Middleware '23)*. Association for Computing Machinery, New York, NY, USA, 71–84. https://doi.org/10.1145/3590140.3629106

[41] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. 2017. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China) *(SOSP '17)*. Association for Computing Machinery, New York, NY, USA, 51–68. https://doi.org/10.1145/3132747.3132757

[42] V. Gramoli, R. Guerraoui, A. Lebedev, C. Natoli, and G. Voron. 2023. Diablo: A Benchmark Suite for Blockchains. In *Proceedings of the Eighteenth European Conference on Computer Systems* (Rome, Italy) *(EuroSys '23)*. Association for Computing Machinery, New York, NY, USA, 540–556. https://doi.org/10.1145/3552326.3567482

[43] V. Gramoli, R. Guerraoui, A. Lebedev, and G. Voron. 2024. Stabl: Blockchain Fault Tolerance. arXiv:2409.13142 [cs.DC] https://arxiv.org/abs/2409.13142

[44] M. Guennewig. 2024. *Blockchain Congestion Facilitates Currency Competition*. CRC TR 224 Discussion Paper Series. University of Bonn and University of Mannheim, Germany. https://ideas.repec.org/p/bon/boncrc/crctr224_2024_549.html

[45] hyperledger. 2018. Hyperledger Caliper — hyperledger.github.io. https://hyperledger.github.io/caliper/. Accessed: 2025-05-02.

[46] M. R. A. Lathif, P. Nasirifard, and H.A. Jacobsen. 2018. CIDDS: A Configurable and Distributed DAG-Based Distributed Ledger Simulation Framework. In *Proceedings of the 19th International Middleware Conference (Posters)* (Rennes, France) *(Middleware '18)*. Association for Computing Machinery, New York, NY, USA, 7–8. https://doi.org/10.1145/3284014.3284018

[47] A. Lebedev and V. Gramoli. 2024. On the Relevance of Blockchain Evaluations on Bare Metal. In *Distributed Ledger Technology*, Naipeng Dong, Babu Pillai, Guangdong Bai, and Mark Utting (Eds.). Springer Nature Singapore, Singapore, 22–38.

[48] Z. Li and P. Mohapaira. 2004. The impact of topology on overlay routing service. In *IEEE INFOCOM 2004*, Vol. 1. 418. https://doi.org/10.1109/INFCOM.2004.1354513

[49] B.Y. Lin, D. Dziubałtowska, P. Macek, A. Penzkofer, and S. Müller. 2023. TangleSim: An Agent-based, Modular Simulator for DAG-based Distributed Ledger Technologies. In *2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 1–5. https://doi.org/10.1109/ICBC56567.2023.10174950

[50] L. Ma, X. Liu, Y. Li, C. Zhang, G. Shi, and K. Li. 2024. GFBE: A Generalized and Fine-Grained Blockchain Evaluation Framework. *IEEE Trans. Comput.* 73, 3 (2024), 942–955. https://doi.org/10.1109/TC.2024.3349654

[51] S. Malwa. 2024. Solana Rolls Out Update to tackle network congestion. https://www.coindesk.com/tech/2024/04/15/solana-rolls-out-update-to-tackle-network-congestion/ Accessed: 2025-05-02.

[52] M. Mazzoni, A. Corradi, and V. Di Nicola. 2022. Performance evaluation of permissioned blockchains for financial applications: The ConsenSys Quorum case study. *Blockchain: Research and Applications* 3, 1 (2022), 100026. https://doi.org/10.1016/j.bcra.2021.100026

[53] D. Mechkaroska, V. Dimitrova, and A. Popovska-Mitrovikj. 2018. Analysis of the Possibilities for Improvement of BlockChain Technology. In *2018 26th Telecommunications Forum (TELFOR)*. 1–4. https://doi.org/10.1109/TELFOR.2018.8612034

[54] A. Miller and R. Jansen. 2015. Shadow-Bitcoin: scalable simulation via direct execution of multi-threaded applications. In *Proceedings of the 8th USENIX Conference on Cyber Security Experimentation and Test* (Washington, D.C.) *(CSET'15)*. USENIX Association, USA, 7.

[55] S. Nakamoto. 2009. Bitcoin: A Peer-to-Peer Electronic Cash System. (May 2009). http://www.bitcoin.org/bitcoin.pdf

[56] B. Nasrulin, M. De Vos, G. Ishmaev, and J. Pouwelse. 2022. Gromit: Benchmarking the Performance and Scalability of Blockchain Systems. In *2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. 56–63. https://doi.org/10.1109/DAPPS55202.2022.00015

[57] R. Neiheiser, M. Matos, and L. Rodrigues. 2021. Kauri: Scalable BFT Consensus with Pipelined Tree-Based Dissemination and Aggregation. In *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles* (Virtual Event, Germany) *(SOSP '21)*. Association for Computing Machinery, New York, NY, USA, 35–48. https://doi.org/10.1145/3477132.3483584

[58] T. Neudecker. 2019. *Characterization of the Bitcoin Peer-to-Peer Network (2015-2018)*. Technical Report 1. Karlsruher Institut für Technologie (KIT). https://doi.org/10.5445/IR/1000091933

[59] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz. 2019. Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities. *IEEE Access* 7 (2019), 85727–85745. https://doi.org/10.1109/ACCESS.2019.2925010

[60] H. Pan, X. Duan, Y. Wu, L. Tseng, M. Aloqaily, and A. Boukerche. 2020. BBB: A Lightweight Approach to Evaluate Private Blockchains in Clouds. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9322354

[61] G. A. Pierro and R. Tonelli. 2022. A Study on Diem Distributed Ledger Technology. In *Proceedings of the 4th Workshop on Distributed Ledger Technology co-located with the Italian Conference on Cybersecurity 2022 (ITASEC 2022), Rome, Italy, June 20, 2022 (CEUR Workshop Proceedings, Vol. 3166)*, Maurizio Pizzonia and Andrea Vitaletti (Eds.). CEUR-WS.org, 33–47. https://ceur-ws.org/Vol-3166/paper03.pdf

[62] J. Polge, S. Ghatpande, S. Kubler, J. Robert, and Y. Le Traon. 2021. BlockPerf: A hybrid blockchain emulator/simulator framework. *IEEE Access* 9 (July 2021), 107858–107872. https://doi.org/10.1109/ACCESS.2021.3101044

[63] A. Qin and E. Livni. 2021. China cracks down harder on cryptocurrency with New Ban. https://www.nytimes.com/2021/09/24/business/china-cryptocurrency-bitcoin.html#:~:text=China%20intensified%20its%20crackdown%20on,for%20newly%20created%20crypto%20tokens

[64] E. Rohrer, J. Malliaris, and F. Tschorsch. 2019. Discharged Payment Channels: Quantifying the Lightning Network Resilience to Topology-Based Attacks. In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroSys)*. IEEE Computer Society, Los Alamitos, CA, USA, 347–356. https://doi.org/10.1109/EuroSPW.2019.00045

[65] D. Saingre, T. Ledoux, and J.M. Menaud. 2020. BCTMark: a Framework for Benchmarking Blockchain Technologies. In *2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA)*. IEEE Computer Society, Los Alamitos, CA, USA, 1–8. https://doi.org/10.1109/AICCSA50499.2020.9316536

[66] R. Saltini. 2019. IBFT Liveness Analysis. In *2019 IEEE International Conference on Blockchain (Blockchain)*. 245–252. https://doi.org/10.1109/Blockchain.2019.00039

[67] M. Schäffer, M. di Angelo, and G. Salzer. 2019. Performance and Scalability of Private Ethereum Blockchains. In *Business Process Management: Blockchain and Central and Eastern Europe Forum*, Claudio Di Ciccio, Renata Gabryelczyk, Luciano García-Bañuelos, Tomislav Hernaus, Rick Hull, Mojca Indihar Štemberger, Andrea Kő, and Mark Staples (Eds.). Springer International Publishing, Cham, 103–118.

[68] J. Sliwinski, Q. Kniep, R. Wattenhofer, and F. Schaich. 2024. Halting the Solana Blockchain with Epsilon Stake. In *Proceedings of the 25th International Conference on Distributed Computing and Networking*. 45–54.

[69] Steam. [n. d.]. https://store.steampowered.com/app/570/Dota_2/ Accessed: 2025-05-02.

[70] P. Szilágyi. 2017. EIP-225: Clique proof-of-authority consensus protocol. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-225 Accessed: 2025-05-02.

[71] B. Toshniwal and K. Kataoka. 2021. Comparative Performance Analysis of Underlying Network Topologies for Blockchain. In *2021 International Conference on Information Networking (ICOIN)*. 367–372. https://doi.org/10.1109/ICOIN50884.2021.9333978

[72] T. Wang, Z. Su, Y. Xia, B. Qin, and M. Hamdi. 2014. NovaCube: A low latency Torus-based network architecture for data centers. In *2014 IEEE Global Communications Conference*. 2252–2257. https://doi.org/10.1109/GLOCOM.2014.7037143

[73] X. Wang, A. Al-Mamun, F. Yan, M. Sadoghi, and D. Zhao. 2019. BlockLite: A Lightweight Emulator for Public Blockchains. arXiv:1905.02157 [cs.DB]

[74] G. Wood. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 151, 2014 (2014), 1–32.

[75] X. Wu, J. Yan, and D. Jin. 2019. Virtual-Time-Accelerated Emulation for Blockchain Network and Application Evaluation. In *Proceedings of the 2019 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation* (Chicago, IL, USA) *(SIGSIM-PADS '19)*. Association for Computing Machinery, New York, NY, USA, 149–160. https://doi.org/10.1145/3316480.3322889

[76] Anatoly Y. 2021. *Solana: A New Architecture for a High Performance Blockchain v0.8.13*. https://solana.com/solana-whitepaper.pdf Accessed: 2025-05-02.

[77] H. Yajam, E. Ebadi, and M. A. Akhaee. 2024. JABS: A Blockchain Simulator for Researching Consensus Algorithms. *IEEE Transactions on Network Science and Engineering* 11, 1 (2024), 3–13. https://doi.org/10.1109/TNSE.2023.3282916

[78] A. Yakovenko. 2020. Tower BFT: Solana's high performance implementation of PBFT. https://medium.com/solana-labs/tower-bft-solanas-high-performance-implementation-of-pbft-464725911e79

[79] A. Yakovenko. 2020. Turbine-Solana's block propagation protocol solves the scalability trilemma. https://medium.com/solana-labs/turbine-solanas-block-propagation-protocol-solves-the-scalability-trilemma-2ddba46a51db

[80] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham. 2019. HotStuff: BFT Consensus with Linearity and Responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing* (Toronto ON, Canada) *(PODC '19)*. Association for Computing Machinery, New York, NY, USA, 347–356. https://doi.org/10.1145/3293611.3331591

[81] J. Zhang, J. Gao, Z. Wu, W. Yan, Q. Wo, Q. Li, and Z. Chen. 2019. Performance Analysis of the Libra Blockchain: An Experimental Study. In *2019 2nd International Conference on Hot Information-Centric Networking (HotICN)*. 77–83. https://doi.org/10.1109/HotICN48464.2019.9063213