The Impact of Network Topology on Performance Metrics and Energy Consumption for Blockchains: Towards Repeatable Benchmarking

VINCENZO P. DI PERNA, University of Urbino, Italy
VALERIO SCHIAVONI, University of Neuchâtel, Switzerland
MIGUEL MATOS, IST U. Lisboa and INESC-ID, Portugal
FRANCESCO FABRIS, University of Trieste, Italy
MARCO BERNARDO, University of Urbino, Italy

As blockchains transition from experimental prototypes to production systems, their evaluation must account not only for performance metrics such as throughput and latency, but also for energy costs as well as the repeatability (same setup, same results) and predictability (stable expectations) of assessment under controlled network conditions. While consensus design is a well-known driver, the role of the underlying network topology is still under-characterized and cloud-based studies are expensive and hard to reproduce.

We present Lilith, a reproducible, cost-aware, and system-agnostic blockchain benchmarking environment that supports both cluster deployments and large-scale emulation, enabling controlled changes of the network topology without altering the application logic. Using Lilith, we assess five blockchains (Algorand, Diem, Ethereum Clique, Quorum IBFT, and Solana) across five network topologies (fat-tree, full mesh, hypercube, scale-free, and torus) under realistic workloads spanning transfer transactions and smart-contract execution (DDoS, FIFA, gaming, GAFAM, PayPal, VISA). In addition to average performance and energy, we quantify run-to-run dispersion and use statistical tools (factorial ANOVA and ICC) to attribute variance across factors; we further present a four-level predictability taxonomy (P0-P3) and place our methodology at level P2.

Our results highlight a clear performance-energy trade-off: under intensive workloads, full mesh, hypercube, and torus tend to deliver higher performance, whereas fat-tree and full mesh yield the lowest energy per committed transaction. At the same time, throughput and latency can exhibit extreme worst-case deviations (exceeding 1,200%) in specific blockchain-topology-workload combinations, while energy is comparatively steadier and primarily workload-driven. Overall, Algorand and Diem provide strong energy efficiency with stable mid-range dispersion, Ethereum Clique shows the narrowest run-to-run spread but remains more energy-hungry, and Quorum IBFT and Solana are the most sensitive to topology and load, incurring increasing energy costs with scale under heavy workloads.

CCS Concepts: • **Computer systems organization** → **Distributed architectures**; • **Networks** → **Network performance analysis**; **Topology analysis and generation**; • **Hardware** → *Power and energy*; • **General and reference** → *Measurement*; *Validation.*

Additional Key Words and Phrases: blockchain benchmarking, distributed ledger technologies, network topology, performance evaluation, energy consumption, repeatability, reproducibility.

Authors' Contact Information: Vincenzo P. Di Perna, University of Urbino, Urbino, Italy, vincenzo.diperna@unicam.it; Valerio Schiavoni, University of Neuchâtel, Neuchâtel, Switzerland, valerio.schiavoni@unine.ch; Miguel Matos, IST U. Lisboa and INESC-ID, Lisbon, Portugal, miguel.marques.matos@tecnico.ulisboa.pt; Francesco Fabris, University of Trieste, Trieste, Italy, ffabris@units.it; Marco Bernardo, University of Urbino, Urbino, Italy, marco.bernardo@uniurb.it.

ACM XXXX-XXXX/2026/7-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Vincenzo P. Di Perna, Valerio Schiavoni, Miguel Matos, Francesco Fabris, and Marco Bernardo. 2026. The Impact of Network Topology on Performance Metrics and Energy Consumption for Blockchains: Towards Repeatable Benchmarking. 1, 1 (July 2026), 33 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Blockchains are distributed ledger systems that record transactions in an immutable and verifiable block sequence across a set of nodes that may not fully trust each other. As their adoption expands well beyond cryptocurrencies – touching finance [11], supply chains [76], healthcare [21], and other domains [48, 82] – a substantial effort has been devoted to understanding key design choices, from consensus protocols [38, 63, 66, 79, 89] to transaction processing capabilities (*e.g.,* transactions per second, TPS) [35]. In parallel, a rich ecosystem of benchmarking and simulation tools has emerged [6, 29, 34, 37, 50, 54, 68, 77] to evaluate blockchain behavior under controlled settings.

Yet, systematic and repeatable blockchain benchmarking is still challenging. One reason is that real-world blockchain deployments are large, heterogeneous, and dynamic: public peer-to-peer networks evolve continuously, often featuring complex and hard-to-observe connectivity patterns. For example, the Bitcoin network alone includes tens of thousands of reachable full nodes [14]. A second reason is practical: running large-scale, long-duration experiments in the cloud is expensive and frequently difficult to reproduce. For deployments of up to 200 high-performance nodes, we estimate costs in the order of 2,000-14,000 USD to keep the infrastructure running continuously across major providers such as Amazon Web Services (AWS), Google Cloud Platform (GCP), Azure, and Alibaba (Fig. 1). Even when costs are acceptable, cloud environments are problematic from a methodological standpoint. They offer limited transparency on the underlying network fabric and limited control on key network properties (*e.g.,* ad-hoc latencies, bandwidth caps, packet loss, and congestion patterns).

Moreover, the network itself is not stable: both geographical placement and time introduce measurable variability that undermines reproducibility across independent runs and several months. Fig. 2 highlights substantial differences among AWS regions, while Fig. 3 shows that link latencies fluctuate over time. Compared to a 2023 dataset [39], our measurements observe a 3% decrease in average latency (from 194 to 188 ms) and an 85.5% increase in average throughput (from 74.6 to 138.3 Mbps). These observations motivate a key point: *experimental repeatability* (reproducing outcomes under identical experimental conditions [1]) and *performance predictability* (delivering constant performance that is not easily degraded by adverse conditions [88]) become hard to assess when benchmarking is performed on opaque and time-varying network substrates, where it is difficult both to attribute performance outcomes to protocol design and to quantify run-to-run deviations [1, 88]. Nevertheless, most blockchain evaluations remain point-estimate-centric and often rely on a single run per configuration, implicitly treating deviations as noise. In practice, performance should be treated as a distribution: even under identical software and network conditions, independent runs can exhibit non-negligible variations. This motivates complementing average performance and energy results with explicit experimental repeatability and performance predictability analyses under controlled settings.

Accurate benchmarking is not only a matter of scientific rigor: it is also necessary to validate performance claims that may diverge from independently observed results. For instance, Solana reports up to 200,000 TPS and sub-second finality [103], whereas independent evaluations observed markedly lower throughput and significantly higher latency [39]. At the same time, performance alone is no longer sufficient as an evaluation target. Large-scale deployments raise sustainability concerns, as energy consumption can become substantial, especially when validation relies on intensive computation and specialized hardware. Energy discussions often focus on consensus choice (*e.g.,* Proof of Work (PoW) [36] vs. Proof of Stake (PoS) [66] and transitions from PoW to PoS [66]) and on mining/validator hardware. However, energy usage is also shaped by the communication substrate: the network topology influences message dissemination, latency, contention, and ultimately the amount

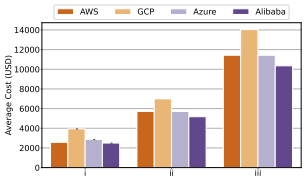


Fig. 1. Estimated weekly cloud cost comparison for three scenarios: (*i*) 10 nodes, 30/60 vCPUs, 64/120 GB RAM; (*ii*) 200 nodes, 4 vCPUs, 8 GB RAM; (*iii*) 200 nodes, 8 vCPUs, 16 GB RAM.

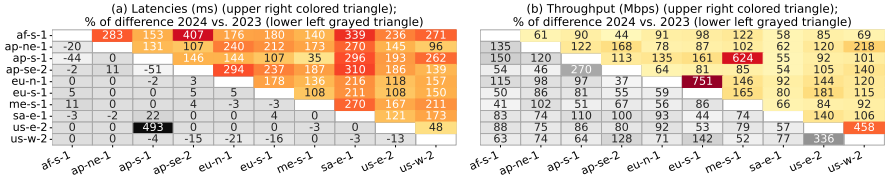


Fig. 2. Latencies (ms) and throughput (Mbps) heatmaps of our 2024 measurements (upper colored triangles); % of difference 2024 vs. 2023 (lower grayed triangles). The 10 AWS regions are: *af-s-1* (Cape Town), *ap-ne-1* (Tokyo), *ap-s-1* (Mumbai), *ap-se-2* (Sydney), *eu-n-1* (Stockholm), *eu-s-1* (Milan),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

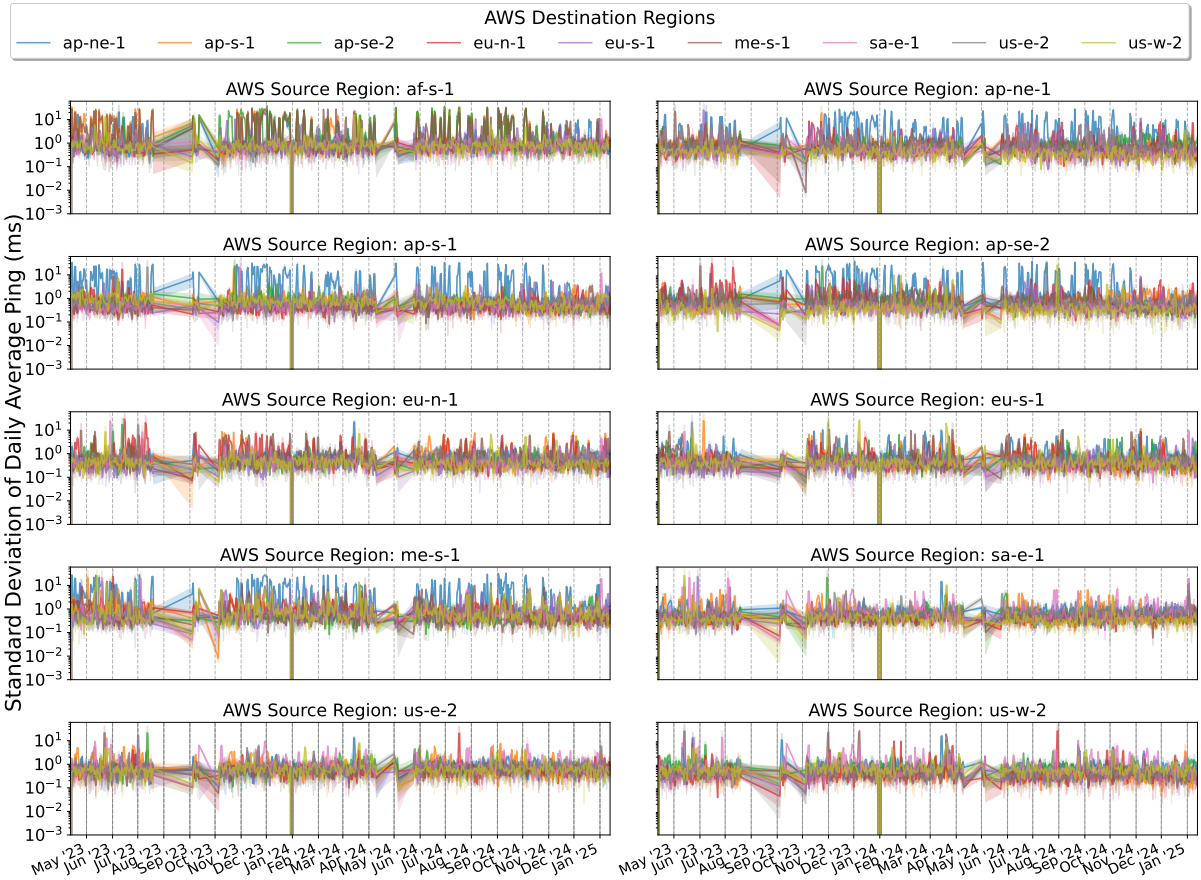


Fig. 3. Long-term ping standard deviation snapshots (Apr '23 – Jan '25) across 10 AWS regions (see Fig. 2).

of work performed per committed transaction. Despite growing attention to energy-aware blockchain analysis and tooling [16, 46, 93], the effect of network topology on energy efficiency remains largely under-explored.

This article argues that network topology should be treated as a first-class experimental factor in blockchain benchmarking [7, 53]. While a limited body of work considers network conditions (*e.g.,* latency effects on propagation and protocol parameters [26, 65] or disruptions such as node failures and network contraction [73, 75]), comprehensive experimental studies spanning multiple real-world blockchain systems and multiple representative topologies are still scarce – with few notable exceptions, such as evidence on hypercube benefits for Bitcoin [91]. We therefore study topology not as an incidental deployment detail, but as a controllable variable that may improve efficiency (performance and/or energy) without changing a blockchain's public or private nature.

To enable controlled and repeatable experimentation, we built Lilith, a network-controlled benchmarking framework that combines *Diablo* [39] (blockchain benchmarking) with *Kollaps* [7] (distributed network topology emulation) and integrates Intel Running Average Power Limit (RAPL) indicators to measure energy consumption. Lilith is system-agnostic at the orchestration and emulation layers, supports both clusters and large-scale emulation, and integrates new backends through thin adapters without changing the orchestrator. Its novelty is to benchmark real blockchain clients under controlled emulation, making topology and link constraints explicit and reproducible. This enables topology-aware analysis of performance, energy, and run-to-run variability under fixed configurations without relying on opaque cloud networking. Using Lilith, we evaluate five blockchain systems – Algorand [38], Diem [13], Ethereum Clique [98], Quorum IBFT [30], and Solana [103] – across five representative topologies – fat-tree, full mesh, hypercube, scale-free, and torus – under realistic workloads derived from prior benchmarking practice [39]. In particular, we consider payment-style transaction processing inspired by PayPal and VISA, an attack injection through a distributed denial-of-service (DDoS) workload, and smart-contract executions (FIFA, Google-Apple-Facebook-Amazon-Microsoft (GAFAM), gaming) modeling bursty requests for stock trades. Besides point estimates, we execute multiple independent runs for each blockchain-topology-workload configuration as well as quantify intra-configuration dispersion with robust indices (*e.g.,* interquartile range (IQR) and normalized standard deviation) and tail behavior via worst-case deviations from the configuration mean. We further attribute variability to experimental factors via factorial analysis of variance (ANOVA) [47] and assess run-to-run reliability via interclass correlataion coefficient (ICC) [49], separating systematic effects (design and topology/workload interactions) from residual noise. Within a four-level performance predictability taxonomy (P0-P3) that we introduce, our methodology targets level P2: quantified experimental repeatability under fixed software versions and fixed, deterministic network constraints and topologies.

Contributions. This journal article consolidates and extends our prior conference results [27, 28] into a unified performance-energy assessment under controlled network conditions. Besides consolidating prior work, the journal version adds (*i*) a four-level predictability taxonomy (P0–P3), (*ii*) a variance-centric benchmarking methodology focused on experimental repeatability and performance predictability under controlled geo-emulated networks, (*iii*) a new large-scale dataset of independent blockchain benchmark runs, and (*iv*) a dispersion-oriented analysis of typical and worst-case variability. More precisely:

- *Network variability evidence.* We report a 2-year cloud characterization trace[1] showing that cloud networking evolves over time and can materially affect benchmark outcomes.
- *Joint performance-energy insights.* We provide a systematic, topology-aware evaluation across blockchains and workloads, highlighting a concrete performance-energy trade-off: under intensive workloads, connectivity-rich topologies (*e.g.,* full mesh, hypercube, and torus) tend to improve performance, whereas fat-tree and full mesh typically minimize energy per committed transaction.
- *Blockchain-level findings.* Across configurations, Algorand and Diem exhibit the most stable run-to-run behavior and lowest energy per transaction; Ethereum Clique remains robust but more energy-demanding; Quorum IBFT is particularly sensitive to network dynamics; and Solana shows high energy demands and operational stress as scale increases under heavy workloads.

[1]Released at https://doi.org/10.5281/zenodo.11457019

- *Experimental repeatability and variance attribution.* We complement average performance-energy results with a variance-centric analysis across repeated runs, reporting typical dispersion and worst-case deviations and attributing variance to blockchain, topology, and workload factors via ANOVA and ICC.
- *Benchmarking taxonomy (P0-P3).* We introduce a lightweight four-level taxonomy to distinguish uncontrolled observations (P0) from controlled testbeds (P1), quantified experimental repeatability (P2), and predictive modeling (P3), then we position our study at level P2.
- *Reusable artifacts.* We release ready-to-use network topologies, traces, and a large-scale dataset of independent blockchain benchmarks (https://doi.org/10.5281/zenodo.17681717) to support repeatable experimentation in addition to the specific blockchain configurations studied here.

Roadmap. Section 2 presents related work while Section 3 discusses background details. Section 4 illustrates Lilith and its measurement pipeline, including energy instrumentation. Section 5 reports performance, energy, and run-to-run variance results, followed by a joint discussion in Section 6. Finally, Section 7 concludes the article with future work.

2 Related Work

Benchmarking frameworks have proliferated to study throughput, latency, and protocol behavior [6, 29, 34, 37, 39, 50, 54, 68, 77]. However, repeatable benchmarking remains difficult because network conditions are rarely treated as a controllable experimental dimension. Moreover, most toolchains and experimental studies are still point-estimate-centric: they commonly report single-run averages (or coarse percentiles) and do not systematically quantify run-to-run dispersion under identical configurations. As a result, repeatability (stability of outcomes under the same setup) and performance predictability (how tightly a fixed configuration confines its performance variability) are rarely treated as first-class evaluation targets [1, 88]. This is problematic because topology and link properties influence dissemination, fork/conflict rates, and queueing/contention patterns, thus they can distort both performance and energy outcomes [7, 53]. While some work considers the effect of latency and propagation on protocol parameters [26, 65] and other studies examine disruptions such as node removals or network contraction [73, 75], comprehensive evaluations across multiple real-world blockchain implementations and multiple representative topologies are still limited, aside from notable evidence on topology effects in specific contexts (*e.g.,* hypercube benefits for Bitcoin) [91].

Table 1 provides an overview of the diversity and features of existing tools. Specifically, we report their publication year, programming language, supported network configurations, orchestration strategies, execution mode, portability, workload support, the presence of configurable network controls, whether its venue offered an artifact evaluation (AE) process, and the status of ACM and IEEE badging [1, 45]. For the last feature, we mark tools published before the introduction of the corresponding artifact and reproducibility schemes (2016 for ACM, 2019 for IEEE) with "–", indicating that badges were not yet available [15, 69]. The legend maps symbols to full, partial, no, or unknown support. Notably, almost all of the listed tools lack ACM/IEEE badges for reproduction/replication, though a few provide archived artifacts [1, 45]. We therefore treat badges as auxiliary signals: our analysis focuses on operational capabilities required by variance-centric evaluation – fine-grained network control, topology enforcement, deterministic orchestration, and support for repeated runs – marking documentation gaps as "data unavailable". This last capability is crucial to move beyond single-run claims and to make variability (and thus predictability) measurable rather than anecdotal.

Many tools prioritize scalability by abstracting away real client behavior, but this can reduce fidelity. For instance, Minichain [99] and BlockLite [96] emulate only PoW dynamics, while several simulators [5, 32, 33] rely on synthetic transaction/block models (*e.g.,* wallet creation, signing, simplified message exchange) and may also support dynamic events [5]. Since they do not execute real client code, these approaches can miss resource hot-spots and internal bottlenecks (*e.g.,* data-structure contention) that materially affect throughput

Table 1. Overview of features supported by blockchain benchmarking tools. Legend: ⁘ data unavailable; ✗ not reported; ◑ partially reported; ✓ fully reported; "–" badge scheme not yet in place (ACM/IEEE since 2016/2019). Abbreviations used: Centr. = Centralized, Distr. = Distributed, Simul. = Simulation, Emul. = Emulation, SC = Smart Contracts, TT = Transfer Transactions. Boldface denotes Lilith, the framework introduced in this article.

									ACM Badges			IEEE Badges		
Tool	Year	Programming Language	Orchestration	Mode	Portability	Workload	Network Controls	AE Process	Artifacts Avail.	Results Reprod.	Results Replic.	Code/Data Avail.	Results Reprod.	Code/Data Reviewed
Shadow-Bitcoin [62]	2015	Python	Centr.	Simul.	✗(Bitcoin)	TT	⁘	⁘	–	–	–	–	–	–
HIVE [31]	2016	Go	Distr.	Simul.	✗(Ethereum)	⁘	⁘	⁘	✗	✗	✗	–	–	–
Simcoin [80]	2016	Python	Centr.	Simul.	✗(Bitcoin)	TT	⁘	⁘	✗	✗	✗	–	–	–
Bitcoin-Simulator [36]	2016	C++	Centr.	Simul.	✓(Bitcoin-like)	TT	◑	⁘	✗	✗	✗	–	–	–
BlockBench [29]	2017	C++,Go	⁘	Simul.	✓(Private)	SC	⁘	✓	✗	✗	✗	–	–	–
CIDDS [50]	2018	Python	⁘	Simul.	✓(DAG-based)	⁘	◑	⁘	✗	✗	✗	–	–	–
Caliper [44]	2018	Java	Distr.	Emul.	✓(Ethereum,Hyperledger)	SC	◑	⁘	✗	✗	✗	–	–	–
Minichain [99]	2019	Python	Centr.	Emul.	✓	⁘	◑	⁘	✗	✗	✗	✗	✗	✗
BlockLite [96]	2019	Java	Centr.	Emul.	✓(PoW-public)	⁘	⁘	⁘	✗	✗	✗	✗	✗	✗
BlockSim-f [32]	2019	Python	Centr.	Simul.	✓	⁘	◑	⁘	✗	✗	✗	✗	✗	✗
SimBlock [9]	2019	Java	Centr.	Simul.	✓	TT	◑	⁘	✗	✗	✗	✗	✗	✗
BlockSim-m [5]	2020	Python	Centr.	Simul.	✓	TT	◑	⁘	✗	✗	✗	✗	✗	✗
Core-Bit-Netw-Simul. [6]	2020	Python	Centr.	Simul.	✓	⁘	◑	⁘	✗	✗	✗	✗	✗	✗
BBB [68]	2020	Python	Centr.	Emul.	✓(Private)	⁘	◑	⁘	✗	✗	✗	✗	✗	✗
SIMBA [33]	2020	Python	Centr.	Simul.	✓	⁘	◑	⁘	✗	✗	✗	✗	✗	✗
BCTMark [77]	2020	Python	Distr.	Emul.	✓	TT	◑	⁘	✗	✗	✗	✗	✗	✗
BlockPerf [72]	2021	C/C++,Python	Distr.	Both	✗(Bitcoin)	TT	✗	⁘	✗	✗	✗	✗	✗	✗
DLPS [34]	2021	Python	Distr.	Emul.	✓	TT	◑	⁘	✗	✗	✗	✗	✗	✗
Gromit [64]	2022	Python	Distr.	Emul.	✓	SC	◑	⁘	✗	✗	✗	✗	✗	✗
Diablo [39]	2022	Go,Python,Perl	Distr.	Emul.	✓	TT,SC	✗	✓	✓	✗	✗	✗	✗	✗
JABS [101]	2023	Java	Centr.	Simul.	✓	TT	◑	⁘	✗	✗	✗	✗	✗	✗
TangleSim [54]	2023	Go,Python	⁘	Simul.	✓(DAG-based)	⁘	✗	⁘	✗	✗	✗	✗	✗	✗
COCONUT [37]	2023	Jinja,Java	Distr.	Emul.	✓	TT,SC	⁘	✓	✓	✗	✗	✗	✗	✗
BBSF [74]	2023	C++,Go	⁘	⁘	⁘	⁘	✗	✗	✗	✗	✗	✗	✗	✗
GFBE [55]	2024	⁘	Distr.	Emul.	✓	SC	✗	⁘	✗	✗	✗	✗	✗	✗
STABL [41]	2024	⁘	Distr.	Emul.	✓	TT,SC	◑	⁘	✗	✗	✗	✗	✗	✗
Hammer [94]	2024	⁘	⁘	⁘	⁘	⁘	✗	✗	✗	✗	✗	✗	✗	✗
RBlockSim [71]	2025	⁘	⁘	⁘	⁘	⁘	◑	✓	✓	✓	✗	✗	✗	✗
Lilith [27]	**2025**	**Bash, Python**	**Distr.**	**Emul.**	✓	**TT, SC**	✓	✓	✓	✗	✗	✗	✗	✗

and latency [33, 72]. SimBlock [9], for example, is event-driven and samples block creation times probabilistically rather than deriving them from node-specific mining difficulty and execution behavior.

On the other side of the spectrum, practical test environments exist but frequently lack either introspection or resource/network reservation capabilities. HIVE [31] (Ethereum testing) cannot inspect the internal client state, whereas Hyperledger Caliper [44] does not provide resource reservation primitives. Several benchmark suites execute real implementations but rely on ad-hoc deployment scripts tightly coupled with a given testbed. BlockBench [29] and Diablo [39] fall into this category: they provide high-fidelity measurements, yet their orchestration and network modeling are not inherently topology-aware and wide-area behavior is not captured without additional support – despite recent attempts in this direction [41, 51]. Other frameworks improve portability (*e.g.,* BCTMark [77] via YAML-described structures) or configurability (*e.g.,* COCONUT [37]), or integrate partial network emulation (*e.g.,* Core-Bitcoin Network Simulator [6] via Pumba [52] plus analysis/plotting). BBB [68], JABS [101], and DLPS [34] address additional limitations in earlier toolchains.

Existing tools rarely combine real-client execution, topology control, integrated energy measurement, and repeated-run variance analysis in one pipeline. Lilith makes network topology a first-class experimental factor.

3 Background

A blockchain is a distributed ledger that stores transactions in an append-only structure. Transactions are grouped into blocks that are cryptographically linked to previous blocks, forming a tamper-evident chain. Since no single trusted party is assumed, nodes rely on a consensus protocol to agree on the block order and on the ledger state. Consensus designs span resource-based mechanisms – such as PoW [36, 63] and PoS [66] – as well as authority- and committee-based approaches like Proof of Authority (PoA) [89] and Byzantine Fault Tolerance (BFT) families [17, 38, 79, 102, 105]. In addition to simple transfers, many platforms support *smart contracts* [98, 106], *i.e.,* programs executed by the blockchain to enforce application logic without intermediaries.

From a deployment standpoint, blockchains are commonly classified as: (*i*) public (permissionless), where any node can join and validate; (*ii*) private (permissioned), with restricted participation and potentially centralized governance; and (*iii*) consortium, jointly operated by multiple organizations [92, 100]. We evaluate five widely used systems spanning these categories: Algorand [38], Ethereum Clique [98], and Solana [103] (public); Diem [13] (private); and Quorum IBFT [30] (consortium).

The remainder of this section introduces the two background elements that motivate our design choices. First, Section 3.1 discusses why energy consumption cannot be treated only as a protocol-level or computation-level issue, but must also be analyzed through the lens of network topology. Then, Section 3.2 introduces a predictability-oriented taxonomy of blockchain benchmarking studies, which clarifies how this work differs from prior evaluations in terms of network control, repeated executions, variance analysis, and predictive strength.

3.1 Energy Consumption and Why Network Topology Matters

Energy has become a first-order concern for large-scale blockchain deployments. For PoW-based systems, energy consumption is dominated by the competitive solution of cryptographic puzzles, where only one miner succeeds and the remaining work is effectively wasted [46]. More generally, validating transactions features at least two components: local computation and communication overhead due to message exchange among nodes [23]. In PoW, computation is so dominant that network energy may appear negligible [23]; however, for many non-PoW systems and for smart-contract-heavy workloads, communication patterns, dissemination mechanisms, and contention increasingly shape overall efficiency. Hence, the network topology – by affecting who talks to whom, how fast information spreads, and how much duplication occurs – can influence both performance and energy, especially at scale and under intensive workloads [39, 91].

Existing work on blockchain energy analysis largely favors post-deployment approaches [8, 16, 23, 43]. A representative example is the Cambridge Blockchain Network Sustainability Index (CBNSI) [16], which reports energy-related indicators for major public blockchains. In contrast, predictive approaches that rely on controlled experiments (testbeds/benchmarks) are less common and, when existing, they frequently rely on manual instrumentation or indirect estimation. For instance, even when using mature benchmarking workflows such as BCTMark [77], energy is often obtained via external measurements and/or proxies (*e.g.,* gas-based estimates) rather than being an integrated, systematic output of the evaluation pipeline [78].

A further limitation is that many predictive studies evaluate simplified or constrained network settings, often driven by IoT/SDN scenarios. Examples include FPGA-based testbeds for estimating Bitcoin energy usage [43] and analyses emphasizing the role of broadcast protocols and network size – issues that become even more pronounced in constrained environments [8]. Other SDN/IoT-oriented frameworks focus on routing comparisons [104] or security aspects [12, 83] without quantifying energy, sometimes by offloading consensus away from devices (thereby reducing actual on-device energy costs). Model-based approaches that incorporate network-related terms (*e.g.,* messages per transaction) provide useful intuition [23], but often cannot explore network scaling or topology diversity due to resource constraints. Interestingly, for some non-PoW systems (*e.g.,* Ripple [18] and Stellar [59]), communication can dominate energy costs, suggesting that network-level factors cannot be ignored in general [23].

3.2 Proposing a Predictability Taxonomy

We structure blockchain benchmarking practice into four levels of performance predictability:

P0 (uncontrolled) covers evaluations with no control over the execution substrate (*e.g.,* mainnet observations), where variability is observable but confounded by evolving environments.

P1 (controlled, unquantified) includes controlled testbeds that may fix software and workloads but typically report single-run averages (or coarse percentiles) without systematic multi-run dispersion analysis.

Table 2. Representative blockchain performance studies and their position in the P0–P3 taxonomy. Legend: ✓ variance reporting; ◑ some distributional analysis (*e.g.,* , percentiles) but not under controlled repeated setups; ◌ data unavailable; ✗ single-run or averages only. $B \times T \times W \times S$ = Blockchains × Topologies × Workloads × Scales (validator-set sizes). Boldface denotes the present work.

Work	Year	Environment	Variance focus	Level	$B \times T \times W \times S$	Runs
[29]	2017	Local testbed, multi-platform	✗	P1	$3 \times 1 \times 6 \times 7$	5
[44]	2018	Cloud/local, cross-platform	✗	P1	$3 \times 1 \times 3 \times 1$	1
[77]	2020	Energy-centric testbed	◑	P1	◌	◌
[74]	2023	Single-machine multi-node testbed	✗	P1	$2 \times 1 \times 2 \times 3$	3
[37]	2023	Latency benchmarking	◑	P1	$7 \times 1 \times 6 \times 4$	◌
[67]	2023	Ethereum mainnet	✓	P0	$1 \times 1 \times 1 \times 1$	1
[56]	2023	Bitcoin traces + queueing model	✓	P0	$1 \times 1 \times 1 \times 1$	1
[39]	2023	Cloud, cross-platform	✗	P1	$6 \times 1 \times 5 \times 2$	◌
[41]	2024	Tail-latency analysis on testbed	◑	P1/P2	$5 \times 1 \times 1 \times 1$	◌
[94]	2024	Cloud testbed	✗	P1	$4 \times 1 \times 1 \times 1$	3
[71]	2025	Parallel/distributed simulator	✗	P1	$1 \times 1 \times 2 \times 4$	5
[27]	2025	Geo-emulated testbed	✗	P1	$5 \times 5 \times 5 \times 2$	3
[28]	2025	Geo-emulated testbed	✗	P1	$5 \times 5 \times 3 \times 2$	3
This work	**2025**	**Geo-emulated testbed**	✓	**P2**	$\mathbf{5 \times 5 \times 3 \times 2}$	**10**

P2 (quantified repeatability) denotes controlled experiments with enforced network constraints and topologies as well as repeated independent runs per configuration, enabling explicit dispersion reporting and variance attribution.
P3 (predictive) requires models that can predict performance distributions *ex-ante* and are validated against measured multi-run outcomes.

This taxonomy highlights why many existing toolchains (Table 2) effectively stop at P0/P1: without enforceable network control and repeated runs, predictability claims remain anecdotal. Our goal is to push topology-aware benchmarking at least to P2 by combining deterministic orchestration, explicit topology and network-constraint enforcement, and repeated runs with integrated energy measurements.

4 Lilith Overview

Lilith is a blockchain benchmarking framework that we developed to support multiple deployment environments including clusters. At its core, Lilith is system-agnostic: it coordinates Diablo's workflow (see Section 4.1) and Kollaps's emulation capabilities (see Section 4.2) without being tied to a specific blockchain family or client. 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. Blockchain-specific support is confined to thin adapters, so new backends can be added without changing the core orchestration or workload engines. Fig. 4 illustrates the execution flow of Lilith, which we detail in Sections 4.3–4.8. In addition to enabling topology-aware performance and energy studies as well as reporting point estimates, we run a variance-centric campaign with repeated independent executions and variance attribution to assess run-to-run dispersion and performance predictability under identical configurations.

Our Lilith framework targets a gap that emerges clearly from the literature: enabling experiments that encompass (*i*) real blockchain clients and realistic workloads, (*ii*) explicit control over network topologies and link properties (latency, bandwidth, loss, jitter, *etc.*), (*iii*) integrated energy measurements without manual instrumentation, and (*iv*) repeated independent runs for quantified repeatability analysis. Lilith combines real-client benchmarking, topology control, energy measurement, and repeated runs for P2-level evaluation.

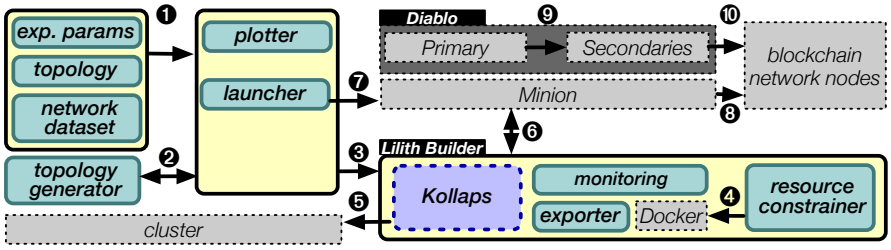


Fig. 4. Architecture and execution flow of a Lilith experiment (dashed contour lines represent existing components).

4.1 The Diablo Benchmark Suite

We selected Diablo [39] as a building block since it is the one matching most of the features in Table 1 and supports the different blockchain types mentioned in Section 3. 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 [39] introduces a set of Perl scripts called *Minion* [40], 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.

4.2 The Kollaps Network Emulator

To assess the impact of network properties on blockchain systems we leveraged Kollaps [7], a decentralized network emulator for large-scale applications that ensures end-to-end properties (*e.g.,* latency, bandwidth, and packet loss). Its fully distributed model guarantees 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 OSPF that calculates routes based on the shortest path tree – by relying on criteria such as latency or hop count. Kollaps supports native processes and virtual machines (VMs) 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.

4.3 Network Topology Generation

The first step in the execution flow of a Lilith experiment (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.

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. In our experiments, we evaluate the five network topologies – fat-tree, full mesh, hypercube, scale-free, and torus. We use these topologies as controlled proxies for overlay properties – fan-out, path diversity, and degree heterogeneity – rather than exact replicas of production networks. Dynamic or hybrid overlays may change exact rankings, but not the structural dimensions compared here. Many blockchains operate over public or permissioned overlays rather than raw Internet routing; thus, modeling fan-out,

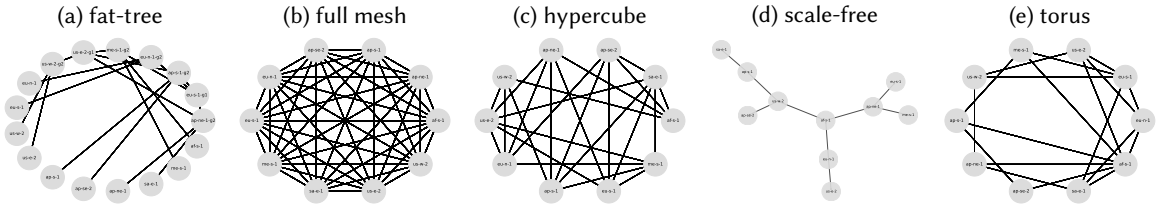


Fig. 5. Real-world network topologies supported by Lilith.

path diversity, and degree distribution is more informative than mirroring physical wiring. In our emulator, each topology is exercised under geography-informed latency constraints and time-varying network conditions representative of wide-area operation, enabling controlled tests of structured fan-out (*e.g.,* fat-tree/torus), dense committee communication (full mesh), and heterogeneous peer connectivity (scale-free). 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.

In a **full mesh** topology 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 a 4-bit binary vector 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 [20]. For our 10-AWS-region setting, we map each region to one gateway/node in the topology and then generate the inter-region links through the preferential attachment algorithm [20], so that nodes with higher degrees are more likely to be selected as new connections. Thus, the 10 regions are encoded as the 10 nodes of the scale-free graph, while measured RTTs are assigned to the corresponding inter-region links. This topology resembles Internet-like (WAN) networks [20].

A **torus** topology connects nodes in a wrap-around grid, enabling efficient data transfer, and is often used in supercomputing [95]. 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.

Table 3(a) 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.

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

Table 3. Selected topologies (a), workloads (b), and 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)	$N-1$	$N(N-1)/2$	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 integrated in Lilith.

Workload	Type	Scenario	Duration (s)	Injected (TPS)
DDoS	Transfer transaction	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 transaction	Constant rate	300	200
VISA	Transfer transaction	Constant rate	300	1,800

(b) Workloads integrated in Lilith.

Blockchain	Consensus	VM	DApp	Block Finality (s)	Claimed TPut (TPS)
Algorand	BBA [19, 38]	AVM	PyTeal [3]	3.3 [4]	7.5K [4]
Diem	HotStuff [105]	MoveVM	Move	100 [70]	60–1K [107]
Ethereum	Clique [89]	geth	Solidity	10–20 [58]	10–15 [81]
Quorum	IBFT [79]	geth	Solidity	2–15 [60]	0.7K–2.5K [25]
Solana	TowerBFT [102]	Sealevel	Solang	12 [39]	65K [22]

(c) Blockchains integrated in Lilith.

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.

4.4 Built-in Configurations for Workloads

Lilith ships with a set of ready-to-use configurations that cover both the demand side (workloads) and the supply side (blockchain backends), including setups that reproduce configurations from prior work [39]. The goal is to enable immediate, comparable experiments without bespoke setup, while still allowing users to tailor profiles to their own questions.

The bundled workloads in Table 3(b) are selected to evaluate blockchain performance under realistic, diverse conditions (not to mirror current Internet traffic). They span (*i*) payment-style transfer workloads at moderate and high transaction rates (PayPal and VISA), (*ii*) bursty and contention-heavy smart-contract workloads derived from application traces (FIFA, GAFAM, and Gaming), and (*iii*) stress-oriented overload conditions (DDoS). Consequently, the injected TPS values are controlled benchmarking inputs rather than claims that all profiles mirror current production traffic. We purposely include traces with short-lived peaks and longer steady windows so that experiments can probe both overload and stability regimes under controlled overlays.

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 [10]. 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 https://www.nasdaq.com/, 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*) [87]. The trace lasts 276 seconds, invoking at an almost constant high update rate of 13,000 injected TPS.

The **PayPal** profile represents a moderate payment workload derived from the 193 TPS average in [61]. We model it as a constant 200 TPS workload over 5 minutes.

Similarly, the **VISA** profile represents a higher-rate payment-style workload. Starting from the 1,770 TPS figure reported in [39], we approximate it with a constant workload of 1,800 injected TPS.

Extensibility is supported by allowing new workloads to be added through the same declarative interface used by the bundled profiles, without changing the orchestrator.

4.5 Blockchains under Test

The default backend set covers heterogeneous blockchain designs in consensus, execution, and networking so that comparisons are not bound to a single family. We include a public PoS chain with fast finality (Algorand), a permissioned BFT design (Diem with HotStuff), two EVM-based stacks with distinct finality models (Ethereum Clique and Quorum IBFT), and a high-throughput PoS+BFT system whose execution layer supports parallel transaction processing (Solana). To ensure consistency, we used the blockchain configurations from [39], with each blockchain version specified by the respective repository commits. Their high-level traits are reported in Table 3(c); here we summarize the rationale for their inclusion.

Algorand [19, 38] uses a pure PoS consensus algorithm, selecting nodes via cryptographic self-sortition to propose and verify 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 [24]. 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 via Algorand at commit `116c06e`.

Diem [13] uses an adapted version of HotStuff [105] 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. Our testing was based on Diem's testnet branch at commit `4b3bd1e`. Though no longer active, Diem provides valuable insights into permissioned infrastructures.

Ethereum [98] is a public blockchain platform for decentralized applications (DApps) and smart contracts. In our experiments, the Clique PoA variant [89] 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 [30] is a permissioned, enterprise Ethereum client that preserves Ethereum JSON-RPC while adding permissioning and optional private transactions via an external transaction manager. Following previous studies [39], we configured it with IBFT [79] at commit `919800f`. IBFT runs a round-based BFT protocol that tolerates up to f Byzantine validators (with total validator set size $n = 3f + 1$). A block is finalized once it gathers a supermajority quorum of at least $2f+1$ votes (*i.e.,* $> \frac{2}{3}$ of the validators), yielding deterministic finality (no chain reorganizations except under validator-set reconfiguration), and improving robustness over PoA-style schemes [89]. We benchmarked the public transaction path and detected commits via JSON-RPC receipt confirmation after block inclusion.

Solana [103] faces forks like Ethereum and requires 30 confirmations to finalize transactions [86]. Blocks are added every 400 milliseconds, with a simplified data structure and EdDSA signature replacing ECDSA. 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`.

For each backend, Lilith bundles minimal adapters (RPC/ABI, transaction encoders, confirmation/finality probes) and default node/client configurations aligned with prior work [39]. This makes Lilith system-agnostic at the orchestration and emulation layers: new blockchains can be integrated by implementing only the thin adapter interface (submit, poll/confirm, encode/decode, deploy), without modifying the orchestrator or the workload engines.

4.6 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.,* VMs, 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 [40] 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) by orchestrating the execution of the 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 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 that collects network usage through `vnstat` [90]. An exporter module further supports post-processing of benchmark results and execution traces. 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 the next sections, allowing performance and network metrics to be checked.

4.7 Metrics Gathering

Lilith measurement pipeline combines client-side traces, validator logs, overlay counters, host-level resource monitors, and (when available) energy probes. Time is aligned by the orchestrator so that per-source timestamps can be merged without *post-hoc* drift correction.

We report five primary signals. First, the *commit ratio* quantifies the fraction of submitted transactions that are durably committed (committed/submitted), making admission failures visible under overload or churn. Second, *throughput* counts committed TPS over sliding windows, with both sustained and peak values. Third, *latency to confirmation* (often called block finality in prior work [39]) measures end-to-end time from issuance to first confirmation at the chain's finality layer. Fourth, *network load* captures per-interface throughput in Mbps and effective overlay utilization. Fifth, *energy* probes based on Intel RAPL provide host-level processor/package energy indicators and cumulative consumption (kWh), normalized as energy per committed transaction for cross-run comparability. These values are a controlled proxy for computational energy, not a full whole-system estimate.

Besides these primaries, Lilith collects host metrics (CPU, memory, disk I/O) to assist diagnosis. All raw signals are exported as CSV/JSON with an explicit schema and bundled with plotting notebooks to reproduce

the next-section figures. To preserve comparability, the P2 campaign mandates dispersion-aware reporting of run-to-run offsets under identical seeds and network constraints, not point estimates alone.

4.8 Testing Configurations

Lilith can run experiments on different testbeds (*e.g.,* single machine, clusters, cloud platforms), where it installs the required containers and the Kollaps agents, enforces resource caps via `cgroups`, and deploys the emulated overlay. Unless noted, nodes run in Docker with pinned images and CPU affinities; analogous VM- or native-process setups are possible but not required. In this article, validator denotes any node participating in the consensus/finality protocol (*e.g.,* PoS validators, PoA signers, or BFT replicas). Unless stated otherwise, experiment scale (*e.g.,* "10 nodes", "40 nodes") refers to validator-set size (number of consensus-participating nodes), while clients only inject workload and seed nodes are auxiliary bootstrap peers not counted as validators.

We vary three classes of factors:

- *Topology and placement*: we instantiate the families in Table 3(a) and assign validators and Lilith's entities to regions/gateways according to the target graph; per-link latency, bandwidth, jitter, and packet loss are taken from the selected network dataset and enforced by Kollaps.
- *System scale and resources*: we tune the number of validators per region and the client fan-out, then we bound host resources (CPU cores, memory limits, and per-container I/O) to emulate different instance classes.
- *Demand and resilience*: we execute the bundled workloads in open- or closed-loop mode and inject controlled perturbations – packet drops, bandwidth caps, latency spikes, link and node failures, and temporary partitions – to observe admission, backlog growth, and recovery.

Fault injections are scripted with absolute times or event triggers (*e.g.,* "after backlog exceeds threshold"), with their effects being reflected in both performance and overlay traces. This design allows us to answer like-for-like questions – *e.g.,* how a change in overlay diameter affects tail latency under the same workload and resource envelope – without confounding cloud variance.

5 Results

We evaluate how network topology influences blockchain behavior by focusing on the key performance indicators introduced in Section 4.7. The results below come from executed experiments on our on-premises geo-emulated cluster. For each configuration, we instantiate the selected topology, dataset, workload, and resource constraints; deploy and bootstrap the backend through Minion; inject the workload; collect performance, network, and energy measurements; and export logs for post-processing and repeated-run analysis. This workflow is applied independently to every blockchain/topology/workload configuration. We distinguish two complementary benchmark campaigns:

- Performance-energy campaign: we use the full matrix reported in Table 3 (five blockchains, five topologies, and six workloads) to derive topology-aware performance evaluations (Section 5.1), while we use a restricted set of workloads (GAFAM, PayPal, VISA) for the energy benchmarks (Section 5.2).
- Variance-centric P2 campaign: to keep the additional overhead minimal yet representative, we select three canonical workloads (GAFAM, PayPal, VISA) and two validator-set sizes (10 and 40).

Each configuration is executed in ten independent runs; for the substantially longer network-dynamics experiments, we report averages over three runs due to time constraints. Ten runs are a practical compromise between statistical depth and campaign breadth. Given the large number of blockchain-topology-workload-scale configurations and the cost of each geo-emulated execution, this budget supports comparative P2-level repeatability analysis and variance attribution, while finer distributional and tail characterization is left to future work. Each run uses pinned software versions, deterministic orchestration, and fixed network constraints and topologies, so that observed dispersion can be attributed to system and factor effects rather than infrastructure drift.

All experiments run on an on-premises cluster of 7 dual-socket servers, each with two 16-core Intel Xeon CPUs (32 hardware threads) and 128 GB RAM, interconnected via a non-blocking 40 GbE switch and running Ubuntu 22.04 LTS (kernel 5.15). This avoids the opacity and time-varying noise of public clouds while retaining realistic WAN conditions by enforcing per-link network constraints (latency/bandwidth) with Lilith.

Our evaluation targets the following research questions (RQs):

RQ1 How does topology affect blockchain performance and which is best for each blockchain?
RQ2 What is the impact on network perturbations on performance, such as packet loss, congestion, node failures, and increased latency, and which blockchain is most affected?
RQ3 How does network topology influence energy efficiency, in terms of, *e.g.,* total kWh and kWh/committed transaction, and what trade-offs emerge among energy, throughput, and latency?
RQ4 How repeatable and predictable are performance and energy outcomes under controlled, geo-emulated topologies, aiming at level P2, and which factors explain the observed variability?

In Section 5.1 we report commit-level performance for the small geo-distributed deployment (10 nodes, one per region), highlighting how workload intensity and overlay topology affect commit ratio, throughput, and latency (Fig. 6). In Section 5.2 we then scale out to 40 nodes (four per region) to expose gateway contention and scalability limits under the same workloads and topologies (Fig. 6). Finally, in Section 5.3 we shift from commit metrics to the underlying communication cost by analyzing aggregate network throughput across platforms, topologies, and workloads (Fig. 7).

5.1 Performance Measurements

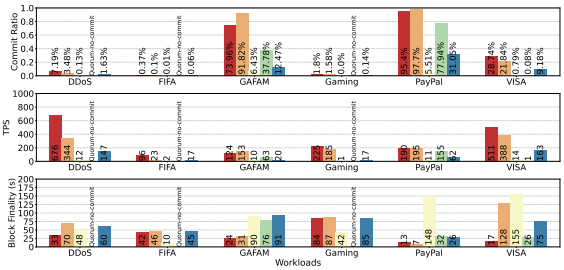
We now quantify how workload and overlay topology shape blockchain performance under geo-distributed deployments. We report the commit ratio, *i.e.,* the fraction of injected transactions that are successfully committed within the experiment window, and the corresponding committed throughput/latency trends (Fig. 6). We first consider a small deployment with one node per region (10 nodes total), then scale to four nodes per region (40 nodes total) to stress gateway contention and protocol scalability. Finally, we complement commit-level results with aggregate network load (Fig. 7) and controlled degradations (packet loss, bandwidth congestion, node crashes, and increased latency) to assess robustness under adverse conditions.

5.1.1 Small Deployment (10 Nodes). We first deploy one node per region (10 nodes total). The corresponding results are shown on the left side of Fig. 6 for each topology. For moderate workloads (PayPal and GAFAM), Algorand and Diem achieve commit ratios above 80%, approaching 100% for PayPal; Diem is typically marginally ahead. Performance is generally constant across topologies, with one notable exception: under the GAFAM workload on a scale-free topology, Algorand experiences a marked throughput reduction (down to 78 TPS), corresponding to roughly a 50% drop compared to other topologies.

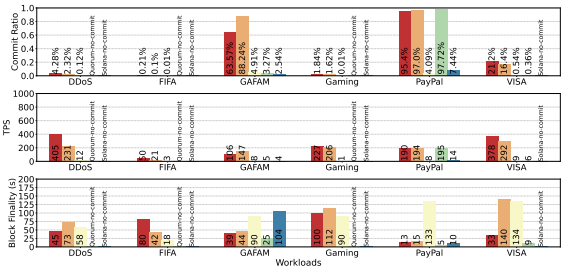
Quorum attains a commit rate above 90% for PayPal in hypercube (Fig. 6e) and torus (Fig. 6i), which we attribute to the combination of lower contention in torus and the high connectivity of hypercube that facilitates transaction retrieval. Solana exceeds 75% commit ratio in full mesh (Fig. 6c) and hypercube (Fig. 6e); in the remaining topologies, congestion reduces its commit rate below 50%.

Under the most demanding workloads, commit ratios collapse across platforms. For Algorand, commit rate stays below 8% for high-rate transfer injections (DDoS, see Fig. 6c) and below 3% for intensive smart-contract workloads (Gaming and FIFA, see Fig. 6a). A plausible explanation is Algorand's large transaction pool (up to 75,000 transactions) [2], which can absorb bursts but still saturates under extreme injection rates.

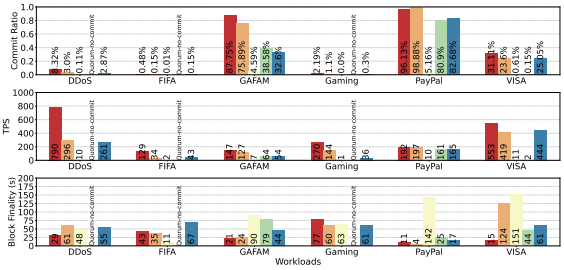
As also reported by prior work [39], Quorum does not commit transactions for the most demanding workloads. We attribute this to a leader bottleneck in the BFT-style protocol used by Quorum, which can saturate memory pools and/or network queues when the system is pushed hard.



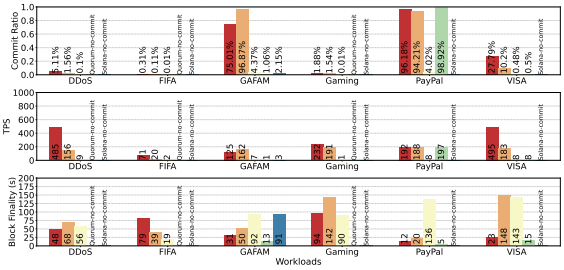
(a) Fat-tree, one node/region.



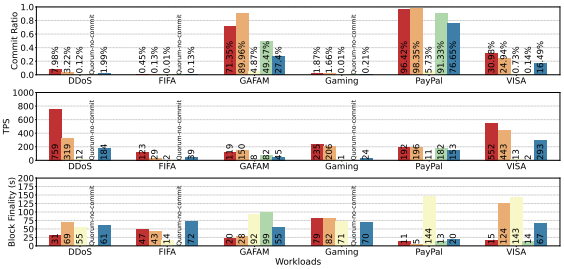
(b) Fat-tree, four nodes/region.



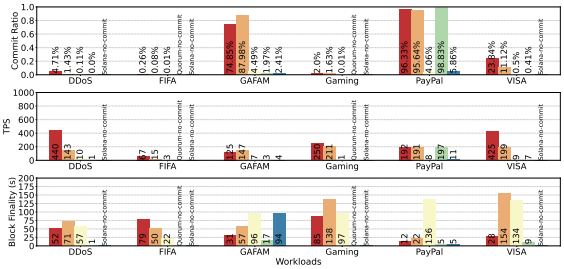
(c) Full mesh, one node/region.



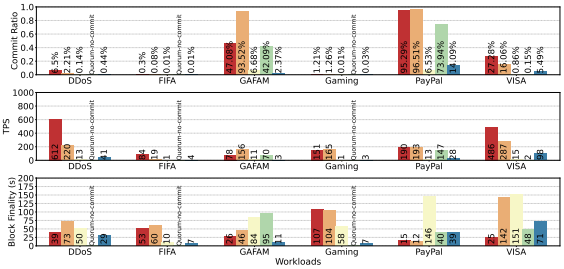
(d) Full mesh, four nodes/region.



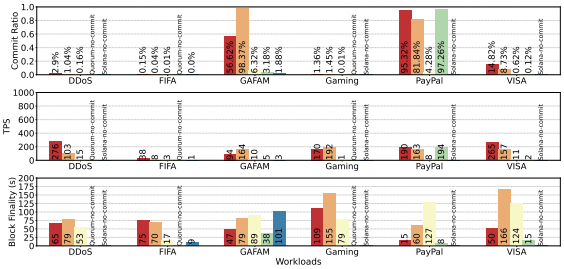
(e) Hypercube, one node/region.



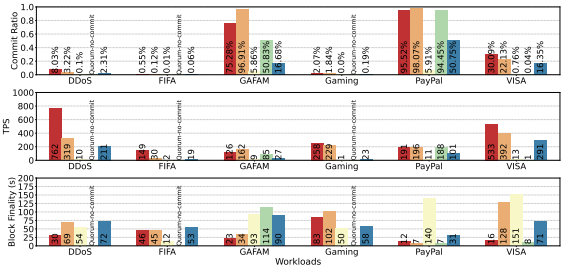
(f) Hypercube, four nodes/region.



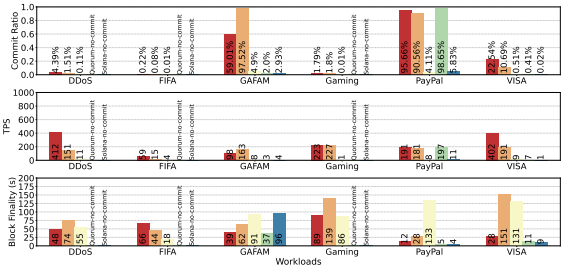
(g) Scale-free, one node/region.



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



(i) Torus, one node/region.



(j) Torus, four nodes/region.

Fig. 6. Blockchain performance using the 2023 AWS dataset (**Algorand**, **Diem**, **Ethereum**, **Quorum**, **Solana**).

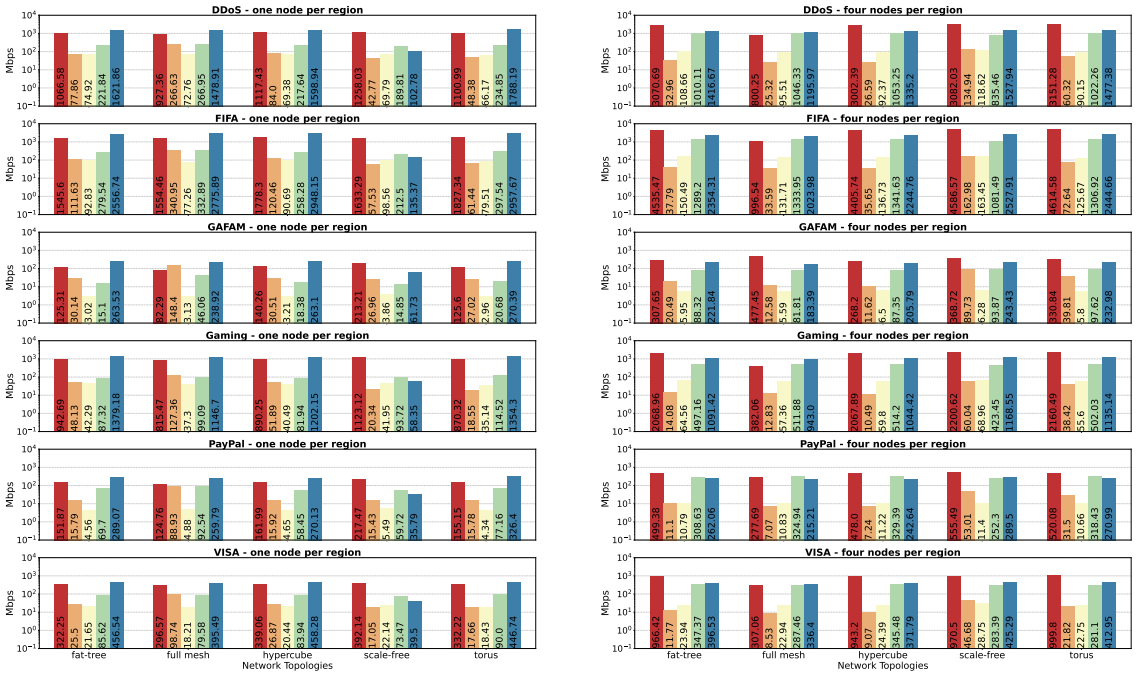


Fig. 7. Network load (Mbps) during workload execution (**Algorand**, **Diem**, **Ethereum**, **Quorum**, **Solana**).

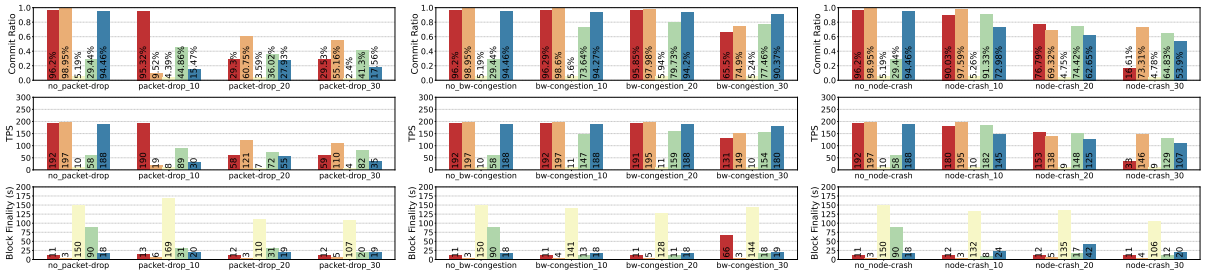
Ethereum maintains a commit rate below 7% (Fig. 6a) and exhibits an unusual latency pattern: for PayPal and VISA it shows higher block latency (≥148 ms) than for highly demanding workloads (DDoS, FIFA, Gaming), where latency can drop below 90 ms. This is largely driven by Ethereum's block production period, which bounds transaction confirmation independently of available bandwidth.

Overall, Algorand and Diem deliver the strongest PayPal results regardless of topology. In the 10-node setting, the VISA workload exceeds the capacity of all platforms, with commit ratio always below 32%. Algorand achieves the highest DDoS results (up to 8% commit rate over 10,000 injected TPS). For smart contracts, Algorand and Diem again lead, reaching about 100 TPS for FIFA, up to 150 TPS for GAFAM, and around 200 TPS for Gaming. Quorum and Solana reach up to 85 TPS for non-intensive smart-contract workloads. Across all systems, intensive smart-contract workloads (FIFA and Gaming) remain challenging and yield poor results.

5.1.2 Scaling the Network (40 Nodes). We then scale to four nodes per region (40 nodes total), which is the largest configuration we can sustain without overloading our cluster. The corresponding results appear on the right side of Fig. 6. As expected, increasing nodes per region increases the number of flows competing for the gateway capacity, amplifying contention.

Quorum remains unable to sustain high workloads at this scale, consistent with the quadratic communication cost of its consensus protocol as the participant set grows. Solana commits transactions reliably only for the GAFAM workload across topologies; for PayPal and VISA it commits only under torus (Fig. 6i) and fails on the others. This behavior is mainly explained by congestion: Solana may drop newly submitted transactions when an RPC node's rebroadcast queue exceeds 10,000 transactions [84], preventing forwarding to the leader; moreover, validators can become unresponsive under high load [57, 85].

Ethereum maintains a relatively low throughput that varies only weakly with network size. Algorand is the least affected by scaling and by the additional contention introduced by larger deployments. One contributing



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

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

factor is message-size asymmetry: credential messages (used for participant authentication) are significantly smaller (100–200 bytes) than block proposals (around 5 MB), enabling faster propagation and helping peers prioritize block proposals more effectively, thereby alleviating congestion.

5.1.3 Network Throughput. We next analyze the network traffic generated during workload execution. Fig. 7 reports aggregate network load (Mbps) by platform, topology, and workload.

Transfer workloads generate substantially less traffic than smart-contract workloads. With one node per region (left side of Fig. 7), PayPal consumes up to 270 Mbps on hypercube, whereas FIFA reaches up to 2,948 Mbps on the same topology. With four nodes per region (right side), network load increases broadly, especially for GAFAM. In this configuration, PayPal reaches up to 470 Mbps across topologies, while FIFA reaches up to 4,405 Mbps on every topology.

At 10 nodes, Solana shows the highest network throughput, followed by Algorand and then Diem. At 40 nodes, Algorand becomes the highest-throughput platform, followed by Solana and Quorum. Ethereum typically exhibits the lowest bandwidth utilization; the next-lowest depends on scale, with Quorum being lower in the 10-node case and Diem being lower in the 40-node case.

5.1.4 Network Degradation. We now study controlled network degradations – packet loss, bandwidth congestion, node failures, and increased latency – and quantify their performance impact. These experiments use a full mesh topology with one node per region. For packet loss, congestion, and node failures, we use the PayPal workload (200 injected TPS for 300 seconds), as it provides a sustained yet simple load profile. The dynamic event is triggered 60 seconds after workload execution starts and reverted 60 seconds later.

For packet loss (Fig. 8a), we apply discrete drop rates (0%, 10%, 20%, 30%) to one third of the links, randomly selected. Throughput degrades for all systems as loss increases; Algorand and Solana remain comparatively robust at 10% loss. Interestingly, Quorum occasionally improves under some loss settings, which we ascribe to the random link selection potentially yielding a more favorable transaction distribution or isolating a congested subset of paths.

For bandwidth congestion (Fig. 8b), we randomly select 10%, 20%, and 30% of the links and reduce their bandwidth to 20% of the original capacity. Performance remains comparatively stable across platforms under this form of congestion, with Solana exhibiting the strongest resilience.

For node crashes (Fig. 8c), we crash 10%, 20%, and 30% of the nodes. Algorand degrades sharply at 30% crashes, whereas Ethereum – despite its lower baseline performance – is the most resilient. Diem, Quorum, and Solana show the expected gradual performance reductions as the fraction of crashed nodes increases.

For the increased-latency experiments (Figs. 9 and 10), we set the network-delay range using the minimum and maximum inter-region latencies observed in our dataset, spanning approximately 30 ms to 500 ms. We use

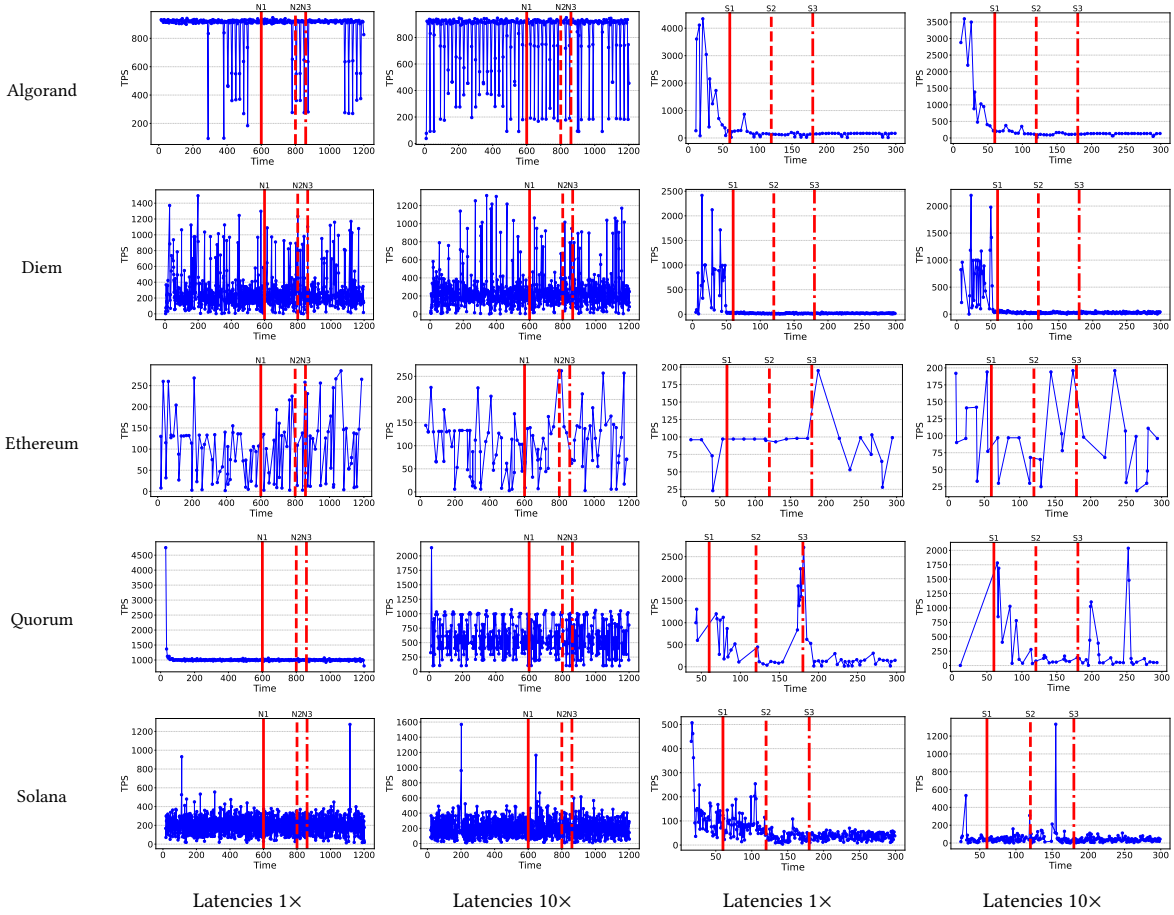


Fig. 9. Benchmark with increasing latencies (PayPal workload). Red lines mark latency variation events (see Section 5.1.4).

Fig. 10. Benchmark with increasing latencies (GAFAM workload). Red lines mark latency variation events (see Section 5.1.4).

extended scenarios for both PayPal and GAFAM. For PayPal, we run a 20-minute experiment: latencies start increasing at minute 10 (N1), reach a peak over 200 seconds (N2), remain at the maximum for 60 seconds (N3), and then return to baseline for the final 6 minutes. For GAFAM, we run a 5-minute experiment: latencies increase after the first minute (S1), peak after another minute (S2), remain at maximum until the third minute (S3), and then gradually return to the initial levels. We repeat both scenarios starting from a baseline factor of 1× and then with a 10× factor, to emulate extreme real-world conditions such as peak transaction periods or market-driven congestion [42]. As latencies increase, blockchains using more traditional consensus mechanisms (*e.g.,* Quorum) tend to recover more slowly. Algorand is also impacted in the PayPal scenario, while Diem, Ethereum, and Solana remain comparatively robust as latencies rise.

5.2 Energy Measurements

We quantify how the five considered topologies influence energy consumption of the five examined blockchains under the three workloads described earlier. Each energy experiment is repeated ten times and we report average

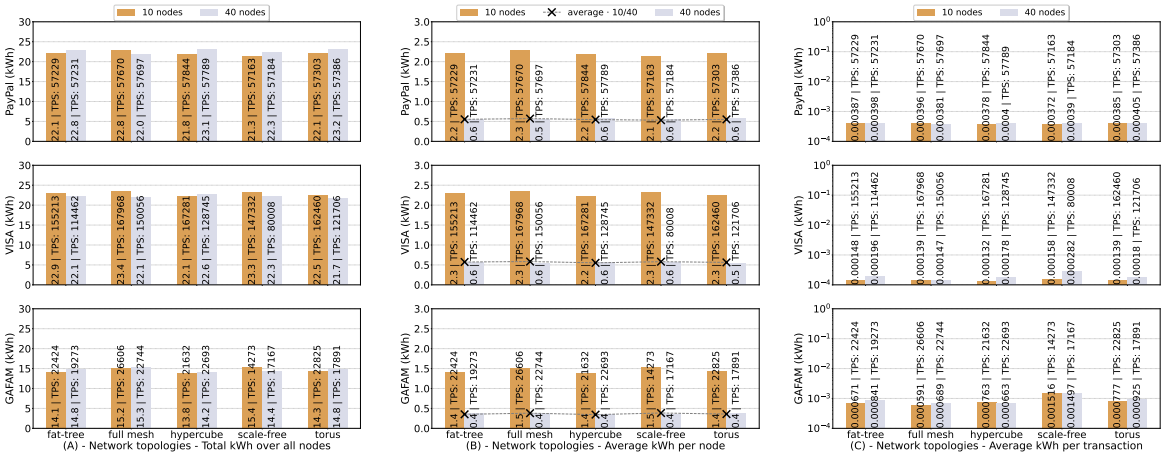


Fig. 11. Algorand energy consumption (kWh): total over all nodes (A), average per node (B), average per transaction (C).

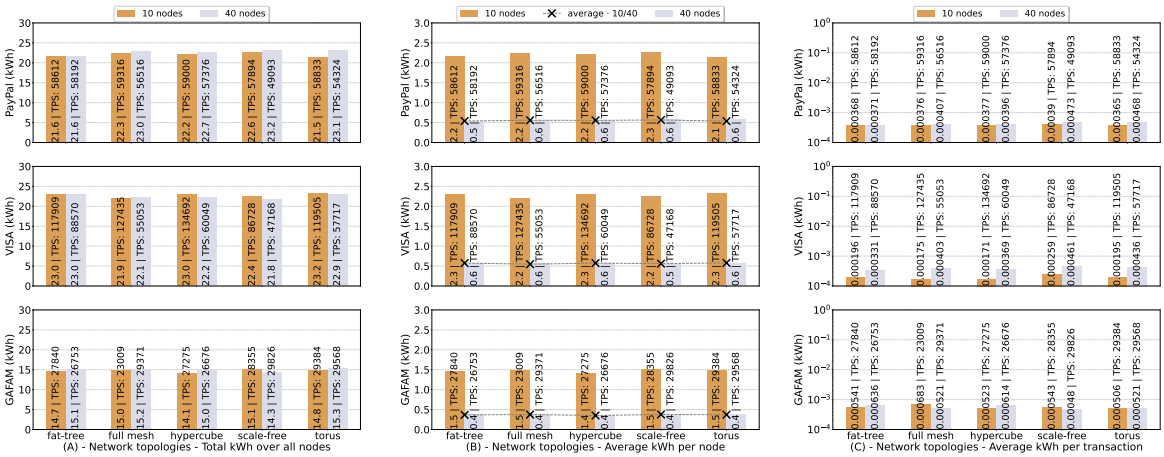


Fig. 12. Diem energy consumption (kWh): total over all nodes (A), average per node (B), average per transaction (C).

values in Figs. 11–16. For each platform (Figs. 11–15), we report: (A) total energy consumed by the network, (B) average energy per node, and (C) average energy per committed transaction. We also annotate bar plots with the achieved TPS, since energy interpretation is meaningful only when paired with delivered throughput. For the per-node view, we explicitly compare the 10-node and 40-node configurations (denoted by *average · 10/40* in the legends) to highlight how per-node energy changes as the network expands.

We also contrast the observed per-node energy at 10 and 40 nodes against a simple linear expectation. For instance, under linear scaling, a consumption of 2 kWh per node with 10 nodes would translate to 0.5 kWh per node with 40 nodes. Finally, Fig. 16 compares average energy per transaction across all blockchains for each workload, while holding topology constant.

5.2.1 Energy Variability as the Network Expands. Increasing the number of nodes does not necessarily imply a proportional increase in total energy consumption. Instead, topology can improve (or degrade) energy efficiency depending on platform and workload.

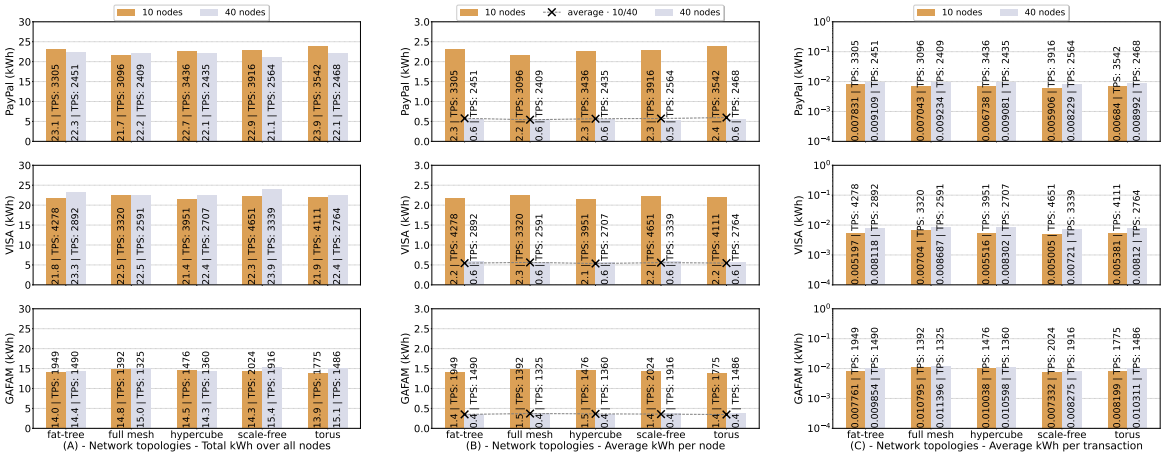


Fig. 13. Ethereum Clique energy consumption (kWh): total over all nodes (A), average per node (B), average per transaction (C).

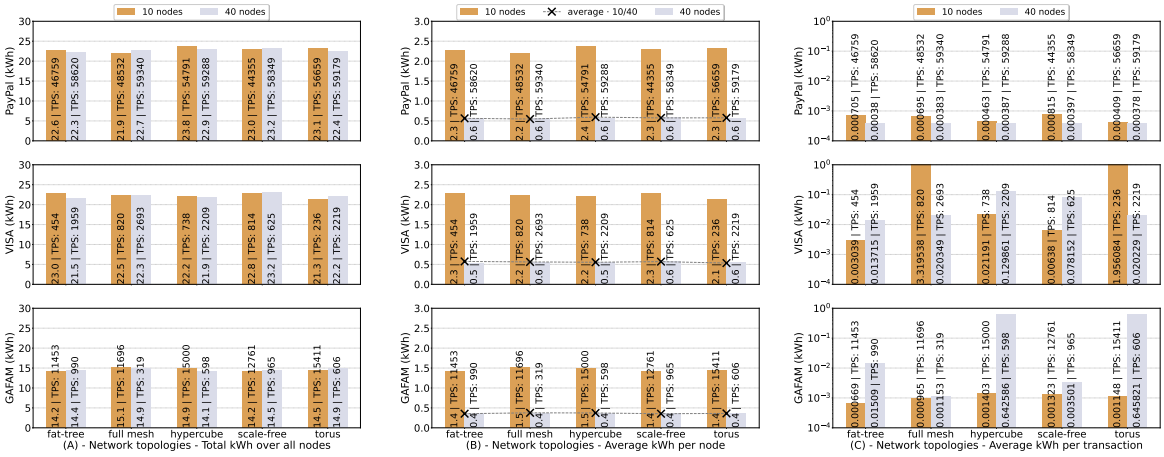


Fig. 14. Quorum IBFT energy consumption (kWh): total over all nodes (A), average per node (B), average per transaction (C).

For Algorand (Fig. 11A–B), full mesh yields the largest reductions in energy as the network grows (down to −17%) across workloads, followed by fat-tree. However, fat-tree reacts poorly to smart-contract workloads, where per-node energy increases by more than 20%. Hypercube and torus also show inefficiencies for transaction-processing workloads.

For Diem (Fig. 12A–B), fat-tree is on average the most efficient across all workloads, with savings up to 35%. Hypercube performs particularly well for transaction processing, reducing energy by 40%, mirroring the trend observed for Algorand. Scale-free is advantageous for smart contracts (−50%) but becomes costly (+45%) under less intensive transfer workloads such as PayPal.

Ethereum (Fig. 13A–B) shows limited opportunities for energy efficiency improvements as the network grows, largely irrespective of topology. Scale-free and torus perform poorly across workloads; hypercube and fat-tree are the least efficient under intensive transaction processing (VISA), with increases up to +40%. For smart contracts, all topologies contribute to per-node energy increases between 10% and 20%.

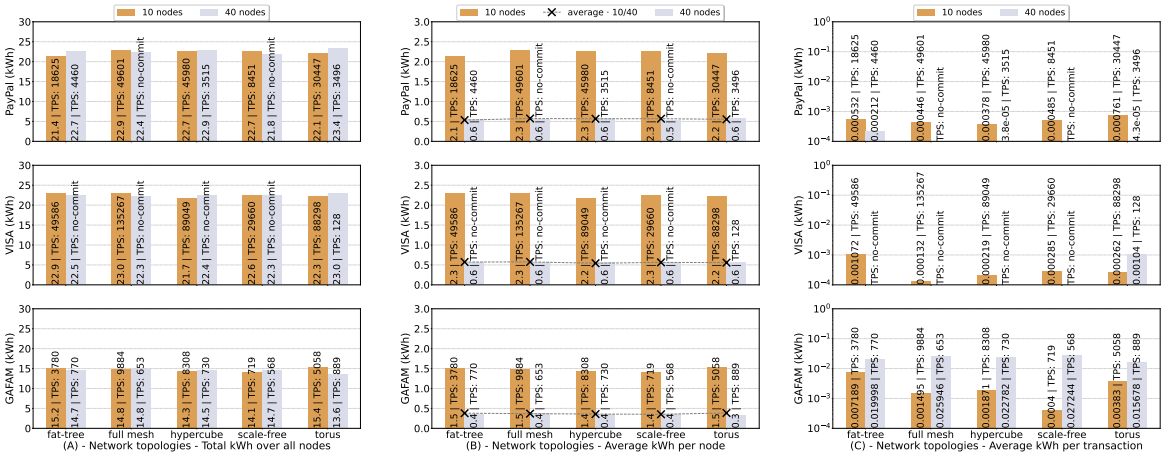


Fig. 15. Solana energy consumption (kWh): total over all nodes (A), average per node (B), average per transaction (C).

For Quorum (Fig. 14A–B), fat-tree is the most energy efficient across workloads, especially for transaction processing (−10% to −20%), followed by full mesh, which shows limited gains/losses. Torus is the least energy-efficient topology, with per-node energy increasing by 15%–25% when scaling from 10 to 40 nodes. Hypercube performs better on smart-contract workloads, but on the transaction-intensive VISA benchmark it degrades, incurring roughly a 15% increase in energy consumption.

Solana (Fig. 15A–B) exhibits severe scaling issues: in many 40-node configurations the number of committed transactions drops to zero, making energy-per-commit not meaningful for most practical interpretations. When measurable, torus remains inefficient for transaction processing (about +20%).

Across platforms, two broad trends emerge: full mesh and fat-tree are the most energy-efficient, while torus and scale-free are less reliable and frequently lead to higher energy consumption, especially at larger scale or under intensive transaction-processing workloads.

5.2.2 Network Topology Impact on Energy per Transaction. Higher energy consumption does not automatically translate into higher throughput. Across Figs. 11–15, we observe a clear negative relationship between energy per commit (kWh/commit) and the number of committed transactions: protocols that sustain higher commit volumes tend to be more energy-efficient on a per-transaction basis. Fig. 16 further enables topology-by-topology comparisons across all platforms.

Algorand and Diem maintain low energy per commit (of the order of 10^{-4}) under hypercube and full mesh (Figs. 11C and 12C), while torus is typically less efficient for the same platforms. Ethereum, in contrast, shows the highest energy per commit among all tested blockchains (typically between 10^{-3} and 10^{-2}), with limited dependence on the specific topology, suggesting protocol-level limitations in energy efficiency as transaction volume grows. Quorum experiences strong increases as workloads intensify; for smart contracts, fat-tree, hypercube, and torus can reach energy-per-commit values of the order of 10^{-1}, with scaling further amplifying these costs. Solana can exhibit energy-per-commit comparable to Algorand and Diem when it commits transactions reliably, but scaling-related failures hinder a precise characterization; in larger configurations, torus-related conflicts likely increase energy per commit, eroding this advantage.

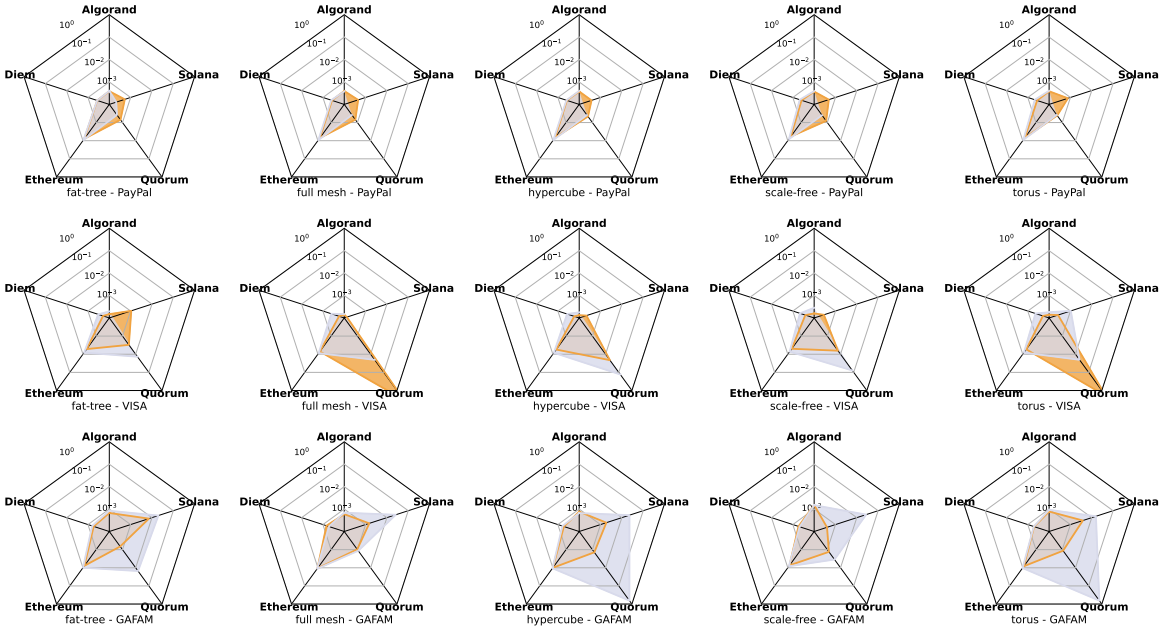


Fig. 16. Comparison of average energy consumption (kWh) per transaction across blockchains, stratified by topology (10 nodes, 40 nodes).

5.2.3 Workload-Specific Insights. For a fixed workload, topology effects are more pronounced.

The PayPal workload (200 TPS over 300 s) is comparatively light. Algorand and Diem keep energy per commit low, especially under full mesh and hypercube; torus shows mild inefficiencies, particularly for Ethereum and Quorum. Overall, fat-tree and full mesh are the most energy-efficient topologies in this regime.

The VISA workload (1,800 TPS over 300 s) substantially increases pressure. Algorand and Diem remain the most efficient, with hypercube and full mesh sustaining better energy profiles at higher transaction volumes. Ethereum becomes strongly inefficient across topologies; for VISA, fat-tree and hypercube are the least efficient for Ethereum, with increases that can exceed 40%. Quorum also increases energy usage under VISA, especially in fat-tree and torus as scale grows.

The GAFAM workload (smart contracts over 180 s) is bursty: it reaches 20,000 TPS early and then stabilizes around 100 TPS. Algorand and Diem again retain low energy per commit, particularly under hypercube and full mesh. Torus is repeatedly inefficient, especially for Ethereum, whose energy rises sharply under burst conditions. Fat-tree shows moderate efficiency here, but its advantage is reduced compared to other topologies when processing burst-heavy smart-contract phases.

5.3 Variance Measurements

We now quantify repeatability and performance predictability under fully controlled, geo-emulated conditions (taxonomy level P2). We execute $n_c = 10$ independent runs for configuration $c = \langle B, T, W, S \rangle$, where B is the blockchain, T the topology, W the workload, and S the validator-set size. To keep the additional overhead minimal yet representative, this P2 analysis focuses on the three canonical workloads shared throughout the article (GAFAM, PayPal, VISA) and the two validator-set sizes (10 and 40). We report three metrics: throughput (TPS), latency to confirmation (s), and total cluster energy (kWh).

5.3.1 Statistical Treatment of Variability. Let $y_{c,r}$ be the value of a given metric for configuration c at run $r = 1, \ldots, n_c$ and let $\bar{y}_c$ be the configuration mean. We characterize typical dispersion via the interquartile range (IQR) and standard deviation (Std), both normalized by the mean:

$$\mathrm{IQR}\%_c = 100 \cdot \frac{Q_{3,c} - Q_{1,c}}{\bar{y}_c} \qquad \mathrm{Std}\%_c = 100 \cdot \frac{\sigma_c}{\bar{y}_c} \tag{1}$$

where $Q_{1,c}$ and $Q_{3,c}$ are the empirical 25th and 75th percentiles and σ_c is the sample standard deviation. To capture tail behavior, we report the worst-case deviation from the mean:

$$\mathrm{WCD}\%_c = 100 \cdot \frac{\max_r |y_{c,r} - \bar{y}_c|}{\bar{y}_c} \qquad \Delta_{\mathrm{abs},c} = \max_r y_{c,r} - \min_r y_{c,r} \tag{2}$$

so that dispersion is visible both in relative terms (percent) and in native units (TPS, s, kWh).

To attribute variance to experimental factors, we perform a factorial ANOVA on per-run observations using a fixed-effects model. Let y_i denote the value of a given metric measured in run i (*i.e.,* one replica of one configuration), and let b_i, t_i, w_i, and s_i denote the levels of the factors blockchain B, topology T, workload W, and size S associated with run i. We fit the following fixed-effects model:

$$y_i = \mu + \alpha_{b_i} + \beta_{t_i} + \gamma_{w_i} + \delta_{s_i} + (\alpha\beta)_{b_i,t_i} + (\alpha\gamma)_{b_i,w_i} + \varepsilon_i \tag{3}$$

Here, μ is the grand mean across all runs; α_{b_i}, β_{t_i}, γ_{w_i}, and δ_{s_i} are the main effects associated with the specific levels of B, T, W, and S in run i; $(\alpha\beta)_{b_i,t_i}$ and $(\alpha\gamma)_{b_i,w_i}$ capture two-way interactions (respectively, how topology and workload sensitivity depend on the blockchain design). All remaining effects, including higher-order interactions not modeled explicitly and run-level noise, are absorbed by the residual term ε_i. Finally, we quantify run-to-run reliability within configurations via the intraclass correlation coefficient (ICC) [49], using the ICC(1,1) convention:

$$\mathrm{ICC} = \frac{\sigma^2_{\mathrm{between}}}{\sigma^2_{\mathrm{between}} + \sigma^2_{\mathrm{within}}} \tag{4}$$

which measures the fraction of total variance σ^2 explained by systematic differences between configurations rather than within-configuration noise.

5.3.2 Typical Dispersion around the Mean. Table 4 summarizes typical run-to-run dispersion using IQR% and Std% (Section 5.3.1). Overall, throughput and latency are markedly more volatile than energy even when focusing on the central part of the distribution. For TPS, Diem and Algorand are the most stable (IQR% $\approx$ 5–6%), Ethereum Clique sits in the middle ($\approx$ 12%), and Solana and Quorum IBFT show much larger spreads (above 30% and 40%). Latency follows a similar ordering, whereas energy is comparatively steadier across chains (IQR% in a narrow $\approx$ 15–18% band). Across configurations, topology often acts as a selector of better/worse regimes within each blockchain rather than as a dominant variance driver; workload choice is more impactful, with steady low-rate PayPal typically benign, bursty GAFAM increasing dispersion, and sustained high-rate VISA amplifying queuing and variability.

5.3.3 Worst-Case Behavior. Typical dispersion hides operational tails. Table 5 reports worst-case run-to-run swings. Solana exhibits the most dramatic excursions (*e.g.,* Δ_{abs} above 680 TPS and relative spikes reaching 1,200% in at least one configuration), while Quorum IBFT reaches similarly extreme relative spikes despite smaller absolute ranges. Latency tails are even more revealing: Quorum IBFT and Solana can exhibit order-of-magnitude slowdowns compared to their configuration averages. Energy remains the least extreme metric, with substantially smaller relative swings than TPS and latency.

Table 4. Overall blockchain-level variability. For each metric, we report IQR% and Std%. The most stable blockchains (minimum values) are highlighted in , the least stable ones in .

	TPS		Lat		kWh	
Blockchain	**IQR**	**Std**	**IQR**	**Std**	**IQR**	**Std**
Algorand	6.11	6.73	5.91	11.73	14.75	12.91
Diem	5.44	6.02	10.61	10.53	15.96	12.11
Ethereum Clique	11.68	11.61	7.40	7.14	15.86	13.93
Quorum IBFT	44.59	56.60	28.91	121.47	18.00	13.68
Solana	34.71	59.58	17.47	44.51	16.21	12.99

Table 5. Worst-case run-to-run swings per blockchain. For each metric we report: the maximum observed range (Δ_{abs}), the largest absolute drop below the mean ($\Delta^{\downarrow}$), the largest absolute increase above the mean ($\Delta^{\uparrow}$), and their percentage counterparts ($\Delta^{\downarrow}\%$, $\Delta^{\uparrow}\%$) relative to the configuration mean. The least stable values (WCD and WCD%, depending on the column) are highlighted in .

	TPS					Lat					kWh				
Blockchain	Δ_{abs}	$\Delta^{\downarrow}$	$\Delta^{\uparrow}$	$\Delta^{\downarrow}\%$	$\Delta^{\uparrow}\%$	Δ_{abs}	$\Delta^{\downarrow}$	$\Delta^{\uparrow}$	$\Delta^{\downarrow}\%$	$\Delta^{\uparrow}\%$	Δ_{abs}	$\Delta^{\downarrow}$	$\Delta^{\uparrow}$	$\Delta^{\downarrow}\%$	$\Delta^{\uparrow}\%$
Algorand	169.40	96.03	94.28	85.57	84.45	42.03	13.69	29.01	36.16	91.18	19.00	6.65	12.86	31.03	54.99
Diem	396.50	232.03	164.47	79.32	56.22	90.90	67.62	39.67	72.29	108.56	13.95	8.48	7.67	39.24	51.73
Ethereum Clique	11.90	7.22	8.22	65.52	76.18	99.48	29.91	79.51	20.66	53.55	16.85	8.23	9.63	37.13	44.17
Quorum IBFT	171.90	135.50	103.65	94.43	1094.52	247.86	79.35	202.99	85.03	928.83	14.18	7.66	9.16	36.79	41.83
Solana	684.80	444.45	320.40	70.29	1200.00	242.40	101.43	171.16	39.46	1200.00	14.22	6.19	8.13	43.06	36.46

Table 6. Variance decomposition and run-to-run reliability. Each entry reports the percentage of variance explained by the corresponding factor or interaction in the factorial ANOVA; ε denotes the residual term. The last column reports the intraclass correlation coefficient (ICC). B: Blockchains; T: Topologies; W: Workloads; S: Scales.

Metric	B	T	W	S	$B \times T$	$B \times W$	ε	**ICC**
TPS	42.7	1.0	10.4	3.1	1.1	22.2	19.5	0.911
Lat	42.9	0.4	6.1	0.3	0.7	22.0	27.6	0.804
kWh	0.0	0.0	64.3	0.0	0.1	0.1	35.5	0.632

5.3.4 Where Variability Comes From. Table 6 decomposes variance via factorial ANOVA and reports ICC. For throughput and latency, the blockchain factor B and its interaction with workload ($B \times W$) dominate, while topology T and size S explain only a small share on their own. This indicates that repeatability is primarily a design-by-demand phenomenon rather than a pure connectivity artifact: performance variability is driven first by blockchain design and then by how each blockchain reacts to workload conditions. Topology still matters, but more as a modulator that selects better or worse operating regimes within a given blockchain than as the primary standalone source of variance. Energy behaves differently: workload W dominates the variance share, confirming that energy dispersion is largely driven by demand characteristics and more by demand intensity and transaction mix than by connectivity alone. ICC values suggest that within-configuration variability is small relative to between-configuration differences (*i.e.,* systematic effects across configurations dominate). However, heavy tails still matter for service-level guarantees, because specific regimes repeatedly trigger large swings. Overall, blockchain type is the main driver of performance variability, workload amplifies/attenuates that variability depending on the platform, and topology remains a secondary but operationally relevant factor.

6 Discussion

This section summarizes the main findings of Section 5 and answers our four research questions, linking performance, robustness under dynamics, energy efficiency, and repeatability/predictability under topology control:

RQ1 **Topology as a performance knob.** Topology is not a neutral deployment detail: it selects materially different performance regimes once the system moves beyond light load. Across blockchains and workloads, highly connected overlays (full mesh and hypercube) more reliably deliver higher throughput and lower latency. Under the most demanding workloads, however, we observe that more structured overlays such as torus can yield better committed throughput for some systems, plausibly by balancing load and avoiding the congestion amplification that dense connectivity can trigger when many concurrent flows compete. Importantly, the optimal topology is not universal: the same overlay can be beneficial for one blockchain (or workload) and detrimental for another, indicating a strong interaction between protocol design (commit path and propagation) and overlay structure.

RQ2 **Robustness under network perturbations.** Dynamic network events affect platforms unevenly, with the ranking differing from the static-topology setting. In our controlled dynamics (packet loss, bandwidth congestion, node crashes, and increased latencies), Diem and Solana tend to be the most sensitive to packet loss, followed by Quorum and Algorand. Despite its low throughput baseline, Ethereum Clique is among the least affected by the tested perturbations, suggesting a robustness profile that trades peak performance for steadier progress under adverse conditions. Protocols with multi-round, leader-driven BFT coordination (*e.g.,* Quorum IBFT) can exhibit slower recovery or pronounced latency inflation as enforced network delays increase, whereas systems with more conservative pacing (*e.g.,* Clique) remain comparatively stable.

RQ3 **Energy efficiency and trade-offs with performance.** Energy outcomes are shaped by the interaction among topology, workload profile, and protocol design rather than by any single factor. Across platforms, fat-tree and full mesh are generally the most energy-efficient choices, especially as workload intensity increases, while scale-free and torus can become inefficient depending on the system and scale. Hypercube performs particularly well for transaction-processing workloads (notably for Algorand and Diem). A key observation is that energy efficiency must be interpreted together with delivered throughput: configurations that fail to commit (or commit very little) can appear arbitrarily inefficient on a per-transaction basis. In our results, Algorand and Diem achieve low energy per committed transaction, while Ethereum Clique exhibits the highest kWh/commit largely independent of topology, reflecting protocol-level limits in throughput that dominate energy-per-commit. Quorum IBFT shows rising energy costs under demanding workloads (and at larger scale). Solana's scaling-related fragility can limit the interpretability of energy-per-commit in the most strained settings. Overall, these findings highlight a practical trade-off space: topologies that maximize performance are often, but not always, aligned with energy efficiency, with the "best" overlay depending on whether the target objective is peak throughput, tail latency robustness, or energy/transaction.

RQ4 **Repeatability and predictability under controlled geo-emulation (P2).** Besides average performance, our campaign shows that repeatability and predictability strongly depend on the blockchain-workload pair, with topology and validator-set size mostly modulating existing fragilities. Even under pinned software versions and fixed geo-emulated network conditions, throughput and latency can display nontrivial run-to-run dispersion, while energy is comparatively steadier. The ANOVA/ICC results show that throughput and latency variability is driven mainly by blockchain design and its interaction with workload. Topology is weaker in isolation, but still affects operating stability, while energy variability is driven primarily by workload. This implies that mean-only reporting can be misleading when topology-driven differences are of the same order as intra-configuration variability. In practice, conclusions should be drawn from trends that persist across multiple topologies and workloads; performance gaps smaller than the observed

dispersion should be treated as inconclusive. Overall stability is highest for Algorand and Diem, whereas Quorum IBFT and Solana are more variance-prone in stressed regimes, with larger swings and occasional failure modes at scale.

Finally, although public systems often exhibit Internet-like, scale-free connectivity and private/consortium deployments are frequently engineered around structured designs (*e.g.,* fat-tree or hypercube), our findings suggest that reorganizing the overlay (or dissemination topology) can improve performance or sustainability without changing the public/private nature of a blockchain. This motivates benchmarking practices that treat topology and link properties as first-class experimental metadata and encourages topology-aware deployment guidelines that balance throughput/latency objectives against energy costs.

6.1 Practical and Design Implications, Usability Scenarios, and Recommendations

Besides the comparative results, Lilith serves as a decision-support framework for protocol developers, network operators, consortium deployers, and benchmark designers. For protocol developers, it enables controlled sensitivity analyses of blockchain behavior under topology changes, resource constraints, and adverse network conditions, clarifying trade-offs among throughput, latency, robustness, and energy efficiency. For operators and deployers, it supports what-if analyses for topology selection, validator placement, and reconfiguration: connectivity-rich topologies may favor peak performance, whereas more structured alternatives, such as fat-tree, can reduce communication overhead and improve energy efficiency in several regimes. For researchers, Lilith combines topology control, workload injection, monitoring, and repeated execution in one pipeline, enabling reproduction of prior studies, fairer assessment of performance claims, and clearer separation between network-substrate and protocol-level effects.

These findings lead to three practical recommendations: treat topology as a first-class experimental and deployment factor; interpret throughput and latency jointly with energy efficiency; and prefer repeated executions over single-run claims when drawing comparative conclusions. More broadly, practitioners should co-tune overlay structure, workload, and operational objectives rather than treating the communication substrate as a fixed background condition. No topology is universally optimal: beneficial structures depend on workload, protocol, and deployment goals. Overall, Lilith is not only a benchmarking tool, but also a methodological aid for transparent, repeatable, and deployment-aware blockchain evaluation.

6.2 Limitations

Lilith is resource-intensive when targeting large-scale configurations, particularly with real clients and containerized deployments. For instance, Solana can require substantial per-node resources (*e.g.,* multiple CPU threads and several GB of RAM), which constrains how far we can scale within a single cluster.

For this study, we limit the evaluation to a 40-node network, which is commonly used in controlled benchmarking and can capture representative trends for throughput/latency in several settings [39, 51], while still enabling systematic topology control. Although this scale is smaller than many real deployments, several effects should generalize qualitatively to larger networks, especially the role of overlay structure in dissemination cost, contention, and performance-energy trade-offs as the validator set grows. At the same time, absolute throughput, latency, and energy-per-transaction values may change quantitatively at larger scales as protocol bottlenecks, coordination costs, and topology-induced path lengths become more pronounced. The results should therefore be read as controlled comparative evidence on topology sensitivity rather than direct quantitative predictions for arbitrary production deployments. Extending the study to larger networks remains future work and will likely require multiple clusters and more heterogeneous hardware.

Additional repetitions would improve statistical confidence, especially for rare tails and finer distributional effects, but ten runs were a practical compromise between statistical depth and the breadth of the geo-emulated experimental matrix across blockchains, topologies, workloads, and validator-set sizes.

Energy is measured by using Intel RAPL counters at the machine level, which enables systematic collection without external instrumentation but does not provide perfect per-container attribution. More precisely, RAPL exposes host-side processor/package energy indicators rather than a complete whole-system measurement covering all hardware, background services, and external infrastructure. Accordingly, the reported energy-per-transaction values are most meaningful for controlled comparisons across topologies, workloads, and platforms, and should not be over-interpreted as deployment-independent end-to-end energy estimates. Finer-grain monitoring at container level is possible [97] and would help isolate overhead from orchestration, monitoring, and emulation components. Moreover, our experiments are performed in an emulated and containerized environment (Docker + Kollaps), which inevitably differs from wide-area bare-metal deployments in timing and scheduling; we mitigate this by focusing on comparative conclusions across topologies, workloads, and platforms as well as by relying on validated tooling [7, 39]. While this work focuses on static topology families, real blockchain overlays may be dynamic, partially reconfigured, or hybrid, especially in public peer-to-peer settings. These dynamics can change the exact quantitative ranking of topologies by affecting peer selection, path diversity, and dissemination over time. However, the main conclusion remains: overlay structure is not a neutral background condition and connectivity patterns materially influence throughput, latency, robustness, and energy efficiency. Lilith already supports dynamic scenarios (*e.g.,* churn and connectivity changes), which we leave for future work together with broader protocol-parameter co-design (*e.g.,* block size and propagation strategies).

7 Conclusions and Future Work

Our experiments indicate that, across five production-grade platforms and several representative transaction-processing and smart-contract workloads, performance-optimal overlays depend on the operating regime: torus can be advantageous under sustained stress, whereas full mesh and hypercube more often enable the highest overall throughput thanks to higher connectivity and effective capacity. Under dynamic network perturbations (loss, congestion, crashes, and latency growth), robustness is strongly platform-dependent: some systems degrade sharply (notably in terms of packet loss), while others remain comparatively resilient albeit at lower baseline throughput.

Energy efficiency is likewise governed by the interaction between topology, workload, and protocol design: fat-tree and full mesh are typically the most efficient choices, Algorand and Diem minimize energy per committed transaction, Ethereum Clique exhibits the highest kWh/commit largely independent of topology, and Quorum IBFT and Solana can become costly and/or fragile in stressed and larger configurations.

Network topology is thus a first-class deployment parameter: even under realistic geo-distributed network conditions, changing the overlay can materially reshape commit throughput, latency, and energy efficiency. Overall, our findings show that topology-aware deployment (and reconfiguration) can serve as an actionable optimization knob to balance peak performance, robustness, and sustainability without changing the public/private nature of a blockchain. Lilith provides a system-agnostic methodology reusable across heterogeneous backends through thin adapters; this journal version enriches our prior work [27, 28] with a variance-centric view of repeatability and predictability.

Future work will extend our experiments to larger and more heterogeneous multi-cluster settings, increase the number of independent repetitions per configuration to strengthen statistical confidence and tail characterization, refine energy attribution beyond machine-level counters, and explore broader dynamic scenarios together with protocol-parameter co-design (*e.g.,* block sizing and propagation strategies).

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] ACM. 2025. Artifact review and badging – current. https://www.acm.org/publications/policies/artifact-review-and-badging-current
[2] Algorand. 2024. Node configuration settings. https://developer.algorand.org/docs/run-a-node/reference/config/
[3] Algorand. 2024. PyTeal Documentation. https://pyteal.readthedocs.io/en/latest/
[4] Algorand. 2024. Why Algorand. https://developer.algorand.org/docs/get-started/basics/why_algorand/
[5] M. Alharby and A. van Moorsel. 2020. BlockSim: An extensible simulation tool for blockchain systems. *Frontiers in Blockchain* 3 (2020), 28:1–28:16. https://doi.org/10.3389/fbloc.2020.00028
[6] L. Alsahan, N. Lasla, and M. Abdallah. 2020. Local Bitcoin Network Simulator for performance evaluation using lightweight virtualization. In *Proc. of the 3rd IEEE Int. Conf. on Informatics, IoT, and Enabling Technologies (ICIoT 2020)*. IEEE Computer Society, 355–360. https://doi.org/10.1109/ICIoT48696.2020.9089630
[7] S. Amaro, M. Matos, and V. Schiavoni. 2025. Kollaps: Decentralized and efficient network emulation for large-scale systems. *IEEE/ACM Trans. on Networking* 33, 1 (2025), 35–50. https://doi.org/10.1109/TNET.2024.3478050
[8] R. Antwi, J. D. Gadze, E. T. Tchao, A. Sikora, H. Nunoo-Mensah, A. S. Agbemenu, K. O.-B. Obour Agyekum, J. O. Agyemang, D. Welte, and E. Keelson. 2022. A Survey on Network Optimization Techniques for Blockchain Systems. *Algorithms* 15, 6 (2022), 1–37. https://doi.org/10.3390/a15060193
[9] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo. 2019. SimBlock: A blockchain network simulator. In *Proc. of the 38th IEEE INFOCOM Workshops (INFOCOM-WKSHPS 2019)*. IEEE Computer Society, 325–329. https://doi.org/10.1109/INFCOMW.2019.8845253
[10] M. Arlitt and T. Jin. 2000. A workload characterization study of the 1998 World Cup Web site. *IEEE Network* 14, 3 (2000), 30–37. https://doi.org/10.1109/65.844498
[11] R. Auer, B. Haslhofer, S. Kitzler, P. Saggese, and F. Victor. 2023. *The technology of Decentralized Finance (DeFi)*. BIS Working Papers 1066. Bank for International Settlements. https://ideas.repec.org/p/bis/biswps/1066.html
[12] S. R. Basnet and S. Shakya. 2017. BSS: Blockchain security over software defined network. In *Proc. of the 3rd Int. Conf. on Computing, Communication and Automation (ICCCA 2017)*. IEEE Computer Society, 720–725. https://doi.org/10.1109/CCAA.2017.8229910
[13] 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*. Technical report. The Libra Association. https://developers.diem.com/papers/diem-consensus-state-machine-replication-in-the-diem-blockchain/2019-06-28.pdf
[14] Bitnodes. 2025. *Global Bitcoin Nodes by Country*. https://bitnodes.io/nodes/all/countries/1d/
[15] R. F. Boisvert. 2016. Incentivizing Reproducibility. *Commun. ACM* 59, 10 (2016), 5–5. https://doi.org/10.1145/2994031
[16] Cambridge Center for Alternative Finance. 2023. Cambridge Blockchain Network Sustainability Index (CBNSI). https://ccaf.io/cbnsi/cbeci
[17] M. Castro and B. Liskov. 1999. Practical Byzantine fault tolerance. In *Proc. of the 3rd Symp. on Operating Systems Design and Implementation (OSDI 1999)*. USENIX Association, 173–186. https://dl.acm.org/citation.cfm?id=296824
[18] B. Chase and E. MacBrough. 2018. *Analysis of the XRP Ledger Consensus Protocol*. Technical Report. arXiv:1802.07242
[19] J. Chen and S. Micali. 2019. Algorand: A Secure and Efficient Distributed Ledger. *Theoretical Computer Science* 777 (2019), 155–183. https://doi.org/10.1016/j.tcs.2019.02.001
[20] 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
[21] H. S. Chena, J. T. Jarrell, K. A. Carpenter, D. S. Cohen, and X. Huang. 2019. Blockchain in healthcare: A patient-centered model. *Biomedical Journal of Scientific & Technical Research* 20, 3 (2019), 15017–15022. https://doi.org/10.26717/BJSTR.2019.20.003448
[22] CoinCu News. 2024. Solana TPS: How Many TPS Can Solana Handle? https://coincu.com/knowledge/solana-tps-how-many-tps-can-solana-handle/
[23] R. Cole and L. Cheng. 2018. Modeling the Energy Consumption of Blockchain Consensus Algorithms. In *Proc. of the IEEE Int. Conf. on Internet of Things (iThings 2018) and IEEE Green Computing and Communications (GreenCom 2018) and IEEE Cyber, Physical and Social Computing (CPSCom 2018) and IEEE Smart Data (SmartData 2018)*. IEEE Computer Society, 1691–1696. https://doi.org/10.1109/Cybermatics_2018.2018.00282
[24] M. Conti, A. Gangwal, and M. Todero. 2019. Blockchain Trilemma Solver Algorand has Dilemma over Undecidable Messages. In *Proc. of the 14th Int. Conf. on Availability, Reliability and Security (ARES 2019)*. ACM Press, Article 16, 8 pages. https://doi.org/10.1145/

3339252.3339255
[25] Cryptoexchange.com. 2024. What is Quorum Blockchain? https://cryptoexchange.com/learning/what-is-quorum
[26] C. Decker and R. Wattenhofer. 2013. Information propagation in the Bitcoin network. In *Proc. of the 13th IEEE Int. Conf. on Peer-to-Peer Computing (P2P 2013)*. IEEE Computer Society, 1–10. https://doi.org/10.1109/P2P.2013.6688704
[27] V. P. Di Perna, M. Bernardo, F. Fabris, S. Amaro, M. Matos, and V. Schiavoni. 2025. Impact of network topologies on blockchain performance. In *Proc. of the 19th ACM Int. Conf. on Distributed and Event-Based Systems (DEBS 2025)*. ACM Press, 122–133. doi:10.1145/3701717.3730540. URL: https://zenodo.org/records/11409100.
[28] V. P. Di Perna, V. Schiavoni, F. Fabris, and M. Bernardo. 2025. Blockchain Energy Consumption: Unveiling the Impact of Network Topologies. In *Proc. of the 7th IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC 2025)*. IEEE Computer Society, 67–76. https://doi.org/10.1109/ICBC64466.2025.11114569
[29] 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 *Proc. of the 43rd ACM Int. Conf. on Management of Data (SIGMOD 2017)*. ACM Press, 1085–1100. https://doi.org/10.1145/3035918.3064033
[30] T. Espel, L. Katz, and G. Robin. 2017. Proposal for protocol on a Quorum blockchain with zero knowledge. *Cryptology ePrint Archive* 2017 (2017), 1–22. https://eprint.iacr.org/2017/1093
[31] Ethereum Foundation. 2024. Ethereum/Hive: Ethereum end-to-end test harness. https://github.com/ethereum/hive
[32] C. Faria and M. Correia. 2019. BlockSim: Blockchain Simulator. In *Proc. of the 2nd IEEE Int. Conf. on Blockchain (Blockchain 2019)*. IEEE Computer Society, 439–446. https://doi.org/10.1109/Blockchain.2019.00067
[33] S. M. Fattahi, A. Makanju, and A. Milani Fard. 2020. SIMBA: An efficient simulator for blockchain applications. In *Proc. of the 50th IEEE/IFIP Int. Conf. on Dependable Systems and Networks (DSN-S 2020)*. IEEE Computer Society, 51–52. https://doi.org/10.1109/DSN-S50200.2020.00028
[34] G. Fridgen, J. Sedlmeir, P. Ross, A. Luckow, J. Lockl, and D. Miehle. 2021. The DLPS: A New Framework for Benchmarking Blockchains. In *Proc. of the 54th Hawaii Int. Conf. on System Sciences (HICSS 2021)*. 6855–6864. https://doi.org/10.24251/HICSS.2021.822
[35] E. Georgiadis. 2019. How many transactions per second can Bitcoin really handle? Theoretically. *Cryptology ePrint Arch.* (2019), 1–416. https://eprint.iacr.org/2019/416
[36] 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 *Proc. of the 23rd ACM SIGSAC Conf. on Computer and Communications Security (CCS 2016)*. ACM Press, 3–16. https://doi.org/10.1145/2976749.2978341
[37] F. C. Geyer, H.-A. Jacobsen, R. Mayer, and P. Mandl. 2023. An end-to-end performance comparison of seven permissioned blockchain systems. In *Proc. of the 24th Int. Middleware Conf. (Middleware 2023)*. ACM Press, 71–84. https://doi.org/10.1145/3590140.3629106
[38] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. 2017. Algorand: Scaling Byzantine agreements for cryptocurrencies. In *Proc. of the 26th Symp. on Operating Systems Principles (SOSP 2017)*. ACM Press, 51–68. https://doi.org/10.1145/3132747.3132757
[39] V. Gramoli, R. Guerraoui, A. Lebedev, C. Natoli, and G. Voron. 2023. Diablo: A benchmark suite for blockchains. In *Proc. of the 18th European Conf. on Computer Systems (EuroSys 2023)*. ACM Press, 540–556. https://doi.org/10.1145/3552326.3567482
[40] V. Gramoli, R. Guerraoui, A. Lebedev, C. Natoli, and G. Voron. 2023. Diablo Blockchain Benchmark Suite. https://diablobench.github.io/
[41] V. Gramoli, R. Guerraoui, A. Lebedev, and G. Voron. 2025. Evaluating Blockchain Fault Tolerance with Stabl. In *Proc. of the 55th IEEE/IFIP Int. Conf. on Dependable Systems and Networks (DSN-S 2025)*. IEEE Computer Society, 182–183. https://doi.org/10.1109/DSN-S65789.2025.00062
[42] M. Guennewig. 2024. *Blockchain Congestion Facilitates Currency Competition.* Technical Report. https://ideas.repec.org/p/bon/boncrc/crctr224_2024_549.html
[43] V. T. Hayashi, F. V. de Almeida, and A. E. Komo. 2021. LabBitcoin: FPGA IoT Testbed for Bitcoin Experiment with Energy Consumption. In *Anais Estendidos do XXI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*. SBC, 90–97. https://doi.org/10.5753/sbseg_estendido.2021.17344
[44] Hyperledger. 2018. Hyperledger Caliper — hyperledger.github.io. https://hyperledger.github.io/caliper/
[45] IEEE. 2025. Overview of IEEE Xplore Digital Library Content - Reproducibility Badges. https://ieeexplore.ieee.org/Xplorehelp/overview-of-ieee-xplore/about-content#reproducibility-badges
[46] M. R. Islam, M. M. Rashid, M. A. Rahman, M. H. S. B. Mohamad, and A. H. B. Embong. 2022. A Comprehensive Analysis of Blockchain-based Cryptocurrency Mining Impact on Energy Consumption. *Int. Journal of Advanced Computer Science and Applications* 13, 4 (2022), 590–598. https://doi.org/10.14569/IJACSA.2022.0130469
[47] M. Janczyk and R. Pfister. 2023. *Factorial Analysis of Variance (ANOVA).* Springer. https://doi.org/10.1007/978-3-662-66786-6_9
[48] R. Karanjai, L. Xu, L. Chen, N. Diallo, and W. Shi. 2024. Decentralized FaaS over multi-clouds with blockchain based management for supporting emerging applications. In *Proc. of the 39th ACM SIGAPP Symp. on Applied Computing (SAC 2024)*. ACM Press, 122–130. https://doi.org/10.1145/3605098.3636029
[49] T. K. Koo and M. Y. Li. 2016. A Guideline of Selecting and Reporting Intraclass Correlation Coefficients for Reliability Research. *Journal of Chiropractic Medicine* 15, 2 (2016), 155–163. https://doi.org/10.1016/j.jcm.2016.02.012

[50] M. R. A. Lathif, P. Nasirifard, and H.-A. Jacobsen. 2018. CIDDS: A configurable and distributed DAG-based distributed ledger simulation framework. In *Proc. of the 19th Int. Middleware Conf. (Middleware 2018 – Posters)*. ACM Press, 7–8. https://doi.org/10.1145/3284014.3284018

[51] A. Lebedev and V. Gramoli. 2024. On the Relevance of Blockchain Evaluations on Bare Metal. In *Proc. of the 7th Int. Symp. on Distributed Ledger Technology (SDLT 2023) (Communications in Computer and Information Science, Vol. 1975)*. Springer, 22–38. https://doi.org/10.1007/978-981-97-0006-6_2

[52] A. Ledenev. 2016. PUMBA: Chaos testing, network emulation, and stress testing tool for containers (GitHub repository). https://github.com/alexei-led/pumba

[53] Z. Li and P. Mohapaira. 2004. The impact of topology on overlay routing service. In *Proc. of the 23rd IEEE Conf. on Computer Communications (INFOCOM 2004)*, Vol. 1. IEEE Computer Society, 1–418. https://doi.org/10.1109/INFCOM.2004.1354513

[54] 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 *Proc. of the 5th IEEE Int. Conf. on Blockchain and Cryptocurrency (ICBC 2023)*. IEEE Computer Society, 1–5. https://doi.org/10.1109/ICBC56567.2023.10174950

[55] 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. on Computers* 73, 3 (2024), 942–955. https://doi.org/10.1109/TC.2024.3349654

[56] I. Malakhov, A. Marin, and S. Rossi. 2023. Analysis of the confirmation time in proof-of-work blockchains. *Future Generation Computer Systems* 147 (2023), 275–291. https://doi.org/10.1016/j.future.2023.04.016

[57] 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/

[58] I. Mashaly. 2019. Geth-POA Tutorial – README: Block Time. https://github.com/ibrahimmashaly/geth-poa-tutorial/blob/1257b0d397edd7bc5a508a36c37c0d8df898ff9a/README.md#block-time

[59] D. Mazieres. 2015. The Stellar consensus protocol: A federated model for Internet-level consensus. *Stellar Development Foundation* 32 (2015), 1–45. https://www.stellar.org/papers/stellar-consensus-protocol.pdf

[60] 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), 1–11. https://doi.org/10.1016/j.bcra.2021.100026

[61] D. Mechkaroska, V. Dimitrova, and A. Popovska-Mitrovikj. 2018. Analysis of the possibilities for improvement of blockchain technology. In *Proc. of the 26th Telecommunications Forum (TELFOR 2018)*. 1–4. https://doi.org/10.1109/TELFOR.2018.8612034

[62] A. Miller and R. Jansen. 2015. Shadow-Bitcoin: Scalable simulation via direct execution of multi-threaded applications. In *Proc. of the 8th USENIX Conf. on Cyber Security Experimentation and Test (CSET 2015)*. USENIX Association, 1–8. https://www.usenix.org/conference/cset15/workshop-program/presentation/miller

[63] S. Nakamoto. 2009. Bitcoin: A peer-to-peer electronic cash system. (2009). http://www.bitcoin.org/bitcoin.pdf

[64] B. Nasrulin, M. De Vos, G. Ishmaev, and J. Pouwelse. 2022. Gromit: Benchmarking the performance and scalability of blockchain systems. In *Proc. of the 4th IEEE Int. Conf. on Decentralized Applications and Infrastructures (DAPPS 2022)*. IEEE Computer Society, 56–63. https://doi.org/10.1109/DAPPS55202.2022.00015

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

[66] 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

[67] M. Pacheco, G. Oliva, G. K. Rajbahadur, and A. Hassan. 2023. Is My Transaction Done Yet? An Empirical Study of Transaction Processing Times in the Ethereum Blockchain Platform. *ACM Trans. on Software Engineering and Methodology* 32, 3, Article 59 (2023). https://doi.org/10.1145/3549542

[68] 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 *Proc. of the 21st IEEE Global Communications Conf. (GLOBECOM 2020)*. IEEE Computer Society, 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9322354

[69] M. Parashar. 2019. Editor's Note: IEEE Trans. on Parallel and Distributed Systems (TPDS) Reproducibility Initiative. *IEEE Trans. on Parallel and Distributed Systems* 30, 8 (2019), 1690. https://doi.org/10.1109/TPDS.2019.2922052

[70] G. A. Pierro and R. Tonelli. 2022. A Study on Diem Distributed Ledger Technology. In *Proc. of the 4th Distributed Ledger Technology Workshop (DLT 2022) (CEUR Workshop Proceedings, Vol. 3166)*. CEUR-WS.org, 33–47. https://ceur-ws.org/Vol-3166/paper03.pdf

[71] A. Pimpini and A. Pellegrini. 2025. RBlockSim: Parallel and Distributed Simulation for Blockchain Benchmarking. In *Proc. of the 39th ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation (PADS 2025)*. ACM Press, 176–185. https://doi.org/10.1145/3726301.3728421

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

[73] 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
[74] K. Ren, J. F. B. Van Buskirk, Z. Y. Ang, S. Hou, N. R. Cable, M. Monares, H. F. Korth, and D. Loghin. 2023. BBSF: Blockchain Benchmarking Standardized Framework. In *Proc. of the 1st Workshop on Verifiable Database Systems (VDBS 2023)*. ACM Press, 10–18. https://doi.org/10.1145/3595647.3595649
[75] E. Rohrer, J. Malliaris, and F. Tschorsch. 2019. Discharged payment channels: Quantifying the Lightning network resilience to topology-based attacks. In *Proc. of the 4th IEEE European Symp. on Security and Privacy Workshops (EuroS&P-W 2019)*. IEEE Computer Society, 347–356. https://doi.org/10.1109/EuroSPW.2019.00045
[76] S. Saberi, M. Kouhizadeh, J. Sarkis, and L. Shen. 2019. Blockchain technology and its relationships to sustainable supply chain management. *Int. Journal of Production Research* 57, 7 (2019), 2117–2135. https://doi.org/10.1080/00207543.2018.1533261
[77] D. Saingre, T. Ledoux, and J.-M. Menaud. 2020. BCTMark: A framework for benchmarking blockchain technologies. In *Proc. of the 17th IEEE/ACS Int. Conf. on Computer Systems and Applications (AICCSA 2020)*. IEEE Computer Society, 1–8. https://doi.org/10.1109/AICCSA50499.2020.9316536
[78] D. Saingre, T. Ledoux, and J.-M. Menaud. 2022. Measuring performances and footprint of blockchains with BCTMark: A case study on Ethereum smart contracts energy consumption. *Cluster Computing* 25, 4 (2022), 2819–2837. https://doi.org/10.1007/s10586-021-03441-x
[79] R. Saltini. 2019. IBFT liveness analysis. In *Proc. of the 1st IEEE Int. Conf. on Blockchain (Blockchain 2019)*. IEEE Computer Society, 245–252. https://doi.org/10.1109/Blockchain.2019.00039
[80] SBA Research. 2017. Simcoin: Blockchain Simulation Framework with Docker and Python. https://github.com/sbaresearch/simcoin
[81] M. Schäffer, M. di Angelo, and G. Salzer. 2019. Performance and Scalability of Private Ethereum Blockchains. In *Proc. of the Blockchain and Central and Eastern Europe Forum (BPM 2019) (Lecture Notes in Business Information Processing, Vol. 361)*. Springer, 103–118. https://doi.org/10.1007/978-3-030-30429-4_8
[82] Z. Shahbazi and Y.-C. Byun. 2021. Integration of blockchain, IoT and machine learning for multistage quality control and enhancing security in smart manufacturing. *Sensors* 21, 4 (2021), 1–18. https://doi.org/10.3390/s21041467
[83] P. K. Sharma, S. Singh, Y.-S. Jeong, and J. H. Park. 2017. DistBlockNet: A Distributed Blockchains-Based Secure SDN Architecture for IoT Networks. *IEEE Communications Magazine* 55, 9 (2017), 78–85. https://doi.org/10.1109/MCOM.2017.1700041
[84] Solana. 2024. Retrying Transactions. https://solana.com/docs/advanced/retry
[85] Solana. 2024. Transaction Confirmation and Expiration. https://solana.com/docs/advanced/confirmation
[86] Solana-Labs. 2022. Solana vote_state.rs source file. https://github.com/solana-labs/solana/blob/master/programs/vote/src/vote_state/mod.rs#L34
[87] Steam. 2013. Dota 2 on Steam. https://store.steampowered.com/app/570/Dota_2/
[88] W. Sun, Z. Xu, W. Ni, and L. Chen. 2025. InTime: Towards Performance Predictability In Byzantine Fault Tolerant Proof-of-Stake Consensus. *Proc. of the ACM on Management of Data* 3, 1, Article 47 (2025), 27 pages. https://doi.org/10.1145/3709740
[89] P. Szilágyi. 2017. EIP-225: Clique proof-of-authority consensus protocol. https://eips.ethereum.org/EIPS/eip-225
[90] T. Toivola. 2014. VnStat – a network traffic monitor for Linux and BSD. https://humdi.net/vnstat/
[91] B. Toshniwal and K. Kataoka. 2021. Comparative Performance Analysis of Underlying Network Topologies for Blockchain. In *Proc. of the 35th Int. Conf. on Information Networking (ICOIN 2021)*. 367–372. https://doi.org/10.1109/ICOIN50884.2021.9333978
[92] G. Tripathi, M. A. Ahad, and G. Casalino. 2023. A comprehensive review of blockchain technology: Underlying principles and historical background with future challenges. *Decision Analytics Journal* 9 (2023), 100344. https://doi.org/10.1016/j.dajour.2023.100344
[93] R. Vaishnavi, C. Athira, C. Pradeep, Sankalpkumar, Eashwara Prasanna, and V. Srikanth. 2024. Energy Efficiency in Blockchain Social Networks. *Int. Journal of Research Publication and Reviews* 5, 3 (2024), 819–825. https://doi.org/10.55248/gengpi.5.0324.0632
[94] G. Wang, Y. Zhang, C. Ying, X. Lit, and G. Yu. 2024. Hammer: A General Blockchain Evaluation Framework. In *Proc. of the 44th IEEE Int. Conf. on Distributed Computing Systems (ICDCS 2024)*. IEEE Computer Society, 391–402. https://doi.org/10.1109/ICDCS60910.2024.00044
[95] T. Wang, Z. Su, Y. Xia, B. Qin, and M. Hamdi. 2014. NovaCube: A low latency Torus-based network architecture for data centers. In *Proc. of the 15th IEEE Global Communications Conf. (GLOBECOM 2014)*. IEEE Computer Society, 2252–2257. https://doi.org/10.1109/GLOCOM.2014.7037143
[96] X. Wang, A. Al-Mamun, F. Yan, and D. Zhao. 2019. Toward Accurate and Efficient Emulation of Public Blockchains in the Cloud. In *Proc. of the 12th Int. Conf. on Cloud Computing (CLOUD 2019) (Lecture Notes in Computer Science, Vol. 11513)*. Springer, 67–82. https://doi.org/10.1007/978-3-030-23502-4_6
[97] M. Warade, K. Lee, C. Ranaweera, and J.-G. Schneider. 2023. Monitoring the energy consumption of docker containers. In *Proc. of the 47th IEEE Computers, Software, and Applications Conf. (COMPSAC 2023)*. IEEE Computer Society, 1703–1710. https://doi.org/10.1109/COMPSAC57700.2023.00263
[98] G. Wood. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 151 (2014), 1–32. https://ethereum.github.io/yellowpaper/paper.pdf
[99] X. Wu, J. Yan, and D. Jin. 2019. Virtual-time-accelerated emulation for blockchain network and application evaluation. In *Proc. of the 27th ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation (PADS 2019)*. ACM Press, 149–160. https://doi.org/10.1145/3316480.3322889

[100] J. Xu, C. Wang, and X. Jia. 2023. A Survey of Blockchain Consensus Protocols. *Comput. Surveys* 55, 13s (2023), 1–35. https://doi.org/10.1145/3579845

[101] H. Yajam, E. Ebadi, and M. A. Akhaee. 2024. JABS: A blockchain simulator for researching consensus algorithms. *IEEE Trans. on Network Science and Engineering* 11, 1 (2024), 3–13. https://doi.org/10.1109/TNSE.2023.3282916

[102] 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

[103] A. Yakovenko. 2021. *Solana: A new architecture for a high performance blockchain v0.8.13.* https://solana.com/solana-whitepaper.pdf

[104] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K. R. Choo. 2020. An Energy-Efficient SDN Controller Architecture for IoT Networks With Blockchain-Based Security. *IEEE Trans. on Services Computing* 13, 4 (2020), 625–638. https://doi.org/10.1109/TSC.2020.2966970

[105] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham. 2019. HotStuff: BFT consensus with linearity and responsiveness. In *Proc. of the 38th ACM Symp. on Principles of Distributed Computing (PODC 2019)*. ACM Press, 347–356. https://doi.org/10.1145/3293611.3331591

[106] J. Zakrzewski. 2018. Towards Verification of Ethereum Smart Contracts: A Formalization of Core of Solidity. In *Proc. of the 10th Int. Conf. on Verified Software, Theories, Tools, and Experiments (VSTTE 2018) (Lecture Notes in Computer Science, Vol. 11294)*. Springer, 229–247. https://doi.org/10.1007/978-3-030-03592-1_13

[107] 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 *Proc. of the 2nd Int. Conf. on Hot Information-Centric Networking (HotICN 2019)*. 77–83. https://doi.org/10.1109/HotICN48464.2019.9063213