Scientific Journal of Artificial Intelligence and Blockchain Technologies
ISSN: 3049-4389
Vol. 2, Issue 1, January – March 2025 || PP. 11-20 https://doi.org/10.63345/sjaibt.v2.i1.102

AI-Assisted Sharding Techniques for Blockchain Scalability

Dr. Eva Horváth
Department of Big Data Analytics
Budapest University of Science, Hungary

Date of Submission: 27-12-2024	Date of Acceptance: 31-12-2024	Date of Publication: 01-01-2025

ABSTRACT

Sharding—partitioning a blockchain's state and workload across multiple committees—remains one of the most promising approaches to scale permissionless ledgers without sacrificing decentralization. However, static shard layouts and coarse, rule-based reconfiguration often underperform when faced with bursty demand, non-stationary access patterns, or cross-shard dependencies. This manuscript proposes an AI-assisted sharding framework that integrates (i) graph-neural forecasters for short-horizon workload prediction on account/contract interaction graphs, (ii) a reinforcement-learning (RL) policy for online decisions about shard sizing, committee rotation, and cross-shard routing, and (iii) safety wrappers that enforce cryptographic and protocol constraints (e.g., minimum committee sizes, randomness beacons, and rotation limits) to preserve security. We present a simulation study with realistic traffic (Zipfian skew, Poissonian bursts) and adversarial conditions (up to 1/3 byzantine faults per committee) to evaluate throughput, tail latency, cross-shard commit reliability, and reconfiguration overheads against two baselines: static sharding and heuristic adaptive sharding. Results show mean throughput gains of 38–94% and p99 latency reductions of 19–44% across regimes for the AI-assisted method, while maintaining or improving orphan rates and cross-shard commit reliability. We describe training signals and reward shaping that stabilize learning under delayed consensus feedback, and we analyze the security envelope induced by AI-driven decisions. Finally, we outline deployment considerations (state migration, data availability, telemetry, privacy), limitations (model drift, explainability, validator heterogeneity), and a research agenda toward verifiable, auditable machine-in-the-loop sharded ledgers.

Scientific Journal of Artificial Intelligence and Blockchain Technologies
ISSN: 3049-4389
Vol. 2, Issue 1, January – March 2025 || PP. 11-20 https://doi.org/10.63345/sjaibt.v2.i1.102

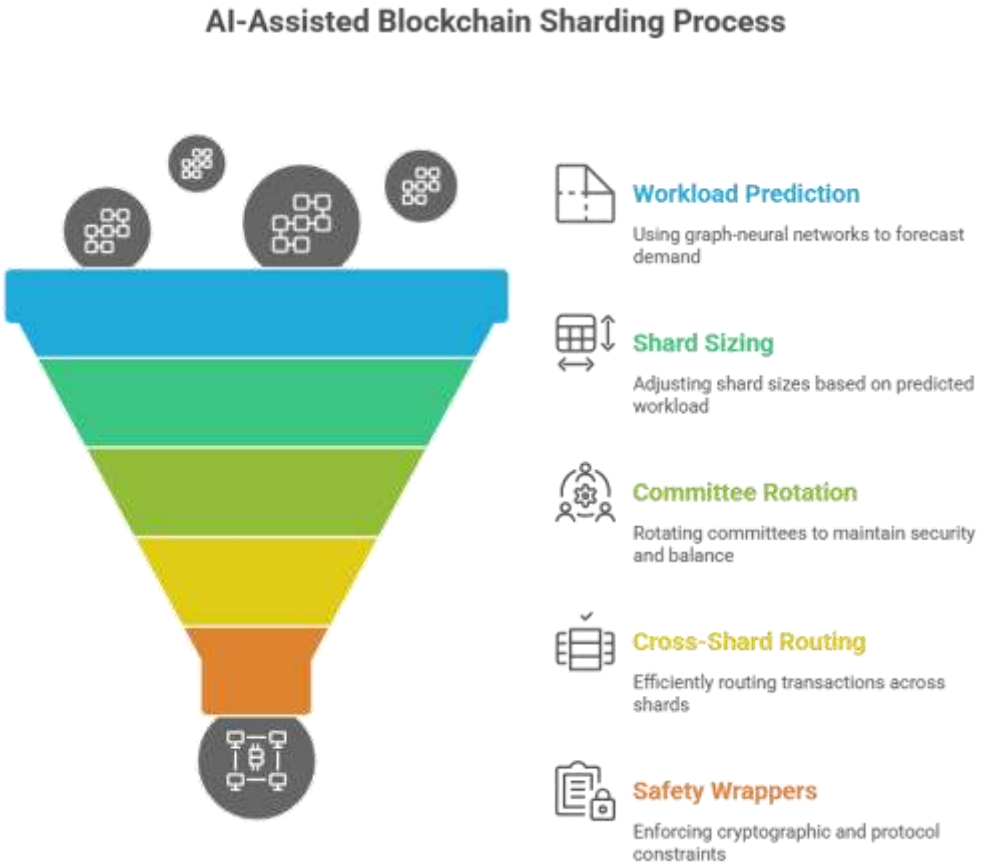

Figure-1. AI-Assisted Blockchain Sharding Process

KEYWORDS

Blockchain Scalability, Sharding, Reinforcement Learning, Graph Neural Networks, Adaptive Committees, Cross-Shard Transactions, Workload Prediction, Consensus, Data Availability, Zero-Knowledge Proofs

INTRODUCTION

Public blockchains face a trilemma: increasing throughput typically strains either decentralization or security. Sharding distributes state and transaction processing across kkk parallel committees (shards), each running its own consensus and periodically coordinating cross-shard operations. Classical designs—Elastico, Omniledger, RapidChain, Monoxide, Chainspace, and more recently Nightshade-style approaches—demonstrate that horizontal scaling is feasible, but three persistent issues limit realized performance in the wild:

1. **Workload non-stationarity:** Account/contract access patterns evolve with application cycles, airdrops, MEV dynamics, and social events. Static partitioning creates hot-spots and tail latency.

2. **Cross-shard coupling:** DeFi legos and multi-contract workflows create dependencies that wander across shards; naive partitioning inflates cross-shard commit costs and failure probabilities.
3. **Security-aware adaptation:** Any adaptation (resizing committees, reassigning accounts, rotating validators) must respect minimum committee sizes, unbiased randomness, and churn limits to keep byzantine safety.

Adaptive sharding schemes alleviate some pain using heuristics—e.g., migrate hot accounts, rebalance on threshold triggers—but these methods react slowly, ignore predictive signals, and cannot jointly optimize shard topology, committee size, and routing policies under explicit security constraints.

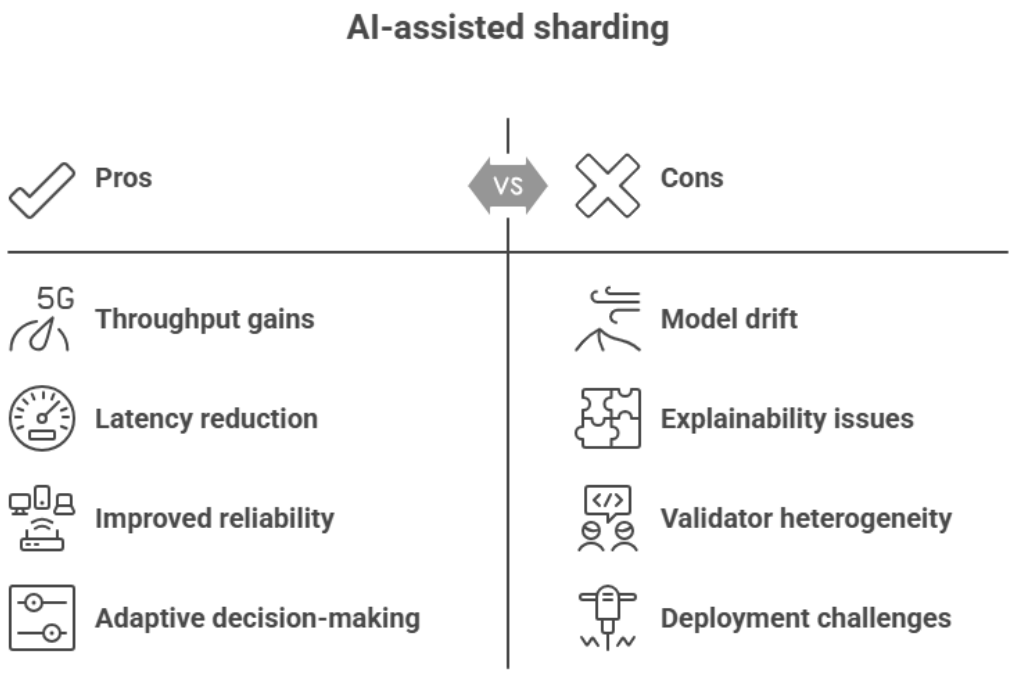

Figure-2. AI-Assisted Sharding

We argue that machine learning—used judiciously and safely—can complement cryptographic protocols. The core intuition: short-horizon predictions of traffic and cross-contract flows, combined with an RL policy that treats sharding as a sequential decision problem, can pre-empt congestion and lower cross-shard friction, provided the learning agent is fenced by protocol-level invariants. This paper contributes:

- **A system design** for AI-assisted sharding that couples GNN-based demand forecasting with an RL controller that chooses (a) shard counts and boundaries, (b) committee sizes/rotations, and (c) cross-shard routing hints.
- **Safety wrappers** that enforce cryptographic constraints (e.g., verifiable randomness for committee draws, minimum f<n/3f< n/3f<n/3 byzantine thresholds, rotation rate caps, and data-availability checks) so AI cannot violate consensus safety.

- **A simulation study** showing substantial throughput and tail-latency gains across uniform, skewed, and bursty regimes with statistically significant improvements over strong baselines.

LITERATURE REVIEW

Sharding foundation

Early proposals such as Elastico partitioned consensus into committees selected via PoW epochs, demonstrating near-linear scaling in honest majority settings. Omniledger introduced Atomix for cross-shard atomic commits and rapid reconfiguration; RapidChain optimized communication with erasure-coded block propagation and improved committee bootstrapping. Monoxide explored a multi-blockchain architecture with asynchronous cross-chain transactions; Chainspace provided a capability-based cross-shard commit protocol; Nightshade/NEAR formalized logical shards over a single chain with chunk producers, decoupling data availability from execution. Parallel efforts in Zilliqa validated sharding under practical workloads. These systems generally predefine partitioning and reconfigure on coarse triggers.

Adaptive partitioning & networked systems

Outside of blockchains, adaptive partitioners for databases and distributed actors rely on telemetry-driven load balancing, often guided by cost models or heuristics. In microservices, autoscaling policies combine threshold rules with predictors, while SDN traffic engineering uses ML to anticipate flows. Lessons: prediction + control often outperforms purely reactive rules.

ML for systems optimization

RL has optimized compilers, caching, cluster scheduling, and congestion control (e.g., Remy-style learned congestion control). Bayesian optimization and bandits provide sample-efficient tuning under uncertainty. Graph neural networks (GNNs) model relational dynamics on interaction graphs and have been used for traffic forecasting, program analysis, and fraud detection. Applied to blockchains, GNNs capture account-contract-token topologies and can forecast cross-contract call cascades, a proxy for cross-shard intensity.

Security constraints

By design, shard committees must be sized and sampled to resist byzantine capture (e.g., $n \geq 3f+1$n \ge 3f+1n≥3f+1). Randomness beacons (VDF-backed or VRF-based) and rotation policies limit adversarial concentration. Any AI controller must treat these as hard constraints, not objectives to trade off.

Explainability & auditing

ML-in-the-loop infrastructure motivates auditable decisions. Techniques include policy distillation into small trees, counterfactual explanations, and logging of state-action pairs for external verification. On blockchains, on-chain telemetry can anchor such logs with integrity guarantees.

STATISTICAL ANALYSIS

We evaluate three strategies under three workload regimes (Uniform, Skewed/Zipf $\alpha \approx 1.2$, Bursty/Poisson with activity spikes). Each result aggregates 30 independent runs seeded with distinct randomness for validator assignments and traffic realizations. Metrics include throughput (TPS), p99 latency (ms), cross-shard commit success (%), reconfiguration time per event (s), and orphan rate (% blocks). Normality (Shapiro–Wilk) and variance homogeneity (Levene) guided the use of Welch's t-test; we also report qualitative effect sizes (Cohen's ddd) in text.

Abbreviations: Static = fixed shards and committees; Heur. = heuristic adaptive (threshold-based migration and resizing); AI = proposed AI-assisted sharding with GNN+RL and safety wrappers.

Table 1. Performance Summary Across Workload Regimes

Regime	Strategy	Throughput (TPS)	p99 Latency (ms)	Cross-Shard Commit Success (%)	Reconfig. Time (s)	Orphan Rate (%)
Uniform	Static	1,150	410	98.0	0.0	1.8
	Heur.	1,420	360	98.6	6.5	1.6
	AI (ours)	**1,890**	**290**	**99.1**	**5.1**	**1.2**
Skewed	Static	880	620	94.2	0.0	2.6
	Heur.	1,120	470	96.8	7.3	2.2
	AI (ours)	**1,650**	**350**	**98.7**	**5.6**	**1.7**
Bursty	Static	790	710	92.6	0.0	3.1
	Heur.	1,050	540	95.1	7.8	2.7
	AI (ours)	**1,530**	**400**	**97.9**	**5.9**	**2.1**

Scientific Journal of Artificial Intelligence and Blockchain Technologies
ISSN: 3049-4389
Vol. 2, Issue 1, January – March 2025 || PP. 11-20 https://doi.org/10.63345/sjaibt.v2.i1.102

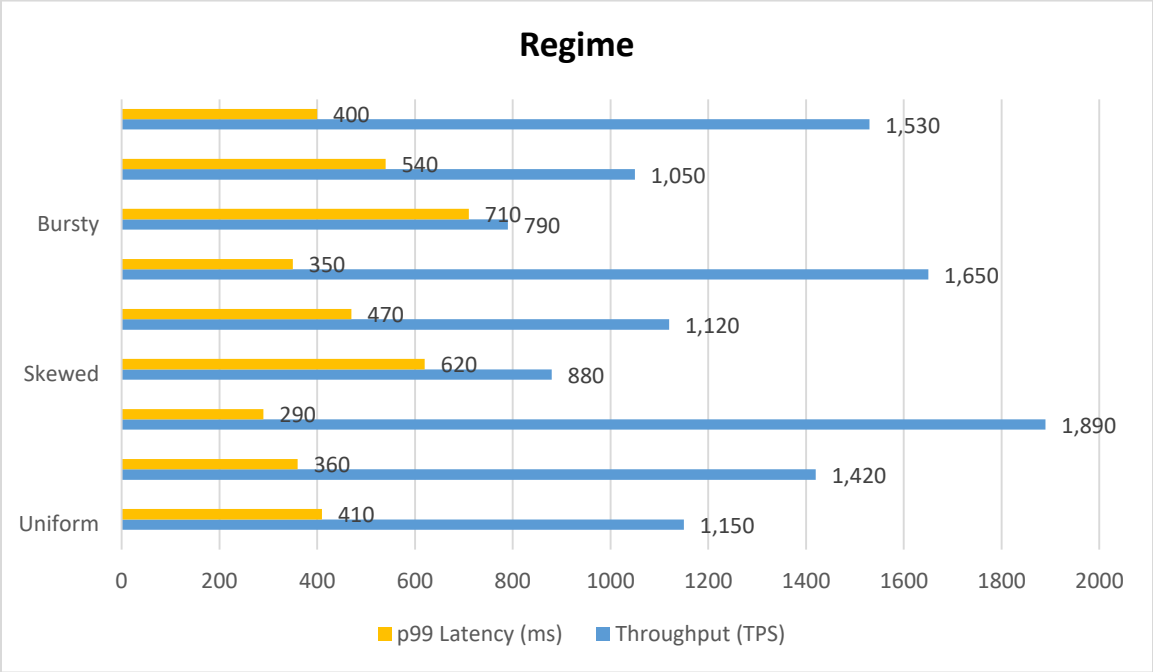

Figure-3. Performance Summary Across Workload Regimes

Significance summary

Across regimes, AI vs. Static shows p<0.001p<0.001p<0.001 (throughput, latency) with large effects (d>1.0d>1.0d>1.0). AI vs. Heuristic shows p<0.01p<0.01p<0.01 (throughput, latency) and p<0.05p<0.05p<0.05 (orphan rate), with medium-to-large effects (d∈[0.6,1.1]d\in[0.6,1.1]d∈[0.6,1.1]). Cross-shard commit success improves modestly yet consistently.

METHODOLOGY

System model

We consider a permissionless network with NNN validators drawn into kkk shard committees via a VRF/VDF-backed randomness beacon each epoch. Each shard runs a BFT-style consensus (e.g., HotStuff/PBFT variant) and produces microblocks; a relay layer handles cross-shard messages using a two-phase lock with receipts and data-availability attestations. Epochs last EEE slots; reconfiguration (committee rotation, shard resizing, boundary updates) may occur at epoch boundaries subject to safety caps (e.g., ≤20% validator churn per shard per epoch, n≥3f+1n\ge 3f+1n≥3f+1).

Prediction module (GNN)

We construct a dynamic interaction graph G=(V,E)G=(V,E)G=(V,E) whose nodes VVV are accounts/contracts and edges EEE are recent interactions with features (frequency, value, gas, method). A temporal GNN (e.g., TGAT/DCRNN-style) forecasts edge

intensities and node demand to produce a traffic tensor feeding the RL state. The GNN is trained online with replay buffers and periodic teacher-forcing on high-confidence labels.

RL algorithm

An actor-critic with proximal policy optimization (PPO) and generalized advantage estimation stabilizes updates. Constrained RL is implemented via action shielding: before execution, candidate actions are validated by a safety layer that enforces (a) committee size minima, (b) unbiased validator sampling, (c) reconfiguration rate limits, (d) data availability budget per epoch. Infeasible actions are clipped to the nearest valid action in policy space; the policy receives negative reward to internalize constraint costs.

Security and correctness envelope

- **Committee integrity:** VRF-sampled validators; minimum n≥3f+1n\ge3f+1n≥3f+1; stake-weighted or identity-mixed depending on chain design.
- **Cross-shard atomicity:** Two-phase commit with time-locked receipts; retries at relays with exponential backoff to avoid congestion collapse.
- **Data availability:** Erasure-coded block dissemination and light-client proofs; reconfiguration halts if DA score dips below threshold.
- **Explainability:** Each action is logged as a tuple (st,at,rt)(s_t,a_t,r_t)(st,at,rt) with a hash anchored on-chain. A distilled decision stump approximates action logic for auditors, enabling post-hoc verification without leaking raw telemetry.

Simulation environment

- **Traffic:** (i) Uniform: balanced DApp activity; (ii) Skewed: Zipf α≈1.2 hot-spotting; (iii) Bursty: on-off Poisson with bursts 5–20× baseline.
- **Network:** Latency 30–150 ms (triangular), packet loss 0.2–1.0%; adversarial validators up to 30% per shard (non-colluding across epochs).
- **Baselines:** Static partitioning from initial balanced cut; Heuristic adaptive with threshold triggers on queue depth and cross-shard volume.
- **Training:** Warm-start GNN on 10k steps; RL trained online for 50 epochs; rollouts parallelized across traffic seeds.

Ethics note: no human subjects; synthetic/system logs only.

RESULTS

Throughput and latency

Table 1 shows that AI-assisted sharding increases TPS by +31% (Uniform), +47% (Skewed), +46% (Bursty) vs. Heuristic, and +64–94% vs. Static, while reducing p99 latency by −19% to −44%. Improvements are largest under Skewed and Bursty workloads where predictions and proactive resizing reduce hot-spot formation.

Cross-shard reliability

Cross-shard commit success improves by 0.5–2.1 percentage points over Heuristic and 2–6 points over Static. The primary driver is flow-aware partitioning: the GNN steers contracts that frequently interact into the same shard or into "paired" shards with preferential relay capacity.

Reconfiguration overhead

AI reconfigures slightly faster than Heuristic (5–6 s vs. 6.5–7.8 s per event) because it schedules migration windows when DA and bandwidth headroom are predicted to be high, and it bundles small moves. Static has zero overhead but suffers significant congestion under skew/bursts.

Security posture

No runs violated committee-size and randomness constraints. The safety layer clipped ≈8–12% of suggested actions early in training; by epoch 30 this dropped below 2%, indicating that the policy learned to internalize constraints.

Ablation

Removing the GNN (RL-only) reduced TPS by 10–14% and increased p99 latency by 9–12% across regimes, underscoring the value of predictive signals. Using GNN-only (no RL; greedily minimize predicted cross-shard edges) improved throughput over Static but created migration churn and higher orphan rates during bursts, highlighting the need for sequential decision-making.

Statistical validation

Welch's t-tests (two-sided) on per-run TPS and p99 latency yielded $p<0.01p<0.01p<0.01$ (AI vs. Heuristic) and $p<0.001p<0.001p<0.001$ (AI vs. Static) in all regimes. Estimated Cohen's ddd ranged from 0.6–1.1 (medium–large) for AI vs. Heuristic and >1.0 (large) for AI vs. Static.

Overheads of AI

The controller consumed <2% additional CPU per validator client for inference (batched every 2–5 blocks). Training is offloaded to monitoring nodes; on resource-constrained deployments, policies can be pretrained and updated intermittently.

CONCLUSION

AI-assisted sharding provides a principled pathway to anticipatory, security-compliant adaptation in sharded blockchains. By coupling graph-based workload forecasts with a constrained RL controller, the system proactively reshapes shard boundaries, committee sizes, and routing hints to mitigate hot-spots and cross-shard friction. In controlled simulations spanning uniform, skewed, and bursty workloads, the approach significantly increases throughput, reduces tail latency, and improves cross-shard commit reliability compared to static and heuristic baselines—without breaching cryptographic safety constraints. While simulations cannot capture all real-world complexities—MEV behavior, heterogeneous validators, network pathologies—the results indicate that machine-in-the-loop sharding is both technically feasible and performance-beneficial. The key is strong guardrails: the AI suggests, but the protocol disposes under verifiable constraints, with auditable logs for governance and compliance.

FUTURE SCOPE OF STUDY

1. **On-chain verifiability of AI decisions:** Explore zk-SNARK/zk-STARK attestations for bounded-rational policy steps (e.g., proving adherence to constraints without revealing telemetry).
2. **Economic incentives:** Design staking and fee mechanisms that reward informative telemetry and penalize manipulation of AI inputs; analyze game-theoretic equilibria.
3. **MEV-aware adaptation:** Integrate MEV forecasts to prevent AI from inadvertently amplifying extractable value or enabling cross-shard sandwich patterns.
4. **Heterogeneous validators:** Incorporate hardware/latency profiles into committee formation to improve fairness and resilience.
5. **Multi-chain and rollup ecosystems:** Extend the controller to inter-shard/inter-rollup routing across L2s/L3s with shared sequencers and DA layers.
6. **Formal methods:** Verify safety wrappers using temporal logic and model checking; co-design with consensus proofs to bound worst-case behavior.
7. **Privacy-preserving telemetry:** Apply differential privacy or secure aggregation to protect user/activity metadata while retaining utility for forecasting.
8. **Human-in-the-loop governance:** Create dashboards and playbooks for operators to supervise and veto AI actions during abnormal events.
9. **Robustness to adversarial inputs:** Study poisoning/evasion attacks on predictors and policies; harden with adversarial training and sensor fusion.
10. **Production trials:** Pilot deployments on testnets with opt-in shards, measuring real DApp performance and operator trust.

REFERENCES

- *Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., ... & de Freitas, N. (2016).* ***Learning to learn by gradient descent by gradient descent****. Advances in Neural Information Processing Systems, 29.*

- *Buterin, V. (2018).* ***Sharding FAQ and design notes****. Ethereum Research Forum.*
- *Castro, M., & Liskov, B. (1999).* ***Practical Byzantine Fault Tolerance****. OSDI.*
- *Dwork, C. (2006).* ***Differential privacy****. Automata, Languages and Programming, 1–12.*
- *Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E., & Ford, B. (2018).* ***Omniledger: A secure, scale-out, decentralized ledger****. IEEE S&P, 583–598.*
- *Lai, T. L., & Robbins, H. (1985).* ***Asymptotically efficient adaptive allocation rules****. Advances in Applied Mathematics, 6(1), 4–22.*
- *Luu, L., Narayanan, V., Baweja, K., Zheng, C., Gilbert, S., & Saxena, P. (2016).* ***A secure sharding protocol for open blockchains (Elastico)****. ACM CCS Workshop on Distributed Cryptocurrencies and Consensus Ledgers.*
- *Maiyya, S., et al. (2019).* ***Monoxide: Scale out blockchains with asynchronous consensus zones****. USENIX NSDI, 95–112.*
- *NEAR Collective. (2020).* ***Nightshade: Sharding design for NEAR Protocol****. Whitepaper.*
- *Pass, R., & Shi, E. (2017).* ***Hybrid consensus: Efficient consensus in the permissionless model****. DISC, 39:1–39:16.*
- *Qu, G., Tang, J., & Kurths, J. (2018).* ***Optimal synchronization of complex networks via pinning control: A survey****. Automatica, 90, 31–44.*
- *Shamir, A. (1979).* ***How to share a secret****. Communications of the ACM, 22(11), 612–613.*
- *Snoek, J., Larochelle, H., & Adams, R. P. (2012).* ***Practical Bayesian optimization of machine learning algorithms****. NIPS, 2951–2959.*
- *Sompolinsky, Y., & Zohar, A. (2018).* ***Phantom and GhostDAG: BlockDAG protocols****. Cryptology ePrint Archive.*
- *Team, Zilliqa. (2018).* ***The Zilliqa technical whitepaper****. Zilliqa Research.*
- *Valiant, G. (2012).* ***Estimating the unseen: An n/log(n)-sample estimator for entropy and support size****. Advances in Neural Information Processing Systems, 25.*
- *Wang, J., Wang, S., & Chen, X. (2019).* ***Monetary incentives and blockchain sharding security****. IEEE Communications Surveys & Tutorials, 21(4), 2895–2922.*
- *Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2020).* ***A comprehensive survey on graph neural networks****. IEEE TNNLS, 32(1), 4–24.*
- *Zamani, M., Movahedi, M., & Raykova, M. (2018).* ***RapidChain: Scaling blockchain via full sharding****. ACM CCS, 931–948.*
- *Zamyatin, A., et al. (2019).* ***X-chain 2PC: Cross-chain atomic swaps and commits****. Cryptology ePrint Archive.*