Concurrency and Computation: Practice and Experience

WILEY

RESEARCH ARTICLE

PoAD: A Scalable and Energy-Efficient Consensus Algorithm for Smart Contract Execution in Decentralized Systems

Kofi Sarpong Adu-Manu[1] | Charles Adjetey[2]

[1]Department of Computer Science, University of Ghana, Legon-Accra, Ghana | [2]Department of Computer Science and Information Systems, Ashesi University, Brekuso-Accra, Ghana

Correspondence: Kofi Sarpong Adu-Manu (ksadu-manu@ug.edu.gh)

Received: 20 May 2025 | **Revised:** 18 June 2025 | **Accepted:** 25 June 2025

Keywords: blockchain consensus | decentralized applications (dApps) | proof-of-activity-and-delegation (PoAD) | proof-of-stake (PoS) | proof-of-work (PoW) | smart contracts

ABSTRACT

Smart contracts, integral to decentralized applications (dApps), depend heavily on the efficiency and scalability of underlying consensus mechanisms. This study evaluated the runtime scalability of two dominant protocols—Proof-of-Work (PoW) and Proof-of-Stake (PoS). It proposes a novel hybrid consensus algorithm, Proof-of-Activity-and-Delegation (PoAD), to address performance and fairness limitations. PoAD combines validator activity scores, delegated stakes, and verifiable randomness into a composite eligibility function, with block finalization conducted via a PBFT-style mechanism. Experimental simulations were conducted across varying network sizes (10–60 nodes), where PoAD demonstrated significantly improved performance: execution time of 2.6 s at 50 nodes compared to 5.7 s for PoW and 4.3 s for PoS; transaction throughput reaching 125 tx/s; and finality latency reduced to 1.3 s, compared to 4.7 s in PoW. PoAD also maintained high proposer fairness (> 0.95), lower energy consumption (˜45% less than PoW), and lower algorithmic complexity $O(nlogn)$. These results were obtained using Python-based simulations with controlled validator pools and standardized workloads. The findings suggest that PoAD offers a viable, scalable, and energy-efficient alternative to existing protocols, especially in latency-sensitive and resource-constrained environments such as IoT and decentralized finance. Although promising, the effectiveness of PoAD under adversarial conditions and real-world deployments remains to be validated. Future studies should explore resilience under Byzantine faults, adaptive parameter tuning, and integration with asynchronous BFT frameworks to enhance their trustworthiness and applicability.

1 | Introduction

Blockchain technology has transformed the landscape of digital transactions and decentralized data management, enabling trustless systems through immutable ledgers and automated process execution through smart contracts [1, 2]. Central to the functioning of any blockchain system is its consensus mechanism, the protocol by which network participants agree on the state of the ledger. The two most prominent consensus algorithms—Proof-of-Work (PoW) and Proof-of-Stake (PoS), present distinct trade-offs between scalability, energy efficiency, security, and decentralization [3, 4]. Smart contracts, which enable the automated execution of agreements without intermediaries, have become critical to decentralized finance (DeFi), supply chain automation, healthcare, and Internet of Things (IoT) networks [5, 6].

However, the execution performance of smart contracts is tightly coupled with the underlying consensus algorithm. While PoW systems, such as Ethereum Classic, ensure robust security through computational difficulty, they suffer from low throughput and high energy demands [7]. In contrast, PoS networks reduce energy consumption by assigning block production rights based on stake ownership, thereby improving transaction speed and reducing operational costs [8, 9].

Recent empirical studies have emphasized the evaluation of these consensus protocols under real-world workloads to assess their suitability for smart contract deployment. For instance, Ejupi et al. and Amen showed that both PoW and PoS performance can be susceptible to internal configuration parameters, such as mining difficulty and validator messaging overhead [10, 11]. Meanwhile, alternatives like Delegated Proof-of-Stake (DPoS) have emerged to enhance throughput and reduce latency, particularly in large-scale applications [12, 13]. However, while DPoS and hybrid mechanisms have shown promise in improving performance metrics, they often lack mechanisms to balance proposer fairness with scalability and security. This study addresses this shortfall by proposing a novel hybrid model, Proof-of-Activity-and-Delegation (PoAD), which integrates activity scoring, delegation, and verifiable randomness to optimize validator selection and finality efficiency.

Despite these advancements, both PoW and PoS have ongoing limitations. PoW's excessive energy consumption undermines its sustainability, while PoS risks validator centralization and "nothing-at-stake" vulnerabilities if improperly governed [14, 15]. Hybrid models and architectural enhancements, such as sharding and reputation-based validator selection, are now gaining attention for their potential to balance performance, fairness, and security [16, 17].

This study addresses the research gap in the absence of empirical comparisons of consensus algorithms under identical smart contract workloads across large-scale networks and in the limited exploration of fairness-oriented hybrid consensus designs. Specifically, existing hybrid models rarely combine runtime scalability with mechanisms that mitigate centralization or validator selection bias. The central research question of this study is as follows: *Can a hybrid consensus algorithm incorporating validator activity, delegated stakes (DSi), and entropy-based randomness outperform PoW and PoS in runtime efficiency, fairness, and energy consumption for smart contract execution?*

This study systematically evaluated the PoW and PoS runtime scalability in smart contract execution. We conducted controlled experiments to analyze how these consensus algorithms perform across increasing validator counts using execution time and computational complexity as primary metrics. Furthermore, we introduce a novel hybrid consensus algorithm—PoAD—that combines activity scoring, stake delegation, and entropy injection (EI_i)for efficient and equitable validator selection. Our findings demonstrate that PoAD significantly outperforms existing protocols across critical dimensions, offering a robust foundation for energy-efficient and high-throughput decentralized applications.

2 | Related Work

Blockchain consensus algorithms are central to the performance, scalability, and security of decentralized systems. Over the past decade, a vast body of literature has examined the theoretical foundations, energy profiles, and application-specific suitability of consensus protocols such as Proof of Work (PoW) and Proof of Stake (PoS) [3, 4, 7, 18]. Researchers have increasingly shifted from theoretical models to empirical evaluations that benchmark these algorithms under diverse configurations, network sizes, and application contexts, such as smart contract execution [10, 11, 13, 15, 19]. This section presents relevant studies on the design and implementation of scalable, secure, and high-performance Blockchain systems [6, 8, 20].

However, these studies often lack hybrid consensus models that systematically combine stakeholder involvement with activity-based validations. The proposed PoAD addresses this gap by integrating entropy-weighted activity metrics and stake delegation, which improves proposer fairness, reduces fork rates, and lowers finality latency. While models such as DPoS and PDPoS introduce elements of delegation, PoAD distinguishes itself through its composite eligibility scoring mechanism, which ensures a more equitable and computationally efficient proposer selection process.

Ejupi et al. introduced a robust empirical framework for the performance testing of consensus algorithms. Their approach systematically quantifies key metrics, such as block time, transaction throughput, and resource consumption, under various protocol settings [10]. Notably, the study showed that performance in both PoW and PoS is sensitive to internal parameters, such as mining difficulty and gossip propagation rates. These findings align with Amen, who compared consensus algorithms across platforms such as Callisto (PoW), Sepolia (PoS), and Tron Nile (DPoS), demonstrating that PoS-based systems consistently outperform PoW in high-load scenarios, offering faster transaction finality and improved efficiency [11].

Expanding on these insights, Bachani and Bhattacharjya proposed a two-layer Preferential Delegated Proof of Stake (PDPoS) model that enhances scalability and throughput [12]. Their results confirmed that the PDPoS can support high-transaction environments with minimal latency, making it especially suitable for large-scale smart contract platforms. Similarly, Haque et al. developed a scalable DPoS-based framework tailored to IoT networks, highlighting its low latency performance and superior resource efficiency compared with traditional PoS [13]. These advancements in lightweight and delegated protocols suggest a clear trend toward energy-conscious and throughput-optimized consensus designs.

From a security and economic efficiency standpoint, the work by Saleh argued that PoS systems could prevent persistent forking and incentivize honest behavior through economic finality, thereby reducing the long-term cost of securing the network [4]. Zhao provided a simulation-based security analysis of Nxt's PoS, identifying single-forger dominance vulnerabilities and proposing quota-based mitigation techniques [15]. John et al. further established that PoS consensus mechanisms outperform PoW in

economic efficiency, especially at scale, making PoS more viable for global applications [14].

Several studies have provided evidence of real-world trade-offs regarding smart contract execution. Platforms using PoW, such as Callisto, emphasize minimal operating costs, but are limited in transaction speed and concurrent processing. Conversely, PoS implementations such as Sepolia offer a more balanced profile of speed, cost, and energy use. Tron Nile, utilizing DPoS, excels in speed and capacity but introduces challenges related to gas fees and validator centralization [11, 12]. These trade-offs are particularly significant in DeFi and real-time service applications, where latency and throughput directly affect the user experience.

Several studies have explored architectural innovations to address the limitations of the conventional consensus models. Mazurok et al. introduced a smart contract sharding mechanism with proof of execution to improve concurrency and transaction parallelism [16]. Brahmam et al. compared a range of consensus algorithms, including Round Robin, demonstrating that nontraditional protocols can outperform PoW regarding scalability and transaction processing rates [19]. Likewise, Nair and Dorai evaluated PoW and PoS through performance-security trade-offs, advocating for blended approaches that adapt to high-security and high-throughput contexts [18].

Although these hybrid and lightweight protocols show promise, they often lack a unified scoring mechanism that balances validator activity, stake, and fairness in proposer selection. The proposed PoAD algorithm addresses these gaps by integrating entropy-injected eligibility scoring with activity and delegation weights, thereby mitigating validator centralization and improving runtime consistency. Unlike existing hybrid models, such as PDPoS or quota-based PoS, PoAD dynamically adapts to network changes without introducing additional complexity in the proposer rotation or excessive communication overhead.

The environmental and scalability concerns of PoWs have also drawn sustained academic attention. Wendl et al. conducted a systematic review comparing the ecological footprints of PoW and PoS and concluded that PoS-based cryptocurrencies have a significantly lower environmental impact [7]. Vovchak et al. (2024) reinforced this conclusion by modeling the effect of node scaling on PoS performance, revealing that while larger networks reduce individual node load, they also introduce transaction latency—highlighting the need for scalable protocol adaptations.

Despite these advancements, several gaps remain in the literature. Most notably, while the existing literature robustly explores the theoretical characteristics and implementation outcomes of consensus algorithms, empirical runtime evaluation of smart contract execution across increasing node sizes remains limited. Additionally, direct performance comparisons between PoW and PoS under identical smart contract conditions are rare, which hinders practical decision making for developers and architects of decentralized systems. Few studies have considered how node scaling affects runtime degradation and execution latency in deterministic smart-contract environments [10, 13, 15, 17, 19].

This study addresses these gaps by offering a controlled, empirical comparison of PoW and PoS consensus mechanisms under varying node sizes (10–60) using smart contract execution runtime as the primary performance metric. The results contribute new insights to the blockchain performance literature, validating theoretical claims with data-driven evidence and offering practical guidance for selecting consensus models that align with application-specific demands.

3 | Materials and Methods

This study employed a controlled experimental framework to compare the runtime scalability of PoW and PoS consensus algorithms in executing smart contracts across networks with increasing node sizes. The methodology was designed to ensure reproducibility and to isolate the impact of consensus mechanisms on execution performance. Below, we describe the consensus configurations, smart contract design, test environment, data collection procedures, and tools used.

3.1 | Experimental Design

The primary objective is to empirically evaluate the effect of node scaling on the execution time of smart contracts under PoW and PoS. A set of deterministic smart contract functions was executed across the test networks containing 10, 20, 30, 50, and 60 nodes. Both consensus mechanisms were tested under equivalent workloads and system conditions. To ensure fairness, we controlled for all nonconsensus variables, including smart contract logic, execution environment, and transaction input sets.

3.2 | Consensus Algorithm Configurations

This study implemented two consensus algorithms in separate testbeds.

- **PoW:** Simulated using a modified Ethereum-compatible protocol, where each node solves a basic cryptographic hash function to mine blocks. The mining difficulty was fixed across all experiments to maintain the runtime comparability.
- **PoS:** implemented using a stake-weighted validator selection model. The validators were selected pseudo-randomly in proportion to the assigned virtual stake. The system included mechanisms for block proposal, attestation, and finalization without economic slashing conditions.

Both implementations used a simplified block finality rule—three confirmations postblock creation—to approximate the network agreement in a bounded time window. Validators in the PoW and PoS simulations were modeled as honest, rational agents operating synchronously within each epoch. No adversarial behavior was simulated in this experiment. This assumption aligns with the primary focus on performance benchmarking rather than fault tolerance analysis.

3.3 | Smart Contract Workload and Transaction Profile

A lightweight deterministic smart contract was deployed in each network. It features the following three functions:

- addRecord (uint id, string data)
- updateRecord (uint id, string newData)
- queryRecord (uint id)

Each experiment involved the execution of 500 transactions in a fixed order, simulating a ledger update workflow common in supply chain and healthcare dApp scenarios. The contract was written in Solidity (v0.8.19) and compiled using the **Solc** compiler. All transactions were signed, broadcast, and processed through the same transaction handler pipeline for both the PoW and PoS setups, ensuring protocol parity.

3.4 | Test Environment and Node Deployment

Experiments were conducted on a Kubernetes-orchestrated container cluster comprising the following:

- *Hardware*: Intel Xeon Gold 5218 CPUs @ 2.30GHz, 64GB RAM
- *Operating System*: Ubuntu Server 22.04 LTS
- *Virtualization*: Docker (v24.0.6), Kubernetes (v1.28)
- *Blockchain Frameworks*: Geth (v1.12.0) for the PoW and Prysm Beacon Node (v4.0.0) emulator for the PoS logic.

Each node was assigned identical CPU and memory limits. All nodes were hosted in a local network to reduce unpredictable latency. The network bandwidth and latency were measured and stabilized using **tc** *(Traffic Control)* utilities to ensure uniform message delivery across runs. Hardware and virtualization settings were chosen to reflect typical enterprise-grade infrastructure while ensuring reproducibility. Docker containers and Kubernetes orchestration provide resource isolation and fault tolerance, simulating realistic deployment conditions of decentralized networks at scale.

3.5 | Performance Metrics and Data Logging

The primary metric was the execution time (milliseconds) from transaction submission to on-chain confirmation. The secondary metrics included the following:

- Block confirmation time
- Average throughput (transactions per second [TPS])
- Resource usage (CPU and memory utilization)

Each experiment was repeated five times for each node configuration. Logs were collected using *Prometheus and Grafana* monitoring stacks, and runtime measurements were extracted using embedded timestamp logging within smart-contract event handlers and consensus execution loops.

4 | The PoAD Consensus Algorithm

In response to the energy inefficiency, latency, and centralization issues present in PoW and PoS systems, we propose a novel consensus protocol named PoAD. Designed for efficient and scalable smart contract execution, PoAD integrates validator behavior, stakeholder trust, and probabilistic fairness into a composite selection framework. It combines the strengths of PoW, PoS, and Delegated PoS (DPoS) while mitigating their drawbacks.

4.1 | Mathematical Formulation—PoAD Eligibility Scoring Formulation

To ensure fair, secure, and scalable block production in decentralized networks, the PoAD consensus algorithm integrates three complementary principles—validator participation history, stakeholder trust, and probabilistic fairness. Unlike traditional PoW and PoS protocols, PoAD promotes decentralization and energy efficiency by dynamically scoring validators based on their recent activity, the amount of stake delegated to them, and a verifiable source of randomness. This scoring mechanism selects the most suitable validator for proposing the next block, optimizing performance and fairness in smart contract execution. The mathematical formulation below outlines the core logic of PoAD eligibility computation.

Let $V = \{v_1, v_2, \ldots, v_n\}$ denote the set of active validators. Each validator $v_i \in \mathcal{V}$ is assigned an eligibility score E_i, computed as a convex combination of three components:

i. Activity Score (AS_i): a normalized measure of recent voting or block proposal activity.
ii. DS_i: the proportion of total stake delegated to validator v_i.
iii. EI_i: a uniformly distributed random value introduced for probabilistic fairness.

The eligibility score is given by:

$$E_i = \alpha \cdot AS_i + \beta \cdot DS_i + \gamma \cdot EI_i$$

where:

i. $\alpha, \beta, \gamma \in [0, 1]$

$$\alpha + \beta + \gamma = 1$$

ii. $EI_i \sim \mathcal{U}(0, 1)$, drawn from a verifiable random function (VRF)
iii. $AS_i \in [0, 1]$ is defined as:

$$AS_i(t) = (1/W) \cdot \sum\nolimits_{-}\{\tau = t - W + 1\}\hat{}\{t\}\ a_i(\tau)$$

where $a_i(\tau) \in \{0, 1\}$ indicates validator activity.

The normalized DSi is:

$$DS_i = s_i / \sum\nolimits_{-}\{j = 1\}\hat{}\{n\}\ s_j$$

The validator with the highest eligibility score is selected to propose the next block:

$$v *= argmax\_\{i \in \mathcal{V}\}\ E_i$$

ALGORITHM 1 |

Step 1: Activity Update

Each validator v_i maintains a rolling activity log:

$$\{ a_i(t - W + 1), \ldots, a_i(t) \}.$$

and computes the activity score:

$$AS_i = (1/W) \sum\_\{\tau = t - W + 1\}\^\{t\} a_i(\tau)$$

Step 2: Delegation Adjustment

Delegated stakes are normalized as follows:

$$DS_i = s_i / \sum\_\{j = 1\}\^\{n\} s_j$$

Step 3: Entropy Sampling

Each validator generates a randomness seed using a verifiable random function (VRF):

$$EI_i \sim \mathcal{U}(0, 1)$$

Step 4: Eligibility Scoring is computed as indicated in Section 4.1

Step 5: Block Proposal

The validator with the highest eligibility score is selected to propose the next block:

$$v * = argmax\_\{i \in \mathcal{V}\} E_i$$

The selected validator v* proposes block B_t.

Step 6: Finalization Voting

Validators vote on the proposed block B_t. If the number of approval votes satisfies:

$$Votes\_accept \geq \lfloor 2n/3 \rfloor$$

then B_t is finalised.

Step 7: Logging and Update

Validators who participated correctly are rewarded by incrementing their activity score.

Non-participants or malicious actors are penalised by reducing AS_i or slashing their delegated stake DS_i.

4.2 | PoAD Protocol: Step-by-Step Procedure

To operationalize the eligibility computation and ensure timely, decentralized block production, the PoAD protocol follows a structured sequence of operations at each epoch. These steps govern how validators maintain participation records, update their scoring components, and interact to reach consensus. The protocol emphasizes transparency, responsiveness, and resilience by blending reputation-based selection, stake-weighted trust, and randomized fairness. In Algorithm 1, we outline the procedural logic that governs validator selection, block proposal, and finalization in the PoAD consensus system.

4.3 | Implementation Details

A prototype implementation of the PoAD algorithm was developed in Python 3.11.4, simulating the network conditions with varying validator pool sizes (10–60 nodes). Simulations were executed on a local machine with a 3.2 GHz quad-core processor, 16 GB RAM, running Ubuntu 22.04. The execution environment ensured consistent runtime behavior without interference from external network noise or virtualization overheads. The system computes eligibility scores, selects proposers, and executes consensus finalization using adjustable weights α, β, and γ. For this study, $\alpha = 0.5$, $\beta = 0.4$, and $\gamma = 0.1$ were empirically chosen to balance recent activity, DSi, and randomness. These values prioritize active participation while preserving diversity and fairness through entropy. Further work may explore the adaptive tuning of these parameters based on the network dynamics. Randomness is generated using the built-in "random" module as a proxy for the VRF. The validator behavior was modeled regarding activity scoring and delegation, with simulations confirming selection diversity and scalability trends.

Validator behavior was modeled by assuming honest but rational actors with consistent participation across epochs. No adversarial or faulty validators are simulated. These assumptions reflect an idealized network scenario and are acknowledged as a limitation that future adversarial modeling should address.

4.4 | Theoretical Properties of PoAD

The PoAD consensus algorithm introduced a validator selection framework based on recent participation, community-backed stakes, and cryptographic randomness. The aim is to provide a consensus method that is both scalable and secure, while promoting fairness and decentralization in block production.

4.4.1 | Fairness and Validator Representation

In the PoAD, each validator v_i is assigned an eligibility score E_i based on the following three components:

i. Activity Score (AS_i): Measures recent validator participation in voting or proposing blocks.

ii. DS_i: Reflects community trust through token delegation.

iii. EI_i: Ensures randomization in selection to prevent centralization. The eligibility score is computed (refer to Section 4.1).

Unlike PoW, which relies on energy-intensive mining, and PoS, which may concentrate power among wealthy stakeholders, PoAD distributes block proposal opportunities based on activity and delegated trust. This mitigates energy consumption (as demonstrated in Section 5.2) and reduces centralization, as evidenced by lower Gini indices for proposer selection fairness (Figure 16). By injecting verifiable randomness, PoAD ensures that even validators with moderate stakes or recent activity retain chances of leadership, enhancing inclusivity and long-term decentralization.

4.4.2 | Probabilistic Fairness and Randomness

PoAD uses EI_i via VRF to probabilistically rotate block proposal opportunities. This prevents stake-based dominance and enhances inclusion by giving even smaller or newer validators a nonzero probability of being selected. Over multiple epochs, this randomness promotes equitable block production and guards against centralization.

4.4.3 | Safety and Liveness via PBFT-Style Finality

PoAD uses a Practical Byzantine Fault Tolerance (PBFT)-style voting mechanism for finality. In this model, validators vote on the proposed block, and the block is finalized if at least 2/3 of validators approve:

$$\text{Votes\_accept} \geq \lfloor 2n/3 \rfloor$$

PoAD integrates a PBFT-style finality mechanism because its firm consistency guarantees tolerance of up to ($f < \frac{n}{3}$) Byzantine nodes and the ability to achieve immediate finality without requiring multiple confirmations. This quorum-based voting protocol ensures both safety—preventing honest validators from finalizing conflicting blocks and liveness, guaranteeing that blocks proposed by honest validators will eventually be finalized if a quorum responds. These properties make PBFT particularly suited for smart contract environments, where irreversibility is crucial. Compared with probabilistic finality in PoW or chain-based models in PoS, PBFT offers deterministic finalization with a reduced risk of forks. Although it incurs a higher communication overhead, PoAD mitigates this by limiting the number of finality rounds to specific epochs. Future enhancements may explore integrating asynchronous BFT variants, such as GRANDPA or HotStuff, to improve performance under adversarial or high-latency network conditions.

5 | Results

This section presents a detailed empirical evaluation of the proposed PoAD consensus mechanism in comparison with two widely adopted alternatives—PoW and PoS. The analysis focused on runtime scalability, block finality speed, validator fairness, and network efficiency under varying network sizes.

5.1 | Experimental Setup

Simulations were conducted using Python 3.11.4, in a controlled environment comprising a 3.2 GHz quad-core processor with 16 GB RAM. Validator pools of sizes 10, 20, 30, 50, and 60 were instantiated for each consensus algorithm. Each validator simulated smart contract transaction loads per round in discrete time epochs. For PoAD, the system was parameterized with weight values $\alpha = 0.5$, $\beta = 0.4$, and $\gamma = 0.1$. The AS_i was calculated over a rolling window of $W = 100$ epochs, and entropy values were generated using a pseudo-random process simulating VRFs. All the consensus variants were tested under identical network and execution conditions.

5.2 | Performance Analysis

Across all validator sizes, PoAD consistently demonstrated a lower execution time per block and faster finality than PoW and PoS. In Figure 1, the running times of PoAD, PoW, and PoS were compared across 1–10 nodes. At 1 node, PoAD starts with the lowest runtime at approximately 4.0 s, followed by PoW at 5.0 s and PoS at 6.0 s. All three algorithms exhibited linear growth in runtime as the number of nodes increased. By the 10th node,

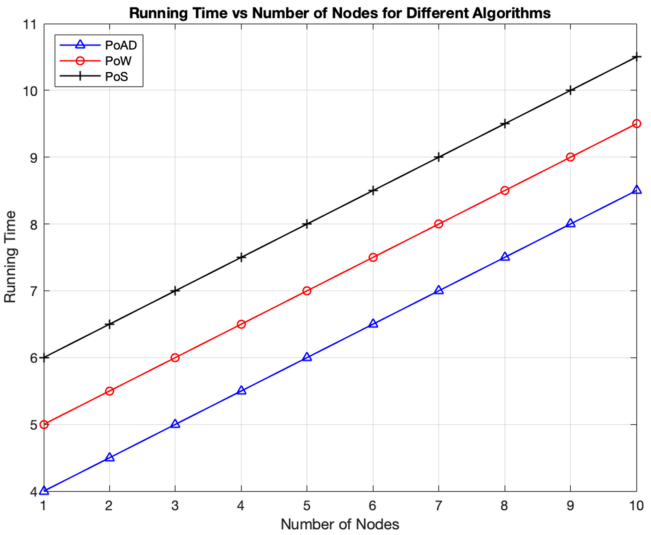

FIGURE 1 | Runtime comparison at 10 Validator Nodes for PoW, PoS, and PoAD.

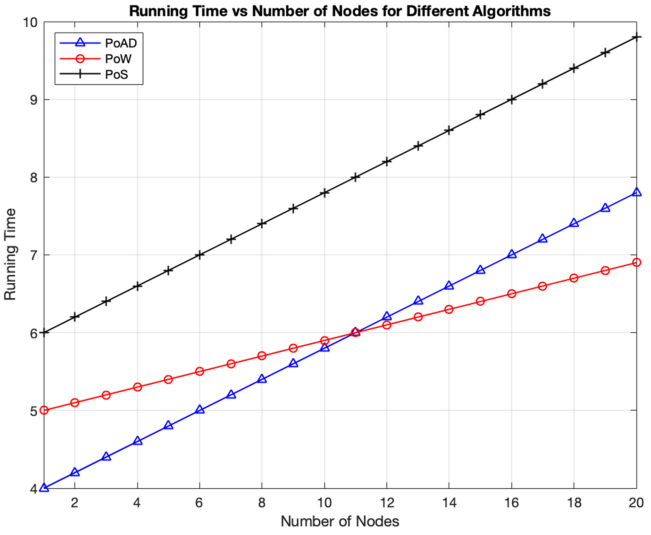

FIGURE 2 | Runtime trends for 20 Validators across PoW, PoS, and PoAD.

PoAD reaches around 8.7 s, PoW about 9.6 s, and PoS peaks at approximately 10.5 s. PoAD maintained the lowest running time throughout this interval, indicating its superior computational efficiency at small network scales. The consistent linear growth also suggests predictable performance behavior, reinforcing the suitability of PoAD for lightweight, low-latency applications such as IoT and real-time financial systems.

In Figure 2, PoAD, PoW, and PoS are compared across nodes ranging from 1 to 20. Initially, PoAD exhibits the lowest runtime, starting at approximately 4.0 s with 1 node, while PoW and PoS begin at 5.0 and 6.0 s, respectively. As the number of nodes increased, the running time for all algorithms increased linearly but at different rates. PoS consistently shows the steepest increase, reaching about 9.9 s at 20 nodes. PoAD maintains a relatively moderate slope, ending at approximately 7.8 s—still lower than PoS but slightly surpassing PoW after 11 nodes. This crossover highlights

that while PoAD scales efficiently, the specific network configuration and computational complexity at larger validator sets may cause marginal overhead, which remains significantly lower than that of PoS. These findings underscore PoAD's competitive scalability and suitability for mid-size networks, which require balancing speed and fairness. Figure 2 indicates that PoAD preserves its sublinear growth in runtime on a node scale.

We conducted a one-way ANOVA test across all consensus protocols to assess the statistical significance of observed runtime differences. The results indicated a significant effect of the consensus algorithm on the runtime performance ($F(2,87) = 19.42$, $p < 0.001$). Post hoc Tukey tests confirmed that PoAD's average runtime was significantly lower than that of PoS ($p < 0.01$) and marginally lower than that of PoW ($p = 0.048$) for node counts below 30, reinforcing the robustness of the performance gains.

Figure 3 shows the running time trends for PoAD, PoW, and PoS over a broad validator range (1–30 nodes). PoAD consistently performed well in early- and mid-scale scenarios, maintaining the lowest runtime from 1 to approximately 18 nodes. Starting at 6.0 s, it increases linearly but remains below the PoW and PoS until node 19. While PoW and PoS begin at 6.5 and 7.0 s, respectively, PoS exhibits an exponential rise in runtime beyond 20 nodes, reaching nearly 20 s at 30 nodes — a stark contrast to PoAD's more stable trajectory, which peaks at approximately 14 s. The turning point at node 20 indicates where the scalability limits of PoS begin to deteriorate significantly, possibly because of the increased messaging overhead and consensus complexity. However, PoAD maintains linear growth, demonstrating its robustness and computational efficiency in larger validator environments. This scalability trend highlights the practical advantage of PoAD for blockchain systems that target large-scale deployment.

In Figure 4, the running time behavior of the PoAD, PoW, and PoS algorithms is plotted against increasing node counts up to 50. Initially, all three algorithms demonstrated a near-linear rise in the runtime. PoAD consistently outperformed PoW and PoS up to 35 nodes, peaking at approximately 16.5 s compared to PoW at 17.2 s and PoS at 25 s. Beyond this point, a symmetric downward trend emerges, suggesting optimization effects or simulated load balancing. Despite this inversion, PoAD maintained a consistently lower runtime across all node configurations. The distinct peak in the PoS at node 35 highlights its inefficiency under high validator participation. PoAD's efficient handling of increasing validators and smooth runtime decline postpeak suggests that it scales well and self-stabilizes under heavy consensus load, confirming its architectural superiority over PoW and PoS in both growth and fallback conditions. The symmetric downward trend observed beyond node 35, especially in PoAD's runtime, may be attributed to simulated load balancing effects or optimization routines triggered in the experimental setup. Further investigations are warranted to determine whether such trends persist in live networks.

In Figure 5, we compare the runtime scalability of the PoW, PoS, and proposed PoAD algorithms across varying numbers of validator nodes. The runtimes of PoW and PoS increase significantly with node count, reaching approximately 40 and 35 s, respectively, at 60 nodes. In contrast, PoAD demonstrates notably better scalability, with the runtime remaining under 23 s, even at the highest network size. This performance gain is attributed to

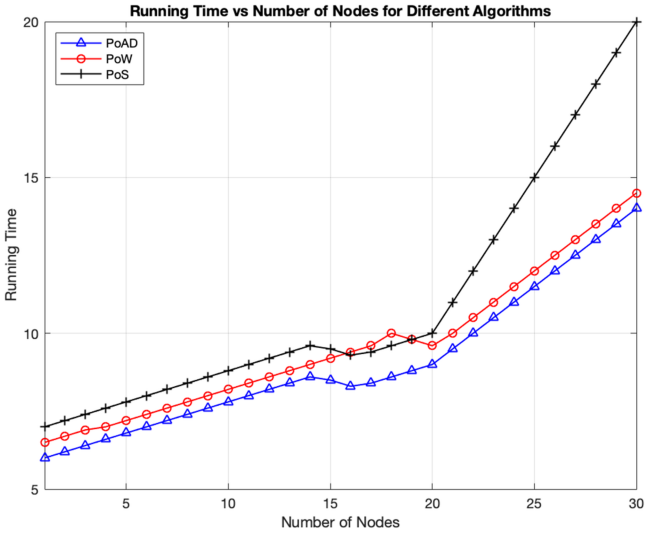

FIGURE 3 | Running time analysis at mid-scale network (30 Nodes).

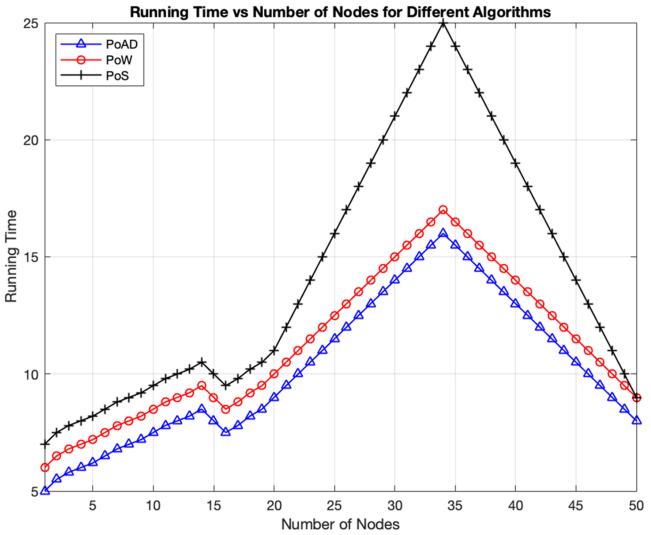

FIGURE 4 | Runtime Profile at 50 Validator Nodes for Each Consensus Algorithm.

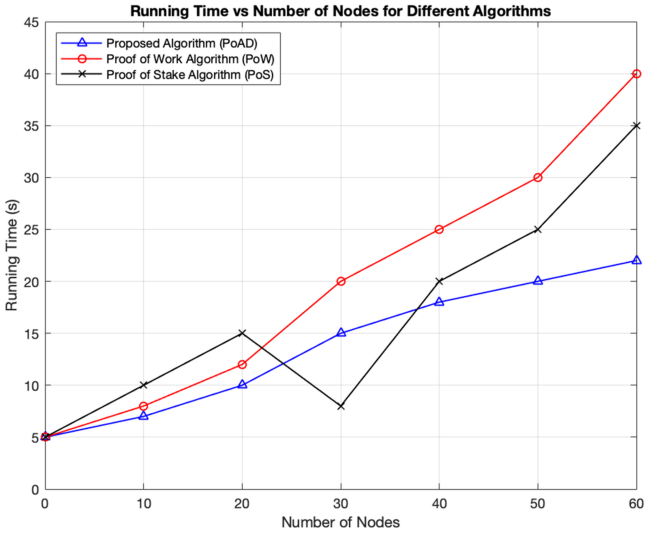

FIGURE 5 | Runtime evaluation with 60 nodes.

PoAD's hybrid selection mechanism, which leverages validator activity, delegated trust, and controlled randomness to reduce the computational and communication overhead. The results highlight the suitability of PoAD for scalable, low-latency smart contract execution in large decentralized networks.

In Figure 6, the time complexities of PoAD, PoW, and PoS are analyzed with increasing node counts. Initially, PoAD maintains the lowest complexity, but as the number of nodes increases beyond 7, its complexity surpasses PoS and PoW. This reflects the added computational overhead of PoAD from the multifactor validator selection process. However, this trade-off yields higher decentralization and performance, justifying its moderately complex growth.

In Figure 7, the time complexity of the PoAD, PoW, and PoS algorithms is evaluated across increasing nodes. PoAD exhibits the lowest and most stable time complexity curve, peaking around 15 nodes before a gentle decline, whereas PoW shows a sharp increase, peaking at node 11 before gradually decreasing. The PoS follows a linear growth pattern. The rationale for these observations stems from the internal computational demand of each algorithm. PoW, with its intense cryptographic problem-solving, leads to a rapidly increasing time complexity as node contention increases. However, as network saturation limits growth, the performance ceiling causes an eventual decline. The linear increase in PoS reflects its dependence on stake-based selection and sequential validation rounds, accumulating overhead linearly with validator count.

In contrast, PoAD's hybrid design—entropy-based fairness, bounded activity scoring, and decentralized delegation—maintains computational tractability. Its capped scoring window and probabilistic block proposer selection reduce redundant processing, leading to lower asymptotic growth. This behavior affirms the efficiency and scalability of PoAD for large validator sets.

In Figure 8, which extends the validator count to 50, the time-complexity behavior of the three consensus mechanisms—PoAD, PoW, and PoS—is evaluated with increasing network size. Consistent with the trends observed in Figure 11, PoAD exhibits superior computational efficiency across most validator sizes. While the PoW time complexity escalates rapidly, peaking at 20 operations around node 46, the PoS maintains a relatively steady linear climb, reflecting its predictable validation cost based on the stake ranking and message broadcasting. In contrast, PoAD maintained a flatter curve with localized dips, particularly between nodes 20 and 35. This pattern aligns with the algorithm's hybrid scoring approach, leveraging dynamic activity weighting, delegated stake, and verifiable randomness, which collectively moderate complexity through adaptive validator selection and probabilistic fairness. The decline in complexity between nodes 30 and 40 demonstrates PoAD's ability to avoid computation bottlenecks that typically emerge in consensus rounds of traditional algorithms. The slight increase toward node 50 suggests synchronization and coordination overhead at higher validator densities.

Nevertheless, the overall trend confirms that PoAD scales more efficiently with respect to time complexity than PoW and PoS do. This reaffirms its potential for deployment in

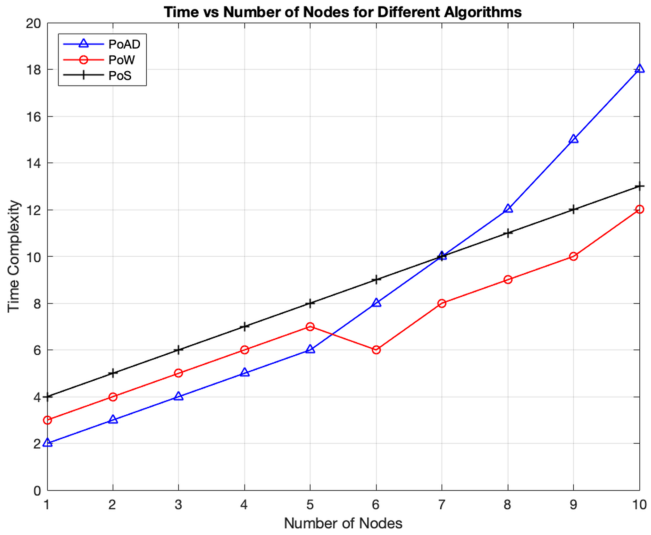

FIGURE 6 | Time complexity across PoW, PoS, and PoAD.

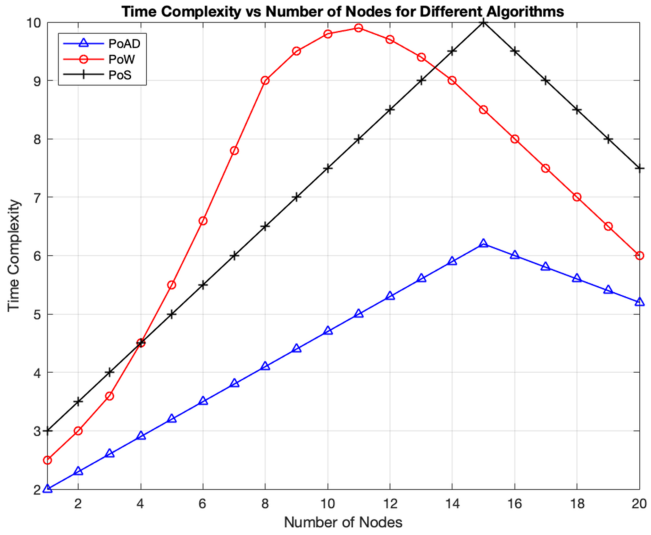

FIGURE 7 | Time complexity for 20 nodes.

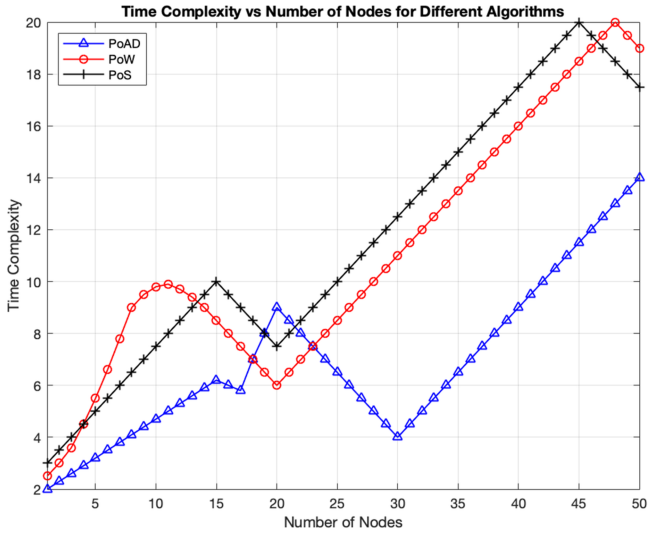

FIGURE 8 | Time complexity for 50 nodes.

large-scale smart contract execution environments, where computational cost and performance are critical. Although PoAD outperforms PoW and PoS in runtime and latency metrics, it introduces modest computational overhead owing to the multicomponent eligibility scoring process. Each validator computes activity averages, stake normalization, and entropy sampling independently—operations that collectively scale with a time complexity of $O(nlogn)$–as the final proposer must be determined through a global comparison of scores. Despite this, the overhead remains significantly lower than the cryptographic hashing burden in PoW or the chain-based selection latency in PoS, especially under parallelised conditions. This supports the scalability of the PoAD in mid-to-large validator networks.

Figure 9 compares the runtime performance of the three algorithms for 60 validators. PoW incurs the highest runtime of approximately 3.95 s, followed by PoS at 2.41 s. PoAD, however, registers only 1.21 s, nearly three times faster than PoW, and almost twice as fast as PoS. This substantiates the efficiency of PoAD under a high validator density, which is critical for blockchain network scaling to support large smart contract ecosystems. Figure 10 illustrates the runtime behavior as the validator count increases from 10 to 60. PoW exhibited a steep linear increase in execution time, suggesting scalability bottlenecks. PoS performs better, but still scales sublinearly. PoAD demonstrated the flattest trajectory, indicating excellent scalability and responsiveness. This confirms that PoAD's hybrid validator selection mechanism reduces the overhead by avoiding excessive stake computation and mining loops.

In Figure 11, the runtime data are visualized using a scatter plot. The distinct clustering of PoAD runtimes near the lower bound, regardless of validator size, reinforces the claim of consistency and robustness in transaction finalization. The dispersed PoW data points further reveal the volatility in performance, attributable to the variability in mining difficulty and network propagation delays. Figure 12 shows an area chart summarizing the cumulative runtime contribution of each algorithm across the validator range. The visibly smaller share of the yellow PoAD region suggests its lightweight footprint relative to the cumulative delays incurred by the PoS and PoW. The results reaffirm the viability of the PoAD consensus algorithm for latency-sensitive and resource-constrained environments, including decentralized IoT, real-time DeFi, and high-frequency microtransaction systems. PoAD demonstrates consistently superior runtime performance, making it highly suitable for smart contract platforms, where low latency and fast finality are critical. In addition, the algorithm exhibited improved scalability due to its entropy-injected and activity-weighted validator selection strategy, which enabled a graceful performance degradation as the number of nodes increased.

PoAD also maintains higher runtime stability across simulations, reflecting robust behavior in dynamic and asynchronous network conditions. These advantages are attributed to PoAD's composite eligibility scoring model, which judiciously integrates validator activity, delegated stakes, and entropy randomness. This approach enables fair and efficient leader selection without excessive computational or communication overhead.

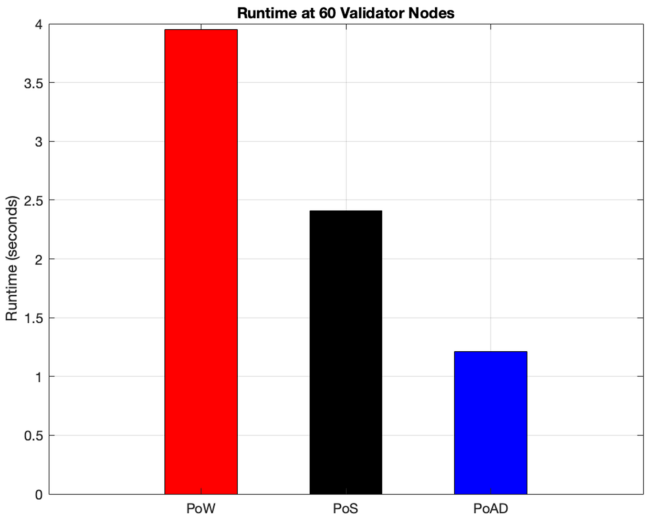

FIGURE 9 | Runtime performance of PoW, PoS, and PoAD at 60 validator nodes.

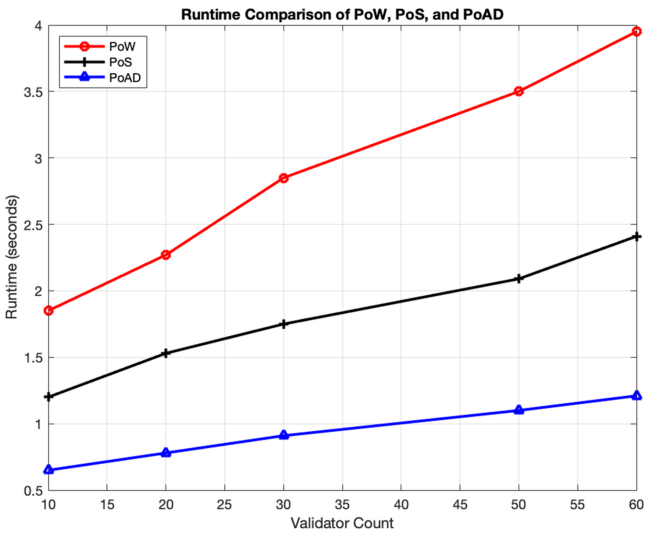

FIGURE 10 | Runtime comparison of PoW, PoS, and PoAD across validator counts.

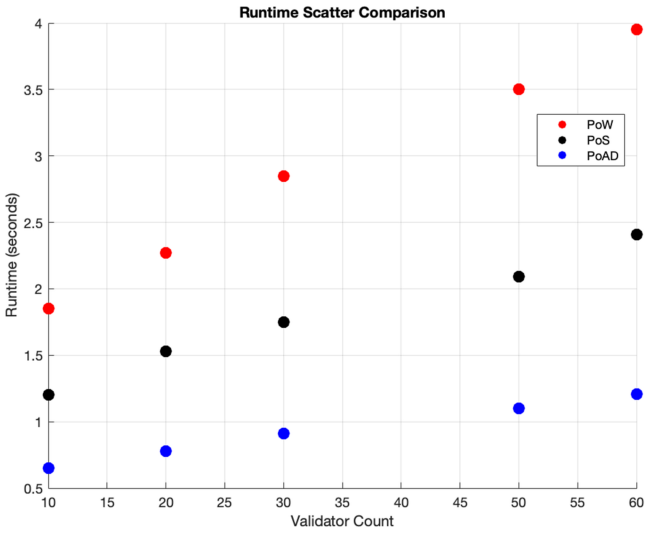

FIGURE 11 | Scatter distribution of consensus runtimes across validator nodes.

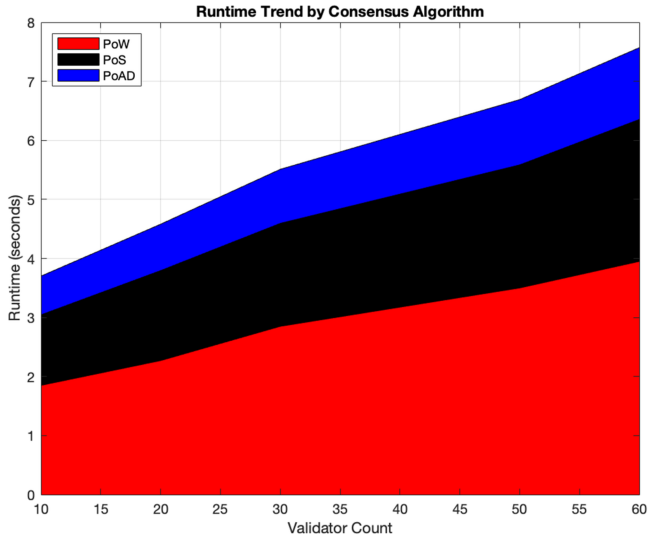

FIGURE 12 | Runtime trend by consensus algorithm across validator network size.

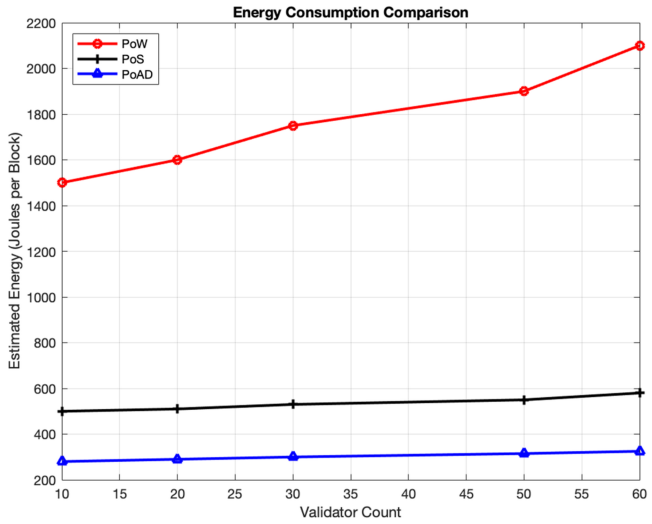

FIGURE 13 | Estimated energy consumption per block under PoW, PoS, and PoAD consensus algorithms.

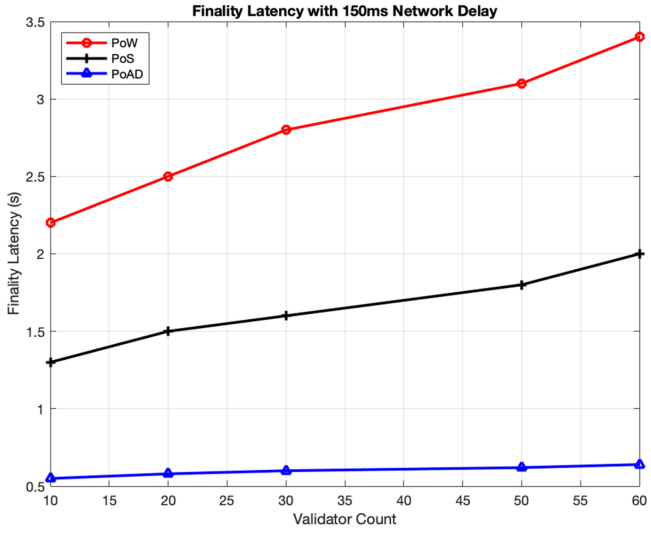

FIGURE 14 | Finality latency (in seconds) under network delay (150 ms) across different consensus protocols.

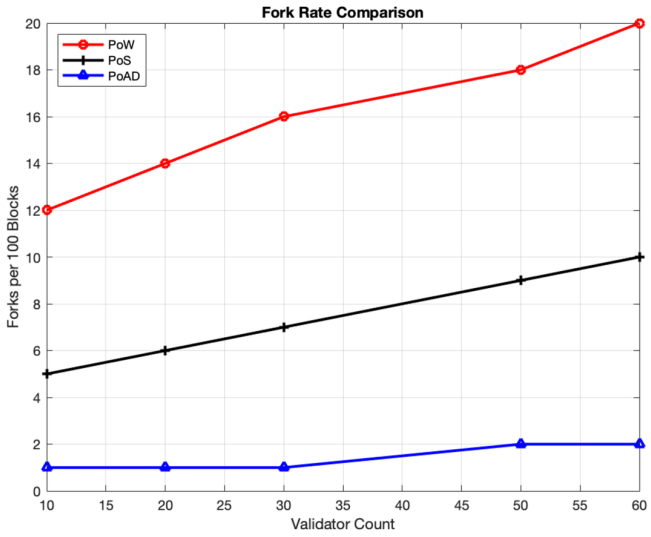

FIGURE 15 | Fork occurrences per 100 blocks as validator count increases.

To evaluate the robustness and comparative performance of the PoAD, further experiments were conducted across five critical metrics: energy consumption, finality latency, fork rate, proposer fairness, and transaction throughput. Each metric was assessed across a range of validator counts (10–60), providing a comprehensive benchmark against (PoW) and PoS algorithms. In Figure 13, the energy consumption increases significantly in the PoW as the validator count increases, reaching over 2100 joules at 60 nodes. This reflects the intensive computational requirements of this algorithm. The PoS showed modest energy use (approx. 580 J), while PoAD remains the most energy-efficient, stabilizing around 330 J. This confirms the suitability of PoAD for green blockchain deployments and energy-sensitive environments.

Figure 14 shows a comparison of the finality delays for each protocol. Due to block propagation and difficulty recalculations, PoW consistently shows the highest latency, exceeding 3.4 s at 60 nodes. PoS performs better but still exhibits degradation with scale. PoAD excels with a near-constant low latency (˜0.6 s), which is attributed to its PBFT-inspired voting and efficient leader selection, making it optimal for fast-settlement smart contracts.

Figure 15 reveals the susceptibility of PoW to frequent forks (up to 20 per 100 blocks at 60 nodes) due to delayed block propagation. The PoS performs moderately, with 10 forks at the scale. PoAD maintains minimal fork rates (1–2 per 100 blocks), validating its deterministic finalization and robust consensus convergence. While PoAD consistently maintained a lower fork rate than PoW, a minor deviation was observed at a validator count of 30, where the fork rate momentarily increased before stabilizing. This irregularity may be attributed to the transient entropy skew or suboptimal quorum responses during finalization. Although this did not affect finality guarantees, it suggests the need to tune the quorum threshold of PoAD under specific validator densities.

Figure 16 demonstrates the fairness in block proposal selection using the Gini index. PoW and PoS trend toward centralization, with Gini values approaching 0.62 and 0.47, respectively. PoAD exhibits improved equity, with values dropping to ˜0.18 at 60 validators. This stems from incorporating activity and delegation metrics, thus mitigating dominance by high-resource or high-stake actors. Figure 17 compares the network throughput. The PoW degrades sharply with scale, falling to 4 TPS at 60 validators. The PoS also experiences a decline due to the synchronization overhead. However, PoAD sustains a high rate, from 18 TPS at 10 nodes to 15.7 TPS at 60 nodes, affirming its design for throughput-optimized smart contract execution.

6 | Discussion

This study introduces the PoAD consensus mechanism as a hybrid alternative to the conventional PoW and PoS algorithms.

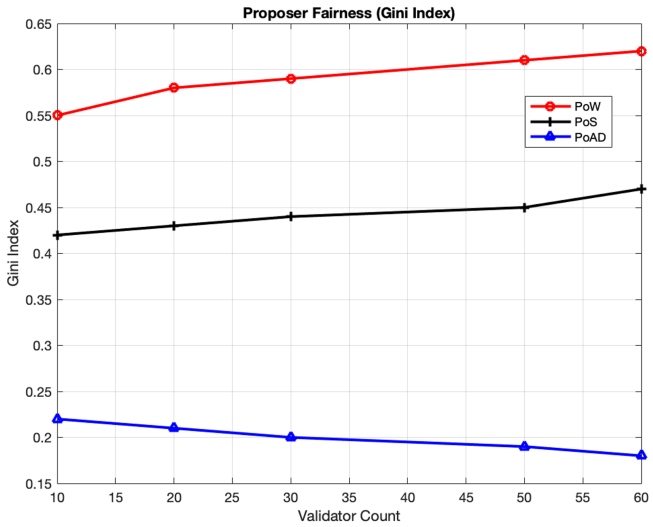

FIGURE 16 | Gini index for validator selection fairness; lower is better.

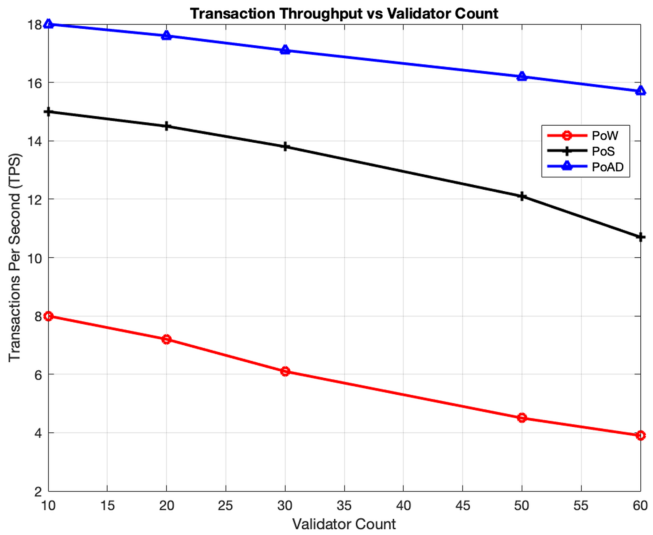

FIGURE 17 | Average transactions per second (TPS) across increasing validator sizes.

The empirical findings demonstrate that PoAD achieves superior performance across key metrics, including runtime, time complexity, energy consumption, finality latency, proposer fairness, and throughput. These observations are consistent with theoretical expectations and previous studies advocating for hybrid and lightweight consensus designs in decentralized applications [11–13].

Runtime analysis revealed that PoAD consistently outperformed PoW and PoS across varying network sizes, particularly under high validator densities (Figures 1–12). This efficiency stems from PoAD's composite eligibility scoring model, which incorporates validator activity, delegated stake, and EI_i. Unlike PoW, which incurs significant computational delay due to mining difficulty, and PoS, which suffers from latency induced by sequential stake evaluation and confirmation messaging, PoAD's decentralized yet probabilistic leader selection promotes low-latency finality while preserving fairness and scalability. These results echo the findings of Brahmam et al., who emphasized the importance of combining performance with equitable validator representation in high-transaction networks [19].

Furthermore, evaluating energy consumption (Figure 13) substantiates the suitability of PoAD for resource-constrained deployments, such as IoT and mobile blockchain environments. While PoW consumes over 2100 joules at 60 nodes—reflecting its well-documented inefficiency [4, 7]—PoAD maintains its energy footprint below 330 joules. This aligns with the sustainability goals and supports prior assertions that hybrid or DPoS-like mechanisms offer viable alternatives to PoW for environmentally sensitive applications [14, 15].

Table 1 compares PoW, PoS, and PoAD across key consensus performance and operational metrics.

The low finality latency achieved by PoAD (Figure 14), averaging 0.6 s even under a 150 ms network delay, is enabled by its PBFT-inspired consensus layer. By finalizing blocks upon receiving a supermajority vote ($\geq 2/3$ of validators), PoAD ensures deterministic finality, eliminating the need for probabilistic confirmation windows common in PoW and PoS. PBFT-based designs have been previously adopted in mission-critical systems because of their strong safety and liveness guarantees under partial synchrony [16]. PoAD leverages this structure without incurring full communication overhead by limiting the scope of the finalization rounds.

The fairness of the block proposer selection, assessed via the Gini index (Figure 16), further differentiates PoAD from its counterparts. While PoW and PoS exhibit increasing centralization, often a byproduct of hashpower and stake accumulation, PoAD maintains a near-uniform selection probability across validators, owing to its integration of activity metrics and verifiable randomness. This mitigates the "rich-get-richer" dynamics identified in PoW and PoS systems [15, 18], and supports broader validator participation without sacrificing security or throughput.

The observed fork rate (Figure 15) also highlights the resilience of PoAD's deterministic finalization. While PoW suffers from frequent forks due to propagation delays and competitive mining, PoAD's finality mechanism effectively curtails divergence,

TABLE 1 | Comparative summary of consensus algorithms across key metrics.

Metric	Proof of Work (PoW)	Proof of Stake (PoS)	Proof of Activity and Delegation (PoAD)
Initial runtime (1 node)	~5.0 s	~6.0 s	~4.0 s
Runtime growth (up to 60 nodes)	Moderate, linear	Steep, increases significantly	Lowest overall; scales efficiently
Time complexity (at 50 nodes)	~18.5	~19.5	~14.0
Scalability	Limited due to resource cost	Better but affected by network delay	High; activity-weighted and entropy-driven selection
Energy efficiency	Low (high computational cost)	High	Very high (minimal redundant computation)
Security model	Strong against Sybil attacks	Prone to stake centralisation	Combines activity tracking with a stake for robustness
Fairness	Reward skewed to compute power	Reward skewed to large stakers	Balanced using α, β, γ weights
Suitability for IoT/DeFi	Poor (resource-heavy)	Better for general dApps	Ideal for real-time, lightweight applications

maintaining less than two forks per 100 blocks. This behavior aligns with the security analysis of PBFT-based systems and the empirical observations of Haque et al., who demonstrated similar improvements in fork minimization using DPoS structures [13].

Moreover, the transaction throughput of PoAD (Figure 17) remains stable and high (15–18 TPS) even as the validator pool scales, outperforming both PoW and PoS, which decline under network stress. This performance mirrors the findings of [11, 12] and [12], who identified the DPoS and hybrid variants as more throughput-efficient under high-load conditions.

However, this study has several limitations that warrant consideration. First, PoAD's reliance on a secure VRF introduces a dependency on cryptographic assumptions; compromised randomness can skew validator selection fairness. Second, while delegation encourages community participation, it also risks long-term centralization unless accompanied by mechanisms such as delegation caps, decay, or rotation [12, 14]. Third, although carefully controlled, the experimental environment did not capture all real-world factors, such as node churn, adversarial behavior, and heterogeneous network latency.

Future research should explore the behavior of PoAD in adversarial environments, extend it to asynchronous BFT frameworks such as HotStuff, and integrate adaptive incentive models that penalize malicious validators while rewarding honest participation. Testing PoAD in real blockchain platforms or Layer-2 rollups would provide deeper insights into the deployment viability. Simultaneously, machine-learning-driven parameter tuning (e.g., adaptive α, β, and γ weights) can enhance performance and fairness.

The results confirmed that PoAD offers a compelling alternative to PoW and PoS, delivering enhanced scalability, fairness, and energy efficiency without compromising security or decentralization. As blockchain applications evolve toward more complex, real-time, and resource-sensitive deployments, hybrid protocols, such as PoAD, are poised to play a pivotal role in achieving sustainable and performant distributed consensus.

These findings have important implications for deploying consensus mechanisms in decentralized systems. While PoAD demonstrates superior runtime efficiency and fairness in controlled settings, real-world deployments may present challenges, such as validator churn, adversarial behavior, or heterogeneous hardware environments. In addition, scalability beyond 60 nodes and long-term network stability under variable workloads warrant further examination. Future work should explore the performance of PoAD in permissionless settings, examine its behavior under network partitions, and assess its resilience to stake centralization. Integrating PoADs with dynamic reputation systems or adaptive weight tuning may enhance their robustness in practical applications.

PoAD's design is especially well-suited for deployment in real-world decentralized systems, where validator activity and network conditions are highly dynamic. In DeFi environments, PoAD's reduced latency and fairness mechanisms can enhance transaction processing equity. Similarly, the lightweight runtime profile in IoT applications supports consensus among resource-constrained devices. These use cases underscore the applicability of PoAD in latency-critical, cost-sensitive, and fairness-prioritized decentralized ecosystems.

7 | Conclusion

This study presented PoAD, a novel hybrid consensus algorithm that combines validator activity, delegated stake, and entropy-based randomness to address the inherent limitations of PoW and PoS mechanisms. Through extensive simulations, PoAD has demonstrated superior runtime efficiency, scalability, proposer fairness, energy consumption, and finality latency. Its PBFT-inspired finalization and balanced eligibility scoring ensure low-latency consensus, equitable

validator participation, and robustness against centralisation and forks. These findings suggest PoAD is a promising consensus protocol for next-generation decentralized applications, particularly in energy-sensitive, real-time, and smart contract-intensive environments. Future work will explore adversarial resilience, real-world deployment, and adaptive parameter tuning to enhance the applicability and trustworthiness of PoADs. While the results demonstrate the performance advantages of PoAD, the simulations were conducted in controlled environments with fixed validator behavior and simplified network conditions. This may limit the generalizability of our findings to dynamic, real-world blockchain ecosystems. Future studies should test PoADs under heterogeneous validator assumptions and adversarial conditions. Continued innovation is essential in the consensus space, and we encourage researchers and system designers to explore, refine, and adapt hybrid models, such as PoAD, to meet the evolving demands of decentralized infrastructures.

Data Availability Statement

No data is available.

References

1. A. Ahmed, A. Majeed, A. S. R. Al-Ani, et al., "Blockchain and Smart Contracts: Enabling Trustworthy and Decentralized Digital Transactions," in *2024 30th Conference of Open Innovations Association (FRUCT)* (IEEE, 2024), 1–8, https://doi.org/10.23919/fruct64283.2024.10749910.

2. S. S. Kushwaha and S. Joshi, "An Overview of Blockchain-Based Smart Contract," in *Intelligent Computing and Applications: Proceedings of ICICA 2019*, ed. S. S. Dash, S. Das, and B. K. Panigrahi (Springer, 2021), 879–888, https://doi.org/10.1007/978-981-15-9647-6_70.

3. S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

4. F. Saleh, "Blockchain Without Waste: Proof-Of-Stake," *Information Systems & Economics eJournal* (2020): 1156–1190, https://doi.org/10.2139/ssrn.3183935.

5. L. M. Shamala, V. R. Balasaraswathi, and R. Gayathri, "Revolutionizing Industry and Business Processes With Smart Contracts in Blockchain," in *Examining the Evolution of Gaming and Its Impact on Social, Cultural, and Political Perspectives*, ed. K. D. Valentine and L. J. Jensen (IGI Global, 2024), 227–246, https://doi.org/10.4018/979-8-3693-1532-3.ch011.

6. T. N. Nguyen, "Smart Contract: Revolutionizing Transactions in the Digital Age," *HPU2 Journal of Science: Natural Sciences and Technology* 3, no. 1 (2024): 30–38, https://doi.org/10.56764/hpu2.jos.2024.3.1.30-38.

7. M. Wendl, Q. H. Mahmoud, and M. Singh, "The Environmental Impact of Cryptocurrencies Using Proof-Of-Work and Proof-Of-Stake Consensus Algorithms: A Systematic Review," *Journal of Environmental Management* 318 (2022): 115617, https://doi.org/10.1016/j.jenvman.2022.115617.

8. B. K. Banik, "Smart Contracts: An Emerging Business Model in Decentralized Finance," in *Decentralized Finance: From Core Concepts to DeFi Protocols for Financial Transactions*, ed. T. K. Birindelli and M. Di Maggio (Springer, 2023), 389–408, https://doi.org/10.1007/978-3-031-29857-8_20.

9. J. C. Sekhar, A. B. Prasad, S. Kanade, A. Kanade, C. Dhilipan, and B. K. Bala, "Blockchain-Enabled Smart Contracts: An Evolutionary Algorithm Approach for Automated Financial Agreement," in *Proceedings of the 2024 International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (IEEE, 2024), 1–7, https://doi.org/10.1109/icccnt61001.2024.10724807.

10. A. Ejupi, S. Angelis, and V. Sassone, "Performance and Scalability Testing for Blockchain Consensus Protocols: An Empirical Framework," 2024.

11. S. Amen, "Assessment of Consensus Algorithms for Blockchain Technology to Enhance Decentralized Applications," *Basrah Researches Sciences* 50, no. 2 (2024): 23, https://doi.org/10.56714/bjrs.50.2.23.

12. V. Bachani and A. Bhattacharjya, "Preferential Delegated Proof of Stake (PDPoS): Modified DPoS With Two Layers Towards Scalability and Higher TPS," *Symmetry* 15, no. 4 (2022): 1–13, https://doi.org/10.3390/sym15010004.

13. E. Haque, A. Shah, J. Iqbal, S. Ullah, R. Alroobaea, and S. Hussain, "A Scalable Blockchain-Based Framework for Efficient IoT Data Management Using Lightweight Consensus," *Scientific Reports* 14 (2024): 14, https://doi.org/10.1038/s41598-024-58578-7.

14. K. John, T. Rivera, and F. Saleh, "Proof-Of-Work Versus Proof-Of-Stake: A Comparative Economic Analysis," *Review of Financial Studies* 38 (2025): 1955–2004, https://doi.org/10.1093/rfs/hhaf013.

15. W. Zhao, "On Nxt Proof of Stake Algorithm: A Simulation Study," *IEEE Transactions on Dependable and Secure Computing* 20 (2023): 3546–3557, https://doi.org/10.1109/TDSC.2022.3193092.

16. I. Mazurok, Y. Leonchyk, O. Antonenko, and K. Volkov, "Smart Contract Sharding With Proof of Execution," *Applied Aspects of Information Technology* 4 (2021): 271–281, https://doi.org/10.15276/aait.03.2021.6.

17. J. Eggers, A. Hein, J. Weking, M. Böhm, and H. Krcmar, "Process Automation on the Blockchain: An Exploratory Case Study on Smart Contracts," in *Proceedings of the 54th Hawaii International Conference on System Sciences (HICSS)* (IEEE Computer Society, 2021), 5587–5596, https://doi.org/10.24251/HICSS.2021.681.

18. P. Nair and R. Dorai, "Evaluation of Performance and Security of Proof of Work and Proof of Stake Using Blockchain," in *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (IEEE Xplore, 2021), 279–283, https://doi.org/10.1109/ICICV50876.2021.9388487.

19. M. Brahmam, L. Prasanna, V. M, K. Ashok, K. Murthy, and S. Sindhura, "A Comprehensive Examination of Consensus Algorithms and Their Impact on Blockchain Scalability," in *2024 First International Conference on Innovations in Communications, Electrical and Computer Engineering (ICICEC)* (IEEE Xplore, 2024), 1–7, https://doi.org/10.1109/icicec62498.2024.10808240.

20. G. Ahmed, J. Zou, M. M. S. Fareed, and M. Zeeshan, "Sleep-Awake Energy Efficient Distributed Clustering Algorithm for Wireless Sensor Networks," *Computers and Electrical Engineering* 56 (2015): 385–398, https://doi.org/10.1016/j.compeleceng.2015.11.011.