

# Collaborative Proof-of-Work: A Secure Dynamic Approach to Fair and Efficient Blockchain Mining

Rizwanul Haque<sup>1</sup>, SM Tareq Aziz<sup>2</sup>, Tahrim Hossain<sup>3</sup>, Faisal Haque Bappy<sup>4</sup>,  
Muhammad Nur Yanhaona<sup>5</sup>, and Tariqul Islam<sup>6</sup>

<sup>1,2</sup> University of Dhaka, <sup>3,4,6</sup> Syracuse University, and <sup>5</sup> BRAC University

Email: mdrizwanul-2018925355@cs.du.ac.bd, smtareq-2018825356@cs.du.ac.bd,  
mhossa22@syr.edu, fbappy@syr.edu, nur.yanhaona@bracu.ac.bd, and mtislam@syr.edu

**Abstract**—Proof-of-Work (PoW) systems face critical challenges, including excessive energy consumption and the centralization of mining power among entities with expensive hardware. Static mining pools exacerbate these issues by reducing competition and undermining the decentralized nature of blockchain networks, leading to economic inequality and inefficiencies in resource allocation. Their reliance on centralized pool managers further introduces vulnerabilities by creating a system that fails to ensure secure and fair reward distribution. This paper introduces a novel Collaborative Proof-of-Work (CPoW) mining approach designed to enhance efficiency and fairness in the Ethereum network. We propose a dynamic mining pool formation protocol that enables miners to collaborate based on their computational capabilities, ensuring fair and secure reward distribution by incorporating mechanisms to accurately verify and allocate rewards. By addressing the centralization and energy inefficiencies of traditional mining, this research contributes to a more sustainable blockchain ecosystem.

**Index Terms**—collaborative mining, blockchain, ethereum, proof-of-work

## I. INTRODUCTION

Blockchain technology, has transformed industries by enabling decentralized and secure platforms. Central to blockchain's functionality are consensus algorithms, which allow independent nodes within the network to achieve agreement on transaction validity, block inclusion, and order without the need for a central authority. By reaching consensus, these algorithms ensure data integrity, resilience, and accuracy across the blockchain, safeguarding the network from errors or malicious attacks [1].

Proof of Work (PoW) is one of the earliest and most widely used consensus algorithms in blockchain renowned for its robust security [2]. It secures networks like Bitcoin [3] by requiring miners to solve cryptographic puzzles to validate transactions and add new blocks, ensuring that tampering or fraudulent activity is prohibitively costly. Miners compete to find a specific nonce that, when combined with the current block's header, generates a hash value below a set difficulty target. This process aligns miners' interests with the network's integrity through block rewards and transaction fees, incentivizing honest participation. Despite its strengths, PoW also presents significant challenges. Its reward structure disproportionately favors strong miners with greater computational resources, consolidating mining power and undermining decentralization [4]. This centralization enables exploitative

behaviors [5]. Additionally, weak miners often waste computational power as they are less likely to win rewards, leading to inefficiencies and substantial energy waste.

Concentrated mining power in PoW systems increases vulnerabilities, such as selfish mining and block withholding attacks, where dominant miners can exploit their position to disrupt operations and gain unfair rewards [6], [7]. Additionally, the substantial energy demands required to secure PoW networks raise concerns about scalability and environmental impact, emphasizing the importance of developing more energy-efficient approaches [8], [9]. To address challenges in efficiency, decentralization, security and fairness recent work explores innovative solutions such as partial contribution rewards [10], low-difficulty transactions [11], energy repurposing for productive tasks [12], and decentralized mining pools [13]. While these approaches offer promising improvements, they come with shortcomings, including misaligned incentives that weaken security, heightened vulnerability to DoS attacks from increased network overhead, centralization risks due to reliance on task managers, and scalability constraints arising from high gas fees and operational costs in blockchain networks. These limitations highlight the need for innovative frameworks that balance efficiency, fairness, and decentralization. To overcome these significant issues, in this work, we focused on the following three research questions.

**RQ1:** *How can we optimize computational resources through dynamic mining pool formation in Collaborative Proof-of-Work (CPoW) systems while ensuring decentralized collaboration among miners?*

**RQ2:** *What strategies can ensure equitable reward distribution in CPoW models for miners with lower capabilities, without compromising incentives for stronger miners?*

**RQ3:** *How can we enhance energy efficiency in mining operations within the CPoW framework, and what trade-offs exist between energy consumption and performance?*

To address these questions, this work introduces a novel Collaborative Proof-of-Work framework, which proposes a dynamic mining pool architecture. Unlike static pools, CPoW enables miners to collaborate flexibly, sharing both computational power and rewards equitably inside the network, thereby promoting decentralization. The main contributions of this work are:

- **Dynamic Mining Pool Formation.** We proposed a flexi-

ble protocol that allows miners to form or leave pools dynamically based on their computational capabilities, ensuring adaptability and resource optimization.

- **Fair And Secure Reward Distribution.** We designed a shared reward system that ensures equitable profit distribution among miners, while incorporating mechanisms to verify contributions and facilitating the secure transfer of rewards.
- **Theoretical Foundation and Security Analysis.** We establish a rigorous theoretical framework for the Collaborative Proof-of-Work (CPoW) model, integrating key concepts, assumptions, and mechanics with a comprehensive security evaluation.
- **Energy Efficiency.** Our approach can reduce the overall energy consumption of mining by distributing computational workloads more effectively through collaboration.

In the remainder of this paper, we review related work in Section II, focusing on existing collaborative mining approaches and their limitations. In Section III, we present the formal foundation, where we define the key concepts and models for our framework. Section IV explores the system workflow of our Collaborative Proof of Work (CPoW) model. In Section V, we conduct a security analysis, followed by a performance evaluation in Section VI. Finally, in Section VII, we conclude with insights on the framework's implications and outline future directions.

## II. RELATED WORK

Proof of Work (PoW) consensus mechanisms, widely used in blockchain networks, face critical issues regarding the fairness of block reward distribution, security against exploitative mining behaviors, and the centralization of mining power.

Hunag et al. [4] investigate the disproportionate acquisition of block mining rewards by large mining entities in PoW systems, resulting in a "rich get richer" dynamic. The authors explain that miners with greater computational power have a higher probability of successfully mining blocks, allowing them to capture more rewards consistently. This cycle concentrates rewards among large miners, undermining decentralization as smaller miners are discouraged from participating, further centralizing power.

Studies by Feng et al [6] and Cheng et al. [7] illustrate that concentrated mining power heightens vulnerabilities in PoW systems, increasing susceptibility to selfish mining and block withholding attacks. When a small number of miners control a significant share of the network's mining power, they can strategically withhold blocks to disrupt network operations and enhance their own rewards at the expense of smaller miners.

The substantial energy demands of PoW systems, which secure networks through intensive computation, are highlighted by Sedlmeir et al. [8] and further discussed by Li et al. [9]. These demands raise scalability and environmental concerns, emphasizing the need for more energy-efficient approaches.

StrongChain introduces "weak solutions" to reward partial contributions in PoW, reducing energy waste and leveling the playing field for smaller miners [10]. However, this approach

departs from PoW's emphasis on full solutions, potentially weakening its security guarantees and encouraging miners to prioritize partial rewards, leading to unintended incentives.

The FruitChains protocol, proposed by Pass and Shi, addresses issues like mining variance and selfish mining by introducing "fruits", transactions with lower mining difficulty that can be appended to blocks mined at higher difficulty [11]. This approach aims to reduce reliance on mining pools by offering more frequent rewards. However, FruitChains introduces new challenges, as low difficulty fruits can lead to high transmission overhead and may even open the network to denial-of-service attacks. Additionally, duplicate fruits are discarded, leading to inefficiencies and increased reward variance. These limitations highlight the complexities involved in modifying traditional Proof of Work (PoW) protocols.

Li et al. [12] introduce the Proof of Neural Architecture Search (PoNAS) consensus, which improves mining efficiency by redirecting computational power from repetitive hash calculations, traditionally used in Proof of Work (PoW), to neural architecture search (NAS) tasks. Instead of solely expending energy on cryptographic puzzles, PoNAS allocates this computational effort toward finding optimal neural network architectures. This shift allows mining energy to contribute both to blockchain security and to advancements in deep learning, addressing PoW's high energy consumption by giving it a dual purpose. However, centralization risks are introduced by relying on a mining pool manager to allocate tasks and manage rewards, creating a single point of control and compromising the blockchain's decentralized nature.

SmartPool is a decentralized mining pool implemented on Ethereum that eliminates the need for a central operator by using smart contracts to manage shares and distribute rewards [13]. Miners submit shares, which are verified probabilistically to reduce gas costs, and rewards are distributed based on these verified shares. While SmartPool enhances decentralization, it faces limitations from Ethereum gas fees and potential scalability issues.

Collectively, these studies highlight significant advancements but reveal a critical gap: existing solutions often prioritize specific aspects such as fairness, energy efficiency, or decentralization, without achieving a unified framework that balances these goals while maintaining robust security and scalability.

## III. FORMAL FOUNDATION

Proof of Work (PoW) relies on cryptographic hash functions to validate blocks in a blockchain. For a block to be accepted as valid, its hash must be lower than a specific threshold set by the network's difficulty level, meaning it must begin with a certain number of leading zeros. The key to achieving this lies in finding a suitable nonce, an arbitrary number that, when combined with the block header and hashed, produces a hash meeting these criteria. In PoW systems, miners must search through a large range of possible nonces,  $N_{\text{total}}$ , repeatedly generating and testing hashes until they find one that meets the requirement. The time  $T_{\text{solo}}$  required by a single miner to

exhaust the nonce range  $N_{\text{total}}$  depends on their computational power  $C_p$ . Since  $C_p^{\text{weak}} \ll C_p^{\text{strong}}$ , this implies  $T_{\text{solo}}^{\text{weak}} \gg T_{\text{solo}}^{\text{strong}}$ . In other words, miners with lower computational power require significantly more time to complete the search than those with higher computational power.

Given that  $C_p^{\text{weak}} \ll C_p^{\text{strong}}$ , the probability  $P_{\text{success}}^{\text{weak}}$  of a weak miner successfully solving the PoW puzzle remains extremely low compared to  $P_{\text{success}}^{\text{strong}}$ . As a result, weak miners will earn rewards  $R_{\text{weak}} \approx 0$  over time, making mining economically unsustainable for them. This causes mining power to become concentrated among strong miners, centralizing control within a few powerful entities. This situation contradicts blockchain's core principle of decentralization, which aims to distribute control and rewards across a wide range of participants. In a decentralized network, all miners should have a fair chance at earning rewards. Concentration of rewards among a few powerful miners undermines this openness and inclusivity.

This would lead weaker miners to collaborate and form groups in an effort to compete. Multiple weak miners can form a collaborative group  $G_{\text{collab}}$ . In this setup, miners within  $G_{\text{collab}}$  divide the nonce search range  $N_{\text{total}}$  into smaller sub-ranges.

Let,  $C_p^{m_i}$  denote the computational power of miner  $m_i$ . The time  $T_{\text{solo}}^{m_i}$  required for a single weak miner  $m_i$  to exhaust  $N_{\text{total}}$  without collaboration is given by:

$$T_{\text{solo}}^{m_i} = \frac{N_{\text{total}}}{C_p^{m_i}}$$

In a collaborative mining setup, each miner  $m_i \in G_{\text{collab}}$  is assigned a portion of the range  $N_i = \frac{N_{\text{total}}}{|G_{\text{collab}}|}$ . The time  $T_{\text{collab}}$  required for the group  $G_{\text{collab}}$  to collectively exhaust  $N_{\text{total}}$  becomes:

$$T_{\text{collab}} = \frac{N_i}{C_p^{m_i}} = \frac{N_{\text{total}}}{|G_{\text{collab}}| \cdot C_p^{m_i}}$$

For a group of  $n$  weak miners, the time savings can be represented as

$$T_{\text{collab}} \approx \frac{T_{\text{solo}}^{m_i}}{n}$$

In a static collaborative mining setup, groups are fixed, with set participants and roles, limiting flexibility. By contrast, dynamic collaborative mining allows miners to join or leave  $G_{\text{collab}}$  as needed, providing greater flexibility. We develop a framework that supports this dynamic collaborative mining, enabling miners to cooperate efficiently and on-demand. This approach increases the probability of successfully mining a block, maximizes rewards, and minimizes wasted computational effort. However, dynamic collaborative mining presents several key challenges, including establishing trust among participants by verifying individual contributions and ensuring fair and secure reward distribution within  $G_{\text{collab}}$ . These challenges are discussed in detail below.

#### A. Trust Dynamics in Contribution Verification

In a collaborative mining group  $G_{\text{collab}}$ , each miner  $m_i$  is expected to contribute a portion of computational power  $C_p^{m_i}$  toward the total computational power  $C_p^{G_{\text{collab}}} = \sum_{i=1}^{|G_{\text{collab}}|} C_p^{m_i}$ . This combined power determines the group's likelihood of successfully mining a block, where the probability of success is  $P_{\text{success}}^{G_{\text{collab}}} \propto \frac{C_p^{G_{\text{collab}}}}{C_p^{m_i}}$ . Each miner should ideally receive a reward  $R_{m_i} \propto \frac{C_p^{m_i}}{C_p^{G_{\text{collab}}}}$ . In a dynamically formed collaborative mining group  $G_{\text{collab}}$ , each miner faces uncertainty regarding the actual computational contributions of others. Without a trusted authority or prior verification, freeriders  $f_j \in G_{\text{collab}}$  could potentially claim rewards without contributing substantial  $C_p^{f_j}$ . This situation creates a scenario where

$$C_p^{f_j} \approx 0 \quad \text{but} \quad R_{f_j} > 0$$

This affects the entire group's effective computational power  $C_p^{G_{\text{collab}}}$ . The presence of freeriders results in

$$C_p^{G_{\text{collab}}} = \sum_{i=1}^{|G_{\text{collab}}|} C_p^{m_i} - \sum_{j=1}^{|F|} C_p^{f_j}$$

where  $F \subset G_{\text{collab}}$  denotes the subset of freeriders. Consequently, the group's mining success probability  $P_{\text{success}}^{G_{\text{collab}}}$  is reduced.

#### B. Challenges in Reward Distribution

Ensuring fair and secure reward distribution remains a critical challenge as collaborative mining evolves from static to dynamic frameworks. While static collaborative mining setups rely on centralized management, dynamic systems must address similar issues but in a decentralized manner. In a static setup, when a block is successfully mined by the group, rewards are typically sent to a single, central etherbase address  $E$ . This etherbase address  $E$  is controlled by the pool operator, who holds the private key  $k_E$  necessary to access and distribute the rewards. However, the pool operator controlling  $k_E$  introduces key issues:

- **Reliance on Operator's Integrity:** Participants must trust the operator to distribute  $R_{m_i}$  fairly. Without oversight, the operator could reduce rewards such that  $R'_{m_i} < R_{m_i}$ , giving miners less than they are due.
- **Single Point of Failure:** If  $k_E$  is compromised, all rewards at  $E$  are at risk. Any loss or misuse of  $k_E$  could prevent miners from receiving  $R_{m_i}$ , effectively reducing rewards to zero:  $R_{m_i} \approx 0$  if  $k_E$  is lost or misused.

In the context of dynamic collaborative mining, the challenge is to address these issues without relying on a central operator to manage  $k_E$ . Instead, a decentralized approach is required to ensure fair reward distribution and safeguard rewards, allowing miners to collaborate effectively.

### IV. SYSTEM WORKFLOW

The Collaborative Proof of Work (CPoW) framework enables secure, fair, and efficient mining within a decentralized, collaborative environment by addressing key challenges in

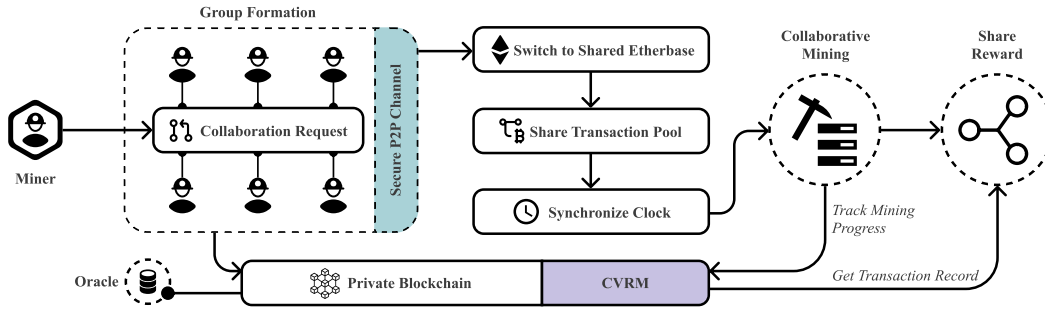


Fig. 1: Collaborative Mining process in CPoW

flexibility, fairness and trust. In CPoW, three core processes, dynamic group formation, contribution management, and decentralized reward distribution work together to overcome the unique obstacles of dynamic collaborative mining.

Dynamic group formation solves the problem of flexibility by allowing miners to form well-balanced collaborative groups based on computational power. Contribution management ensures fairness by accurately tracking each miner's work, verifying their computational effort to ensure that only those who contribute benefit from the group's efforts, thereby preventing any undue advantage for freeriders. Finally, decentralized reward distribution removes the need for a central authority to manage and distribute rewards by enabling fair, direct, and secure allocation to each participant, eliminating the risk of withheld rewards and single point of failure.

This section presents a comprehensive overview of how these processes function together in the CPoW framework to create a dynamic collaborative mining environment as depicted in Figure 1.

#### A. Group Formation and P2P Setup

The collaborative mining process in CPoW begins with group formation as shown in Fig.1. Initially, each node calculates its local hashrate using a dedicated module (Alg.1 lines 1-2). The hashrate represents each node's computational power, specifically its ability to solve cryptographic puzzles by finding a suitable nonce a number that, when combined with the block header and hashed, produces a result below the network's difficulty target. A node seeking collaboration broadcasts a request message to nearby nodes (Alg.1 lines 3-4). Upon receiving this request, neighboring nodes evaluate the sender's hashrate and respond with an acceptance message if they find the hashrate compatible (Alg.1 lines 5-8). Nodes prioritize accepting the first collaboration request with a similar hashrate, ensuring a balanced and efficient mining group.

Once a group is formed, members establish peer-to-peer (P2P) connections and add the collaborating peer list to their trusted peer set, enhancing group connectivity (Alg.1 lines 9-10). To facilitate cooperation, group members switch their local etherbase, the account designated to receive mining rewards to a shared etherbase (Alg.1 line 11) as illustrated in Fig. 1. The etherbase is the address where mining payouts

are sent. The switch to a shared etherbase ensures rewards can be pooled and fairly distributed among all collaborators.

As depicted in Fig.1 a shared transaction pool is established for collaborative mining, while each node's local pool remains active to handle new transactions independently (Alg.1 line 12). Transactions from the local pool are ordered within the shared pool, which uses the Ricart-Agrawala mutual exclusion algorithm [14] to manage the insertion of transactions in a coordinated manner. This mechanism prevents conflicts by ensuring mutual exclusion during transaction handling. To address potential clock synchronization issues, collaborators share their timestamps prior to mining, as outlined in Fig. 1, and employ Berkeley's algorithm to compute a synchronized timestamp that is used in the block being mined (Alg.1 line 13).

The group then proceeds to divide the nonce range between the collaborators. The total range is divided equally among collaborators, with a slight overlap between assigned ranges for contribution verification purposes. Each miner searches within their designated range to find the desired hash using the Dagger Hashimoto [15] algorithm (Alg.1 lines 14 - 15).

#### Algorithm 1 Group Formation and Collaborative Mining

```

1: for each  $node \in net$  do
2:    $node.calcHash()$ 
3:   if  $node.reqCollab()$  then
4:      $node.sendReq(neighbors)$ 
5:     for each  $nbr \in neighbors$  do
6:       if  $nbr.matchHash(node)$  then
7:          $nbr.acceptReq()$ 
8:          $group.add(nbr)$ 
9:   if  $group.ready()$  then
10:     $peers \leftarrow group.connectPeers()$ 
11:     $group.setSharedEthBase()$ 
12:     $group.sharePool(peers)$ 
13:     $group.syncClocks(peers)$ 
14:     $range \leftarrow group.splitRange(peers)$ 
15:     $group.startMining(range)$  so

```

#### B. Contribution Management with CVRM

To securely manage mining activities, we employ a **private blockchain** system featuring the Contribution Verifier and

---

**Algorithm 2** Contribution Management with CVRM

---

```
1: group.startDKG()
2: for each  $m \in \text{group}$  do
3:    $\text{share} \leftarrow m.\text{generateShare}()$ 
4:    $\text{CVRM.storeShare}(m, \text{share})$ 
5:  $\text{group.pubKey} \leftarrow \text{DKG.getPublicKey}()$ 
6:  $\text{group.etherbase} \leftarrow \text{DKG.getEtherbase}()$ 
```

---

Reward Manager (CVRM) as shown in Fig. 1. During group formation, collaborators initiate a Distributed Key Generation (DKG) protocol [16] to create individual private key shares. Each member registers their private key share with the CVRM (Alg.2 lines 1-4). Threshold cryptographic algorithms [17] enable secure group operations that require only a subset of key shares. These key shares collectively generate a shared public key (Alg.2 line 5). From this shared public key, a shared etherbase address is derived, serving as the collective account for mining rewards (Alg.2 line 6). This approach allows secure and decentralized management of mining rewards, ensuring that control is shared among collaborators. It enhances trust, transparency, and reliability, allowing rewards to be distributed fairly without relying on any single member.

### C. Reward Tracking and Distribution

We utilize an oracle to monitor etherbase addresses continuously, collecting data on balances and activities (Alg.3 line 1) as illustrated in Fig. 1. When it is time to distribute rewards, each collaborator is required to submit proof of their contribution to the CVRM (Alg.3 lines 2-3). This proof consists of a mapping of nonce to hash values they calculated during mining. This mapping allows the CVRM to verify each collaborator's work by cross-checking the overlapping portions of nonce ranges between collaborators. If the hashes from the overlapping sections match between collaborators, the proof is accepted, and the collaborators are allowed to withdraw their rewards. However, if discrepancies are detected in the submitted proofs, the CVRM recalculates the hashes for the overlapping nonces to identify inconsistencies. This recalculation helps the CVRM determine the honest collaborators by verifying whose proof aligns with the expected hash outputs (Alg.3 line 4).

---

**Algorithm 3** Reward Tracking and Distribution

---

```
1: oracle.monitorEtherbase(group.etherbase)
2: for each  $c \in \text{group}$  do
3:    $\text{proof} \leftarrow c.\text{generateProof}()$ 
4:   if  $\text{CVRM.verify}(\text{proof})$  then
5:      $\text{rewardAmount} \leftarrow \text{CVRM.reward}(c)$ 
6:      $\text{CVRM.withdraw}(c, \text{rewardAmount})$ 
```

---

Once the proof of contribution has been verified, each collaborator submits a transaction specifying the address to which they wish to transfer their rewards. The CVRM then reviews the transaction details, signs the transaction with

its authority key, and returns the signed transaction to the collaborator (Alg.3 lines 5-6). This signed transaction allows the collaborator to securely transfer their reward to the specified address, ensuring that the distribution process is both authenticated and controlled by the CVRM, maintaining the integrity and fairness of the reward allocation.

## V. SECURITY ANALYSIS

In the collaborative mining framework of CPoW, secure and fair reward distribution is critical. The Contribution Verifier and Reward Manager (CVRM) plays a central role by verifying contributions and ensuring that only honest collaborators receive rewards in a secure manner. This section demonstrates the framework's resilience against dishonest behavior and its effectiveness in preserving reward integrity.

### A. Analysis of Contribution Verification Process

The verification process uses nonce-to-hash mappings and overlapping nonce ranges between collaborators to maintain reward integrity and promote honesty. By cross-checking these overlaps, the Contribution Verifier and Reward Manager (CVRM) can efficiently verify contributions without recalculating costly hash functions. Since collaborators don't know with whom they share overlaps, nor the specific overlapping nonce range collusion is virtually impossible, ensuring secure and fair reward distribution.

The effectiveness of this verification mechanism is formalized in the following statement, which demonstrates how the contribution verification mechanism helps preserve both reward integrity and collaborative honesty. An analysis follows to confirm these properties.

**Claim:** *Given a set of collaborators  $C = \{c_1, c_2, \dots, c_n\}$ , if each collaborator  $c_i$  submits a proof of contribution  $p_i$  verified by the CVRM, then only honest participants  $C_{\text{honest}} = \{c_i \in C \mid p_i \text{ is valid}\}$  are allowed to access rewards.*

**Analysis:** Each collaborator  $c_i$  has a nonce range  $N_i$ , where  $N_i = \{n_{i,1}, n_{i,2}, \dots, n_{i,n}\}$ . The proof  $p_i$  submitted by  $c_i$  includes a mapping of each  $n_{i,j} \in N_i$  to its corresponding hash  $H_{i,j} = f(n_{i,j})$ , where  $f$  is a cryptographic hash function. This hash function is applied to each nonce in an attempt to solve the cryptographic puzzle essential for mining a successful block.

To verify the accuracy of each  $p_i$ , the CVRM checks overlapping nonce ranges  $N_{i,j} = N_i \cap N_j$  between two collaborators  $c_i$  and  $c_j$ .

For each pair of collaborators  $c_i$  and  $c_j$  with an overlapping nonce range  $N_{i,j} = N_i \cap N_j$ , the CVRM verifies that

$$H_i(x) = H_j(x) \quad \forall x \in N_{i,j}$$

Here,  $H_i(x) = f(x)$  and  $H_j(x) = f(x)$  are the hash values submitted by  $c_i$  and  $c_j$  for each  $x \in N_{i,j}$ . If this condition holds for all  $x \in N_{i,j}$ , the CVRM validates the contributions of both collaborators.

If a collaborator  $c_i$  submits a falsified proof  $p_i$ , discrepancies in the hashes for overlapping nonces  $N_{i,j}$  will reveal inconsistencies. In cases of discrepancy, the CVRM recalculates the

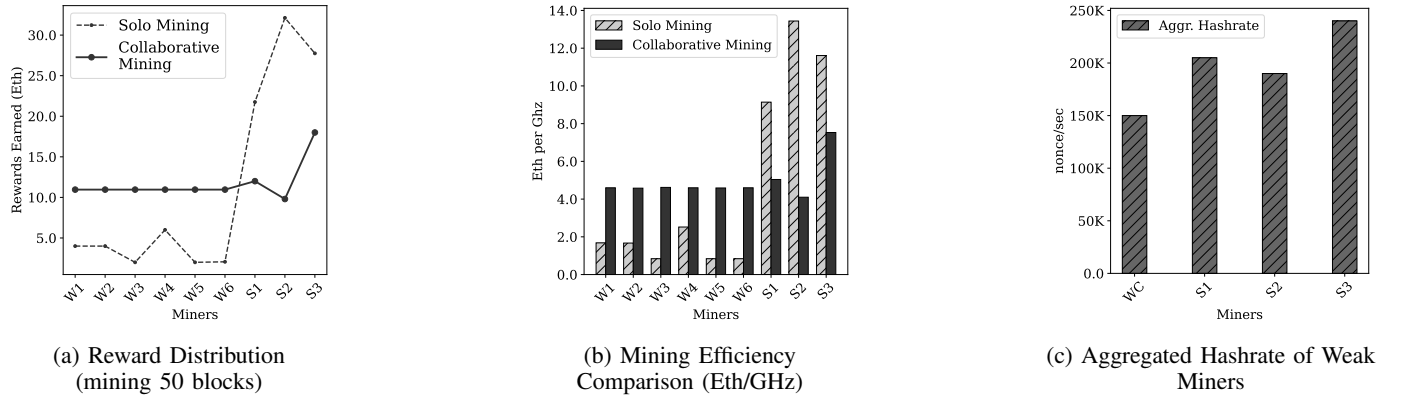


Fig. 2: Performance Comparison between Solo and Collaborative Mining

expected hash values for  $N_{i,j}$  to determine the correct hashes  $H_{\text{true}}$ , identifying the dishonest collaborator. Therefore, any collaborator with an invalid proof  $p_i$  is excluded from  $C_{\text{honest}}$  and is denied access to rewards.

### B. Analysis of Secure Reward Distribution Framework

Building on the contribution verification process, secure reward distribution is crucial to maintaining the integrity of the Collaborative Proof of Work (CPoW) framework. In the collaborative mining framework, secure reward distribution is achieved through controlled access to the private key  $K$ . The CVRM ensures that  $K$  is only reconstructed for verified participants, preventing unauthorized access. This process is formalized as follows:

**Claim:** Let  $C_{\text{eligible}} = \{c_i \in C \mid p_i \text{ is valid}\}$  denote verified collaborators. The CVRM reconstructs  $K$  only when  $\forall c_i \in C_{\text{eligible}}, p_i$  is valid. No collaborator or CVRM component holds  $K$  independently.

**Analysis:**  $K$  exists in distributed shares  $K_1, K_2, \dots, K_m$  across the collaborators  $c_i$ .  $K$  is formulated as

$$K = \text{Combine}(K_1, K_2, \dots, K_m)$$

by the CVRM only for signing after verification. If  $C_{\text{eligible}} \neq \emptyset$ , then  $K$  is temporarily reconstructed and used to sign reward transactions  $T_i$  for each  $c_i \in C_{\text{eligible}}$ . Unauthorized reward access is prevented since  $K$  is inaccessible outside verified signing, ensuring rewards  $R_i$  are distributed only to  $C_{\text{eligible}}$ . Together, these mechanisms (verification and controlled key access) uphold the integrity and fairness of the reward distribution process.

## VI. PERFORMANCE EVALUATION

To assess the performance of our proposed collaborative mining process, we conducted a series of experiments in a controlled environment, implementing our approach on the Ethereum (Geth version 1.11.6 [18]) network. The configuration included Ubuntu 22.04 (LTS) on a 64-bit architecture with an AMD Ryzen 5 3550H CPU, with a base speed of 2.10 GHz, 4 cores, and 8 threads. A total of nine nodes were set up, with six designated as weak miners and three as strong

miners. To simulate the weak miners, we introduced a 5 ms delay before verifying each nonce value.

### A. Reward Distribution Analysis:

In solo mining, weak miners earned an average of 3.34 ETH, with considerable variability (standard deviation of 1.62), indicating inconsistent earnings (Figure 2a). However, in collaborative mining, their average reward increased to 10.96 ETH, with much lower variability (standard deviation of 0.01). This shift suggests that collaborative mining offers a more equitable distribution of rewards.

Strong miners also showed a change, earning an average of 11.40 ETH in solo mode but only 5.04 ETH in collaborative mining. This indicates that while strong miners earned less collaboratively, the overall fairness improved for all miners.

### B. Efficiency Comparison:

We also calculated Ethers per GHz to assess mining efficiency (Figure 2b). Weak miners achieved an average of 1.4 Ethers per GHz in solo mining, while in collaborative mining, their efficiency rose significantly to 4.6 Ethers per GHz, reflecting consistent performance among participants. Strong miners' efficiency decreased from 13.94 Ethers per GHz in solo mining to 6.0 Ethers per GHz in the collaborative setting, highlighting a shift in how resources were allocated.

### C. Hashrate Analysis:

In our configuration, strong miners have about eight times the hashrate of weak miners. Strong miners achieve around 200,000 nonces per second, while weak miners have less than 30,000 nonces per second. Despite only three strong miners, they contribute 81.3% of the total computation power, while weak miners contribute just 19.7%. This centralization limits the chances for weak miners to successfully mine a block, presenting a challenge to their participation.

Combining the hashrates of weak miners brings it closer to that of an individual strong miner (Figure 2c). This significantly enhances the chances of weak miners successfully mining a block. Collaboration of weak miners proves to be an effective strategy, enhancing their overall contribution to the mining process.



TABLE I: Feature Comparison With Existing Approaches

|                     | Dependency on Pool Manager | Dependency on Strong Miners | Reward Security                         | Resource Requirements               | Scalability | Subspace Overlapping | Prob. of Wasted Effort | Cost          |
|---------------------|----------------------------|-----------------------------|-----------------------------------------|-------------------------------------|-------------|----------------------|------------------------|---------------|
| [19]                | Yes                        | No                          | No (Distributed by Pool Miners)         | High                                | Low         | No                   | No                     | High          |
| [20]                | Yes                        | Yes                         | No (Distributed by Pool Miners)         | No                                  | Low         | No                   | No                     | No Extra Cost |
| SmartPool [13]      | No                         | No                          | Yes                                     | Heavy dependency on smart contracts | Scalable    | Yes                  | Yes                    | High          |
| P2Pool [21]         | No                         | Yes                         | No (Pool's Reward address set manually) | ShareChain                          | Low         | Yes                  | Yes                    | No Extra Cost |
| <b>Our Approach</b> | No                         | No                          | Yes                                     | No                                  | Scalable    | No                   | No                     | No Extra Cost |

#### D. Feature Comparison with Existing Approaches

Table I presents a comprehensive comparison of our approach against existing solutions, evaluating eight critical features that impact the overall effectiveness and practicality of mining pool implementations. Our approach demonstrates several significant advantages over existing solutions. Unlike traditional approaches [19] and [20], our system eliminates dependency on pool managers, reducing centralization risks.

In terms of security, our approach matches SmartPool [13] in providing robust reward security, while avoiding the vulnerabilities of solutions [19], [20], and P2Pool, which rely on potentially risky distribution mechanisms through pool miners or manual reward address configurations. Unlike SmartPool's reliance on smart contracts and P2Pool's ShareChain requirements, our design minimizes resource consumption without sacrificing security. It matches SmartPool's scalability while overcoming the limitations and subspace overlapping issues that affect [19], [20], and P2Pool, ensuring more efficient use of computational resources. Cost-wise, it maintains the "No Extra Cost" advantage of [20] and P2Pool while avoiding the high operational expenses of [19] and SmartPool.

#### VII. CONCLUSION

In this paper, we introduced a dynamic collaborative mining approach to enhance fairness and efficiency within the Ethereum network. Our collaborative PoW allows for dynamic mining pool formation and optimizes energy consumption by sharing workloads, addressing the environmental concerns of traditional PoW systems. This framework promotes greater decentralization and inclusivity by enabling miners with lower computational power to participate effectively, reducing the dominance of high-powered entities in blockchain mining. Its fair reward distribution ensures equitable compensation for weaker miners, fostering economic fairness and wider participation in the blockchain ecosystem. Overall, this research has broad implications for improving energy efficiency, equity, and inclusivity in decentralized systems.

#### REFERENCES

- [1] I. Bashir, "Blockchain consensus an introduction to classical, blockchain, and quantum consensus protocols," *Evolution*, 2019.
- [2] J. Xu, C. Wang, and X. Jia, "A survey of blockchain consensus protocols," *ACM Comput. Surv.*, vol. 55, no. 13s, Jul. 2023. [Online]. Available: <https://doi.org/10.1145/3579845>
- [3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," May 2009. [Online]. Available: <http://www.bitcoin.org/bitcoin.pdf>
- [4] Y. Huang, J. Tang, Q. Cong, A. Lim, and J. Xu, "Do the rich get richer? fairness analysis for blockchain incentives," in *Proceedings of the 2021 international conference on management of data*, 2021, pp. 790–803.
- [5] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 3–16. [Online]. Available: <https://doi.org/10.1145/2976749.2978341>
- [6] C. Feng and J. Niu, "Selfish mining in ethereum," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 1306–1316.
- [7] Y. Cheng, Z. Xu, and S. Yao, "The evolutionary equilibrium of block withholding attack," *Journal of Systems Science and Information*, vol. 9, no. 3, pp. 266–279, 2021.
- [8] J. Sedlmeir, H. U. Buhl, G. Fridgen, and R. Keller, "The energy consumption of blockchain technology: Beyond myth," *Business & Information Systems Engineering*, vol. 62, no. 6, pp. 599–608, 2020.
- [9] J. Li, N. Li, J. Peng, H. Cui, and Z. Wu, "Energy consumption of cryptocurrency mining: A study of electricity consumption in mining cryptocurrencies," *Energy*, vol. 168, pp. 160–168, 2019.
- [10] P. Szalachowski, D. Reijnders, I. Homoliak, and S. Sun, "{StrongChain}: Transparent and collaborative {Proof-of-Work} consensus," in *28th USENIX Security Symposium (USENIX Security 19)*, 2019, pp. 819–836.
- [11] R. Pass and E. Shi, "Fruitchains: A fair blockchain," in *Proceedings of the ACM Symposium on Principles of Distributed Computing*, ser. PODC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 315–324. [Online]. Available: <https://doi.org/10.1145/3087801.3087809>
- [12] B. Li, Q. Lu, W. Jiang, T. Jung, and Y. Shi, "A collaboration strategy in the mining pool for proof-of-neural-architecture consensus," *Blockchain: Research and Applications*, vol. 3, no. 4, p. 100089, 2022.
- [13] L. Luu, Y. Velner, J. Teutsch, and et al., "Smartpool: Practical decentralized pooled mining," in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 1409–1426.
- [14] G. Ricart and A. K. Agrawala, "An optimal algorithm for mutual exclusion in computer networks," *Communications of the ACM*, vol. 24, no. 1, pp. 9–17, 1981.
- [15] Ethereum, "Dagger hashimoto." [Online]. Available: <https://github.com/ethereum/wiki/wiki/Dagger-Hashimoto>
- [16] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure applications of pedersen's distributed key generation protocol," in *Topics in Cryptology—CT-RSA 2003*. Springer, 2003, pp. 373–390.
- [17] C. Stathakopoulou and C. Cachin, "Threshold signatures for blockchain systems," *Swiss Federal Institute of Technology*, vol. 30, p. 1, 2017.
- [18] Ethereum, "Release azimir (v1.11.6) · ethereum/go-ethereum." [Online]. Available: <https://github.com/ethereum/go-ethereum/releases/tag/v1.11.6>
- [19] M. J. Mihaljević, L. Wang, S. Xu, and M. Todorović, "An approach for blockchain pool mining employing the consensus protocol robust against block withholding and selfish mining attacks," *Symmetry*, vol. 14, no. 8, p. 1711, 2022.
- [20] B. Li, Q. Lu, W. Jiang, T. Jung, and Y. Shi, "A collaboration strategy in the mining pool for proof-of-neural-architecture consensus," *Blockchain: Research and Applications*, vol. 3, no. 4, p. 100089, 2022.
- [21] P2Pool. P2pool, decentralized monero mining pool. Accessed: 2024-11-15. [Online]. Available: <https://p2pool.io/>