Computer Networks 214 (2022) 109118

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Green-PoW: An energy-efficient blockchain Proof-of-Work consensus algorithm

Noureddine Lasla [a,*], Lina Al-Sahan [a], Mohamed Abdallah [a], Mohamed Younis [b]

[a] Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Qatar
[b] University of Maryland, Baltimore County, Baltimore, MD, USA

ARTICLE INFO

Keywords:
Blockchain
Consensus algorithm
Proof-of-Work
Energy-efficiency

ABSTRACT

This paper opts to mitigate the energy-inefficiency of the Blockchain Proof-of-Work (PoW) consensus algorithm by rationally repurposing the power spent during the mining process. The original PoW mining scheme is designed to consider one block at a time and assign a reward to the first place winner of a computation race. To reduce the mining-related energy consumption, we propose to compensate the computation effort of the runner(s)-up of a mining round, by granting them exclusivity of solving the upcoming block in the next round. This will considerably reduce the number of competing nodes in the next round and consequently, the consumed energy. Our proposed scheme divides time into epochs, where each comprises two mining rounds; in the first one, all network nodes can participate in the mining process, whereas in the second round only runners-up can take part. Thus, the overall mining energy consumption can be reduced to nearly 50%. To the best of our knowledge, our proposed scheme is the first to considerably decrease the energy consumption of the original PoW algorithm. Our analysis demonstrates the effectiveness of our scheme in reducing energy consumption, the probability of fork occurrences, the level of mining centralization presented in the original PoW algorithm, and the effect of transaction censorship attack.

1. Introduction

Nakamoto's Blockchain protocol [1], also known as proof of work (PoW), is the first to achieve consensus in a permission-less setting, where anyone can join or leave during the protocol execution. Its main security design goal is to prevent *Sybil attacks* by relying on a computational cryptographic puzzle-solving process. The protocol has proven its robustness since its first application (Bitcoin) in 2009. However, besides being the most trusted and secure public consensus algorithm, PoW is considered as a computation-intensive voting-based consensus process. For instance, PoW-powered Bitcoin mining consumes massive power that could suffice for a small country, like Denmark [2]. Moreover, as estimated by Digiconomist [3], Bitcoin usage emits about 33.5 MtCO2e annually, as of May 2018. The main reason behind the PoW excessive energy is to make attacks against the blockchain network very expensive.

In a nutshell, PoW is a leader election protocol that designates among network participants (miners) one leader that will append the next block to the chain. To attract more participants to join and maintain the network, and at the same time demotivate them from cheating, an honest miner can be elected to receive a very attractive reward if it can solve a computationally arduous puzzle. The idea behind making a difficult puzzle in PoW consensus algorithm is to bound the economic capacity of an adversary to successfully undermine the network, for instance, to prevent double spending attack and rewriting the block-history [4,5]. Bounding the capacity of users based on their computation (energy) is only a *sufficient* condition to prevent security attacks and not *necessary*. In the literature, there have been several attempts to rationalize the consumed energy in PoW by either bounding the miner economic capacity using alternative more energy-efficient mechanisms [6–8], or by recycling the wasted energy spent in solving the puzzle to also serve for other useful tasks [9]. However, such rationalization still cannot meet the same security level as the original PoW, i.e., they introduce new vulnerabilities compared to the original Nakamoto's consensus. Proof-of-Stake (PoS) [6,10], for instance, is a greener form of distributed consensus where validators (akin to miners) do not have to use their computation power but only proof the ownership of an amount of stake (bond) in order to vote on new blocks. Such an approach suffers from a Nothing-at-Stake problem, where validators can stake for different blocks supporting multiple

* Corresponding author.
E-mail addresses: lasla.noureddine@gmail.com (N. Lasla), lialsahan@hbku.edu.qa (L. Al-Sahan), moabdallah@hbku.edu.qa (M. Abdallah), younis@umbc.edu (M. Younis).

https://doi.org/10.1016/j.comnet.2022.109118
Received 24 July 2021; Received in revised form 20 May 2022; Accepted 15 June 2022
Available online 26 June 2022
1389-1286/© 2022 Elsevier B.V. All rights reserved.

forks in order to maximize their chances of winning a reward [11]. Some advanced PoS implementations to overcome the Nothing-at-Stake problem, such as Casper [12], are still in the testing stage and not yet deployed in large scale networks.

In this paper, we propose a new consensus algorithm that rationalizes the computational overhead of the original PoW and reduces its overall energy consumption to nearly 50% without degrading its security level , i.e., it keeps, at least, the same security properties as the original Nakamoto's consensus. The main idea is to factor the power spent during one mining round, in not only electing which node will write the current mining block but also selecting a small subset of miners $\mathcal{M}'$ that will be allotted exclusivity to mine in the next round (power-save round). Therefore, we modify the original mining scheme by defining a new participation rule in a mining race. Unlike the original PoW, the race in the next round will be among only a small subset of miners $\mathcal{M}'$, and thus less energy will be consumed. The other miners keep waiting until the block gets created before resuming again the mining process. Because only a small subset of miners will participate in a power-save round, it is possible that a deadlock happens if the involved $|\mathcal{M}'|$ miners fail to generate the block. This may occur, for instance, if the miners in $\mathcal{M}'$ get disconnected from the network. To ensure *liveness* and avoid such a situation, we introduce a time-out after each normal race round. If the nodes that are not participating in the mining during a power-save round, do not receive a new block after a specific period of time, they automatically resume the mining in order to create the missing block.

The contributions of this paper can be summarized as follow:

- In order to reduce the energy consumption in PoW, Green-PoW considers mining rounds in pairs and adjusts the election mechanism during the first mining round in the pair, to allow the election of a small subset of miners that will exclusively mine in the second round. Assuming the total network hashing power is equally distributed among miners, a significant energy saving up to 50% could thus be achieved.
- Given that only a small subset of miners can participate in the even-numbered mining rounds, Green-PoW reduces the probability of fork occurrences. Therefore, the total fork occurrence will be considerably decreased compared to the original PoW algorithm.
- Green-PoW can help in reducing the level of mining centralization presented in PoW. This kind of centralization can happen when a small portion of the miners, e.g., mining pools in the case of Bitcoin, holds the majority of the network hash-rate. In Green-PoW, a winner during the first round cannot participate in the second round mining, thus miners that hold the majority of the network hash-rate will no longer have the same potential for dominating block generation.
- We demonstrate the effectiveness of Green-PoW algorithm in reducing the overall energy consumption, during mining, by conducting a stochastic analysis where the mining is modeled as a Poisson process.

The remaining of this paper is organized as follows. Preliminaries about mining in PoW are presented in Section 2. Section 3 reviews the existing work to decrease the energy consumption in PoW. The detailed description of our proposed consensus algorithm can be found in Section 4. The consistency and liveness of the Green-PoW consensus algorithm are discussed in Section 5. Sections 6 and 7 analyze the security properties and reports the performance results of Green-PoW, respectively. Finally the paper is concluded in Section 8.

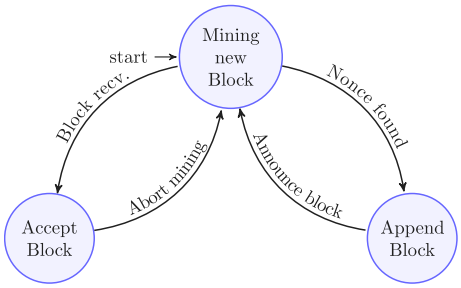


Fig. 1. The state diagram of a miner in the original PoW.

2. Preliminaries

In this section, we present the two main core components of Bitcoin blockchain, namely, the block mining process and difficulty adjustment. In addition, we discuss the security concerns about the fundamental design.

2.1. Proof-of-work

2.1.1. Block mining

In Bitcoin, miners participate in a PoW based consensus in order to maintain the consistency and integrity of a public distributed ledger, which is an ever-growing chain of a tamper-proof data structure called blocks. Each block records a list of transactions, previous block hash, Merkle root, timestamp, hash target, and a nonce. Blocks are chained by storing the hash of the predecessor block in the current block header. Miners participate in an incentivized race to forge a new valid block, by following a brute force approach to find a 32-bit *nonce* value that yields a block hash less than the target, which is derived from an adjustable value called mining difficulty, used to maintain the equilibrium between the block generation rate and the invested computational power over-time [1] (more details in Section 2.1.2).

The mining process in the original PoW is illustrated in Fig. 1. When the mining race begins, miners start competing for forming a valid block; the first miner that finds the nonce is considered as the leader of the current round for creating the new block. Such a miner announces the block to the rest of the network to get rewarded with a newly generated Bitcoins. Every other miner that receives the new block generated by the winning miner immediately desists mining the current block and start mining the next one.

2.1.2. Mining difficulty

Mining difficulty *(D)* is a measure of how difficult it is to find a *Nonce* of a valid block. It is reevaluated every two weeks based on the average block time of the previous 2016 blocks, to maintain a fixed block generation time (10 min in Bitcoin) [1]. The difficulty will increase when the average block time is less than the expected, as it indicates that the network's computational power has increased and miners have become capable of generating new blocks in less than 10 min. Sometimes the network experiences a plummet in the total computational power when a set of miners with a significant hashing capability depart the network thus, the mining difficulty will decrease. Eq. (1) and (2) shows the relation between the previous average block time and the difficulty level, where F is the factor used to recalculate

the difficulty of the next two weeks. It represents the ratio of the expected average block time of the network T_E to the actual recorded average block time T_{Avg}.

Based on how the difficulty adjustment works, Bitcoin pushes miners to invest in more powerful mining rigs, to be qualified for competing with other miners under the incriminating mining difficulty. Despite the operation cost, miners continue to spend more because of the expected remuneration when winning the mining race. This feature makes PoW based blockchain a power-hungry system that burns energy indefinitely as long as the network of miners is growing.

$$F = \frac{T_E}{T_{Avg}} \tag{1}$$

$$D_{(i+1)} = D_{(i)} \times F \tag{2}$$

2.1.3. Chain forks

Blockchain may witness inconsistencies that cause branching in the chain, this phenomenon is called forks, where more than one valid block is broadcasted to the network simultaneously [13]. Consequently, nodes get confused when updating the chain and each will end up adopting different blocks as the chain head. Network propagation delays are mostly the reason for fork occurrence, which has been validated empirically in [13,14]. Other factors also play a role in causing forks such as the time needed to generate a block, network bandwidth, and the number of connections each node establishes with the network. Blockchain nodes resolve a fork in the next block period by adopting the longest chain, which is identified by calculating the cumulative number of expected hashes performed to generate each block in the chain. The frequent occurrence of forks allows adversaries to conduct various attacks (discussed further in Section 6) that could degrade the security of the blockchain network.

3. Related work

In this section, we survey studies made so far in order to mitigate the energy-inefficiency of the PoW algorithm. There are two main categories of solutions. In the first category, the goal is to decrease the energy usage of the original PoW by either recycling the power spent during the mining process in serving other useful real problems rather than solving a useless puzzle, or modifying the consensus protocol flow while maintaining the cryptographic puzzle element unchanged. On the other hand, the second category follows completely different and consensus algorithms such as Proof of Stake (PoS) [6], Proof of Elapsed Time (PoET) [7], and Proof of Retrievability (PoR) [8]. While this class of solutions can achieve considerable energy saving, yet it cannot reach the same security level as the well tested PoW. Our proposed Green-PoW consensus mechanism can be classified in the first category of solutions, which aims to decrease the energy consumption of PoW. Therefore, the focus in this section is on setting Green-PoW apart from competing in such a category. Specifically, we focus on how energy consumption can be decreased during mining and how this affects the overall security of the proposed protocol in terms of mining centralization and fork occurrences.

- Energy saving Vs. mining centralization: The idea of turning the meaningless proof of work into useful tasks to solve actual computational problems of general utility has been studied in several work [9,15–17]. REM [9] replaces the wasted computations in performing the conventional PoW by executing useful workloads outsourced by clients. REM relies on the availability of Software Guard Extensions (SGX) technology at the miners' infrastructure, where a trusted execution environment is utilized to preserve the integrity and confidentiality of the workload. Miners get rewarded based on the number of instructions they can process successfully. However, REM falls short in overcoming the centralized nature of the SGX attestation process where accessing Intel's server is mandatory in order to validate clients' workloads.

Unlike REM, A. Shoker [16] and S. King [17] have proposed two distinct schemes that invest the mining effort to solve public problems that do not require confidentiality. Examples of the considered public problems include the generation of large prime numbers and performing complex matrix operations. In both schemes, the difficulty of solving a public problem can be adjusted, which allows it to be an appropriate replacement for the hashing processes done by the traditional PoW. Similarly, the study in [15] focused on constructing a Proof of Useful Work by replacing the traditional PoW puzzle with useful mathematical tasks that are easily verifiable, such as orthogonal vectors, 3SUM (problem of finding if a given set of n real numbers contains three elements that sum to zero), and a pairs shortest path. However, such a class of systems support only a predefined set of problems and cannot provide solutions for customized ones, which limits the utility of the processed workload. In addition, there is no significant practical value in solving these mathematical problems that justify the significant amount of computation power using in the mining.

Following a similar approach, Felipe et al. [18] have presented a proof-of-learning consensus mechanism that substitutes the PoW puzzle by machine learning tasks. The goal is to re-purpose the wasted computational power to build a decentralized system for crowd-sourcing machine learning models. In their approach, nodes are classified into suppliers, trainers, and validators. Supplier nodes publish machine learning tasks and provide the training and testing data, whereas trainers perform the training work in an attempt to win the block. Validator nodes, which are chosen randomly, are responsible for ranking the published machine learning models based on their performance, then pick the best as proof of the work. By design, the models, testing, and training data are all public and the scheme does not provide any confidentiality guarantees, which makes the usability of such an approach questionable. In order to preserve the privacy of both training and testing data, Qu et al. [19] have considered the use of federated learning. This enabled crowd-sourcing the workload required to train a model while using miner's local training data. Furthermore, the verification of the model's accuracy is done using a Homomorphic encryption-based verification scheme to prevent the disclosure of testing data to the network.

Daian et al. [20] have proposed outsourcing PoW to defeat several frequent cyberattacks such as spam emails and DDoS to recycle the inevitable wasted power. This can be achieved by utilizing the difficulty of solving the cryptographic problem as a challenge for the clients in order to benefit from a service. The mining process is decomposed into two parts, namely, inner and outer puzzles. The inner puzzles are solved by the service clients named workers that provide solutions to the outsourcer to be verified. The outer puzzle process the solutions provided by the workers in order to find the overall solution of the PoW. Consequently, the cost of solving the inner puzzle controls the number of requests that can be submitted by the client in a short period of time. Implementing outsourced PoW in practice is restricted by the resources used at the client's infrastructure (e.g. PCs, mobile phones). As lightweight clients will not be capable of solving the puzzle as fast as other power-full machines, this may delay or prevent them from reaching the service. Such a concern hinders the fairness of the network and obstructs the reliability of the service.

Even though this class of solutions succeed in reducing energy wastage by computing useful work, it can lead to mining decentral-

ization and thus, fails to preserve the same security properties of the original Nakamoto's consensus.

- Energy saving Vs. fork occurrences: One prominent work that modified the consensus protocol to optimize PoW is Bitcoin-NG [21]. Bitcoin-NG altered the conventional PoW consensus by segregating the election of a block creator from processing transactions. This necessitates dividing Bitcoin's block structure into two new types, key blocks for electing a leader for the next epoch, and micro blocks generated by the epoch's leader and consist of network's transactions. This change in design led to faster transaction validation as micro blocks do not require including proof of work, alongside to faster key block propagation due to its small size. While Bitcoin-NG's main purpose is to improve transaction throughput and latency, it implicitly reduces the energy consumption compared to the original Bitcoin since not all the created blocks require PoW. However, Bitcoin-NG suffers from frequent micro block forks that occur whenever a new leader is elected for the new epoch because the current leader will carry on generating micro blocks and propagate it to the network while being unaware that another miner already solved a new key block and started generating micro blocks simultaneously. This behavior increases fork occurrences and makes the network susceptible to fork-related attacks, e.g., double-spending. To mitigate this issue, the authors have proposed a new type of transaction that can be issued by nodes that witnessed two conflicting transactions to report an attempt of double spending and motivate the investigation of the fraudulent miner rewarding. Yet, if the double-spending attack is not noticed during the maturity window of the attacker key block or before the attacker spent the revenue, the double spending will indeed occur and the involved leader will not be penalized. Based on the aforementioned concern we can conclude that although Bitcoin-NG is an enhanced version of Bitcoin in terms of transaction throughput and energy optimization, it has diminished the level of security offered by the original Bitcoin protocol.

In a more recent work, Tsabary et al. [22] proposed to reduce the total energy spent in PoW-based cryptocurrencies by bounding the electricity expenditure while claiming to achieve similar security properties as the classical PoW. The intuition behind their Hybrid Expenditure Blockchain (HEB) is to break the direct relation between the amount of expended computational power for mining and the value of the respective cryptocurrency used for rewarding. To achieve the energy reduction goal, HEB enables miners to consume their winning coins as part of the mining process. By doing so, miners increase their winning probability, and consequently their reward, without using additional computational power. The internally-spent coins during mining are redistributed while ensuring incentive compatibility for miners.

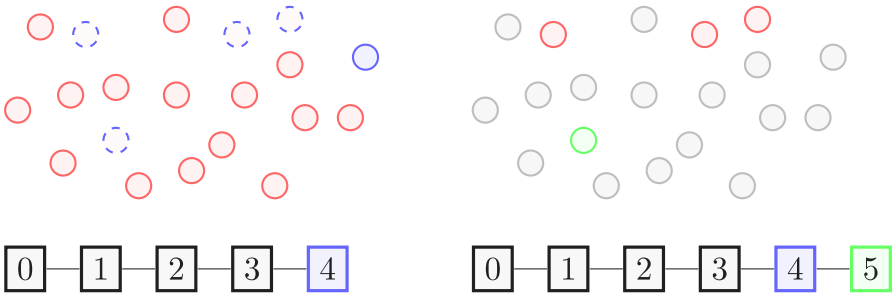
In summary, the surveyed studies fall short in effectively addressing the power consumption of PoW. Existing approaches either replace the crypto-puzzle with different types of useful work which adds complexity to the consensus process or alter the ledger's structure and consensus flow drastically, which degrades the network's security. Table 1 compares the surveyed studies in terms of energy consumption and security level. We define three energy and security levels, i.e., **high, moderate** and **low**. We refer to a high energy consumption when the consumed energy during mining is similar to that of the original Nakamoto's consensus. Moderate and low, when the mining process requires moderate or no extra work, respectively. For the security level, it is high when the solution presents better or similar security properties compared to the original Nakamoto's consensus. Moderate and low, when additional vulnerabilities are incurred by the new solution. Green-PoW opts to avoid these shortcomings by achieving dramatic energy reduction while sustaining the security properties of the PoW based consensus methodology.

Algorithm 1 Green-PoW Consensus Algorithm.

```
1: init :
   b = 1; ρ = 1 {b : block number, b : ρ : round}
   runnerUp = false {I am a runner-up}
2: loop
3:   if ρ == 1 then
4:     nonce =FindBlockNonce(b)
5:     if nonce then
6:       {nonce found}
7:       AppendNewBlock(b)
8:       AnnounceBlock(b)
9:       b = b + 1
10:      ρ = 2 {Enter power-save mode}
11:    else
12:      if ValidBlockReceived(b) then
13:        AppendNewBlock(b)
14:        b = b + 1
15:        ρ = 2
16:        ContinueFindBlockNonce(b)
17:        if nonce then
18:          runnerUp = true
19:          AnnounceRunnerUpBlock(b)
20:        else
21:          if RunnerUpBlockReceived(b) then
22:            AbortMining(b) {Enter power-save mode}
23:          end if
24:        end if
25:      end if
26:    end if
27:  else
28:    {second round: ρ = 2}
29:    if runnerUp then
30:      nonce =FindBlockNonce(b)
31:      if nonce then
32:        AppendNewBlock(b)
33:        AnnounceBlock(b)
34:      else
35:        if ValidBlockReceived(b) then
36:          AppendNewBlock(b)
37:          AbortMining(b)
38:        end if
39:      end if
40:      b = b + 1
41:      ρ = 1
42:      runnerUp = false
43:    else
44:      {not a runner-up}
45:      if ValidBlockReceived(b) then
46:        AppendNewBlock(b)
47:        b = b + 1
48:        ρ = 1 {Exit power-save mode}
49:      end if
50:    end if
51:  end if
52: end loop
```

4. Energy-efficient consensus algorithm

Green-PoW is an energy-efficient consensus algorithm that reduces the computation load to nearly 50% compared to the original Bitcoin's PoW algorithm, without affecting the other properties of the system. The algorithm divides time into epochs, where each epoch consists of two consecutive mining rounds. Let τ_i denote the epoch of time corresponding to the creation of blocks number $2i$ and $2i+1$, and ρ_i^1 and



(a) First mining round ρ_2^1. (b) Second mining round ρ_2^2.

Fig. 2. Illustrating the mining process of Green-PoW in one epoch of time (τ_2). The blue node is the winner of the first round (w_1) and generates the block number 4 (blue square). Red nodes are losing participants in the mining. The dashed blue nodes are second place winners ($\mathcal{M}_2^2$), which will be the only nodes to handle the block in the next round (ρ_2^2). The green node is the winner of the second round which generates block number 5 (green square). Gray nodes are idle miners that do not participate in the mining during the second round $\{\mathcal{M} \setminus \mathcal{M}_2^2\}$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Related work summary.

Solution	Energy consumption	Security Level
REM [9]	**[Moderate]** The consumed energy is equivalent to the energy required to execute the useful workload.	**[Moderate]** Rely on a central trusted execution environment. Yet REM claims that their statistics-based security framework eliminates this concern.
Proof of useful work [15–17]	**[High]** The required work to be done by the miners is computationally heavy (time consuming).	**[High]** The security level of this category is similar to the original Nakamoto's consensus.
Proof of learning [18,19]	**[Moderate]** The consumed energy is equivalent to that required to train and test a machine learning model.	**[Moderate]** The security of the protocol relies on the set of validators which are randomly chosen to evaluate the accuracy of the trained models.
Bitcoin-NG [21]	**[Moderate]** The consumed energy is reduced compared to the original Bitcoin since not all the created blocks require PoW.	**[Low]** The protocol is vulnerable to double-spending attacks.
PoS [6]	**[Low]** The protocol relies on the staked cryptocurrencies to prevent Sybil attack without incurring any extra work.	[Moderate] Vulnerable to the Nothing-at-Stake attack.
PoET [7]	**[Low]** Energy waste-free decentralized consensus.	**[Low]** Rely on a central trusted execution environment (SGX).
PoR [8]	**[Moderate]** The protocol consumes energy equivalent to that required for manufacturing and operating the storage units.	**[Moderate]** Rely on the security of the retrievable file storage and on the security of the miner's private keys storage.

Table 2
Definition of the used notation.

Symbol	Description
τ_i	ith mining epoch
ρ^1, ρ^2	First and second mining round
m_i	Miner i
$\mathcal{M}$	Set of all miners
$\mathcal{M}_i^2$	Set of runners-up during the ith epoch
w	Winner miner
r	Runner-up miner
η	Short additional mining time, during ρ^1, after selecting the first runner-up
D^1, D^2	Mining difficulty during ρ^1 and ρ^2
$\mathcal{P}$	Total hashing power of the network
h_i	Hashing power of m_i
λ	Mining rate, i.e., the average generated blocks per second
E_i	Energy consumption during mining by m_i
E	Total energy consumption across the network during mining
E_{save}	Energy saving by Green-PoW compared to the original PoW

ρ_i^2 denote the first and second mining round, respectively, within the epoch τ_i. During ρ_i^1, the mining process to create a new block follows the same Bitcoin mining steps, where the set of all miners, which is denoted as $\mathcal{M}$, can participate. In addition, a very small subset of miners, denoted as $\mathcal{M}_i^2$, is elected during this same mining round to be the only eligible participants for mining the next block in ρ_i^2. Note that $\mathcal{M}_i^1$, which refers to the set of miners that can participate in ρ_i^1, is equal to $\mathcal{M}$ and both are used interchangeably. All the other miners $\mathcal{M} \setminus \mathcal{M}_i^2$, during ρ_i^2, pause until the considered block gets appended before starting a new mining epoch (τ_{i+1}). An illustrative example of the mining process during one epoch of time is given in Fig. 2. The detailed description of each step of the algorithm is given in the remaining of this section. Table 2 enlists all keys and defines the notation.

4.1. Runner(s)-up election

In the original PoW, when a puzzle-related block is solved by some miner, all the other network nodes desist the mining of that block and immediately start mining the next block. In Green-PoW, if a valid block is found and the first place winner is elected, the race will continue between miners to also determine the runner-up, i.e., the node that has the second place in the same block race. We denote by ω_i the first place winner, and r_i the runner-up. Such runner-up (r_i) will be the only eligible node to mine in ρ_i^2, and all the other nodes enter in *mining-save mode* until the end of ρ_i^2. As illustrated in Fig. 3, when a miner m_j receives a block from the first place winner it continues the mining of the same block. Miner m_j either (1) finds the nonce before receiving a block from another miner claiming the second place; in this case node m_j will broadcast an announcement to the entire network that it is the runner-up, and then immediately starts the mining of the next block, or (2) receives a block from another node m_k and subsequently, adds it to its runners-up list ($\mathcal{M}_i^2$) and then enters the *mining-save mode* for one round. Instead of switching directly to the mining-save mode, m_j continues mining for a very short period of time with the aim of solving the block and joining the $\mathcal{M}_i^2$ list. The next section provides more details about the importance of such continued mining after receiving a block from the miner claiming the second place. $\mathcal{M}_i^2$ serves in the second round to make sure that any received block is originated from a valid runner-up.

Fig. 3 illustrates a simplified case where a miner will automatically switch to *mining-save mode* if it receives a block from another node claiming the second place. Algorithm 1 also summarizes the different

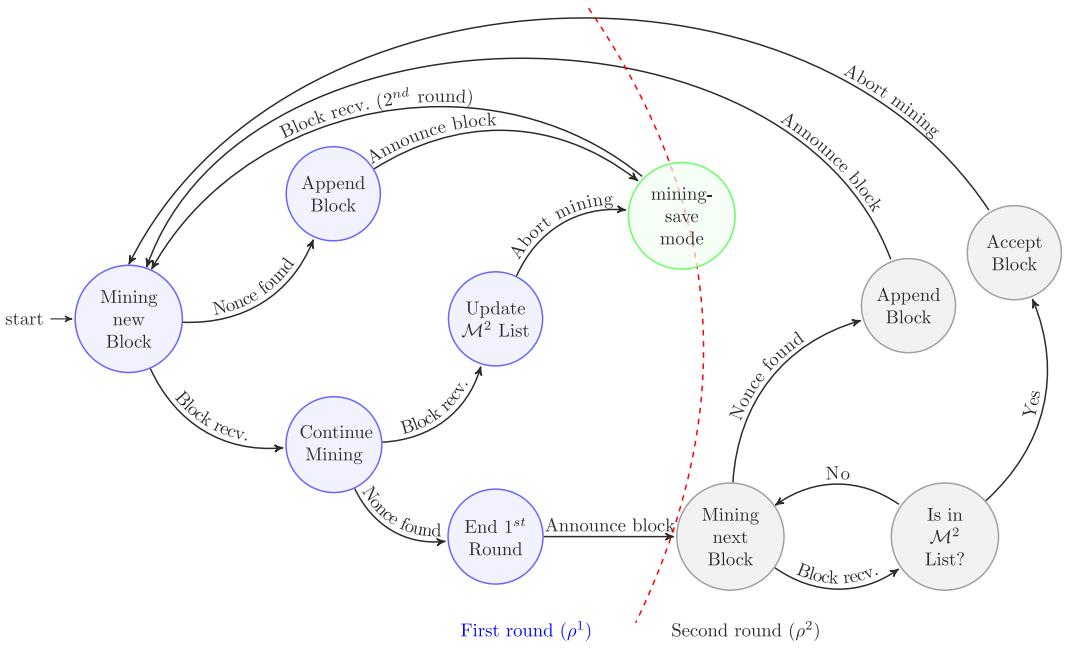


Fig. 3. The state diagram of a miner in Green-PoW.

steps of Green-PoW consensus algorithm executed by nodes during mining. As illustrated in Fig. 3 and Algorithm 1, a miner can be in one of the following four mining states; (1) mining a block to win the first place in ρ^1, (2) continuing mining a block to win the second place in ρ^1, (3) mining a block to win the first place in ρ^2, or (4) mining-save mode. According to Algorithm 1, when ρ is equal to 1, the miner starts searching for a valid nonce for the next block b. If the nonce is found before receiving a valid block from another node, a new block is appended to the chain and announced to the other nodes in the network; such a wining miner also sets ρ to 2, increment b by 1, and enters the power-saving mode for one round (during $\rho = 2$). Otherwise, if the miner receives a valid block from another node, it appends the received block, changes the mining round ρ to 2, and continues mining with the aim to become a runner-up for the same block b. If the miner successfully finds a valid nonce for b, it announces its runner-up block to the others and immediately starts mining the next block (for $b = b+1$) in the second round ($\rho = 2$). Otherwise, if the miner receives the block from another node claiming to be a runner-up for b, the miner aborts mining and switches the power-saving mode for one round (during $\rho = 2$). The miner leaves the power-saving mode only when it receives a valid block from a runner-up. When a valid block is found during $\rho = 2$, the round ρ is reset to 1 and b is incremented by 1.

Liveness: Because of the distributed and asynchronous nature of the network, it is possible to have multiple nodes that consider themselves as runners-up when they find the block at nearly the same time. A similar situation can happen for the first place winner and leads to network fork that is solved later by the longest chain rule [13]. For the runner-up election, this situation will not cause any problem as any node that successfully mines the block in the first round will be considered as a potential runner-up and can participate in the mining race during ρ_i^2. In this case, the set of multiple runners-up $\mathcal{M}_i^2 = \{r_i^1, r_i^2, \ldots, r_i^l\}$. It is worth noting that having multiple nodes as second-place winners does not affect the system inconsistency, but on the contrary, it improves system *liveness* as it increases the chance that a block gets mined during the second round.

In order to engage a sufficient number of participants in ρ_i^2 and improve the system *liveness*, in Green-PoW, even if one node has already claimed to be a runner-up, the other nodes can still continue mining. The decision of a node to continue mining the first round's block depends on the probability of winning the race in ρ_i^2. A miner that solves the first round's block very late, will have a very small chance to win as the others have started the mining earlier. Thus, a stubborn miner will not have any advantage through continued mining when other miners have already obtained a valid membership to mine in the second round. It is better for such a stubborn miner to stop the mining to save its energy and be ready for mining in the next epoch. For the sake of simplicity, when a node receives the first announcement of a runner-up, it continues mining for only a very short period of time η with the hope of quickly finding the nonce and joining ρ_i^2. The value of η is subject to liveness and energy trade-off and is expected to be determined based on the rate of block generation in the network. In the next section, we will introduce another liveness parameter and will elaborate more on how η could be tuned.

Runners-up list inconsistency: Green-PoW is based on Nakamoto's consensus algorithm which can only provide a probabilistic finality. Therefore, there is no certainty that all nodes will agree on the same list $\mathcal{M}^2$ of runners-up. This situation may lead to a fork if the winner in the second round does not exist in the $\mathcal{M}^2$ of some nodes. However, as in Bitcoin, this will be automatically resolved by the longest chain rule.

4.2. Second round timeout

Since only nodes that successfully mine the first round block may be part of $\mathcal{M}_i^2$, the number of potential miners in ρ_i^2, i.e., $|\mathcal{M}_i^2|$, is naturally limited. While this is advantageous from an energy conservation point of view, it is possible that the system goes to a deadlock and the next block does not get generated. This can happen for instance if miners in $\mathcal{M}_i^2$ are isolated from the rest of the network, inadvertent in case of network segmentation, or intentionally by an adversary who launches an eclipse attack [23]. To mitigate this problem and ensure system *liveness*, Green-PoW employs a time-out at the beginning of each ρ_i^2. The time-out should be greater than the average time needed to generate a new block. If a block announcement is not received before the time-out, inactive nodes quit the *mining-save mode* and start immediately ρ_i^2. However, because of the asynchronous nature of the network and the malicious behavior of some nodes, the introduction of time-out may lead to some special cases that we discuss in the following:

- A malicious node that is not eligible to participate in ρ_i^2, may try to start the mining before the time-out in order to get an advantage (receive a reward) over other non-participating nodes. However, this is very risky for the attacker as the success of such manipulation depends on the probability that no block will be received from eligible nodes ($\mathcal{M}_i^2$); in such a case the attacker could be wasting a lot of energy if it does not win.
- Due to network asynchronicity, it is possible that some nodes will receive a valid block from an honest ineligible participant while their time-out is not yet ended. In this situation, these nodes will initially reject the block, but because the majority will accept it, the block will appear in the longest chain and the minority will end up accepting it.

It is worth noting that both the time-out and η help in defining a good balance between system liveness and energy-efficiency. Growing η improves system liveness by increasing $|\mathcal{M}_i^2|$, yet it diminishes energy-efficiency as more miners will participate in ρ_i^2. By using the time-out, Green-PoW can ensure liveness, but it may also increase the energy consumption if $|\mathcal{M}_i^2|$ is very small, i.e., η is very short, as timing-out ρ_i^2 is likely to happen frequently. Therefore, a careful selection of η is crucial for Green-PoW to ensure the desired energy-efficiency. More hints about the typical choice of the time-out and η parameters are given in Section 7.

4.3. Second round mining difficulty

As presented in Section 2.1.2, given that the total mining power of the network can change over time, the mining difficulty is dynamically adjusted to ensure that blocks are generated at a nearly constant rate. In Green-PoW, because the total hash power decreases drastically in each second round mining, compared to the first round, a new difficulty level defined specifically for the second round is required. Let D^1 and D^2 denote the difficulty level to consider during ρ^1 and ρ^2, respectively. A block is considered as valid only if its hash value respects the target hash (e.g., in the block header), which is calculated based on the difficulty level of the corresponding mining round. The difficulty level is initially set to the minimum value and updated every T period of time (every 2 weeks in Bitcoin). Using Eq. (1) and (2) presented in Section 2.1.2, the new difficulties D_j^1 and D_j^2 for the jth T_j period can be calculated as follow:

$$D_j^1 = D_{j-1}^1 \left(\frac{T_E}{T_{Avg^1}} \right) \tag{3}$$

$$D_j^2 = D_{j-1}^2 \left(\frac{T_E}{T_{Avg^2}} \right) \tag{4}$$

where T_E is the expected average block time (10 min in Bitcoin), and T_{Avg^1} and T_{Avg^2} are the actual average block-time calculated over the last T_j, for blocks generated in ρ^1 and ρ^2, respectively. Note that because it may happen that some blocks in the second round are generated with the participation of all network miners, in case of timing out, these blocks will be considered for the adjustment of D^1 but not D^2. To distinguish between blocks generated using a small subset of miners from those using all network miners, the target value in the header of each block will be verified.

The mining in ρ^2 is similar to the mining in the original PoW with the exception that only miners in the corresponding $\mathcal{M}^2$ can participate, and use the second mining difficulty D^2. When a valid block is formed and propagated to the entire network, as shown in Fig. 3, the other active miners in $\mathcal{M}^2$ stop the mining of the current block and start new mining epoch. The passive miners ($\mathcal{M} \setminus \mathcal{M}^2$) also leave their *mining-save mode* and join the others in a new epoch.

4.4. Why only two rounds?

For increased energy saving, one may consider a large number (k) of mining rounds instead of only two, where during the first round we select all subsets of miners that will be responsible of mining the upcoming $k-1$ mining-rounds. However, using k in excess of two would introduce new issues that are difficult to resolve. To better explain, we use the same notation as in Section 4, where ρ^i denotes the ith round, and w_i, and $\mathcal{M}^i$ refer to the winner and the miners of round i, respectively.

In an asynchronous network it is generally difficult to achieve consistency about a particular event and agree about when to start/end it. In our case, achieving agreement about the event of selecting the ith subset of miners, $\mathcal{M}^i$, becomes difficult for $k > 2$. In fact, for $k = 1$, as in the original Nakamoto's algorithm, miners reach consensus on when to start competition in a new mining round by referring to the time when a first valid block is received, declaring a winner w_1. Any inconsistency about w_1 will lead to a fork that will be resolved later using the longest chain rule. For $k = 2$, as in Green-PoW, the consensus algorithm needs to also select during ρ^1, a subset of miners, called runners-up, that will have the exclusivity of mining the block of ρ^2. The runners-up selection will start when w_1 is declared, and remains open until one of the already selected runners-up resolves the second-round's block and announces it. To be considered as a runner-up, a miner needs to successfully complete (resolve the block of) ρ^1 during the selection period.

Following the same logic, for $k = 3$, the selection of $\mathcal{M}^3$ would start when w_2 is declared. However, this raises two problems. First, potential miners that resolve the block of ρ^1 and qualified as runners-up may decide to delay their block announcement until the beginning of ρ^3 in order to have an early start and get more chances to become w_3. This will diminish the number of miners participating in the second-mining round, i.e., the size of $\mathcal{M}^2$, and may lead to a deadlock situation, where all runners-up are waiting for ρ^3 and ρ^2 never complete due to lack of miner participation. Second, the increase of k will progressively reduce the number of qualified miners, where the winners of all previous $i-1$ ($i < k$) rounds are not allowed to participate in ρ^i, i.e., $\notin \mathcal{M}^i$; in other words, the list of those excluded winners will significantly grow and most probably becomes inconsistent among the different network nodes. Such inconsistency may lead to multiple forks that are difficult to resolve.

To overcome the previous concerns, one solution would be to find a way to select all the $k-1$ sub sets of miners beforehand, during ρ^1 and before starting any of the next $k-1$ rounds. However, from a security perspective, this will give an adversary enough time to plan for an attack to eclipse/isolate/compromise one or more of the future sub-sets of elected miners and undermine the entire network.

4.5. Fairness

In Green-PoW, the node that successfully mines a block in the first round is not eligible to compete in the second round. This choice might harm fairness as the first round winner needs to pause for one round before joining the mining race again. However, such a design choice is important to achieve the desired energy saving as well as reducing the effect of transaction censorship attack and mining centralization (see discussion in Section 6). When few miners have disproportionally high computation power, they can dominate the system and lead to mining centralization. Green-PoW prevents such a scenario and enables less powerful miners to participate and grow the chain, which instruments democracy, again without much harm to fairness. For instance, as illustrated in Fig. 4, the most powerful miner (miner 29), will lose only 5% of its share of mined blocks. Moreover, miners that cannot participate in the second round are not wasting their energy as they will switch to power-saving mode during that round.

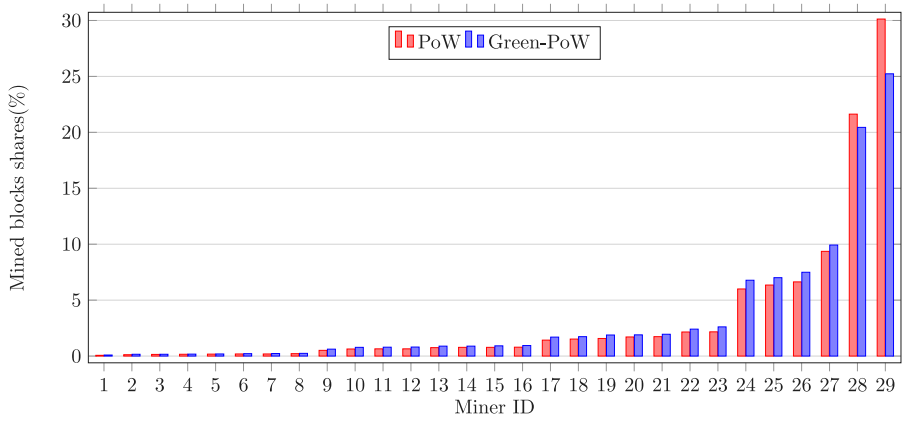


Fig. 4. Shares of miners in Ethereum Vs. Green-PoW based Ethereum, calculated using real data of the latest 7438 blocks imported from the main Ethereum network.

5. Green-PoW consistency and Liveness

The basic security properties of Nakamoto's blockchain consensus protocol were rigorously formulated in [24,25]. These properties include consistency and liveness that can be derived from the following three attributes: (i) chain-growth, (ii) chain-quality and (iii) common-prefix; the attributes capture the resilience of the underlying data structure of the blockchain in the presence of an adversary aiming to subvert the protocol security properties. Both chain-growth and chain-quality refer to liveness, whereas common-prefix implies consistency (safety). These three properties are parameterized by T and can be defined as follow:

1. **Chain-growth:** At any point in the execution of the protocol, the chain of the honest miners grows by T blocks in the last $O(T)$ rounds with very high probability (in T).
2. **Chain-quality:** Any sequence of roughly $\Theta(T)$ blocks must contain a sufficient fraction of honest work contributed by honest miners.
3. **Common-prefix**: For any T, with overwhelming probability (in T), at any two rounds r and s with $r < s$, all but the last T blocks in the chain of any honest miner i at r must be a prefix of the chain of an honest miner j at s.

The studies by Garay et al. [24] and Pass et al. [25] formally proved that Nakamoto's consensus protocol satisfies the safety and liveness properties as long as the mining difficulty (p) is appropriately set as a function of the maximum network delay (Δ). The proposed Green-PoW can be viewed as a special case of Nakamoto's consensus protocol that also *preserves the standard properties of chain growth, chain quality and common prefix*. The correctness of such a statement can be informally proven by highlighting the main difference between Green-PoW and Nakamoto's consensus protocol and how this difference does not affect the consistency and liveness properties.

To save energy, Green-PoW defines two types of mining rounds. In the first mining round, similar to the Nakamoto's consensus, all miners can participate in the mining process to generate new blocks. However, in the second-round, only runners-up that are selected during the first round can take part of the mining process. Intuitively, the limited number of miners that can participate in the second-round might affect the *chain-growth* property. In other words, the probability of generating a new block during the second-round is proportional to the effective size of the runners-up list, i.e., $|M^2|$. As discussed in Section 4.2, the limited size of the set M^2 may lead to deadlock situation if all the runners-up fail to generate a new block during the second-round. To avoid this situation and make sure that the chain is continually growing, Green-PoW defines a time-out at the beginning of each second-round. If the network does not hear any block from a runner-up before the time-out, the remaining miners in $\{M \setminus M^2\}$ leave the mining-save mode and start the mining process which ensure *chain-growth* property.

The *chain-quality* attribute guarantees that the chain of honest nodes do not contain long sequences of adversarial blocks. As Green-PoW follows the same protocol as Nakamoto's consensus to produce blocks during the first round, the quality of blocks is preserved during that round. During the second round, given that runners-up, which have the exclusivity to produce blocks in the second-round, are elected during the first-round following the same Nakamoto's consensus rules, the chain-quality during the second-round is also preserved.

In Green-PoW, the probability of fork occurrences decreases compared to the original PoW consensus algorithm (as will be analyzed in Section 6.5). It is clear that the reduction of fork occurrences improves the system *consistency*, i.e., common-prefix. One might argue that the consistency of Green-PoW can be affected by the limited number of participants in the second-round, where an adversary can compromise the runners-up. However, as will be explained in Section 6.2, even if the number of runners-up is small, the chance for an adversary to succeed is little. Therefore, we believe that consistency in Green-PoW is stronger than Nakamoto's consensus.

6. Security analysis

The main goal of Green-PoW is to considerably reduce the energy consumption of the original Nakamoto's algorithm, without degrading its security. Nonetheless, Green-PoW also improves some security properties such as reducing fork occurrences, mining centralization, and the effect of censorship attack. Addressing other known attacks against Nakamoto's algorithm, such as self-mining, is out of scope and is not a design objective for Green-PoW.

6.1. Sybil attack resistance

An adversarial miner can conducts a Sybil attack [26] by creating multiple fake identities in order to gain a disproportionate mining share of the network. These fake identities appear to be distinct miners, whereas in fact they are all controlled by a single node. The use of a PoW mechanism in the original Nakamoto's consensus is mainly to prevent such an attack. Green-PoW preserves the Sybil attack resilience of the original Nakamoto's consensus algorithm. In Green-PoW, a malicious miner may try to perform a Sybil attack in order to violate the established Green-PoW rule that prohibits a first-round winner from participating in the second mining round. To break such a rule, a malicious node may try to use two fake identities, one reserved to compete in the first round to become a block winner, and a second identity to gain runners-up designation and qualify for the mining in the second round. However, such an attack cannot succeed as the malicious node in this case needs to split the mining power between the

two identities which significantly diminishes the probability of winning both the first-round block and runner-up membership.

Another strategy by the adversary is to use the whole miner power to win the first block and after that starts again the mining using the second fake identity to become a runner-up. However, such an attack scenario requires the malicious miner to at least have double the computation power of each other node in the Blockchain network, which contradicts the basic PoW assumption where the *majority of the computing power* must be held by honest miners. Thus, this strategy would fail since the malicious node in this case needs to compete with the other miners that have started the mining much earlier. Consequently, the chance for the malicious miner to also win the second place before the others is very low. Finally, we note that the presence of a super node in the Blockchain network that can dominate and win all mining attempts is a fundamental challenge that Nakamoto's algorithm, and obviously Green-PoW, cannot overcome. We reiterate that Green-PoW is optimizing the energy profile of the original Nakamoto's algorithm and is not geared to tackle any security vulnerabilities.

6.2. Undermining runners-up

In Green-PoW, a newly selected runner-up in the first round can start the second-round mining immediately. Therefore, the only available amount of time for an adversary to perform its attack against a runner-up is after the designation of such a runner-up and before the end of the second mining round, i.e., before one of the runners-up finds a valid second round's block. Fortunately, this time is very limited and is about 10 min and 20 s in average for Bitcoin and Ethereum networks, respectively. For the attack to succeed, an adversary needs to undermine all runners-up within such a small time window, which is difficult to achieve since the runners-up are not all known at the same time but rather are gradually picked during the selection phase.

6.3. Transaction censorship

A transaction censorship attack [27] happens when adversarial miners prevent the inclusion of certain transactions in their mined blocks. Such an attack might lead to significant financial damages, e.g., by censoring all financial transactions sent from a target company. This type of attacks can only be carried out by powerful miners, such as mining pools, that control the mining of most blocks. The potential of transaction censorship exists in public blockchains because of the limited size of the block and the asynchronous nature of the network, which makes it difficult to verify which set of transactions a particular block must include [21,28]. Usually, miners select only a subset of transactions from their pool of pending transactions to not exceed the block-size limit and tend to prioritize transactions with higher fees to maximize their profit. Such freedom on selecting transactions to include in a block gives adversarial miners the opportunity to censor some transactions from being added to the next block, even those with high fees.

By design, PoW extenuates this concern because the censorship time t_c is bounded by the average block generation time $1/\lambda$, and restricted by the fact that a malicious miner m_c must be the winner. However, a powerful attacker that may successfully mine and win k consecutive blocks will delay the inclusion of some urgent transactions for a longer time, i.e., $t_c = k/\lambda$. In Green-PoW, a malicious miner m_c can still censor transactions for t_c during ρ^1, yet because the winner miner in ρ^1 cannot participate in ρ^2, the effect of such an attack is limited to only one mining round. As a result, Green-PoW can reduce the censorship time to nearly 50%, which guarantees users better transaction time and reduces the intensity of a potential denial of service attack launched by powerful miners. The only case when a winner in ρ^1 can also participate and win in ρ^2, is after timing out ρ^2. However, this can only happen under specific conditions, such as separating the set $\mathcal{M}^2$ from the rest of the network, which is difficult to be controlled by an attacker.

6.4. Mining centralization

In PoW-based networks, such as Bitcoin or Ethereum, the mining power is concentrated among a relatively small number of miners (pools) which makes the crypto-system highly susceptible to censorship or even 51% attacks [29]. Less powerful miners are usually unfortunate to generate a new valid block in the presence of other superior miners. This will eventually lead to a monopoly based system, where a small percentage of the network earns the highest rewarding shares. By design, Green-PoW can reduce the monopoly of powerful miners, since generating consecutive blocks by the same miner is likely not possible. A miner that wins the mining race in ρ^1 is not allowed to participate in ρ^2, and consequently gives the other nodes the chance to win with less competition. The only case when it is still possible that a winner in ρ^1 also wins in ρ^2, is after timing out ρ^2.

To better illustrate the impact of Green-PoW on mining shares distribution, we have imported the latest 7437 blocks (50 days) from the Ethereum main network. We plot in Fig. 4 the corresponding shares of each miner as the ratio of its mined blocks relative to the total block count, i.e., 7437. In the same plot, we also include the corresponding shares of each miner when applying Green-PoW to the same blocks. In Green-PoW, two, three, four, or more consecutive blocks are highly improbable to be mined by the same miner. It is worth mentioning that two consecutive blocks could be generated by the same miner under a special condition, where a miner wins the second round block and the next block in the first round of the following epoch. For this reason, we subtracted one block from only 50% of two consecutive blocks cases. As shown in Fig. 4, the corresponding shares of the most powerful miners, i.e., miner numbers 28 and 29, in Green-PoW are reduced, compared to the case of the original PoW. Such an impact limits the dominance that the most powerful miners may have on the network. For example, the most powerful miner had a 5% reduction in its share, which is redistributed among other nodes. The reduction can be more significant if the computing power varies widely among the nodes. Subsequently, better share distribution between miners could be achieved.

It is important to note that Green-PoW does not completely eliminate the mining centralization problem presented in the original Nakamoto's consensus. In the scenario where the network has two most powerful miners colluding together to generate blocks, with a high probability, they will alternatively win the first and the second round.

6.5. Fork occurrences

As discussed in Section 2.1.3, a fork in blockchain can happen when multiple miners find a block almost simultaneously. More generally, a fork occurs whenever a miner m_j finds a block while another miner m_i has already formed a valid block without being aware of it. This situation is likely to occur in a network with a large number of competing miners and having a long propagation delay. This effect has been well studied in the literature; specifically Decker and Wattenhofer [30] have conducted a theoretical analysis and presented an approximate model to predict the rate at which forks can occur. For a newly found block b_i by m_i, the probability of fork occurrence, i.e., conflicting blocks will be found by other miners before being aware of b_i, is estimated by (i) determining the number of unaware miners at time t, and (ii) the probability that each unaware miner will find a conflicting block during that time. Given a ratio of unaware miners $u(t)$ about b_i during time t, the probability of having a fork (F) on the network can be expressed as follow:

$$Pr[F > 0] = 1 - (1 - P_b)^{\int_0^{\infty} u(t)\ dt} \tag{5}$$

Where P_b is the probability of a block being found by the network at a given time t. From the formula, it is clear that the fork rate is proportional to the ratio of unaware miners and thus proportional to the total number of miners in the network.

In Green-PoW, during the second round, the ratio of active miners is very small compared to the first round. Therefore, the ratio of unaware miners u is also very small. Subsequently, Green-PoW helps diminish the fork rate by reducing the number of unaware miners during the second mining round.

7. Performance evaluation

In this section, we evaluate the power consumption of Green-PoW using stochastic analysis. As modeled in the Bitcoin white paper [1] and in [31–33], the Bitcoin mining process could be well-approximated as a Poisson process with a deterministic rate λ which represents the mining rate or the *average time* between block-arrival events. In the following, we first formulate the mining process in both PoW and Green-PoW as a Poisson process and then assess the power saving achieved by Green-PoW.

7.1. Average power saving in Green-PoW

Let $\mathcal{M}$ denote the set of n miners in the network $\mathcal{M} = \{m_1, m_2, \dots, m_n\}$. Each miner $m_i \in \mathcal{M}$ has a fraction h_i of the total hashing power in the network $\mathcal{P}$, so that it mines a new block at a rate of $h_i\lambda$, where $\sum h_i = 1$. As explained in Section 2.1.2, the difficulty of finding a block is dynamically adjusted to ensure that a block is generated every $1/\lambda$ seconds in expectation with a rate λ ($\lambda = 1/600$ in Bitcoin). The inter-arrival times of consecutive blocks follow Exponential distribution with the same rate parameter λ, whose cumulative distribution function is:

$$\Pr[T \leq t] = 1 - e^{-\lambda t} \tag{6}$$

In PoW, each miner m_i spends on average $1/\lambda$ and consumes energy E_i which is proportional to its hashing power h_i and can be expressed as:

$$\mathrm{E}_i = \frac{1}{\lambda}\, h_i\, \mathcal{P} \tag{7}$$

Therefore, the average total energy E consumed by the network to generate a block is inversely proportional to the block generation rate λ:

$$\mathrm{E} = \mathcal{P}\frac{1}{\lambda} \sum_{i=1}^{n} h_i = \mathcal{P}\frac{1}{\lambda} \tag{8}$$

In Green-PoW, a block is either generated during the first or the second round. In the first round, compared to PoW mining, additional energy is consumed in order to select the second-place winners. This additional energy depends on the number of second-place winners and the time they need to complete the mining and form a valid block. Assuming that m_f is the first winner, $m_s, m_{s+1}, \dots, m_{s+k}$ are k runners-up, and $t_s, t_{s+1}, \dots, t_{s+k}$ the respective time needed by each of the runner-up to find the block. Thus the average total energy $\mathrm{E}_{1^{\mathrm{st}}}$ consumed by the network during the first round can be expressed as follow:

$$\mathrm{E}_{1^{\mathrm{st}}} = \mathcal{P}\left(\frac{1}{\lambda} + \sum_{i=s}^{k} t_i(1 - h_f + \sum_{j=i-1}^{k-1} h_j)\right), \tag{9}$$

where $h_{s-1} = 0$

In the second round, the average consumed energy is proportional to the time needed to generate a block ($1/\lambda$) and the total hashing power of the runners-up. For simplicity, we do not consider the scenario where the set of runners-up fail to generate a block, and other miners start the mining process after the timeout:

$$\mathrm{E}_{2^{\mathrm{nd}}} = \mathcal{P}\frac{1}{\lambda} \sum_{i=s}^{k} h_i \tag{10}$$

From Eqs. (8), (9) and (10), the power saving in Green-PoW can be, therefore, expressed as follow:

$$\mathrm{E}_{\mathrm{save}} = 2\ \mathrm{E} - (\mathrm{E}_{1^{\mathrm{st}}} + \mathrm{E}_{2^{\mathrm{nd}}}) \tag{11}$$

Table 3
Simulation parameters.

Parameter	Value
# blocks	100,000
# miners	[100, 200, 300]
# second winners	$[1, 2, \dots, 10]$
Hashing power dist.	% of miners having 50% of total hash power [2%, 5%, 10%, 20%, 50%]

7.2. Experimental setup

In order to determine the time needed to select k runners-up and thus, calculate the energy spent in the first and the second round, we basically used the inverse function of the CDF in Eq. (6) and feed it different probability values p from a Uniform$(0, 1)$ distribution to generate the blocks inter-arrival times t:

$$t = -\frac{1}{\lambda}\log(1 - p) \tag{12}$$

The time when a runner-up finds a valid block, during the first round, can be estimated as follow:

$$t = -\frac{1}{\lambda(1 - h_{prev})}\log(1 - p) \tag{13}$$

where h_{prev} is the sum of the hashing power of all its predecessor runners-up including the first winner of the round. The time t is increasing for every newer runner-up as the ratio of the total network power is decreasing $(1 - h_{prev})$.

We conduct extensive simulation and average the power saving in Green-PoW over 100,000 blocks. We consider three network sizes with 100, 200, and 300 miners, and different hashing power using Uniform and Normal distribution. Table 3 summarizes the simulation parameters used to assess power saving.

7.3. Results

Figs. 5 and 6 show the impact of the number of runners-up and the size of the network on the total energy consumption in Green-PoW. Fig. 5 illustrates the ratio of power saving in Green-PoW with respect to the original PoW when varying the number of second round contenders for different network sizes. When only one node mines the block in the second round, the saving power is nearly 50% regardless of the size of the network. However, for a larger number of winners, the saving drop to nearly 32%, 41%, and 44% for networks of 100, 200, and 300 nodes, respectively. We also evaluate the total energy consumption in PoW and Green-PoW during the first and the second round and plot the results in Fig. 6. For a network of 100 miners, as shown in the figure, in Green-PoW the energy consumption during the second round is nearly 10% of that of the first round; such dramatic energy saving is due to the fact that only few nodes are participating in the mining process during the second round. In PoW, the average energy consumption is almost constant and is 8–10 times more than the second round of Green-PoW. Green-PoW consumes more energy than PoW in the first round since the nodes continue mining the same block in order to determine the runners-up. Nonetheless, the average of the first and second rounds is about 30–50% less than PoW.

We also assess in Fig. 7 the impact of a different distribution of the hashing power on the energy-saving in Green-PoW. We consider a network of 200 miners and engage 5 nodes to mine in ρ^2. We distribute the hashing power among network miners by varying the percentage of miners that hold 50% of the total network hash power, while assuming the remaining power is equally distributed among the other 50% of the network. When 50% of the network, i.e., 100 miners, equally hold 50% of the hashing power, this means that all miners have exactly the same portion of hashing power (0.5%). As illustrated in the figure, when a small portion of miners (2%) holds most of the hashing power (50%),

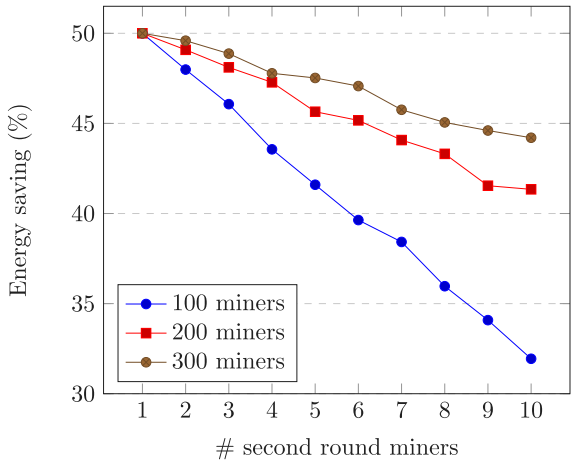


Fig. 5. Energy saving ratio Vs. number of second round miners.

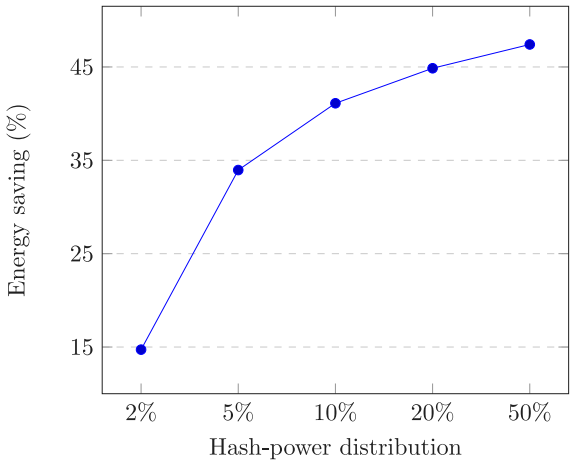


Fig. 7. Energy saving ratio Vs. hash-power distribution.

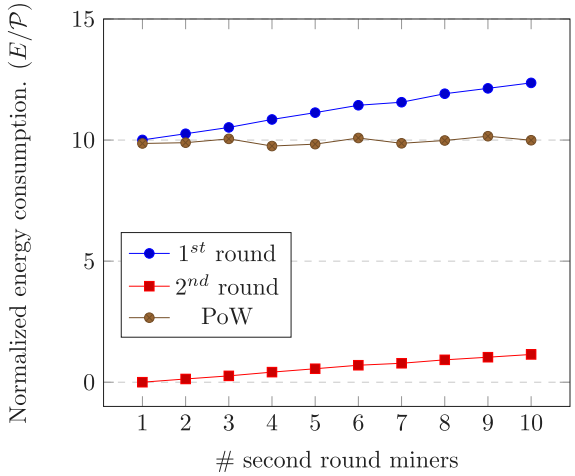


Fig. 6. Normalized energy consumption Vs. number of second round miners.

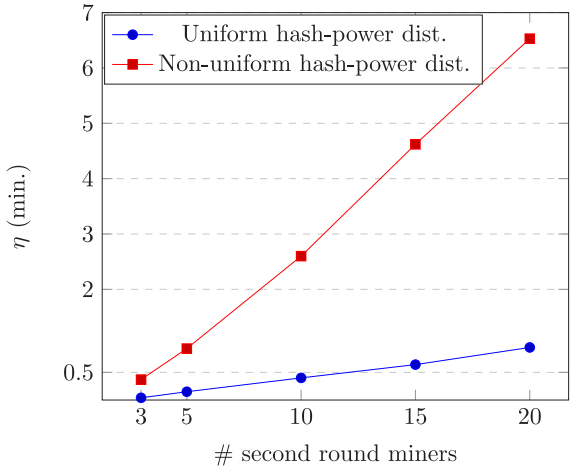


Fig. 8. Average time (η) between the first and last considered runner-up to be include in M_i^2.

the energy-saving in Green-PoW is minimal, however, when the power is equally distributed among miners, Green-PoW achieves its maximal saving. This is mainly due to the fact that the energy-saving in Green-PoW depends on the mining power of the second round contenders. When some of them have high power, per Eq. (4) the mining difficulty will be increased and consequently, more energy needs to be spent in order to find the second round block. Vise versa, when they have small hashing power, less energy will be consumed in order to find the block in the second round as the mining difficulty will be reduced.

7.4. Time-out and η selection

As we discussed in Section 4, the time-out and η are two important parameters that help in striking a good balance between system liveness and energy-efficiency. The η parameter defines the additional time a particular node needs to spend in mining during ρ_i^1 after hearing from the first considered runner-up. This time can be set as a function of the number of miners that we want to have in the second round, i.e., a function of $|\mathcal{M}_i^2|$. To capture the effect of η, we plot in Fig. 8 the time needed in order to have a specific size of $\mathcal{M}_i^2$. We consider the same simulation parameters as before, and we plot the time between the first and last considered runner-up, when having, 3, 5, 10 15, and 20 miners in the second round. We also consider different distributions of the hashing power in the network. In the case of uniformly distributed hash power among miners, the value of η does not increase much with the number of second round miners; however, when the distribution is not uniform, specifically, when 50% of the power is held by only 5% of the miners, η increases significantly. This is because more time is needed to wait for less-powerful nodes to mine a block and be able to join other miners in ρ_i^2.

We also plot in Fig. 9 the required time for a block to be mined in the second round. As discussed previously, the inter block generation (mining) time follows Exponential distribution with the same rate parameter λ (1/10 in Bitcoin). Using Eq. (13) we plot the mining time between two consecutive blocks (time between the first round block and second round block) for different probability, and consider two values of λ; i.e., 1/10 and 1/5. A safe time-out can be chosen as the duration of time ensuring that a block will be mined with a high probability. For instance, for a block to be mined with a probability between [0.7, 0.9] a network with $\lambda = 1/10$ needs to wait for a time between [12, 23] min. Whereas a network with $\lambda = 1/5$ needs to wait for a time between [6, 12] min. Therefore, a typical time-out can be chosen from this interval. Note that having different hash power distribution will not affect the block generation time, as the defined difficulty D_i^2 in Eq. (4) ensures that a block is mined at a constant rate on the average (10 min in Bitcoin).

The η parameter is important to ensure that a minimum number of runners-up will be competing in the second round so that a block most probably will be generated before the specified time-out value passes. On the other hand, the value of η should not be too large

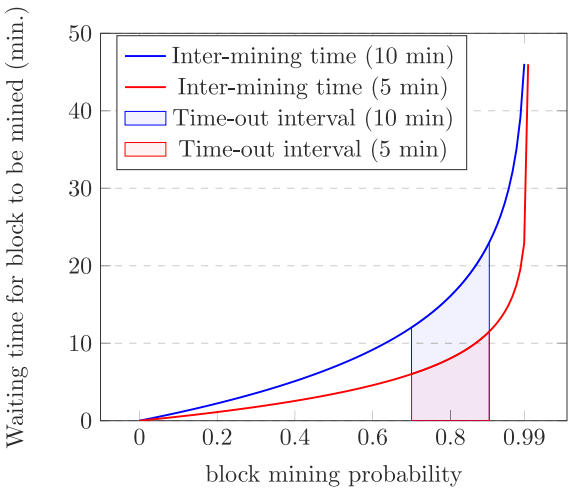


Fig. 9. Time to wait for a block to be mined in the second round Vs. the corresponding probability.

in order to avoid wasting the miner's energy unnecessarily. Based on the conducted experiments, the network hash-power distribution needs to be considered before setting the value of η. Regarding the time-out parameter, its selection depends only on the rate at which blocks are generated (e.g., 1 block every 10 min in Bitcoin); yet it should not be too long to ensure that blocks are on average generated at the same rate. We have used the Bitcoin network as a baseline; for other blockchain networks, the designers can reference the relative performance to Bitcoin to decide on appropriate setting.

8. Conclusion

In this paper, we have proposed a novel and energy-efficient consensus algorithm, called Green-PoW, for a public blockchain. In our algorithm, the overall energy consumption during mining is reduced by up to 50% compared to the original PoW. Green-PoW achieves its goal by taking advantage of the energy spent during one block mining to also elect a small number of miners that will exclusively mine the next block. In Green-PoW, time is divided into epochs that consist of two mining rounds. The first round is similar to mining in the original PoW with the exception that a small additional power is spent in order to qualify a subset of miners to exclusively contend in the second round. In the second round, where most of the mining power is saved, only the elected miners during the previous round have the right to participate and compete for forming a new block. To validate the performance of Green-PoW, extensive simulations have been conducted to mainly assess the energy saving compared to the original PoW. The results demonstrated the efficiency of the solution where up to 50% of the mining energy can be saved for a large network with equally distributed hashing power. We also have studied key security properties and shown the advantage of Green-PoW in reducing fork occurrences, the effect of censorship attack, and mining centralization.

CRediT authorship contribution statement

Noureddine Lasla: Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Writing – review & editing. **Lina Al-Sahan:** Validation, Writing – original draft, Writing – review & editing. **Mohamed Abdallah:** Investigation, Writing – review & editing. **Mohamed Younis:** Conceptualization, Writing – reviewing & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, Tech. rep., 2008.
[2] S. Deetman, Bitcoin could consume as much electricity as Denmark by 2020. 2016, 2017, URL https://Motherboard.vice.com/en_us/article/bitcoin-could-consume-as-much-electricity-as-denmark-by-2020. Retrieved March 18.
[3] C. Mora, R.L. Rollins, K. Taladay, M.B. Kantar, M.K. Chock, M. Shimada, E.C. Franklin, Bitcoin emissions alone could push global warming above 2 C, Nature Clim. Change 8 (11) (2018) 931–933.
[4] I.M. Ali, M. Caprolu, R. Di Pietro, Foundations, properties, and security applications of puzzles: A survey, ACM Comput. Surv. (ja) http://dx.doi.org/10.1145/3396374.
[5] X. Li, P. Jiang, T. Chen, X. Luo, Q. Wen, A survey on the security of blockchain systems, Future Gener. Comput. Syst. 107 (2020) 841–853.
[6] C. Badertscher, P. Gaži, A. Kiayias, A. Russell, V. Zikas, Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 913–930.
[7] S. Bano, A. Sonnino, M. Al-Bassam, S. Azouvi, P. McCorry, S. Meiklejohn, G. Danezis, Consensus in the age of blockchains, 2017, arXiv preprint arXiv:1711.03936.
[8] A. Miller, A. Juels, E. Shi, B. Parno, J. Katz, Permacoin: Repurposing bitcoin work for data preservation, in: 2014 IEEE Symposium on Security and Privacy, IEEE, 2014, pp. 475–490.
[9] F. Zhang, I. Eyal, R. Escriva, A. Juels, R. Van Renesse, {REM}: REsource-efficient mining for blockchains, in: 26th {USENIX} Security Symposium ({USENIX} Security 17), 2017, pp. 1427–1444.
[10] I. Bentov, A. Gabizon, A. Mizrahi, Cryptocurrencies without proof of work, in: International Conference on Financial Cryptography and Data Security, Springer, 2016, pp. 142–157.
[11] A. Baliga, Understanding blockchain consensus models, Persistent 2017 (4) (2017) 1–14.
[12] Ethereum 2.0 specifications, 2019, https://github.com/ethereum/eth2.0-specs, Accessed: 2019.
[13] Y. Shahsavari, K. Zhang, C. Talhi, A theoretical model for fork analysis in the bitcoin network, 2019.
[14] L. Alsahan, N. Lasla, M. Abdallah, Local bitcoin network simulator for performance evaluation using lightweight virtualization, in: 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), IEEE, 2020, pp. 355–360.
[15] M. Ball, A. Rosen, M. Sabin, P.N. Vasudevan, Proofs of useful work, IACR Cryptol. ePrint Arch. 2017 (2017) 203.
[16] A. Shoker, Sustainable blockchain through proof of exercise, in: 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), IEEE, 2017, pp. 1–9.
[17] S. King, Primecoin: Cryptocurrency with prime number proof-of-work, 2013, p. 6, July 7th 1.
[18] F. Bravo-Marquez, S. Reeves, M. Ugarte, Proof-of-learning: A blockchain consensus mechanism based on machine learning competitions, in: 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON), IEEE, 2019, pp. 119–124.
[19] X. Qu, S. Wang, Q. Hu, X. Cheng, Proof of federated learning: A novel energy-recycling consensus algorithm, 2019, arXiv preprint arXiv:1912.11745.
[20] P. Daian, I. Eyal, A. Juels, E.G. Sirer, (Short paper) Piecework: Generalized outsourcing control for proofs of work, in: International Conference on Financial Cryptography and Data Security, Springer, 2017, pp. 182–190.
[21] I. Eyal, A.E. Gencer, E.G. Sirer, R. Van Renesse, Bitcoin-ng: A scalable blockchain protocol, in: 13th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 16), 2016, pp. 45–59.
[22] I. Tsabary, A. Spiegelman, I. Eyal, Heb: Hybrid expenditure blockchain, 2019, arXiv preprint arXiv:1911.04124.
[23] E. Heilman, A. Kendler, A. Zohar, S. Goldberg, Eclipse attacks on bitcoin's peer-to-peer network, in: 24th {USENIX} Security Symposium ({USENIX} Security 15), 2015, pp. 129–144.
[24] J. Garay, A. Kiayias, N. Leonardos, The bitcoin backbone protocol: Analysis and applications, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2015, pp. 281–310.
[25] R. Pass, L. Seeman, A. Shelat, Analysis of the blockchain protocol in asynchronous networks, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2017, pp. 643–673.
[26] J.R. Douceur, The sybil attack, in: International Workshop on Peer-to-Peer Systems, Springer, 2002, pp. 251–260.
[27] F. Winzer, B. Herd, S. Faust, Temporary censorship attacks in the presence of rational miners, in: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE, 2019, pp. 357–366.
[28] B. Kaiser, M. Jurado, A. Ledger, The looming threat of china: An analysis of chinese influence on bitcoin, 2018, arXiv preprint arXiv:1810.02466.
[29] C. Ye, G. Li, H. Cai, Y. Gu, A. Fukuda, Analysis of security in blockchain: Case study in 51%-attack detecting, in: 2018 5th International Conference on Dependable Systems and their Applications (DSA), IEEE, 2018, pp. 15–24.

[30] C. Decker, R. Wattenhofer, Information propagation in the bitcoin network, in: IEEE P2P 2013 Proceedings, IEEE, 2013, pp. 1–10.
[31] D. Kraft, Difficulty control for blockchain-based consensus systems, Peer-to-Peer Netw. Appl. 9 (2) (2016) 397–413.
[32] D. Fullmer, A.S. Morse, Analysis of difficulty control in bitcoin and proof-of-work blockchains, in: 2018 IEEE Conference on Decision and Control (CDC), IEEE, 2018, pp. 5988–5992.
[33] R. Bowden, H.P. Keeler, A.E. Krzesinski, P.G. Taylor, Block arrivals in the bitcoin blockchain, 2018, arXiv preprint arXiv:1801.07447.

Noureddine Lasla received the B.Sc. and M.Sc.degrees in 2005 and 2008 from the University of Science and Technology Houari Boumediene (USTHB) and the Superior Computing National School (ESI), respectively, and received his Ph.D. degree in 2015 from the USTHB, all in computer science. He was a Researcher at CERIST research center, Algiers, Algeria from 2010 to 2017, and then a Research Scientist with the Qatar Mobility Innovations Center (QMIC), Doha, Qatar, from 2017 to 2018. Since 2018, he has been a Postdoctoral Research Fellow in the Division of Information and Computing Technology at Hamad Bin Khelifa University, Qatar with expertise in distributed systems, network communication and cyber security.

Lina Al-Sahan received her master degree in 2020 from Hamad Bin Khalifa University (HBKU), Doha, Qatar. Since 2019, she has been a Research Assistant in the Division of Information and Computing Technology at Hamad Bin Khelifa University. Her main research interests include cyber security and networking.

Mohamed Abdallah received his B.Sc degree from Cairo University in 1996. He received his M. Sc. and Ph.D. degrees from the University of Maryland at College Park in 2001 and 2006, respectively. From 2006 to 2016, he held academic and research positions at Cairo University and Texas A&M University at Qatar. Currently, he is a founding faculty member with the rank of Associate Professor at College of Science and Engineering at Hamad bin Khalifa University (HBKU). His current research interests include wireless networks, wireless security, smart grids, optical wireless communication and Blockchain applications for emerging networks. Dr. Abdallah has published more than 150 journals and conferences and four book chapters, and co-invented four patents. Dr. Abdallah is the recipient of the Research Fellow Excellence Award at Texas A&M University at Qatar in 2016, the best paper award in multiple IEEE conferences including IEEE BlackSeaCom 2019 and the IEEE First Workshop on Smart Grid and Renewable Energy in 2015, and the Nortel Networks Industrial Fellowship for five consecutive years, 1999–2003. Dr. Abdallah professional activities include an associate editor for IEEE Transactions on Communications and IEEE Open Access Journal of Communications, Track co-chair of the IEEE VTC Fall 2019 conference, a technical program chair of the 10th International Conference on Cognitive Radio Oriented Wireless Networks, and a technical program committee member of several major IEEE conferences.

Mohamed F. Younis is currently a professor in the department of computer science and electrical engineering at the University of Maryland Baltimore County (UMBC). He received his Ph.D. degree in computer science from New Jersey Institute of Technology, USA. Before joining UMBC, he was with the Advanced Systems Technology Group, an Aerospace Electronic Systems R&D organization of Honeywell International Inc. While at Honeywell he led multiple projects for building integrated fault tolerant avionics and dependable computing infrastructure. He also participated in the development of the Redundancy Management System, which is a key component of the Vehicle and Mission Computer for NASA's X-33 space launch vehicle. Dr. Younis' technical interest includes network architectures and protocols, wireless sensor networks, embedded systems, fault tolerant computing, secure communication and distributed real-time systems. He has published about 300 technical papers in refereed conferences and journals. Dr. Younis has seven granted and three pending patents. In addition, he serves/served on the editorial board of multiple journals and the organizing and technical program committees of numerous conferences. Dr. Younis is a senior member of the IEEE and the IEEE communications society.