Research Article

Energy Efficient Multiprocessing Solo Mining Algorithms for Public Blockchain Systems

Zeeshan Raza,1 Irfan ul Haq1, Muhammad Muneeb1, and Omair Shafiq2

1Department of Computer and Information Sciences (DCIS), Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad 45650, Pakistan
2School of Information Technology, Carleton University, Ottawa, ON, Canada

Correspondence should be addressed to Zeeshan Raza; zeeshan.raza@yahoo.com

Received 30 March 2021; Revised 29 June 2021; Accepted 30 August 2021; Published 31 October 2021

Academic Editor: Jiwei Huang

Copyright © 2021 Zeeshan Raza et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Blockchain as a decentralized distributed ledger is revolutionizing the world with a secure design data storage mechanism. In the case of Bitcoin, mining involves a process of packing transactions in a block by calculating a random number termed as a nonce. The nonce calculation is done by special nodes called miners, and all the miners follow the Proof of Work (PoW) mining mechanism to perform the mining task. The transaction verification time in PoW-based blockchain systems, i.e., Bitcoin, is much slower than other digital transaction systems such as PayPal. It needs to be quicker if a system adapts PoW-based blockchain solutions, where there are thousands of transactions being computed at a time. Besides this, PoW mining also consumes a lot of energy to calculate the nonce of a block. Mining pools resulting into aggregated hashpower have been a popular solution to speed up the PoW mining, but they can be attacked by using different types of attacks. Parallel computing can be used to speed up the solo mining methods by utilizing the multiple processes of the contributing processors. In this research, we analyze various consensus mechanisms and see that the PoW-based blockchain systems have the limitations of low transaction confirmation time and high energy consumption. We also analyze various types of consensus layer attacks and their effects on miners and mining pools. To tackle these issues, we propose parallel PoW nonce calculation methods to accelerate the transaction verification process especially in solo mining. We have tested our techniques on different difficulty levels, and our proposed techniques yield better results than the traditional nonce computation mechanisms.

1. Introduction

Blockchain has introduced a new transaction/data storage mechanism that provides better transparency, is more secure, enables business between untrusted parties, and helps in reducing fraud [1]. The use of blockchain technology is not limited to the cryptocurrencies now but is also being used in other industries like transportation, automotive industry, supply chain management [2], healthcare [3, 4], and agriculture sector [5, 6]. Blockchain offers advantages like transparency and immutability, but it also has some limitations specially when the PoW is used in solo mining. Proof of Work (PoW) [7, 8] is one of the first blockchain mining algorithms popularized by Bitcoin, and now many blockchain technologies use it for transactions confirmation.

The principle behind PoW is to solve a mathematical puzzle, and a reward is given to the miners who solve this complex problem. In PoW mining, miners need to pack transactions in a block and use a brute-force mechanism to find a nonce, which satisfies a given difficulty level. All the miners are given equal opportunity to find the nonce, and in case of success, they are given mining rewards as well as transaction fees.

As mentioned in the literature, such as [9], the mining process works approximately as shown in equation (1). Symbol + is used to denote the concatenation of strings. The cryptographic problem of computing a double SHA256 hash has to be solved by a miner denoted by M.

\[ s = SHA256(SHA256(n + h + s' + x)) \] (1)
Here, $s < x$, where $n$ is a nonce value, $h$ is a double SHA256 hash over the transactions that miner $M$ intends to incorporate into its next block $B$ (in the case of Bitcoin, $h$ is the Merkle root), $s'$ is the solution for the block at the head of the blockchain at miner $M$, and $x$ is the current level of difficulty (leading zeros). If $s \geq x$, then $n$ is incremented, and $s$ is recomputed up to the point that a solution is found with $s < x$. If miner $M$ finds a valid solution $s$, miner $M$ appends block $B$ to the blockchain at miner $M$ and broadcasts $(B, n, h, x)$ to the blockchain peer-to-peer network.

Nonce values are numeric values that are concatenated with block data before performing the mining task. The block data is then given to a hashing function SHA256 to get a hash of a targeted difficulty level. If the targeted difficulty level is not achieved, then the nonce value is incremented and again concatenated with block data before applying SHA256. The whole process continues until a hash that satisfies the given difficulty level is achieved. Once nonce is found, the block is broadcasted to the blockchain network.

The nonce finding process in typical PoW is described in Figure 1.

PoW is a complex task that takes much more time and consumes a huge amount of energy [10, 11]. First blockchain application based on PoW is Bitcoin, and according to Bitcoinist “the average cost to mine a bitcoin in Serbia is about 3,100 US Dollars” [11]. It is also mentioned that “mining of a single bitcoin block consumes energy, which is enough to power more than 28 homes in the United States for one day” [11].

A popular method to increase the chances of success is to make a pool of mining resources by collaborating with other miners and dividing the reward consequently. The consolidated pool of resources builds more hashpower and increases their chance to succeed, but it decreases the chances of success if a miner wants to perform solo mining. Besides, these attackers can attack miners and mining pools as mentioned in Table 2 to influence the public blockchain systems.

In this paper, we propose algorithms to parallelize nonce computation of the PoW solo mining by assigning different ranges of nonce values to multiple processes. The proposed algorithms not only speed up the PoW mining process but also reduce energy consumption. Rather than applying pure brute force through aggregated hashpower, our proposed algorithms based on parallel computing [29] follows the principle of divide and conquers and subdivides a complex task into subtasks, which are then processed in parallel on a single machine. All the processes try to solve the subtasks of nonce computation in parallel at the same time, and once a processor finds a solution, it is communicated to all other processes. The proposed parallel PoW is much faster than the sequential version and saves energy, thus reducing the carbon footprint. Scientific contributions of this paper include the following:

Interleaved algorithm: multiple processes try to find the suitable nonce from different nonce intervals/ranges.

Progressive algorithm: multiple processes are assigned consecutive nonce values, and in case of failure in an iteration, jump to the next nonce value (a jump of 8 values in case of 8 processes) in next iteration and test the proposed parallel algorithms on different difficulty levels.

The rest of the paper is organized as follows: The literature review is discussed in Section 2, and the proposed solo mining algorithms are given in Section 3. Section 4 presents the performance analysis of proposed techniques, and the paper is concluded in Section 5.

2. Literature Review

Scalability is the main issue in PoW-based public blockchain systems, and researchers are working on different solutions based on hashing algorithms and consensus mechanisms to tackle this issue. Some related work typically related to the scalability issue of PoW is discussed in this section.

The work in [30] introduced a new protocol to improve blockchain performance by changing the chain data structure to Graph Chain. Their parallel mining mechanism is focused on the selection of leaders among $M$ miners. By solving a puzzle, miners can become leaders, and leaders are assigned duty for a period. The authors suggested that transaction confirmation speed can be increased by allowing more leaders to do PoW mining tasks in parallel. They only focused on the miners’ selection process and did not work on the improvement of nonce computation.

In [31], the authors proposed an accelerated process of PoW mining. They built a process that carries out selection of a manager, work distribution, and a reward system. In their method, miners can use the same transaction data (same transaction hash) but cannot use the same nonce values; thus, no multiple miners can carry out the same mining task. Same as the study in [30], the focus of [31] is on miners selection.

The work in [32] also proposed a parallel mining protocol. In their protocol, each transaction is connected to at least two transactions that are verified, and all miners need to verify any new transactions. Their focus was also on miners but not on the actual mining process.

2.1. Blockchain Consensus Algorithms

Consensus algorithms [33, 34] are decision-making processes for a group, where participants of a group support the decision that is best for everyone. To overcome the scalability issue of PoW, other consensus mechanisms are also introduced, which are discussed with the help of the comparative analysis provided in Table 1.

Each consensus mechanism shown in Table 1 has its strengths and weaknesses and could be used in different types of blockchain systems. But the most common consensus algorithm in public blockchains is PoW, which has time and energy consumption issues associated with it. In this research, our focus is on scalability of such blockchain systems, which intend to use PoW specially in solo mining.
2.2. Attacks on the Blockchain Consensus Layer. Security is an essential feature in a variety of systems and is very important issue in blockchain-based systems. There are a variety of attacks, which are used by the attacker to attack the blockchain data, blockchain networks, and their communication, consensus mechanism, miner/mining pools, and smart contracts [56, 57], i.e., Malleability Attack [58], Quantum Attack [59], Distributed Denial of Service (DDoS) Attack [60], Eclipse Attack [61], Sybil Attack [62], Border Gateway Protocol (BGP) Attack [63], Phishing Attack [64], Selfish Mining Attack [65], Consensus Delay Attack, Double Spending Attack [15], Integer Overflow Attack [66], and Short Address Attack [67].

As our research work is based on consensus mechanism and mining, we focus on the consensus layer attacks and highlight the research work done to mitigate them. In selfish mining attack [65], a selfish miner in a mining pool uses selfish strategies to get more rewards than the honest miners. The selfish miner tries to withhold a validated block and stops its broadcasting to mining pool network but continues to mine the next block. In this way, a selfish miner demonstrates more PoW than other miners of the mining pool. Doubles pending [15] is a digital cash scheme flaw, in which the attacker spends a digital token more than once, as a digital token is associated with a digital file that can be duplicated. In bribery attack, some miners act rationally and accept bribes from attackers to increase their reward. Refund attack is a Payment Protocol attack that could affect the BIP70 protocol merchants [22]. In block withholding attack, a miner who has mined a block does submit it but abandons it and causes a mining pool and loses all bitcoin rewards. In balance attack [26], attackers select miners that have the same mining capabilities, and rather than entering into mining competition with other miners, they defer the messages between the selected miners. Consensus Delay is the time between the block propagation to block storage after the consensus [68]. In a supply chain system, attackers can use consensus delay attack to interrupt the information propagation and, hence, can easily create a double-spent transaction by refuting the vote of an authorized user [13].

Table 1: Survey of blockchain consensus mechanisms.

<table>
<thead>
<tr>
<th>Consensus mechanisms</th>
<th>Pros</th>
<th>Cons</th>
<th>Blockchain systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Proof of work” [10]</td>
<td>Less risk of 51 percent attack and better security</td>
<td>Energy consumption, low transaction confirmation time and centralization of miners</td>
<td>Bitcoin [7], Litecoin [35] and Ethereum [36]</td>
</tr>
<tr>
<td>“Proof of stake” [37]</td>
<td>Energy efficient, more decentralized and reduces the threat of 51 percent attack</td>
<td>Not fully decentralized and nothing at stake problem (we do not lose anything by behaving badly)</td>
<td>PIVX (private instant verified transaction), NavCoin [38], ARDOR, and Stratis [39]</td>
</tr>
<tr>
<td>“Leased proof of stake” [40]</td>
<td>Fair usage and lease coins</td>
<td>Decentralization issue</td>
<td>Waves [41]</td>
</tr>
<tr>
<td>“Delegated proof of stake” [42]</td>
<td>Energy efficient and scalable</td>
<td>Double spending attack</td>
<td>Lisk [43], Ark [44], EOS [45], and BitShares</td>
</tr>
<tr>
<td>“Proof of capacity” [46]</td>
<td>Cheap, efficient, and distributed</td>
<td>Favor big fishes</td>
<td>Burstcoin</td>
</tr>
<tr>
<td>“Proof of importance” [47]</td>
<td>Transaction partnership and vesting/harvesting</td>
<td>Decentralization issue</td>
<td>NEM (new economy movement) [48]</td>
</tr>
<tr>
<td>“Proof of activity” [49]</td>
<td>Reduces the threat of 51 percent attack</td>
<td>Greater energy consumption</td>
<td>Decred [50] and Espers [51]</td>
</tr>
<tr>
<td>“Proof of elapsed time” [52]</td>
<td>Cheap participation</td>
<td>Need for specialized hardware and not good for public blockchain</td>
<td>Hyperledger sawtooth [53]</td>
</tr>
<tr>
<td>“Practical byzantine fault tolerance” [54]</td>
<td>Less energy consumption</td>
<td>Feasible for a small group of nodes, communication gap and sybil attack</td>
<td>Hyperledger fabric [55]</td>
</tr>
</tbody>
</table>

Figure 1: Typical proof of work mining mechanism.
In Table 2, we did a comparative analysis of different attacks, which could eventually influence the working of consensus mechanism and mining process.

Our research work could be beneficial in blockchain systems, which avoids the mining pools but rather wants fast solo mining protocols. If a system adopts solo mining, then definitely it will have to bear the consequences of more time and energy consumption. We proposed two parallel commuting based techniques to make the solo mining faster and cheaper in terms of energy consumption.

3. Proposed Parallel Algorithms for Nonce Calculation

Two main performance indicators of PoW include transaction verification time and energy consumption and both are tackled in our proposed parallel processing-based algorithms. In our proposed techniques, we did not alter the core working of PoW, but building upon the existing research, we proposed different nonce value selection methods before performing the PoW mining on multiple processes.

First of all, the performance of PoW mining as described in Algorithm 1 is tested. Algorithm 1 is tested on different difficulty levels, and the time taken to compute the nonce value on each difficulty level is computed as shown in 3; then, we compared it with our proposed parallel PoW techniques. For experimental purposes, the string “Parallel Computing” is used in place of Merkle Tree (in case of bitcoin), as even a simple string is enough to validate performance of proposed solution. We did not change the string in the whole process but varied the difficulty levels to calculate the mining time at each difficulty level for PoW and parallel PoW techniques.

The pseudocode of the PoW is given in Algorithm 1. In our naive implementation, a block of blockchain contains the following fields:

- nonce: int
- index: int
- blockData: string
- timeStamp: dateTime
- hash: string
- previousHash: string

We have computed the results on 8 difficulty levels on available Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz CPU machine as well as on a machine with built-in NVIDIA GeForce RTX 2060 GPU.

Table 3 is generated by implementing Algorithm 1 on the CPU; it contains difficulty levels, nonce values, and time taken in seconds to perform the mining task on constant block data.

As shown in Table 3, there is a significant amount of change in the nonce values and time taken in seconds as we increase the difficulty levels from 1 to 8. Time is almost constant from difficulty levels 1 to 4 but rapidly changes from difficulty levels 5 to 8, which shows that PoW mining becomes difficult on higher levels of difficulty.

To address the time consumption issue as shown in Table 3, we have introduced two parallel processing-based PoW mining algorithms and done multithreading and multiprocessing in both methods as shown in Figure 2.

3.1. Interval/Interleaved Algorithm. In the proposed interleaved algorithm, the nonce values are divided into multiple ranges/chunks, and the multiple processes/threads perform mining on different nonce ranges in a parallel fashion. The workflow and proposed mechanism of the interleaved approach are shown in Figure 3.

The pseudocode of the interleaved approach is given in Algorithm 2.

3.2. Multiprocessing Interleaved Approach. In this approach, we make 8 nonce value ranges as given in Figure 4. These ranges are given to multiple CPU processes to compute the nonce value. Once a process finds a nonce that satisfies the difficulty level, other processes stop working. The nonce value is then propagated to the blockchain network, and in case of acceptance, the reward is given to the miner. The results of the multiprocessing interleaved approach are given in Table 4.

3.3. Multithreading Interleaved Approach. We performed the mining task by giving subarrays to multiple threads to calculate the hash of a given difficulty level. Once a thread is successful in getting the resultant nonce, the whole process stops. Results computed by the multithreading interleaved approach are given in Table 5.

3.4. Progressive Approach. In this proposed approach, the PoW mining process is divided into 8 processes, and each process is assigned nonce values from 1 to 8. If a nonce value does not satisfy the desired difficulty level, it is updated by adding 8 in the previous value, and the mining process is repeated until one of the processes gets the desired nonce value. Once the solution is found, all other processes stop working, and the nonce value is propagated to the whole network. The workflow and proposed mechanism of parallel proof of work progressive approach are given in Figure 5.

The pseudocode of a progressive approach is given in Algorithm 3.

3.5. Multiprocessing Progressive Approach. In the multiprocessing progressive approach, the PoW mining process is assigned to 8 different processes. Each process initially assigned the nonce values from 1 to 8, and in the iteration, it updates the nonce value by adding 8 into it. If a process finds a nonce value that satisfies the given difficulty level, all other processes stop working, and the nonce value is propagated to the blockchain network. Table 6 contains the results of a multiprocessing progressive approach.

3.6. Multithreading Progressive Approach. In the multithreading progressive approach, 8 different threads perform the mining. Table 7 contains the results computed by the multithreading progressive approach.
3.7. **GPU-Based Interleaved Approach.** For GPU-based results and analysis, we picked the available laptop machine with built-in NVIDIA GeForce RTX 2060 GPU and calculated the time consumption and energy consumption of both PoW and our best performing multiprocessing interleaved algorithm. The selection of nonce value ranges and increment in nonce values is done on CPU, while GPU is used for parallel implementation of the SHA256 algorithm. Figure 6 elaborates the process.
Initialization

Block Data = Block Index + Merkle Tree + Previous Hash + Timestamp + …
NonceRangeSet = Range1, Range2, …, RangeN.

BlockHashes

BlockHash1 = SHA256 (Block Data + First Nonce Value in Range1)
BlockHash2 = SHA256 (Block Data + First Nonce Value in Range2)

\[ \text{}; \]
BlockHashN = SHA256 (Block Data + First Value in RangeN).

Process1

while BlockHash1 does not satisfy the difficulty level do.

BlockHash1 = SHA256 (Block + (Nonce Value + 1)).

Process2

while BlockHash2 does not satisfy the difficulty level do.

BlockHash2 = SHA256 (Block + (Nonce Value + 1)).

\[ \text{}; \]
ProcessN while BlockHashN does not satisfy the difficulty level do.

BlockHashN = SHA256 (Block + (Nonce Value + 1)).

Solution

Broadcast the solution to the Blockchain network (Nonce value, the Merkle root and the Mined Block).

Algorithm 2: Interleaved approach.

Figure 3: Interleaved approach workflow and mechanism.

Table 4: Multiprocessing interleaved approach results (CPU).

<table>
<thead>
<tr>
<th>Difficulty level</th>
<th>Nonce value</th>
<th>Time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5200000003</td>
<td>0.0099937915802100195</td>
</tr>
<tr>
<td>2</td>
<td>6500000027</td>
<td>0.009989261527197266</td>
</tr>
<tr>
<td>3</td>
<td>5200000299</td>
<td>0.0039988136322012484</td>
</tr>
<tr>
<td>4</td>
<td>5200000299</td>
<td>0.0039975643157958984</td>
</tr>
<tr>
<td>5</td>
<td>5200000299</td>
<td>0.003997802734375</td>
</tr>
<tr>
<td>6</td>
<td>5200000299</td>
<td>0.003997802734375</td>
</tr>
<tr>
<td>7</td>
<td>260538310</td>
<td>2.2236204147338867</td>
</tr>
<tr>
<td>8</td>
<td>1015800503</td>
<td>462.1195707321167</td>
</tr>
</tbody>
</table>

Figure 4: Nonce value ranges.

Table 5: Multithreading interleaved approach results (CPU).

<table>
<thead>
<tr>
<th>Difficulty level</th>
<th>Nonce value</th>
<th>Time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5200000003</td>
<td>0.0059969425201416016</td>
</tr>
<tr>
<td>2</td>
<td>6500000027</td>
<td>0.005996227264404297</td>
</tr>
<tr>
<td>3</td>
<td>5200000299</td>
<td>0.03896665573120117</td>
</tr>
<tr>
<td>4</td>
<td>5200000299</td>
<td>0.08294916152954102</td>
</tr>
<tr>
<td>5</td>
<td>5200000299</td>
<td>0.04797053337097168</td>
</tr>
<tr>
<td>6</td>
<td>5200000299</td>
<td>0.07795337097168</td>
</tr>
<tr>
<td>7</td>
<td>260538310</td>
<td>7.604280948638916</td>
</tr>
<tr>
<td>8</td>
<td>1015800503</td>
<td>2235.2516202926636</td>
</tr>
</tbody>
</table>
4. Performance Analysis of Proposed Techniques

The code of the proposed interleaved and progressive techniques is done in Python programming language on both CPU and GPU machines.

4.1. Data Used for Evaluation. We generated data set by implementing the PoW and proposed parallel PoW algorithms on different difficulty level and evaluated the proposed algorithms. Table 3 for PoW and Tables 4–7 consist of the nonce values and time consumption at different difficulty levels from 1 to 8.
4.2. CPU-Based Results. On CPU, we performed the proposed parallel techniques and compared them with PoW.

4.2.1. Time Consumption. From our findings of Tables 3–5, multiprocessing interleaved approach is the best one. It performs way better than other proposed multithreading and multiprocessing approaches. It takes just 2.22 seconds as compared to 7.60 seconds in the multithreading interleaved approach and 556.78 in the traditional PoW [69] approach (with serial execution) for difficulty level 7. In the case of difficulty level 8, results are even better: 462.11 in multiprocessing interleaved approach, 536.25 in the case of multiprocessing progressive approach, and 2209.52 seconds in the case of Algorithm 1-based PoW mining.

The multiprocessing interleaved approach performs better than the multithreading progressive approach, but its performance is not up to the mark as compared to Algorithm 1-based PoW and multiprocessing progressive approach. Mining is a complex task, so any multithreading approach will not work better, but if we have a simple problem of finding a solution in any domain, we can improve performance through a multithreading interleaved approach.

The multithreading progressive approach performs better than Algorithm 1-based PoW and multiprocessing progressive approach; this is because processes can perform better on complex tasks. It can be seen in Table 6 that as the difficulty levels increase the performance of the multiprocessing progressive approach, it becomes better as it takes 536.28 seconds to mine a block of 8 difficulty level as

<table>
<thead>
<tr>
<th>Difficulty level</th>
<th>Nonce value</th>
<th>Time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>0.0019996166229248047</td>
</tr>
<tr>
<td>2</td>
<td>29</td>
<td>0.015625476837158203</td>
</tr>
<tr>
<td>3</td>
<td>1985</td>
<td>0.003113746430664062</td>
</tr>
<tr>
<td>4</td>
<td>9433</td>
<td>0.008995532989501953</td>
</tr>
<tr>
<td>5</td>
<td>166408</td>
<td>0.08794569969177246</td>
</tr>
<tr>
<td>6</td>
<td>27688014</td>
<td>14.330108404159546</td>
</tr>
<tr>
<td>7</td>
<td>260538310</td>
<td>132.9916124343872</td>
</tr>
<tr>
<td>8</td>
<td>1015800503</td>
<td>536.2866580486298</td>
</tr>
</tbody>
</table>

Table 6: Multiprocessing progressive result (CPU).

<table>
<thead>
<tr>
<th>Difficulty level</th>
<th>Nonce value</th>
<th>Time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>0.00000000000000000</td>
</tr>
<tr>
<td>2</td>
<td>29</td>
<td>0.015624523162841797</td>
</tr>
<tr>
<td>3</td>
<td>1985</td>
<td>0.03897666931152344</td>
</tr>
<tr>
<td>4</td>
<td>9433</td>
<td>0.05296826362609863</td>
</tr>
<tr>
<td>5</td>
<td>166408</td>
<td>0.889079323516846</td>
</tr>
<tr>
<td>6</td>
<td>27688014</td>
<td>38.5037682056427</td>
</tr>
<tr>
<td>7</td>
<td>260538310</td>
<td>838.6467852592468</td>
</tr>
<tr>
<td>8</td>
<td>1015800503</td>
<td>2861.7225003242493</td>
</tr>
</tbody>
</table>

Table 7: Multithreading progressive approach results (CPU).
compared to 2861.72 seconds in the multithreading progressive approach and 2209.52 seconds in traditional PoW.

In the multithreading progressive approach, the time taken for mining is almost similar to Algorithm 1-based PoW for 1–4 difficulty levels, but the performance of the multithreading progressive approach is not better for higher difficulty levels; this is because the threads work better on simple tasks and in case complex problems performance of multithreading decreases.

We can conclude from CPU processing that the multiprocessing interleaved approach is the best one and reduces the time consumption of the PoW mining process. Comparative analysis of all mentioned techniques is given in Figures 7–9.

4.2.2. Energy Consumption. Intel(R) Core(TM) i7-3770 CP consumes the power of 146.4 W wattage [70], and if we multiply 0.1464 by the time consumed in hours in the mining process, we can get the energy consumption in kWh.

We did the energy consumption comparison of Algorithm 1-based PoW, multiprocessing progressive, and multiprocessing interleaved approach. For difficulty levels
from 1 to 6, the energy consumption of all three is almost equal, but if we go to higher difficulty levels of 7 and 8, then there is a significant decrease in energy consumption in the proposed techniques. The performance of the multiprocessing progressive approach is 4 times better than PoW for 7 and 8 difficulty levels, and multiprocessing interleaved performed almost 5 times better than traditional PoW in case of energy consumption. Figure 10 elaborates the energy consumption in the CPU.

4.3. GPU-Based Results. Time consumption and energy consumption results show that the proposed multiprocessing interleaved approach is the best performing technique on both CPU and GPU.

4.3.1. Time Consumption. Similar to CPU, GPU produced the best results in the interleaved approach. We performed interleaved technique by varying the number of processes and analyzed that GPU produced better results with 32 parallel processes. Figure 11 shows the PoW and interleaved results, while Figure 12 shows the change in results with the change of the number of processes. We did the experiment with 8, 16, 24, 32, and 40 processes but obtained optimal results with 32 processes. This is because the CPU GPU synchronization in task completion for 32 processes is a perfect match, while in 8, 16, 24, and 40, one of them has to wait for the other to start its assigned tasks.

4.3.2. Energy Consumption. NVIDIA GeForce RTX 2060 GPU consumes the energy of 480 wattage [71], and by multiplying 0.480 with the time consumed (in hours) in the mining process, we can get the energy consumption in kWh.

We picked some high values for nonces to calculate energy consumption at it was almost zero for smaller nonce values. Figure 13 shows the results of the experiment of energy consumption in kWh and the same as CPU, GPU also performing 4 times better in the multiprocessing interleaved approach.

5. Conclusion

Parallel processing is used to solve numerous complex problems and can also be used in blockchain mining to make it faster. PoW mining is a famous blockchain mining protocol and is being used in many blockchain-based systems; however, due to its complex nature, it has time and energy consumption issues. To improve these issues, we built two parallel processing-based PoW solo mining algorithms, in which multiple processes solve a complex mathematical puzzle on different nonce values. Pool mining can be affected by different types of attacks; hence, in our proposed algorithms, we encourage fast solo mining instead of pool
mining. The results indicate that the proposed solo mining algorithms perform much better than sequential PoW mining. We conclude that if a public blockchain system chooses to adopt solo mining with better performance, our proposed multiprocessing interleaved algorithm could prove to be beneficial.

**Data Availability**

There is no data provided for publication in this paper.

**Conflicts of Interest**

The authors declare that they have no conflicts of interest.

**References**


[65] I. Eyal and E. G. Sirer, “Majority is not enough: bitcoin mining is vulnerable,” Financial Cryptography and Data Security,


