Proof of Game (PoG): A Proof of Work (PoW)'s Extended Consensus Algorithm for Healthcare Application

Adarsh Kumar and Saurabh Jain

Abstract Advancement in blockchain technologies during the past decade has attracted tremendous interests from academia, research community and the industry. A blockchain network is a peer-to-peer, decentralized and immutable distributed ledger system for transactional records. With an increase in the number of blockchain-based applications, it becomes a powerful technology for decentralized data processing and consensus mechanisms based blockchain networks. In this work, a single- and multiplayer bit challenging and incentivized consensus mechanisms for blockchain networks are used in proposing a "proof-of-game (PoG)" protocol for resource variant blockchain networks. Bit verifier PoG is designed to be memory dependent and CPU independent mechanism for time efficiency and resource independence. In results, it is observed that the number of blocks mined using this protocol is proportional to the number of participants associated with blocks. Further, it is observed that the priority of a blockchain increases exponentially with an increase in the number of blocks mined, and the number of blocks mined decreases exponentially with an increase in computational challenge.

Keywords Blockchain · Consensus algorithm · Proof of concepts · Trusted environments · Cryptography · Cryptocurrency

1 Introduction

Blockchain, developed by Satoshi Nakamoto in 2008, acts as a transaction ledger of the cryptocurrencies. According to [1], there are 2238 cryptocurrencies today. Among these cryptocurrencies, ten popular cryptocurrencies are bitcoin, ethereum, ZRP, litecoin, bitcoin cash, eos, binance coin, bitcoin sv, tether and stellar. Blockchain

A. Kumar (✉) · S. Jain
School of Computer Science, University of Petroleum and Energy Studies, Bidholi, Dehradun, India
e-mail: adarsh.kumar@ddn.upes.ac.in

S. Jain
e-mail: saurabh.jain@ddn.upes.ac.in

© Springer Nature Singapore Pte Ltd. 2021

D. Gupta et al. (eds.), *International Conference on Innovative Computing and Communications*, Advances in Intelligent Systems and Computing 1165,
https://doi.org/10.1007/978-981-15-5113-0_2

is constructed with a sequenced interconnection of blocks. Thus, interconnections and blocks are an integral component of a blockchain. Decentralized, transparent, open-source, autonomous, immutability and anonymity are the key elements of blockchain design. According to Zheng et al. [2], the block structure consists of a block header and block body. Block header contains the name of the block, block version number, block's hash value, previous block's hash value, consensus algorithm's target difficulty value, creation time (timestamp) of the block, Merkle root of transactions and nonce value (i.e., a random counter value). The block body contains multiple transactions and their records. A block may contain a large number of transactions depending upon block and transaction size. A transaction contains a transaction header and payload. Transaction header consists of a transaction hash value, block number/index, transaction number in the block, creation time (timestamp) of the transaction, sender and receiver's identifications and a digital signature over transaction's hash value. Payloads contain multiple data chunks. Figure 1 shows a typical structure of a large number of blocks interconnected in a blockchain. Each of these interconnected blocks has multiple transactions. For example, Fig. 1 shows that there are four participants (alice, bob, trent and carol) who want to establish transactions with the owner of a block (den). Each of the four participants used their public keys in generating the signatures and verified the transaction. Each transaction inside block contains transaction amount (TA), unspent amount (UA) and gas amount (GA). Hash of a transaction between two or more parties constitutes leaf nodes in the Merkle root tree. Parent and its sibling nodes in the Merkle root tree are computed using hashing operation over child's transaction hashes. Figure 2 shows an example of 4-level Merkle root tree formulated using similar operations. There are eight transaction messages (M_0 to M_7) whose 4-level Merkle root tree is constituted.

Figures 1 and 2 shows a simplified block and transaction structure of a blockchain. However, the consensus algorithm's difficulty level value is an important parameter for a tampered proof blockchain network. The consensus algorithm's difficulty level value is measured using various concepts. For example, proof of stake model (PSM),

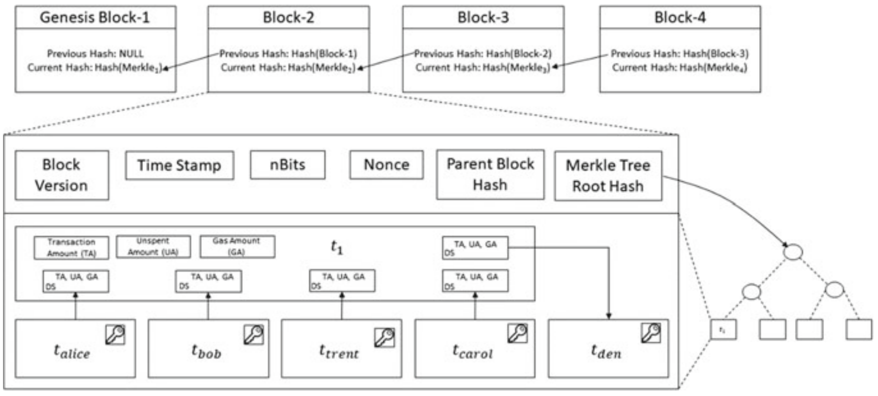

Fig. 1 Block structure and blockchain construction

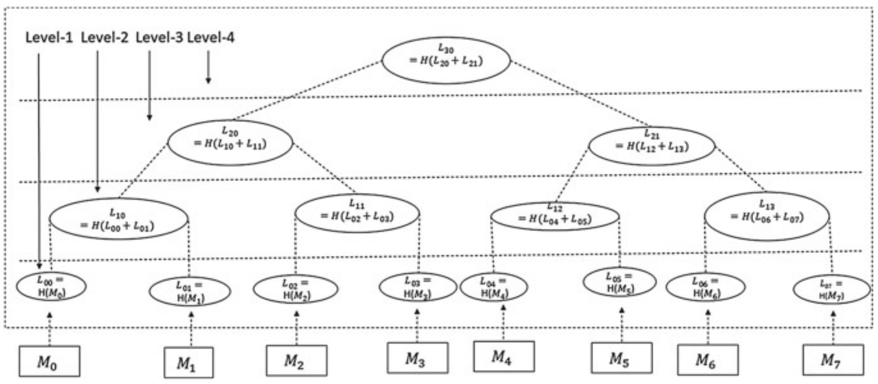
Level-1 Level-2 Level-3 Level-4
$L_{30} = H(L_{20} + L_{21})$
$L_{20} = H(L_{10} + L_{11})$
$L_{21} = H(L_{12} + L_{13})$
$L_{10} = H(L_{00} + L_{01})$
$L_{11} = H(L_{02} + L_{03})$
$L_{12} = H(L_{04} + L_{05})$
$L_{13} = H(L_{06} + L_{07})$
$L_{00} = H(M_0)$
$L_{01} = H(M_1)$
$L_{02} = H(M_2)$
$L_{03} = H(M_3)$
$L_{04} = H(M_4)$
$L_{05} = H(M_5)$
$L_{06} = H(M_6)$
$L_{07} = H(M_7)$
M_0 M_1 M_2 M_3 M_4 M_5 M_6 M_7

Fig. 2 Merkle root tree in block

proof of elapsed time (PET), Byzantine fault tolerance algorithm (BFTA), practical Byzantine fault tolerance algorithm (PBFTA), SIEVE consensus protocol (SCP), cross fault tolerance (XFT), federated Byzantine agreement (FBA), ripple consensus protocol (RCP), Stellar Consensus Protocol (SCP), etc., are popular. In this work, another consensus difficulty level concept "Proof-of-Game (PoG)" is introduced for both resourceful and resource-constraint devices in the blockchain network. PoG is derived from game theory concepts incorporated for blockchain network in recent years. PoG allows participants based on the level of game solved to create new blocks inside the blockchain. PoG can be one player or multiple player games. A single-player PoG is preferable if participants are interested to construct a blockchain in which single stakeholder is selected for permitting new blocks to join the blockchain network (similar to the specialized case of PSM). Whereas, multi-player PoG uses multi-participant playing with participant interested to insert block in the blockchain network. The level and difficulty of the game played in PoG vary with the availability of resources. High difficulty level means more resources are visible publicly and the level of confidence over the blockchain network is high.

The rest of this work is organized as follows: Sect. 2 provides state-of-the-art consensus algorithms for both private (permissioned) and public (limited permission or permissionless) blockchain technology. Here, the use of game theory in blockchain technology is explored in depth. In Sect. 3, the PoG concept is introduced for blockchain technology. Section 4 explains the PoG integrated architecture for the healthcare system. Section 5 performs statistical and experimental analyses of the proposed consensus algorithm. Finally, Sect. 6 concludes this work by summarizing the contributions.

2 Literature Survey

In the literature, several consensus algorithms are proposed [3, 4]. According to [3], special measures required for a consensus mechanism suitable for a secured and sustainable blockchain network include incentive-compatible sub-stages with tolerance to Byzantine and unfaithful faults, avoid adversaries with cumulated computational power (e.g., botnets, mining pools, etc.) and maintains a balance between processing throughput and network scalability. Consensus algorithms and game theory concepts derived for these algorithms are discussed as follows:

- *Proof of Work (PoW)*: In PoW, a cryptographic challenge instead of charging real money is put forward to the challenger. In the blockchain, if the challenger is able to solve the challenge, then it is allowed for him to add a block in the blockchain. An idea of PoW is not new in blockchain; it is used in various other applications like [5]: stopping spamming emails, preventing unauthorized access, measuring capabilities of challenger. PoW is used in various blockchain technologies. For example, the main network of Ethereum (Homestead) uses the PoW consensus model called EthHash [6]. EthHash is used to counter attacks through mining centralization in Bitcoin. In mining centralization, a large number of ASICs were used to perform hashing operations at very high rates [6]. This action allowed corporations having powerful computing resources to control the Bitcoin network by creating mining pools. PoW is a time-consuming process. In this process, each block generator has to process a cycle with nonce values and an algorithm. Back's "hash-cash" [7] is another widely known example of proof-of-work. In "hash-cash," the challenge in front of a sender (challenger) is to apply an algorithm that produces a string whose cryptographic hash starts with a certain number of zeroes. Although, it is expensive to complete this operation, but it is cheap in verification. The "hash-cash" puzzle can be applied to a specific set of email recipients through a nonce, timestamp and recipient email address. If the freshness factor generated through a nonce, timestamp and recipient email address is validated as a string value, then email is sent to specific recipients. Similarly, "hash-cash" is extended is various other applications. Automated metering to the website is another extension of "hash-chash" application [8]. In [9], PoW is used to mitigate distributed denial-of-service attacks. In [10], it is used in a P2P network for uniform user behavior using the incentive-based game. In [11], PoW is used for cloud and fog computing network with a novel mathematical model having lesser iteration to converge to the consensus solution. The proposed approach is efficient in terms of time and memory consumption and suitable for the Internet of Things (IoTs). Apart from PoW, there are various consensus algorithms like PSM, PET, BFTA, PBFTA, SCP, XFT, FBA, RCP, SCP which are popular for putting the challenge to the challenger.
- *Game Theory in Blockchain*: In recent studies [12], game theory has been applied in various scenarios of blockchain. In [12], the concept of supervised machine learning algorithm and game theory is proposed for the detection and stoppage of majority attacks in the blockchain. Supervised machine learning helps intelligent

software agents in activity-based anomaly detection that is extended for majority population size attack detection in a blockchain. Nakamoto protocol [4] for the permissionless blockchain network proposes the concept of incentivizing the participants based on token supply and transaction tipping. Stone [13] incorporated the game-theoretical analysis of consensus nodes prioritizing blocks according to its sizes. In this analysis, it is observed that consensus nodes prefer blocks of large size over small because of a long delay in incorporation and validation which may lead a block to isolation. This isolation can lead to Denial of Service (DoS) or Distributed Denial-of-Service (DDoS) attack. Large gas price for large block size is also not preferable for blockchain because of the physical constraints of a network [13]. Additionally, various game-based models have been adopted for the payoff function of miners [14–19]. For example, evolutionary game, hierarchical game, auctions, etc.

3 Proposed PoG Approach

This section gradually derives the concept of PoG. Algorithm 1 shows the steps followed in constructing a blockchain. Developed blockchain accepts block number, user data, previous block hash, timestamp and presents block's hash value. The SHA256 hashing algorithm is used for linking the blockchain. GBlock() and PreviousHASH() are two functions used for genesis block and calculating a hash value of the previous block. Blockchain() and AddBlock() functions are used for developing a blockchain and adding blocks to the blockchain. Line 62 to line 67 shows the steps followed in creating blocks and adding to blockchain.

Algorithm 2 is an extension of algorithm 1 with PoW. In PoW, various challenges [6, 7–11] can be put to challenger before allowing him to add a block in the blockchain. Algorithm 2 shows an example of PoW with a generation of hash and operation. Hash value could be generated using any hashing algorithm whereas operation should be modular arithmetic infinite field. The generated hash value is verified by existing blockchain participants using ValidateChallenge() function. If the challenge is generated within a specified time period, verified by blockchain participants and linked with a correct block, then the process of generating blockchain using PoW starts in line 35 to line 39.

Algorithm 1 Regular Blockchain

Goal: To create a blockchain without proof of game

```
Index <- BlockCounter()
BBlock <- null
UData <- UserData()
PHash <- PreviousHash(BBlock)
BTime <- CurrentTimestamp()

BlockCounter <- function() {
 index <- index+1
}

UserData <- function() {
  # User entered data
}

GBlock <- function(block) {
  BBlock <- block
   BTime <- TTime()
   pHash <- null
  Hash <- SHA256(BBlock)
}

PreviousHash <- function(block) {
if {block=="Genesis" || Index=="0"}{
  return "0"
else if (block == "null") {
  return "null"
else{
  return Hash(PreviousBlock)
   }
 }
}

Hash <- function(block) {
  return SHA256(block)
}

Blockchain <- function(block) {
Index <- BlockCounter()
UData <- UserData()
PHash <- PreviousHash(BBlock)
BTime <- CurrentTimestamp()
}

AddBlock <- function(block) {
Index <- block.index
UData <- block.data
PHash <- block.PreviousHash(BBlock)
BTime <- CurrentTimestamp()
return Blockchain((Index, UData, PHash, BTime))
}

GBlock(0,CurrentTimestamp(),"Genesis","0",SHA256(0,CurrentTimestamp(),"Genesis","0")) #
    =>Creates Genesis Block
FirstBlock = new Blockchain(Index,UData,PHash,BTime,SHA256(Index,UData,PHash,BTim))
Blockchain(FirstBlock) # =>Creates First Block
SecondBlock = new Blockchain(Index,UData,PHash,BTime,SHA256(Index,UData,PHash,BTim))
Blockchain(SecondBlock) # =>Creates Second Block
ThirdBlock = AddBlock(Index,UData,PHash,BTime),SHA256(Index,UData,PHash,BTime))
```

Algorithm 2 Extending a regular blockchain with 'Proof of Work'

Goal: To create a blockchain with 'Proof of Work' and without 'Proof of Game'

```
Index <- BlockCounter()
BBlock <- null
UData <- UserData()
Challenge <- GenerateChallenge()
PHash <- PreviousHash(BBlock)
BTime <- CurrentTimestamp()

GenerateChallenge <- function(Value, Operation) {
Value <- GenerateValue()
Operation <- GenerateOperation()
ChallengeTime <- CurrentTimestamp()
}

ValidateChallenge <- function(TimeThreshold) {
CurrentTime <- CurrentTimestamp()

if {parent != SHA256(previousBlock) ||  !ValidTransaction(transaction) || (CurrentTime -
    ChallengeTime >= TimeThreshold)}{
  return "False"
else{
  return "True"
   }
 }
}

GBlock(0,CurrentTimestamp(),"Genesis","0", SHA256(0,CurrentTimestamp(),"Genesis","0",)) #
     =>Creates Genesis Block
NewBlock = new Blockchain(Index,UData,PHash,BTime,SHA256(Index,UData,PHash,BTim))
GenerateChallenge(Value, Operation) # =>Creates a Challenge
ValidateChallenge(TimeThreshold) # =>Validate a Block
Blockchain(NewBlock) # =>Creates New Block
```

Now, a group of participants may delegate rights to one or a group of participants for performing "adding blocks" activity. In this activity, one or group of participants authorized to perform this activity is known as Authority and concept is called "Proof-of-Authority (PoA)." Algorithm 3 shows the steps following in PoA before creating a blockchain. In PoA, signatures of participants authorized to perform this activity is verified. Although PoA is an additional overhead to PoW, but it does not require all participants to be active before allowing any participant to add a block. A minimum threshold of active participants is acceptable for allowing any new block in the blockchain. PoA consensus mechanism can be extended with "Proof-of-Ownership" (PoO) as shown in Algorithm 4. In PoO, it is assumed that a blockchain can be owned by a specific set of participants and those participants are allowed to create new blocks with and without the use of the challenge. If a new participant is interested to be part of blockchain, then its identity is verified with ownership credentials before appending it to stakeholders list.

Algorithm 3 Proof of Authority

Goal: To identify and validate blockchain authority using 'Proof of Authority'

```

PoA <- function(Identity,ParticipationProof) {
  return SIGNATURE(Identity,ParticipationProof)
}
```

Algorithm 4 Proof of Ownership

Goal: To identify and validate blockchain authority using 'Proof of Authority'

```
storedOwnership <- signature
stakeholderList = []
PoO <- function(Identity) {
  While(OwnershipIdentityrequest(Identity)) {
  identity <- OwnershipIdentityrequest(Identity)
  stakeholderList <- stakeholderList.append(identity)
  }
  return SIGNATURE(Identity,stakeholderList)
}

if (storedOwnership == PoO(Identity)) {
  continue
}
else {
  exit
}
```

Algorithm 5 presents the PoG concept introduced in the blockchain. In PoG, existing participants play a game with the new block participant based on the availability of resources. Initially, all participants are considered to be honest and ready to share their computational powers. If a new participant is interested to add a block, then it has to reveal its computational power, signature and block information. The existing blockchain participants will start the game as per the resources of the new participants. If new participants have enough resources, then it is put in the resourceful category and a game of heavy computations is played. Otherwise, the lightweight game is preferred and the participant is put in the resource-constraint category. In a resourceful category, a multiplayer game asks the new participant to verify a random bit and its position in a challenge. A challenge could be derived from PoW, PoA or PoO. In this work, the challenge is implemented for three consensus algorithms (PoW, PoA or PoO); however, it may be extended for other algorithms. In the lightweight game, the random bit position of the hash value is verified. If a new participant is able to verify the challenge then it is considered as the winner and allowed to add the block. The honesty of participants is the key success of this consensus mechanism; however, a historical fair play record is required to be maintained in order to increase the reliability and security of the existing blockchain.

Algorithm 5 Proof of Game (PoG)

Goal: To identify and validate blockchain authority using 'Proof of Game'

```

PoG <- function(player) {
  if (player == 'single') {
    Execute PoW() || PoA() || PoO()
    Lightweight_Game_Single(bit_value, position_value)
  }
  else if (player == 'multi') {
    While(Participant)) {
      group_sign=SIGNATURE(Identity,stakeholderList)
      Lightweight_Game_multi(bit_value, position_value,participant_list)
    }
  }
}

Lightweight_Game_Single <- function(player) {
bit, position <- Challenge(bit,position)
if(verify(bit,position)=='valid') {
  return(bit,position)
  }
else {
  return 'invalid'
  }
}

Lightweight_Game_multi(bit_value, position_value,participant_list){
bit, position <- Challenge(bit,position)
While(Participant <- participant_list)) {
bit_p,position_p <- Participant.bit, Participant.position
if(verify(bit,bit_p, position, position_p)=='valid') {
  return(bit,position)
  }
else {
  return 'invalid'
  }
}
```

4 PoG Integrated Proposed Healthcare Application Model

The healthcare industry faces numerous challenges in their system including billing frauds, mistakes, interruption of operations, leakage of information and transaction processing time. The solution to this problem is a blockchain-based proper billing system. With the help of blockchain technology, transparency and provisions to audit records are available for all processings wherever there is any type of payment involved. Further, a well-defined policy-based automated payment processing system can be derived using smart contracts. As the healthcare industry is majorly dependent on insurance thus, smart contract developed for providing health insurance to the patients would develop a useful and strong system. Blockchain-based

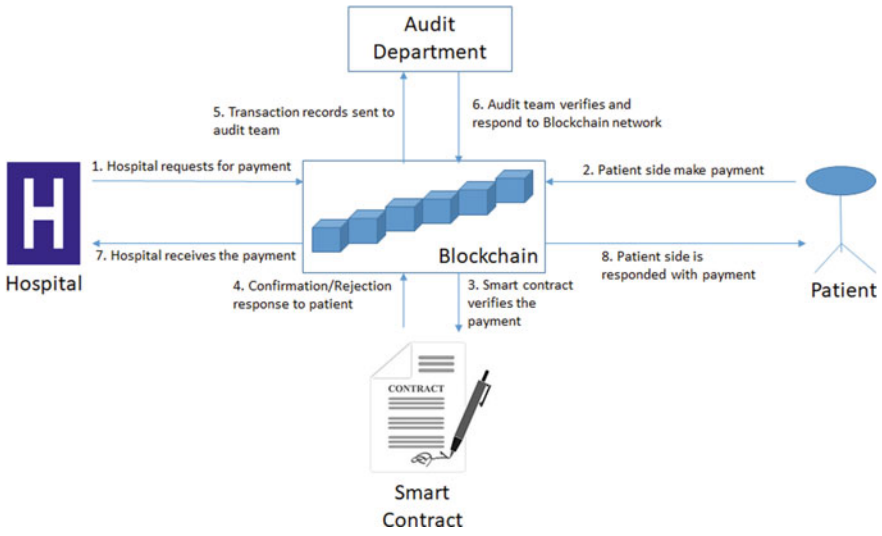
Audit Department
5. Transaction records sent to audit team
6. Audit team verifies and respond to Blockchain network
1. Hospital requests for payment
2. Patient side make payment
Blockchain
7. Hospital receives the payment
8. Patient side is responded with payment
Hospital
Patient
4. Confirmation/Rejection response to patient
3. Smart contract verifies the payment
CONTRACT
Smart Contract

Fig. 3 PoG integrated smart contract for healthcare system

billing and insurance applications will provide a reliable source of trustable information at reduced cost and within the stipulated time period. As blockchain provides immutable records, chances of fraud or attacks will be much lesser.

Figure 3 shows a high-level scenario of integrated PoG consensus algorithms in the healthcare system. There are hour major parties in this system: patient side, hospital side, blockchain network side and the audit team. The hospital initiates the payment request whenever patient arrives in hospital. After depositing a security amount, the blockchain network executes PoG-based smart contract and confirms the transactions as per hospital policies. If a smart contract confirms the transaction, then a permanent record of this transaction is stored in the transaction block of the blockchain network. The data from the transaction block is forwarded to the audit department for verification of any transaction. After random verifications from the audit department, the blockchain network make e-payment to hospital and sends a confirm message to patient side.

5 Analysis

In this section, the behavior of PoG is analyzed for attack detection.

- **Majority-Attack Detection**: After successful forking several blocks, it is exponentially impossible for an attacker to break PoG and produce a tampered chain. This can be proved using Chernoff bound and independence as follows:

Let M represents the population size of honest participants uses PoG for consensus establishment, m represents the population size of dishonest participants able to break PoG for tampered blockchain creation. Now, both sides will try to maximize their chance of success as follows:

$$\text{Chance}_M^{\text{Success}} = \text{Maximize}\left\{\{\text{Uniform}(0,1)\}^M\right\} \tag{1}$$

$$\text{Chance}_m^{\text{Success}} = \text{Maximize}\left\{\{\text{Uniform}(0,1)\}^m\right\} \tag{2}$$

Further, if there are N-blocks created then probability of successful blockchain construction is

$$P_{\text{construction}}^h = \sum_{i=1}^{h} \text{Chance}_M^{\text{Success}} - \text{Chance}_m^{\text{Success}} \tag{3}$$

According to Chernoff bound and independence, the probability of an attacker to tamper a blockchain can be computed as follows:

$$\begin{aligned} P_{\text{tamper}}\left(\text{Chance}_N^{\text{Success}} \le 0\right) &\le \min_{s>0} E\left[e^{-s\text{Chance}_N^{Success}}\right] \\ &= \min_{s>0} \prod_{t=1}^{N} E\left[e^{\text{Chance}_M^{\text{Success}}(t)}\right] E\left[e^{\text{Chance}_m^{\text{Success}}(t)}\right] \\ &= \min_{s>0} \left(E\left[e^{\text{Chance}_M^{\text{Success}}(t)}\right] E\left[e^{\text{Chance}_m^{\text{Success}}(t)}\right]\right)^N \end{aligned} \tag{4}$$

Since $M > m$, thus, there exist an $s > 0$ such that the product of inner expectation is less than 1. In conclusion, the success probability of an attacker in succeeding the attack decreases with an increase in the number of blocks.

- **The Priority of Blocks in the Blockchain**: As the number of players playing the game within the blocks of blockchain increases, so as the priority of blockchain. Each participant associated with any block of a blockchain has an equally likely probability of playing a game. Thus, the expected number of blocks mined by N-participants is proportional to the total number of participants associated with blocks, as shown in Fig. 4.
- **Block Round Time and Confirmation Time**: a block's round and confirmation time are found to be much lesser than 10 min (Bitcoin's block time) because of variation in the availability of resources during challenge generation and confirmation. As compared to block propagation time (6.5 s) observed in the Bitcoin network, the proposed scheme has a comparable time period because a block is propagated only if the game round is successful in its execution. Block mined process includes block round time and confirmation time. Figure 5 shows the variation in a number of blocks mined with an increase in the number of participants associated with blocks. Results show the number of blocks mined is

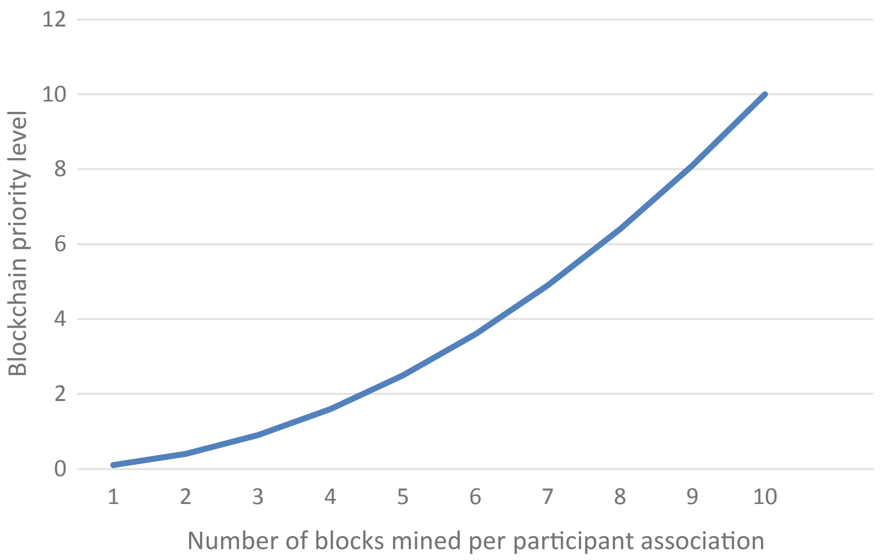

Fig. 4 Variation in blockchain priority level with an increase in the number of blocks mined per participant association

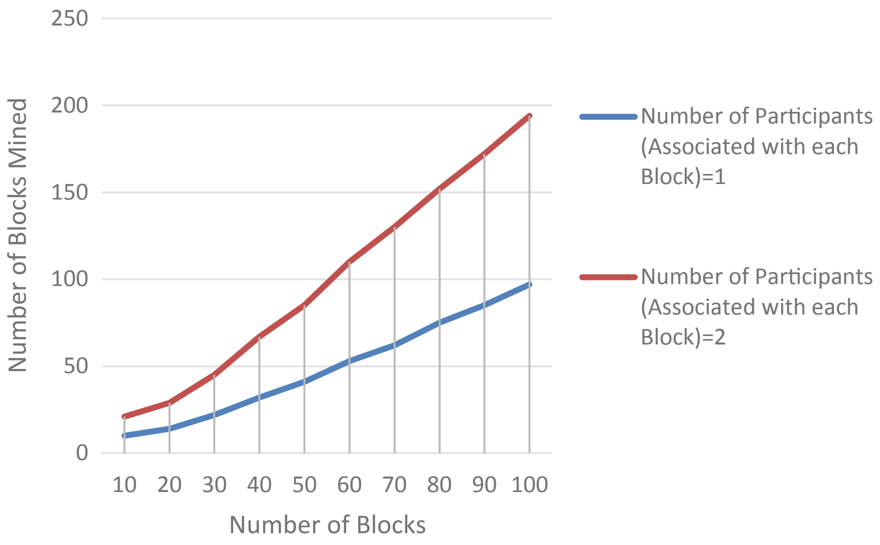

Fig. 5 Variation in the number of blocks mined with an increase in the number of participants associated with blocks (calculated using statistical formula)

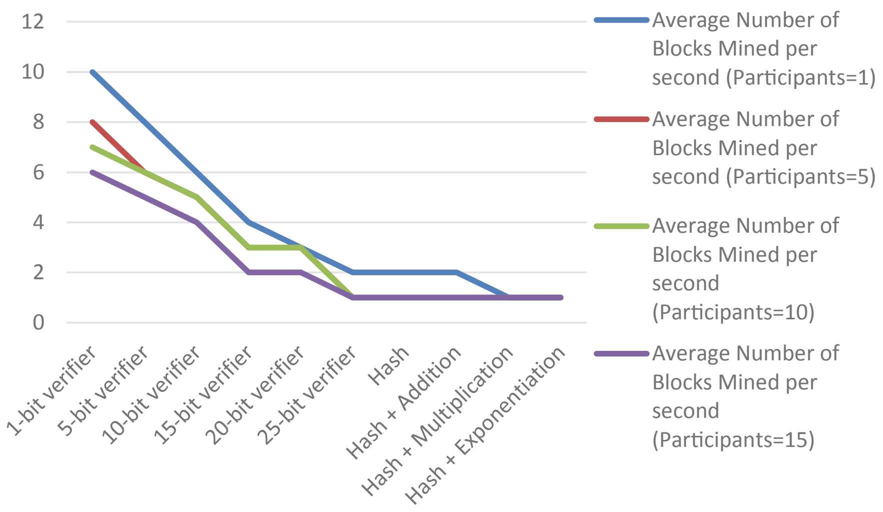

Fig. 6 Variation in the number of blocks mined per second with variation in block challenge

comparatively higher for two participants as compared to one participant. This difference increases with an increase in the number of blocks mined.

Figure 6 shows the variation in the number of blocks mined per second with variation in block challenge. Multiple challenges are considered for comparative analysis. For example, 256 bits of hash value output is considered for 1, 5, 10, 15, 20 and 25-bits verification. A comparative analysis of the number of blocks with increases in bits verification and modular arithmetic operations shows that the number of blocks mined decreases exponentially with an increase in the computational challenge over a system with processor: Intel Core i5-7200U CPU@2.5 GHz, 4 GB RAM and 64-bit operating system.

6 Conclusion

In this work, the PoG concept is introduced with a specific emphasis on designing methodologies for both resourceful and resource-constraint networks. PoG put challenges to challenger if the challenger is interested to give prior information about the availability of resources else memory access time is assumed to provide resource independent results. Further, PoG is a bit generator and verifier game between existing and new participants. The new participant is considered as the winner and allowed to participate in the blockchain if bits are verified in the stipulated time period. The experimental scenario shows that the number of blocks mined in PoG integrated blockchain decreases exponentially with an increase in the number of bits verified and the number of participants. The honesty of the participant is found to be the backbone of success for the PoG consensus mechanism. In the future, PoG will be extended to statistical PoG (SPoG) having an honesty verification model before and after playing a game for adding new blocks.

References

1. All Cryptocurrencies. https://coinmarketcap.com/all/views/all/
2. Z. Zheng, S. Xie, H. Dai, X. Chen, H. Wang, An overview of blockchain technology: architecture, consensus, and future trends, in *2017 IEEE International Congress on Big Data (BigData Congress)*. IEEE (2017), pp. 557–564
3. W. Wang, D.T. Hoang, Z. Xiong, D. Niyato, P. Wang, P. Hu, Y. Wen, *A Survey on Consensus Mechanisms and Mining Management in Blockchain Networks*. arXiv preprint arXiv:1805.02707 (2018), pp. 1–33
4. S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*. http://bitcoin.org/bitcoin.pdf (2008)
5. B. Laurie, R. Clayton, Proof-of-work proves not to work; version 0.2, in *Workshop on economics and information, security*
6. A. Baliga, Understanding blockchain consensus models, in *Persistent* (2017)
7. A. Back, *Hashcash* (1997). http://www.cypherspace.org/adam/hashcash/
8. M.K. Franklin, D. Malkhi, Auditable metering with lightweight security, in *Financial Cryptography* (1997), pp. 151–160
9. D. Mankins, R. Krishnan, C. Boyd, J. Zao, M. Frentz, Mitigating distributed denial of service attacks with dynamic resource pricing, in *Proceedings of 17th Annual Computer Security Applications Conference (ACSAC 2001)* (2001)
10. A. Serjantov, S. Lewis, Puzzles in P2P systems, in *8th CaberNet Radicals Workshop,* Corsica, Oct 2003
11. G. Kumar, R. Saha, M.K. Rai, R, Thomas, T.H. Kim, Proof-of-work consensus approach in blockchain technology for cloud and fog computing using maximization-factorization statistics. IEEE Internet Things J. (2019)
12. S. Dey, Securing majority-attack in blockchain using machine learning and algorithmic game theory: a proof of work, in *2018 10th Computer Science and Electronic Engineering (CEEC)* (2018). IEEE, pp. 7–10
13. A. Stone, An examination of single transaction blocks and their effect on network throughput and block size. Self-published Paper, (Jun. 2015) [Online]. Available http://ensocoin.org/resources/1txn.pdf
14. X. Liu, W. Wang, D. Niyato, N. Zhao, P. Wang, Evolutionary game for mining pool selection in blockchain networks, in *IEEE Wireless Communications Letters* (2018), pp. 1–1
15. Z. Xiong, S. Feng, D. Niyato, P. Wang, Z. Han, Optimal pricing based edge computing resource management in mobile blockchain, in *2018 IEEE International Conference on Communications (ICC)*, Kansas City, Kansas, May 2018
16. Y. Jiao, P. Wang, D. Niyato, Z. Xiong, Social welfare maximization auction in edge computing resource allocation for mobile blockchain, in *2018 IEEE International Conference on Communications (ICC)*, Kansas City, Kansas, May 2018
17. N. Houy, The bitcoin mining game. Ledger J. **1**(13), 53–68 (2016)
18. N. Houy, *The Bitcoin Mining Game*. Available at SSRN 2407834 (2014)
19. A. Kiayias, E. Koutsoupias, M. Kyropoulou, Y. Tselekounis, Blockchain mining games, in *Proceedings of the 2016 ACM Conference on Economics and Computation,* ACM (2016), pp. 365–382