MMOG User Participation Based Decentralized Consensus Scheme and Proof of Participation Analysis on the Bryllite Blockchain System

Jusik Yun[1], Yunyeong Goh [1], Jong-Moon Chung[1*], OkSeok Kim[2,3], SangWoo Shin[2,3], Jin Choi[2,3], and Yoora Kim[2,3]

[1] School of Electrical and Electronic Engineering, Yonsei University
Seoul, Republic of Korea
[e-mail: e-mail: awp212@yonsei.ac.kr, rhdbsdud@yonsei.ac.kr, jmc@yonsei.ac.kr]
[2] HanbitSoft, Seoul, Republic of Korea, [3] Bryllite, Hong Kong
[e-mail: jade@hanbitsoft.co.kr, swoows@hanbitsoft.co.kr, program@t3.co.kr, yoora@t3.co.kr]
*Corresponding author: Jong-Moon Chung

Received April 11, 2019; revised May 25, 2019; accepted June 13, 2019; published August 31, 2019

Abstract

Proof of Work (PoW) based blockchains have limitations in throughput, time consumption, and energy efficiency. In these systems, a miner will consume significant time and resources to obtain a reward for contributing to the blockchain. To overcome these limitations, recent research on blockchains are focused on accelerating the speed, scalability, and enhancing the security level. By enhancing specific procedures of blockchain system, the level of data integrity supported by the blockchain can become more robust, and efficient. In this paper, a new blockchain consensus model based on the Bryllite Consensus Protocol (BCP) is proposed to support a hyper-connected massively multiplayer online game (MMOG) ecosystem. The BCP scheme enables users to participate directly in new consensus processes through a Proof of Participation (PoP) algorithm. In this model, the consensus algorithm has a simpler form while maintaining high security level. In addition, because the BCP scheme gives users an equal chance to make a contribution to the blockchain, rewards are distributed in an equal fashion, which motivates user participation. The analysis of the proposed scheme is applied to the ***Bryllite*** consortium blockchain system (homed in Hong Kong), which is a new blockchain network developed for international game industries, gamers, and game events.

Keywords: Blockchain, Bryllite, Bitcoin, Decentralized Consensus Algorithm, Ethereum, Proof of Participation.

A preliminary version of this paper was presented at ICONI 2018, where it received the Outstanding Paper Award. This research was supported by the MSIT (Ministry of Science and ICT) of the Republic of Korea, under the ITRC (Information Technology Research Center) support program (IITP-2019-2018-0-01799) supervised by the IITP (Institute for Information & communications Technology Promotion).

http://doi.org/10.3837/tiis.2019.08.015 **ISSN : 1976-7277**

1. Introduction

Blockchain is a distributed ledger system that ensures data integrity, which was initially proposed in a paper on Bitcoin by Satoshi Nakamoto [1]. A blockchain used for Bitcoins would need to maintain a decentralized database structure without being under the control of a centralized organization by sharing a ledger with mining nodes. To maintain data integrity, each node stores the same block and uses data chaining and hash functions. The method used to reliably determine the node that generated the block is based on the consensus algorithm. In the Proof of Work (PoW) consensus algorithm, mining nodes use SHA-256 to find a hash output value less than the target hash value corresponding to a predefined difficulty level. Then, the mining node that finds the corresponding hash value is selected to generate a block that is added to the blockchain. The difficulty of the hash calculation is adjusted so that one block is generated every 10 minutes for a Bitcoin, while all other competing mining nodes waste a huge amount of electric energy due to the computation process of the hash calculation. Meanwhile, with the continuous development of blockchain technology, the application range has expanded in to fields such as finance, game industry, and social network services (SNSs). Especially, the application of blockchains in the game industry can help build a hyper-connected game ecosystem for various massively multiplayer online games (MMOGs) supported by a common blockchain service network. In networks supporting MMOGs, there are numerous users and MMOG service providers, therefore, network scalability is essential [2]. Blockchain technology is now being linked across various industries, and consensus algorithms such as PoW are inefficient and unsuitable to satisfy user quality of service (QoS) requirements in networks supporting MMOGs, which need quick transactions. Therefore, in this paper, a novel proof of participation (PoP) consensus mining scheme based on user participation is proposed, and is applied to the Bryllite game blockchain system and its performance is analyzed [3].

The unique features of the proposed blockchain algorithm are characterized by the following two aspects. First, not only the miners but also the users (clients) can participate in the consensus process and receive a compensation. This motivates the user to join the blockchain network. Second, only one hash computation needs to be conducted for each user during the PoP process, which is opposing to the repetitive hash computations that were conducted by all miners in the PoW mechanism. In addition, the probability of receiving individual users block rewards is fairly equal, which promotes continuous monitoring and participation to the blockchain by its users.

In this paper, the proposed PoP based consensus algorithm is named the Bryllite Consensus Protocol (BCP). The following sections of this paper are organized as follows. Section 2 presents the taxonomy of blockchain consensus algorithms, the system model of the BCP is provided in section 3, the BCP algorithm's analysis is in section 4, the simulation results and discussion are in section 5, and the paper is concluded in section 6.

2. Taxonomy of Blockchain Consensus Algorithms

In this section, the characteristics of blockchains and the corresponding consensus algorithms are described. Blockchains can be classified into permissionless or permissioned depending on the network purpose.

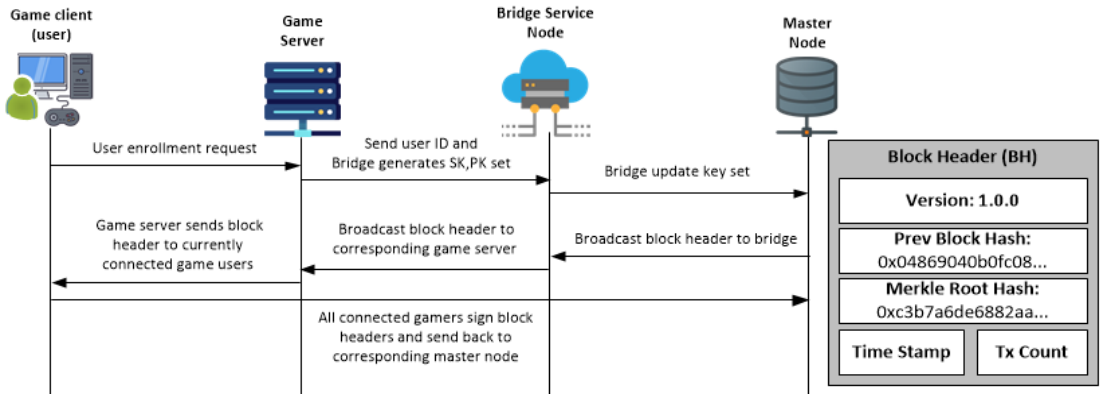

Fig. 1. Overall process of the BCP PoP

2.1 Permission-less Blockchain

In a permissionless blockchain, there are no specific restrictions to participate in the network. Since network members are not pre-validated, a strict consensus algorithm is used to verify untrusted distributed ledgers of participants [4]. PoW is a typical consensus algorithm used in permissionless blockchains. When PoW is used, block miners solve the SHA-256 based crypto puzzle consuming an enormous amount of hash calculations, which is a means of defending against well-known Sybil attacks [5]. This expensive method of consensus is a mechanism triggered by the nature of the network that any user can participate in the blockchain without permission. Such blockchains are typically classified as a public blockchain, thus, public blockchains are generally permissionless blockchains. Permissionsless blockchains are focused on decentralization, in which Bitcoin and Ethereum are represenative examples.

2.2 Permissioned Blockchain

In a permissioned blockchain, the distributed ledger is confirmed by pre-validated users. Since the trust of the participants is guaranted, there is less motivation to use a resource consuming consensus algorithm such as PoW. The block mining process is performed using a bounded asynchronous consensus algorithm, where the Practical Byzantine Fault Tolerance (PBFT) is a representative example [6]. Such schemes are much faster than permissionless consensus algorithms, such as PoW, but these schemes have the disadvantage that the message complexity increases signifincantly when the network is expanded [7]. Therefore, Byzantine Fault Tolerance (BFT) based consensus algorithms experience scalability issues. Consortium or private blockchains belong to the category of permissioned blockchains. In particular, these consortium blockchains appear in the form of a prospective blockchain that are mainly used in various service chain models by many companies [8-10].

3. BCP System Model

The Bryllite Consensus Protocol (BCP) has been proposed to serve as a new type of blockchain network for the game industry ecosystem, enabling cryptocurrency exchange between companies, conferences, consortiums, and gamers. The BCP system architecture can be classified into four major components as shown in **Fig. 1**.

1) **Master Node** (MN): A mining node that participates in the consensus of the Bryllite blockchain. One MN is assigned to each game.
2) **Game Client**: A game user that plays the game(s).
3) **Game Server**: A game server will manage the local game database, and also store and manage the transactions that occur during the game(s). However, game servers do not belong to the blockchain network, instead they interact with the Bryllite blockchain network via the bridge service node described below.
4) **Bridge Service Node**: The bridge service node manages bidirectional communication between the MN and game servers. It also manages the asymmetric keys (i.e., public and private keys) of the game client.

BCP consists of two steps: PoP and Majority Voting. In the PoP process, each MN selects a representative game client with a minimum hash value. In the Majority Voting process, each MN proposes a hash value for its representative game client, and the MN that proposes the lowest output hash value is chosen as the block generation node, which will receive a mining compensation for its contribution. Transactions from users are propagated in the network and accumulated in the transaction pool (e.g., Mempool in Bitcoin). In the proposed blockchain model, each MN generates a candidate block based on the consensus interval (T), and the winning miner is selected by the PoP based on majority vote, which decides the block to connect to the blockchain.

3.1 Proof of Participation (PoP)

The overall process of the PoP is described in the following, which is based on an individual MN. The symbols and abbreviations used in the text are summarized in **Table 1**.

Table 1. Symbols and Abbreviation

Symbol	Description
G_k	kth game server
B_k	kth bridge node
M_k	kth MN
N_k	Active game client connected to the kth game server
$C_{k,i}$	Client i of game k
$H(\cdot)$	Hash function
$PK_{k,i}$, $SK_{k,i}$	Public and private key of game client i of game k
M_k^{pk}, M_k^{sk}	Public and private key of M_k
Ψ_k	Block header (BH) of M_k
$\Psi_{k,i}$	BH signed by client i of game k
Ψ_{k^*}	BH with minimum hash value in game k (Representative BH of MN k)
Ψ_{V^*}	BH selected from majority vote
Ξ_k	Block generated by M_k
$Sign(\Psi, SK_{k,i})$	Sign of game client i of game k in the BH
$S_{k,i}$	Signature of client i of game k
$V(S_i, \Psi, PK_{k,i})$	Verification of signature of client i of game k through the public key
T	Consensus interval
P_k	Set of Ψ received at the propose step in M_k
V_k	Set of Ψ received at the vote step in M_k
C_k	Set of messages received at the commitment step in M_k

A) Background Key Management Process

A game client will register to the game server to participate in a game. Then the server generates a private key using the private key generator (PKG), which is returned to the user. The generated public and private key set of the entire user is managed by the bridge service node. When a client registers or deletes its own account, the same process is performed in the bridge service node.

B) Block Header (BH) Distribution

A MN will generate a BH Ψ for use in each round of the consensus and send it to the bridge service node. The block header contains the block version, previous block hash, the Merkle root hash (i.e., combined hash value of the transactions), time-stamp, Tx count (i.e., maximum transaction count in one block), and gamer signature. The gamer signature consists of a signature and public key. Initially, the gamer signature is set to null and it will be signed by the client in the PoP process. The structure of the block header is shown in **Fig. 1**. First, the bridge service node sends the received block header to the game server, which is represented in (1). The game server k sends Ψ_k to the active game client $C_{k,i} \in N_k$, where (k, i) denotes client i of game k in (2).

$$M_k \underset{\Psi_k}{\rightarrow} B_k \underset{\Psi_k}{\rightarrow} G_k \tag{1}$$

$$G_k \underset{\Psi_k}{\rightarrow} C_{k,i} \in N_k \tag{2}$$

C) Client signature

After the active client receives the Ψ_k that has the gamer signature space as null, the client hashes the received BH through the hash function $H(\cdot)$ based on SHA256, which has strong collision resistance. Each game client generates a signature using its private key (SK) with a hash value in the BH ($H(\Psi_k)$). The public key generator (PKG) can create a large scale of random unique numbers to be used in assignment to many game clients. In (3), the signature of game client i is represented as $S_{k,i} = Sign(H(\Psi_k), SK_{k,i})$. The ith game client update pair of the signature and public key is denoted as $(S_{k,i}, PK_{k,i})$ in the gamer signature space in (4). Finally, the client will send $\Psi_{k,i}$ to the MN, where in (5), $\Psi_{k,i}$ represents the signed block header of client i in game k.

$$S_{k,i} = Sign(H(\Psi_k), SK_{k,i}) \tag{3}$$

$$\Psi_{k,i} = \Psi_k \cup (S_{k,i} + PK_{k,i}) \tag{4}$$

$$\sum_{i=1}^{N_k} c_{k,i} \in N_k \underset{(\Psi_{k,i})}{\longrightarrow} M_k \tag{5}$$

D) Selection Representative Client

The MN receives the block headers that include the signature and PK pairs sent by all users connected to the game. Using $PK_{k,i}$, the signature $S_{k,i}$ can be decrypted to $H(\Psi_k)$, which is the value sent by client i that represents $V(S_{k,i}, H(\Psi_k), PK_{k,i})$. The MN can identify the signature from all clients if $V(S_{k,i}, H(\Psi_k), PK_{k,i}) = H(\Psi_k)$ is satisfied. For a signature that has been verified, client i^* with the minimum hash value is designated as the representative of M_k, as defined in (6). In addition, the signed block header of client i^* (i.e., Ψ_{k^*}) is the representative block header of M_k, as presented in (7).

$$i^* = arg\min_i H(\Psi_{k,i}) \tag{6}$$

$$\Psi_{k^*} = \Psi_{k,i^*} \tag{7}$$

Each node compares the $H(\Psi_{k,i})$ values of all clients created in each game and sets the block header that has the lowest hash as the representative BH of each game. The overall PoP process is described in **Pseudocode 1**. In the PoP process, MNs present their BH in every round, and the users sign the BH and returns it to the MN. Since the BH information received from the MN is fixed, there is only one possible signed block that the user can submit, depending on its private key. Thus, users cannot attempt mining to achieve a smaller hash value, which is why there is no mining process computation burden in the BCP scheme. In addition, it needs to be noted that if a malicious user attempts to perform a Sybil attack in the BCP PoP process, it will be nearly impossible, as the bridge server node will easily notice irregular user account changes (based on multiple keys being used) and can easily apply user authentication procedures along with gamer profile status checks (e.g., game log monitoring, minimum duration of game playing time, etc.) to prevent Sybil attacks.

Pseudocode 1. PoP operation process

Algorithm: Proof of Participation (PoP)

Ψ: Block Header; *PK*/*SK*: Public/Private Key
Broadcasting Block header to game client
For every consensus
  Generate Ψ, send to bridge service node
  Bridge service sends Ψ to game server
  Server delivers Ψ to connected game client
  Signature Generation at the Game Client
  For each game *k*
  Input: Ψ, PK_k= $(PK_{k,1}, PK_{k,2}, \cdots, PK_{k,N_k})$,
    SK_k= $(SK_{k,1}, SK_{k,2}, \cdots, SK_{k,N_k})$
    N_k: number of active clients
  Generate $H(\Psi)$
   For each game client *i*
    Generate $S_{k,i} = Sign(\Psi_k, SK_{k,i})$
    Output: $\Psi_{k,i}$
    Send output to MN *k*
  Signature Verification
  Input: $S = (S_{k,1}, S_{k,2}, S_{k,3}, \cdots, S_{k,N})$, *PK* set
  $PK = (PK_{k,1}, PK_{k,2}, \cdots, PK_{k,N})$
   For each client *i*
    If $V(S_{k,i}, H(\Psi_k), PK_{k,i}) = H(\Psi_k)$,
     Generate $H(\Psi_{k,i})$
    Else
     Delete Ψ of client *i*
  Generate hash set $H(\Psi_{k,i})$
  Find $i^* = arg\min_i H(\Psi_{k,i})$
  Output: MN selects client i^* and decides representative $\Psi_{k^*} = \Psi_{k,i^*}$.

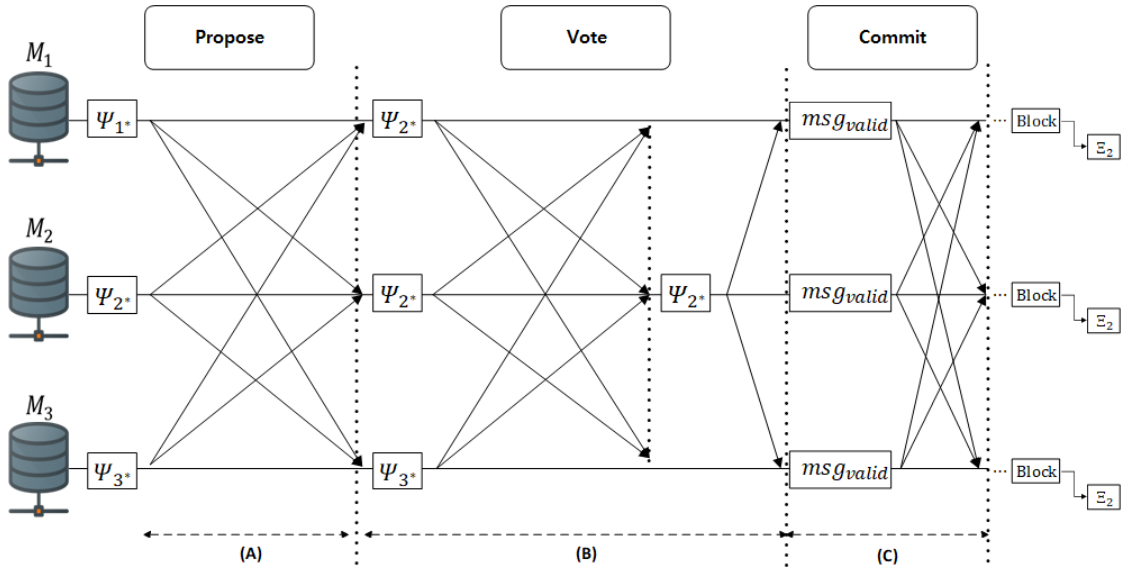

Fig. 2. Example of a majority voting process based on the Propose, Vote, and Commit phase

3.2 Majority Voting

In the majority voting process, each MN broadcasts its representative BH and votes for the minimum hash output. After verifying which BH has the minimum hash value, that block is selected and added to the blockchain, and a reward is given to the provider of that block. When sending and receiving a BH or when voting, pairs of public keys (PKs) and private keys (SKs) of each MN (M_k^{sk}, M_k^{pk}) are used in the certification process.

A) Propose

In the Propose phase, the MNs broadcast their representative BH. Each MN selects a representative client i^* and delegate BH Ψ_{k^*}. The MNs of each game k signs Ψ_{k^*} with its own private key, M_k^{sk}, and propagates it to all other MNs, as presented in (8).

$$\sum_{k=1}^{n} M_k \xrightarrow[Sign(\Psi_{k^*}, M_k^{sk})]{} \sum_{k=1}^{n} M_k \tag{8}$$

Next, each MN receives all block headers from other MNs satisfying $V\left(M_k^{sk}, \Psi_{k^*}, M_k^{pk}\right) = \Psi_{k^*}$. Then, MN k creates its own BH set $P_k = \{\Psi_{i^*} | i = 1, \cdots, n\}$.

B) Vote

After receiving the BH in the Propose phase, in the Vote phase, each MN will find a BH that has the minimum hash value, as presented in (9), and denote it as the voted candidate of M_k (i.e., K_v). After the candidate is determined, each MN re-broadcasts its candidate value to all other nodes, as presented in (10).

$$k_v = arg\min_{i^*}\{ H(\Psi_{i^*}) \in P_k \} \tag{9}$$

$$\sum_{k=1}^{n} M_k \xrightarrow[Sign(\Psi_{K_V}, M_k^{sk})]{} \sum_{k=1}^{n} M_k \tag{10}$$

In the same way, each MN receives all candidate BHs from other MNs satisfying $V\left(M_k^{sk}, \Psi_{K_V}, M_k^{pk}\right) = \Psi_{K_V}$. Then, the MN k generates its own voting candidate set $V_k = \left\{\Psi_{k_V} \middle| k = 1, \cdots, n\right\}$. For example, if M_1 determines the candidate to be Ψ_{2^*} in the voting step, then $\Psi_{1_v} = \Psi_{2^*}$. Since each node selects a candidate BH based on the minimum hash value in P_k, the K_v selected by each node should be equal if there is no malicious situation. Based on the assumption that more than half of the miners are honest, and the PKI system is secure, the voting decision is made using the majority rule. If the value of Ψ_{k^*} in set V_k of node k exceeds the majority, this ensures that the majority of MNs have identified Ψ_{k^*} as the minimum hash value in this round. If $\{Number\ of\ \Psi_{k^*}\ votes | \Psi_{k^*} \in V_k, \text{for } \exists k\} > \frac{n}{2}$ is satisfied, then Ψ_{k^*} will represent Ψ_{V^*}.

C) Commit

In the Commit phase, all MNs verify the block candidate's integrity and finally determine the block that will be connected to the blockchain. The MN that had proposed the final candidate BH Ψ_{V^*} broadcasts the total block data including the transaction list to all other MNs. After receiving the blockchain data, every node verifies the previous block hash and gamer signature in the block. If the block is valid, the MN suggests that this block is valid and broadcasts a verification message (msg_{valid}) to all other nodes. If not, it broadcasts a message claiming that this block is not valid ($msg_{invalid}$). These procedures are expressed in (11).

$$\sum_{k=1}^{n} M_k \xrightarrow[(msg_{valid}\ or\ msg_{invalid})]{} \sum_{k=1}^{n} M_k \tag{11}$$

After receiving the message whether this block is valid or invalid, MNs make a final decision. MN k will store the msg received from the other nodes in its message set C_k. If the number of msg_{valid} exceeds more than half, then the transaction is confirmed, and the MNs connect that block to their own blockchain. The algorithm of the majority voting process is described in **Pseudocode 2**. Finally, the representative user of the MN that proposed the minimum hash value in the majority voting process receives a block compensation.

Pseudocode 2. Majority voting process

Algorithm: Majority Voting

Ψ_{k^*} : BH of minimum hash value in game k
For each MN k **(Propose)**
MN broadcasts Ψ_{k^*} to other nodes
Receive BHs from other nodes
Generate $P_k = \{\Psi_{1^*}, \Psi_{2^*}, \Psi_{3^*}, \ldots, \Psi_{n^*}\}$
For each MN k **(Vote)**
Input: $P_k = \{\Psi_{1^*}, \Psi_{2^*}, \Psi_{3^*}, \ldots, \Psi_{n^*}\}$
Find $k_v = arg\ \min_k\{ H(\Psi_{k^*}) \in P_k \}$
Vote on Ψ_{K_v} and broadcast to network
Receive voted BH and generate V_k
For each Ψ_{k^*}
If $\{number\ of\ \Psi_{k^*} \in V_k\} > \frac{n}{2}\ \ for\ \exists k$
Set $\Psi_{V^*} = \Psi_{k^*}$
Output: Ψ_{V^*}
M_V broadcasts entire block to network
For each MN k (**Commit)**
Input: Ψ_{V^*} , Total block data of Ψ_{V^*}
Verify integrity of block candidate
If block is valid → Generate msg_{valid}
Else → Generate $msg_{invalid}$
Output: msg_{valid} or $msg_{invalid}$
MN broadcasts that message to other MNs and receive messages
Set of received messages C_k is made
Input: C_k, Entire block of MN M_V
If $\{number\ of\ msg_{valid} \in C_k\} > \frac{n}{2}\ \ for\ \exists k$
Validate and confirm this consensus
Connect block of M_V to its own blockchain

As example of the majority voting process is described in **Fig. 2**. In this Figure, three MNs $M_1, M_2,$ and M_3 perform a majority voting process. In the Propose phase, all MNs broadcast their representative block header $\Psi_{1^*}, \Psi_{2^*,}$ and Ψ_{3^*}. In this example, the hash of Ψ_{2^*} is a global minimum hash value. Then, in the Vote phase, each MN votes to elect the BH that has the minimum hash value. But, there may be cases in which a node has problems in communication or a node may intentionally vote maliciously (to hinder the blockchain process) on a BH that does not have the minimum hash value. In this example, M_3 votes for its own representative BH. However, because Ψ_{2^*} is selected by the majority rule vote decision in this example. In the Commit phase, the validity of the block generated by M_2 is checked, where if it is valid, Ξ_2 (i.e., the block generated by M_2) is connected to its own blockchain.

An advantage of the BCP scheme is that it depends on a majority vote mechanism. In comparison, the PBFT algorithm (for asynchronous blockchain networks) has an obligation to select a leader, where more than two-thirds of nodes have to be honest to ensure a successful consensus [4]. In the BCP model, the block message transmitted by each MN during the consensus process includes the signature of each MN's private/public key. Therefore, message modulations is impossible unless the entire blockchain is hacked and manipulated. In addition, the BCP model does not need a leader election process, therefore, it does not have to satisfy the strict consensus bound of PBFT. Rather, the PoP model is a competitive-based algorithm, such as PoW, which selects a block generator

based on the minimum hash value, but is more economical. In the BCP model, if an attempt to manipulate the selection process to make a malicious miner to be selected as the block generator, the number of false game users participating in the game must be increased rapidly. However, this can be prevented by periodic checking of the game log (e.g., play time, user identification) within the game server by the network management system.

4. Analysis of the BCP Algorithm

The proposed system has the following stable and fairness properties.

Lemma 1. There is practically only one minimum hash output value, which is determined on a random basis in the BCP process. □

Proof. The output hash value is $H(\Psi_{V^*})$ and the client's SK and PK are fixed. However, MNs have different Merkle root hashes depending on their combination of transactions, and the hash of a BH per round for each MN are randomly determined. In other words, the output hash values of each MN are all different because hash values have a very low collision probability based on the generalized birthday problem of a n bit hash function. In (12), P (N, n) is the probability that among N inputs that are independent (which are n bits each and given that $k \ll 2^n$), at least two will have the same hash output.

$$\mathrm{P}(N,n) = 1 - \prod_{k=1}^{N-1}\left(1 - \frac{k}{2^n}\right) \approx 1 - \prod_{k=1}^{N-1} e^{-\frac{k}{2^n}} = 1 - e^{-\frac{N(N-1)}{2^{n+1}}} \tag{12}$$

If a hash algorithm such as SHA-256 or SHA-512 is used, the collision resistance of the security requirement is satisfied. In addition, based on the avalanche effect of the hash output [11], the probability that any hash output will be the minimum among all N participants is $1/N$. ■

Lemma 2. Clients participating in each game receive a fair block generation reward. □

Proof. Let $\mathrm{X}_{k,i}^1$ denote the event that client i of game k ($C_{k,i}$) will be selected as the minimum hash in the PoP, and $\mathrm{X}_{k,i}^2$ denotes the corresponding event majority voting process. Then, the probability of $\mathrm{X}_{k,i}^1$ is $P(\mathrm{X}_{k,i}^1) = \frac{1}{N_k}$ by *Lemma 1*. The probability of $\mathrm{X}_{k,i}^2$ is expressed as $P(\mathrm{X}_{k,i}^2) = \frac{N_k}{\sum_{k=1}^{m} N_k}$, where m is the total number of game servers. Let the block generation reward be R, then the expected reward of client $C_{k,i}$ can be expressed as (11).

$$E(R_{k,i}) = P\left(\mathrm{X}_{k,i}^1\right)P\left(\mathrm{X}_{k,i}^2\right)R = \frac{R}{N_k}\frac{N_k}{\sum_{k=1}^{m} N_k} = \frac{R}{\sum_{k=1}^{m} N_k} \tag{13}$$

Therefore, regardless of which user participates in which game, all clients participating in each game has the same expected reward, which is a new fairness property. ■

4.1 Complexity and scalability analysis

In this section, the complexity analysis of BCP and delegated mining blockchains are provided. Since users can receive block rewards through PoP, scalability analysis in reference to an increase in number of users is important. In addition, since BCP is aimed at a highly-connected ecosystem of MMOGs, it is essential to consider connection of multiple games. As the number

of users increases, the number of MNs will correspondingly increase. To overcome complexity issues due to an increasing number of blockchain miners, Delegated Proof of Stake (DPOS) based algorithms (e.g., EOS [12]) confine mining nodes to maintain scalability in blockchains. Therefore, the message complexity of delegated mining is also anlyzed. The analysis of the existing system and the model using delegated mining are summarized in lemmas 3 and 4, respectively.

Lemma 3. *The message complexity M(n) in each consensus round is* $O(3n^2 - 3n - 1)$. □

Proof. In the Propose phase, each MN broadcasts its representative block header to the other *n*-1 nodes. Therefore, the total message complexity of the Propose phase is $O(n(n-1))$. In the Vote phase, each MN selects and broadcasts the BH that has the minimum hash among the received BHs during the Propose phase. Thus, the message complexity becomes $O(n(n-1))$. In the Commit phase, the majority voted node (M_V) broadcasts a block to the other *n*-1 nodes, and each node broadcasts the verification results back through the network, resulting in a message complexity of $O(\mathrm{n}-1)$ and $O((n-1)^2)$, respectively. Therefore, the total message complexity in each consensus round is $O(n(n-1)+n(n-1)+(n-1)+(n-1)^2) = O(3n^2-3n-1)$.

■

Lemma 4. *The message complexity of BCP in delegation mode is* $O(2d^2 + (n-3)d - 1)$, *where d is the number of delegated miner nodes.* □

Proof. In delegation mode, *d* delegation nodes are selected from *n* MNs. The selected *d* nodes only participate in the consensus process of BCP. The (*n*-*d*) MNs that do not participate in consensus should broadcast their representative BH to the *d* mining nodes. Therefore, the complexity of this process is $O((n-d)d)$. The *d* mining nodes compare the received (*n*-*d*) different BH hash values and select the BH that has the minimum value as the representative BH. Next, the consensus process proceeds among the *d* mining nodes, therefore, the complexity is given as $O(3d^2 - 3d - 1)$ based on lemma 3. As a result, the final complexity in delegation mode is given by $O(nd - d^2 + 3d^2 - 3d - 1) = O(2d^2 + (n-3)d - 1)$. In addition, the inequality $O(2d^2 + (n-3)d - 1) \leq O(3n^2 - 3n - 1)$ always holds if $d \leq n$.

■

The message complexity graph is shown in **Fig. 3**. Each performance curve segment represents the message complexity by changing the ratio of the delegation node while keeping *n* constant. The message complexity is proportional to the delegation percentage, and the message complexity increases as the number of MNs increases.

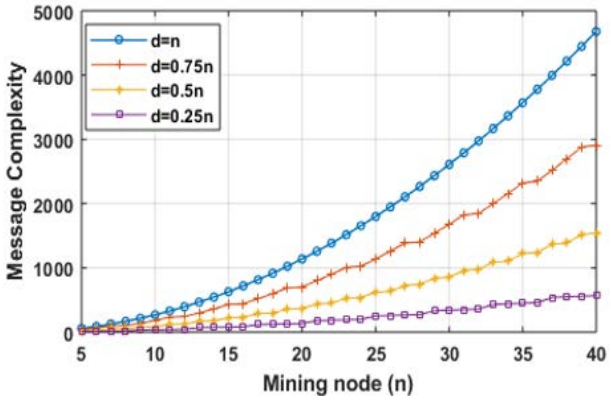

Fig. 3. Message complexity of delegated mining

5. Simulation Result and Discussion

In this section, the TPS and latency simulation of the BCP algorithm is analyzed based on the Bryllite blockchain [13]. First, the TPS of the BCP algorithm was analyzed. The system was set such that one block could be generated per consensus period *T*. In BCP, a transaction has a fixed size of 128 bytes. Therefore, if the block size is 128 KB, the maximum number of transactions that can be contained in one block is 1024. Therefore, dividing this by the period *T* (30 seconds) results in 1024/30 = 34. Therefore, the maximum TPS according to the block sizes 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, and 4 MB is determined as 34, 68, 136, 273, 546, and 1092, respectively. The maximum TPS and average TPS are shown in **Table 2**.

Next, the latency is analyzed. The latency is defined as the total time required for the BCP algorithm to reach consensus and successfully generate a block to add on to the blockchain. Generally, if the BCP algorithm verifies the MN with the minimum hash value within *T* seconds, the consensus process succeeds. Otherwise, consensus fails and the overall TPS performance drops. Therefore, latency is an indicator of the performance against the network scalability, and each cycle of operation aims for completion within the time limit of *T* seconds. The experiment results of the average latency based on an increasing number of nodes is presented in **Fig. 4**. The results show the trend of how the latency increases as the number of MNs increase from 50 to 500. The MN serves the role of block validator in the BCP algorithm and each MN serves one game each. If the block sizes are 512 KB and 1 MB, the average latency corresponds to about 18 seconds. When the block size is expanded to 4 MB, the average latency grows up to 25 seconds for the case of 500 MNs. When the block sizes are larger, the transaction time increases proportionally, as it takes more time to verify and commit to a block. Instead, if the block Commit process is completed within the consensus period *T*, then the TPS increases because more transactions can be processed. The experiment results in **Table 2** show how the average TPS increases in proportion to the block size. However, the difference between maximum and average TPS also increases as the block size increases. This is because the latency increases as the block size increases. If the maximum latency exceeds the consensus period *T*, the consensus will be continued in the next round, which will result in the average TPS value decreasing. Therefore, the block size and latency are in a trade-off relation. In addition, approximately up to 500 games can be connected in the PoP based blockchain under the time limit *T* set in the experiments.

Table 2. TPS experiment results based on block size

Block Size	128 KB	256 KB	512 KB	1 MB	2 MB	4 MB
Maximum TPS	34	68	136	273	546	1092
Average TPS	33	59	128	246	482	945

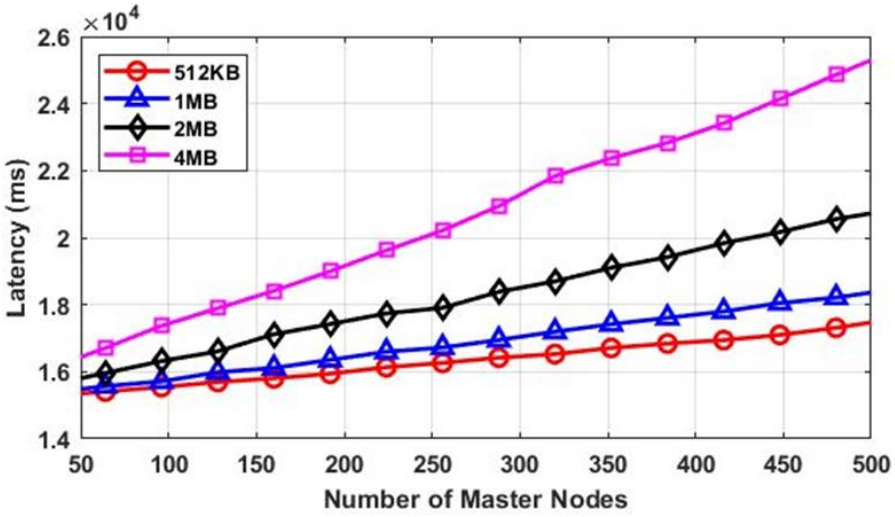

Fig. 4. Latency experiment results based on number of MNs

6. Conclusions

In this paper, the BCP blockchain system is proposed, which is uses the PoP instead of the PoW mechanism. The BCP system model was implemented on the new ***Bryllite*** blockchain international game network. The proposed BCP scheme encourages user participation to the blockchain network by giving block rewards to individual users in a fair fashion. Future work will focus on building more secure and flexible fast blockchains to support massive user networks, game companies, and gamers that need instantaneous real-time transactions.

References

[1] Satoshi Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. Article (CrossRef Link)

[2] M. Ghaffari, B. Hariri, S. Shirmohammadi, and D. T. Ahmed, "A Dynamic Networking Substrate for Distributed MMOGs," *IEEE Trans. Emerging Topics in Computing*, vol. 3, no. 2, pp. 289-302, Jun. 2015. Article (CrossRef Link)

[3] “Bryllite Platform Beyond the Game Boundaries,” Oct. 2, 2018. Article (CrossRef Link)

[4] T. Neudecker and H. Hartenstein, "Network Layer Aspects of Permissionless Blockchains," *IEEE Commun. Surveys & Tutorials*, vol. 21, no. 1, pp. 838-857, 1st quarter, 2019. Article (CrossRef Link)

[5] J. R. Douceur, “The Sybil attack,” in *Proc. of 1st Int. Workshop Peer-to-Peer Syst.*, pp. 251-260, 2002. Article (CrossRef Link)
[6] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance,” in *Proc. of the Third Symposium on OSDI*, vol. 99, pp. 173-186, 1999. Article (CrossRef Link)
[7] K. Yeow, A. Gani, R. W. Ahmad, J. J. P. Rodrigues, and K. Ko, “Decentralized Consensus for Edge-Centric Internet of Things: A Review, Taxonomy, and Research Issues,” *IEEE Access*, vol. 6, pp. 1513-1524, 2017. Article (CrossRef Link)
[8] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling Localized Peer-to-Peer Electricity Trading Among Plug-in Hybrid Electric Vehicles Using Consortium Blockchains," *IEEE Trans. Industrial Informatics*, vol. 13, no. 6, pp. 3154-3164, Dec. 2017. Article (CrossRef Link)
[9] J. Gu, B. Sun, X. Du, J. Wang, Y. Zhuang, and Z. Wang, "Consortium Blockchain-Based Malware Detection in Mobile Devices," *IEEE Access*, vol. 6, pp. 12118-12128, 2018. Article (CrossRef Link)
[10] B. Shahzad and J. Crowcroft, "Trustworthy Electronic Voting Using Adjusted Blockchain Technology," *IEEE Access*, vol. 7, pp. 24477-24488, 2019. Article (CrossRef Link)
[11] “Avalanche effect,” *Wikipedia*, Dec. 4, 2018. Article (CrossRef Link)
[12] “EOS.IO Technical White Paper,” *Github*, 2018. Article (CrossRef Link)
[13] Bryllite test simulation, *Github*, 2019. Article (CrossRef Link)

Jusik Yun (awp212@yonsei.ac.kr) received a B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, Republic of Korea, in 2016. He is currently pursuing a combined M.S. and Ph.D. degree in electrical and electronic engineering at Yonsei University, where he is also a researcher of the Communications and Networking Laboratory (CNL). His current research interests include blockchain, trust systems, and machine learning.

Yunyeong Goh (rhdbsdud@yonsei.ac.kr) received a B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, Republic of Korea, in 2019. He is currently pursuing a combined M.S. and Ph.D. degree in electrical and electronic engineering at Yonsei University, where he is also a researcher of the CNL laboratory. His current research interests include blockchain, trust systems, and augmented reality.

Jong-Moon Chung (jmc@yonsei.ac.kr) received B.S. and M.S. degrees in electronic engineering from Yonsei University and Ph.D. in electrical engineering from the Pennsylvania State University. Since 2005, he has been a professor in the School of Electrical and Electronic Engineering at Yonsei University, where he is currently the Associate Dean of the College of Engineering. From 1997 to 1999, he was an assistant professor and instructor at the Pennsylvania State University. From 2000 to 2005, he was with the Oklahoma State University as a tenured associate professor. He is the Vice President of the IEEE Consumer Electronics Society, Editor of the *IEEE Transactions on Vehicular Technology*, Associate Editor of the *IEEE Transactions on Consumer Electronics*, Section Editor of the *Wiley ETRI Journal*, and Co-Editor-in-Chief of the *KSII Transactions on Internet and Information Systems*.

Ok-Seok Kim (jade@hanbitsoft.co.kr) is an expert in game and software development with 18 years of experience. He has been in charge of game server / client development at Gargamel.com a game portal subsidiary of Hanbitsoft Co., Ltd. as well as at Freecal. He went on to work at SK Telecom and developed their SMS billing system. At LG MC Labs he developed a feature phone SNS client and has worked on a variety of projects beyond gaming. In 2010, he joined TCOMMS and successfully developed the Liveicon service. He is a veteran developer who has been involved in the Bryllite platform since its inception.

SangWoo Shin (swoows@hanbitsoft.co.kr) graduated from Kyungwon College in Architecture. In 2003, began developing feature phone games. And progressed to client and server for online PC games, as well as iOS and Android nativer app development. He has 15 years of experience developing mobile games on Unity game engine , using a wide range of platforms and programming languages. His most notable work was as Gravity's main programmer was Ragnarok Mobile Thieves. He was the main server programmer at Sonov for the game Club Starking. At Hanbitsoft, he has developed the mobile soccer simulation game FCMM, the tank strategy simulation game Hagane Orchestra, and fitness apps Run Day and Fit Day.

Jin Choi (program@t3.co.kr) graduated from Dong-A University, Busan, Korea in Industrial Engineering. He joined T3 Entertainment, where he developed and led server, DB, and security teams for the dance battle online game, Club Audition. The game was launched in 10 different countries in Asia, Europe and South America. He served as a specialist in the development of MMORPG Hellgate London, FCM a soccer simulation game, and online fishing game Grand Mer. In addition, he was the team Manager and Server Programmer for the online game Audition3, MMORPG Camon Hero, and Mythos. Currently, he lead the AR team that is developing an augmented reality training simulator for disaster response. He is also involved in blockchain, cryptocurrency, and the Bryllite Project.

Yoora Kim (yoora@t3.co.kr) has a total of 18 years of gaming business experience. She graduated from Hankuk University of Foreign Studies in International Trade. She went to work at T3 Entertainment as the Marketing Team Lead for the game, Audition. As a game marketing specialist, she held the title of Chief Marketing Officer (CMO). After the acquisition by HanbitSoft in 2008, she served as the Head of Business, before becoming the CEO of HanbitSoft and HanbitSoft Japan Corporation. Kim Yura led the global success of the dance game, Audition. She maintained a network of more than 700 million game users, with business partners in China, Southeast Asia, North America and Europe. In addition, Hanbitsoft has been actively promoting the 4th industry capability of AR, VR, and drone technology. She was also appointed as the CEO of Bryllite Hong Kong to oversee Bryllite's ICO in Hong Kong.