STUDYING THE APPLICABILITY OF PROOF OF REPUTATION(POR) AS AN ALTERNATIVE CONSENSUS MECHANISM FOR DISTRIBUTED LEDGER SYSTEMS

Oladotun Aluko[1] and Anton Kolonin[2]

[1]Novosibirsk State University, Novosibirsk, Russia
[2]Aigents Group, Novosibirsk, Russia

ABSTRACT

Blockchains combine several other technologies like cryptography, networking, and incentive mechanisms in order to support the creation, validation, and recording of transactions between participating nodes. A blockchain system relies on a consensus algorithm to determine the shared state among distributed nodes. An important component underlying any blockchain-based system is its consensus mechanism, which determines the characteristics of the overall system. This thesis proposes a reputation-based consensus mechanism for blockchain-based systems which we term Proof-of-Reputation(PoR) that uses the liquid rank algorithm where the reputation of a node is calculated by blending the normalized ratings by other nodes in the network for a given period with the reputation values of the nodes giving the ratings. The nodes with the highest reputation values eventually become part of the consensus group that determines the state of the blockchain.

KEYWORDS

Consensus, Distributed Ledger Technology, Blockchain, Reputation, Social Computing.

1. INTRODUCTION

In the last couple of years, blockchain has received a significant amount of attention from the industry and academia alike and quite rightly so due to the success of cryptocurren- cies. While cryptocurrencies are the most popular use case for blockchain technology, there is a plethora of application domains. A blockchain system is, fundamentally, a distributed system that relies on a consensus algorithm to determine shared state among distributed nodes. In blockchain speak, this shared state called a chain is a public or private record of all transactions or digital events that have been created and shared among participating nodes. A blockchain system is, fundamentally, a distributed system that relies on a consensus algorithm to determine shared state among distributed nodes. In blockchain speak, this shared state called a chain is a public or private record of all transactions or digital events that have been created and shared among participating nodes [1].

The main objective of any blockchain-based system is to maintain a live decentralized transaction ledger while defending against attacks from malicious Byzantine actors that may try to game the system. The reliability and the integrity of the entire blockchain system as a whole depend largely on the consensus model employed. The applicability of any consensus mechanism is based on three key properties: safety, liveness and fault tolerance [2].

David C. Wyld et al. (Eds): BIoT, DKMP, CCSEA, EMSA - 2021
pp. 41-53, 2021. CS & IT - CSCP 2021 DOI: 10.5121/csit.2021.110804

Distributed consensus among nodes geographically distributed has been a widely studied research topic in distributed systems, however, with the advent of blockchain, it has received more attention as blockchains are a type of distributed system. Most blockchain-based systems targeting different application domains with an array of unique requirements have introduced a corresponding consensus mechanism suited for its particular uses. As a result of this, several consensus algorithms have emerged with different properties and capabilities.

The common approach to building consensus among participants is evidence-based. The basic concept of a proof-based consensus algorithm is that among many nodes joining the network, the node that performs sufficient testing is given the right to add a new block to the chain and gets a reward [3]. A large number of these methods are still vulnerable to the games of the participants on the network. The most popular proof-based algorithm is the PoW (Proof of Work) consensus algorithm, which powers the Bitcoin cryptocurrency, where each participant in the system votes by the total amount of computing power that the participant controls at the time of the vote. The obvious disadvantage of this approach is that anyone with the most computing power can essentially take over a significant portion of the system. In addition, it has been known to consume a significant amount of resources. a lot of electricity [4]. Proof of Stake (PoS) is an energy-saving alternative to PoW. PoS requires nodes to demonstrate ownership of a particular stake as it is believed that nodes with more coins are less likely to attack the network.

As the nature of peer-to-peer (P2P) networks is open and dynamic, the security risk within that environment is greatly increased mostly because nodes can join and leave the network at will. Thus, it is important to have a system that can check against malicious behaviour. One way to minimize risks associated with this type of open communities is to use community-based reputations. Historically, reputation systems have been employed to facilitate trust between entities [5]. The reputation of a node defines an expectation about its behavior, which is based on other nodes' observations or information about the node's past behavior within a specific context at a given time.

The Proof of Reputation (PoR) consensus algorithm is about decentralizing reputation such that the reputation dynamics of each member in the system can be measured and tracked [6]. This consensus algorithm can be seen as the application of a reputation systems model to the blockchain. PoR adopts the concept of the balance of power and designs a decentralized incentive system that ensures that new and existing users have the same opportunity to receive rewards from the system. PoR can prevent the distribution of power from being centralized due to incomplete incentive designs. It is particularly useful in the design of social decentralized applications.

The main contributions of this work are described as follows:

- First, in our reputation-based consensus mechanism, the reputation of a node is not simply calculated by the value of the direct rating given by other nodes but by blending together a normalized set of ratings and the corresponding reputation values of the node providing the rating at a given period in time. The behaviour of a node affects its overall reputation value
- Second, our reputation-based consensus mechanism is based on the following principles: 1) The liquid nature of the reputation values. The reputation value computed for a node is based on the reputation value of the node providing the rating. 2) The temporal scoping of reputation so that reputation values collected by members in the past are less contributing to the current reputation value. 3) The openness of all reputation values to all members in the community so that audits can be performed.

- Third, we use a side chain to store the reputation values of all the nodes without the use of a third party to manage reputation
- Finally, we develop an experimental implementation and evaluate its performance in terms of security and the throughput of the system.

2. REVIEW OF RELATED WORK

2.1. Consensus Algorithms

Blockchains solve the Byzantine Generals Problem [7] which is a problem associated with distributed systems. This problem is addressed with the method of verifying the transactions by many distributed nodes. The data can be delivered between different nodes usually through a broadcast message. However, some nodes may be maliciously attacked, which could lead to a situation where changes are made to communication contents. Every node in the network needs to distinguish the information that has been tampered and obtain the consistent results with other normal nodes. This is usually done through a consensus algorithm.

Consensus is central to the Blockchain Technology [8], [9]. It has been studied for well over three decades. Consensus protocols have been historically known to enable consensus to be reached about a shared among a set of distributed nodes. The design of a consensus protocol is a challenging task and so it is usually common to make assumptions under which the protocol is proven to function properly. These assumptions eventually influence the characteristics of the consensus protocol. In fact, it's the case that most applications of the Blockchain Technology usually roll out their corresponding consensus algorithm to fit the specific use case for the Technology [10], [11], [12].

Proof-of-Work (PoW) is by far the most widely used consensus mechanism for blockchains introduced by Bitcoin [2], [13]. In PoW nodes acting as miners vote by the amount of computing power they possess by trying to solve a computational challenge. The first node to solve the challenge validates and adds a new block of transactions to the blockchain and gets a reward for this action. However, the generation of blocks requires the use of a huge amount of computational power and introduces delay for block confirmation, resulting in low efficiency and low transaction throughput.

Proof-of-Stake (PoS) was proposed as an alternative to PoW. With PoS, nodes who like to participate in the block creation process must prove ownership of a certain amount of coins. In addition, they are required to lock a certain amount of currency, called stake, to participate in the block creation process. A variation of PoS is the Delegated-Proof-of-Stake(DPoS), in which miners are elected by other nodes. The stake of the nodes are used as the weighting parameter for votes.

2.2. Reputation-Based Consensus Algorithms

Reputation has been defined as a quantity derived from the underlying social network which is globally visible to all members of the network [14], [15], [16]. Reputation systems have historically been known as a means of harnessing reputation data in some form. They work by facilitating the collection, aggregation and distribution of data about an entity. This data can thereafter be used to characterize and predict that entity's future actions [17], [18]. Essentially, by referring to the reputation data, users within a network are able to decide whom they will trust, and to what degree. In addition to above, a reputation system is a socially corrective mechanism, as the incentive of positive reputation and the disincentive of negative reputation will generally

encourage good behavior over the longer term. Upon the collection of reputation data by a reputation system, it can be shared amongst users which in turn can be used to evaluate other users before making decisions about intended or future interactions, without ever having to have previously interacted. Examples of the practical application of this system can be found on eCommerce websites like Amazon or eBay where reputation attributed to a seller is influenced by ratings through previous transactions. Another use case is in government where a country like China incentives the behavior of the citizens through a social credit score system. Earlier works proposed possibilities of applying a reputation-based model to distributed computing. One such was described by [19]. The downside was that the approach was not completely decentralized.

Recent studies have introduced reputation systems into the blockchain space to improve efficiency and reliability. [20] proposed a reputation-based consensus mechanism for peer- to-peer networks where reputation serves as the incentive for good behaviour and the node with the highest reputation gets to publish a new block. At the end of each interaction between two nodes, feedback is generated by the service requester and broadcast to the entire network. On reaching the set threshold, nodes start to calculate a ranking list after which the node with the highest ranking publishes the new block and other nodes verify the integrity of the newly published block. [21] also proposed a reputation-based consensus mechanism based on the proof-of-work consensus algorithm. In their approach, a miner's voting power is given by its reputation. The reputation for each miner is computed based on the total amount of valid work a miner has contributed and also the regularity of that contribution over a given period. [22] proposed the Blockchain Reputation-Based Consensus(BRBC) mechanism in which a node in the network must have a reputation score higher than a set threshold to be able to publish a new block. Also, a judge is randomly selected that is responsible for updating node reputation values. However, none of these address the behaviour of a node on a transactional basis as it interacts with other nodes in the network.

3. SYSTEM OVERVIEW AND THREAT MODEL

In our approach, we consider that the network is untrustworthy and unreliable, which means messages in the network can be delayed, duplicated or lost in some cases. Furthermore, the nodes are heterogeneous and failure at a node does not cause the failure of another node.

We also assume that there's a social community that affords nodes the ability to vote about different aspects of the system. Each node i is identified by a public key pki similar to a wallet in regular blockchains like Bitcoin. This public key has a corresponding secret key ski with which it can use to append its signature to transactions. Each transaction group is made up of two nodes [i, j]. Node j gives the rating while node i is the recipient of this rating usually with respect to an interaction between them. The transaction is said to be completed only after it is appended to the blockchain.

During the consensus phase, a node can be a leader of the consensus group or simply a member of the consensus group. For smaller networks, all the nodes in the network can be part of the consensus group. For larger networks, it's impractical to have all nodes as members of the consensus group. In those instances, a subset of nodes within the network that have the highest reputation corresponding to at least two-thirds of the entire reputation values in the network should be used. The leader for the consensus round can then be selected at random.

In addition, we assume that there is a malicious node within the network that may cause the failure or misbehaviour of a number of nodes. In order for the system to be safe and live, we assume that if F nodes eventually become faulty, at least 3F + 1 nodes remain honest.

4. THE CONSENSUS MECHANISM

4.1. Consensus Group

In our scheme, we assume N nodes in the network, an individual node is represented as p_i, $i \in N$. The computation performed by network nodes happens in rounds during which a node sends messages, receives them and thereafter performs some local computation on the received message [23]. Each node i is identified by a key pair, pk_i which is the public key and sk_i is the corresponding secret key. At the end of every interaction between nodes, rating values are generated with respect to the service. A node will usually function in one of two possibilities: either as the recipient node or as the rater node for the particular interaction. Whenever a rater node gives a rating, it broadcasts the details of that transaction to the entire network. We denote this interaction where rater node is i and node j is the recipient node as follows:

$$Transaction = (E_{sk_i}, pk_j, r) \qquad (1)$$

$$\{r : 0 < r < 1\}$$

where pkj is the public key of the recipient node, r is the rating given by the rater node which is a value between 0 and 1, and Eski is the encrypted transaction data signed using the rater node's secret key. Transactions generated between nodes for a round k are added to a list of pending transactions waiting to be appended to the chain during the consensus phase.

At the start of every consensus round, consensus group members need to be selected and added into a consensus group. We denote this consensus group for a round k as Gk. The consensus group members are selected from the nodes with the highest reputation values for which their collective reputation scores are over 50% of the total reputation values of the entire network. With this approach, the size of Gk will vary depending on the reputation distribution in the network. A node that is part of the consensus group is denoted as:

$$pk_i \in G_k \qquad (2)$$

To proceed, a new leader L_k for the round k is selected. After the leader for the round k is selected, it serves the following functions:

- Packaging all valid transactions from the list of pending transactions to a $Block_k$
- Calculating the new reputation values for all network nodes for $Reputation_k$ the round k using data from transactions in the transaction list
- Broadcasting the commit message to the consensus group G_k

4.2. Leader Selection

We use a random function to select the leader for the consensus group for the round G_k. By doing this, there is no deterministic guarantee for which node will be selected as leader for the consensus round k. As such, all nodes that have been selected as members of the consensus group for the round k have equal chance of being selected. The leader's public key is broadcasted to the consensus group before the start of the consensus phase. The leader L_k packages all the transactions T_i for a recent time window in a specific order and adds them into a block. Afterwards, the leader sends a commit message to the consensus group G_k. This message contains

the newly packaged $Block_k$, the leader's public key pk_L, reputation list for the round k, and also a hash of the $Block_k$ generated using the leader's secret key:

$$< Block_k, pk_L, Hash(Block_k), ReputationList > \quad (3)$$

4.3. Block Publication

After a commit message is sent by the leader L_k to the consensus group G_k, the consensus group determines the final block to be published for the current round k. Each node $pk_i \in G_k$ checks the commit message sent by the leader L_k which contains both the $Block_k$ and $Hash(Block_k)$. First, the node checks the pkL in the commit message to see if it matches the pkL that was broadcasted upon the leader selection earlier; otherwise, it can ignore the commit message. It then proceeds to check the integrity of the hash using the pkL. Afterwards, it checks the validity of the transactions within the $Block_k$. If this process completes, it sends a new commit message back to the consensus group G_k. This process continues with every node in the consensus group G_k. Upon completion, each node pk_i that successfully verifies the $Block_k$ and the *ReputationList* sends a verification commit back to the consensus group G_k.

$$< VERIFY, Block_k, Hash(Block_k), ReputationList > \quad (4)$$

The consensus group waits until at least a certain amount of consensus members sends in this message. This message constitutes a consensus group vote. We formalize this using a social choice function. For a set of nodes in the consensus group for round G_k, each node has an associated weight w assigned which is equivalent to its reputation value from the previous round $k - 1$. During the consensus process, there's a minimum quota which has to be reached for decisions to be made. We set this quota at two-thirds of the total weight in the consensus group:

$$d(G_k) = \begin{cases} 1 & \sum_{i \in G_k} w_i > q \\ 0 & \text{otherwise} \end{cases}$$

where $d(G_k)$ represents the decision of the consensus group. Whenever the $d(G_k)$ is 1, it means consensus for round k has been reached.

5. REPUTATION SYSTEM

A node's reputation is defined by an evaluation of the ratings it receives from others in the past. These ratings reflect the degree of trust that other nodes have on a specific node based on their past interactions. Reputation-based systems generally rely on feedback to evaluate a node. This feedback is generally in terms of the amount of satisfaction a node receives by interacting with another node in the network. [24] pointed out that when considering reputation information, the source of information and the context need to be accounted for. We define the reputation principles for approach adapted from [25] as follows:

- The liquid nature of the reputation values. The reputation value computed for a node is based on the reputation value of the node providing the rating.
- The temporal scoping of reputation so that reputation values collected by members in the past are less contributing to the current reputation value.

- The openness of all reputation values to all members in the community so that audits can be performed.

Let s_i denote the reputation for a recipient node *i*. All nodes in the network start with a default reputation value determined on system initialization. During node interactions, a node's reputation value is determined by the liquid rank algorithm [6]. This approach can be used as a predictive metric to evaluate a node's behaviour. For each round, a node can receive multiple unique ratings:

$$s_{i1...n} = \{s_{i1}, ..., s_{in}\} \tag{5}$$

where the range of s_i is *[0, 1]*. Values s_i are then normalised as follows:

$$S_{i,n} = \frac{S_{i,n} - min_i(S_{i,n})}{max_i(S_{i,n}) - min_i(S_{i,n})} \tag{6}$$

We slightly modify the normalization of the rating values to prevent null values from the set of ratings as follows:

$$S_{i,n} = \frac{(S_{i,n} - min_i(S_{i,n})) + 1}{(max_i(S_{i,n}) - min_i(S_{i,n})) + 1} \tag{7}$$

Furthermore, we define the ratings matrix *S* to be $[s_{ij}]$. After each round, these ratings will be generated for all nodes in the network. To compute new reputation values for a node for the round *k*, we blend these ratings with the rater reputation values from the previous round $k - 1$. We denote this as:

$$P = \overrightarrow{S} * \overrightarrow{R} \tag{8}$$

where $\overrightarrow{S} = [s_{ij}]$ and $\overrightarrow{R} = [r_{in}]$. r_{in} corresponds to the rater node providing the rating.

To compute the reputation value for the round *k*, we then blend the initial node's reputation value with the current rank generated from the ratings.

$$R_{i,k+1} = \alpha * P + (\alpha - 1) * R_k \tag{9}$$

where α is a constant determined on system initialization. The value is set between 0 and 1. It determines what portion of the equation to give more priority to. If the value is set closer to 1, it means that the newly generated reputation value will give more priority to the ratings *P* and less priority to the previously generated reputation value. This is what we want as this aligns with the reputation principles stated earlier. It helps to reduce the impact of nodes that change behaviour over time. Further, to prevent reputation values from hopping, we clamp the values using a sigmoid function as follows:

$$R'_{i,k+1} = \frac{R_{i,k+1}}{\sqrt{1 + (R_{i,k+1})^2}} \tag{10}$$

6. REPUTATION STORAGE

Most existing reputation mechanisms use a central server for the storage, management and sometimes distribution of reputation values among network nodes. In our approach, we do not require a central server but the reputation value for each node in the network is managed through a reputation side chain connected to the main transaction chain. The structure of the reputation chain is such that it has a header which contains meta information about a specific block and then the reputation values for all the network nodes:

$$ReputationBlock_i = (ReputationBlockHeader_i, ReputationList_i) \quad (11)$$

The $ReputationList_i$ contains a list of all network nodes with their associated reputation values for the most recent round. This serves as a lookup data structure for future uses. The $ReputationBlock_i$ as well as the transaction block use the standard blockchain block structure with a hash of all the transactions for the round, a previous hash, timestamp and transactions.

A new reputation block is created along with a normal transaction block during the consensus phase. So for a consensus round k, a $Block_k$ which is added to the transaction chain corresponds to a $ReputationBlock_k$ which is added to the reputation chain. As stated in section 4, part of the duties of the consensus group is to validate the reputation calculation generated by the leader L_k. After the consensus is reached, the leader broadcasts the new $ReputationBlock_k$ to the entire network and as such the reputation value of all the nodes in the network is visible to all other nodes.

7. EXPERIMENTS AND RESULTS

For our prototypes, we built an experimental protocol that implements the protocol. Thereafter, the nodes were deployed on AWS EC2 remote server running on 16GB RAM with Amazon's t3 processor. In the experiment, we set up 1,000 nodes. We used a default initial reputation value of 0.2. We set the value of α to 0.6, the effect of that is that we give priority to recently generated reputation values.

To simulate the effect of a Wide Area Network, we impose a round trip latency of 200ms. While it's unlikely that this will be the case in reality, the average of network delays across the entire network will average out to a close enough value.

In terms of the throughput, Figure 1 shows how the throughput values change as we vary the number of network nodes from 500 to 1,000 nodes. As the network size increases, so does the throughput because as more nodes join the network, more messages are being transferred in the network.

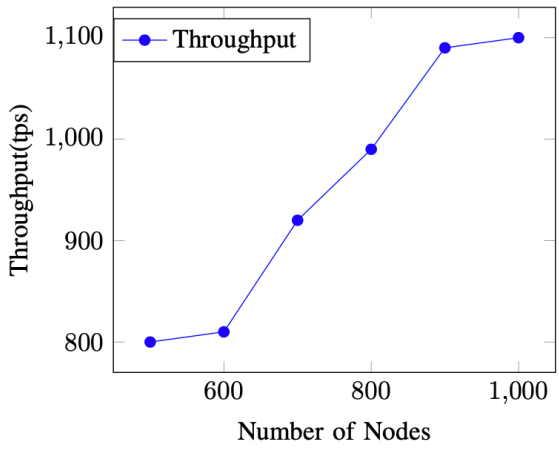

Figure 1: Throughput vs Number of network nodes

Also, we vary the number of transactions in a single block, we vary them between 100 and 500 to measure the average time it takes for a new block to be produced. Figure 2 shows that it takes more time for a new block to be produced as the transactions in a block increase. This is so because more transactions are now in a single block and so it takes more time for those transactions to be processed.

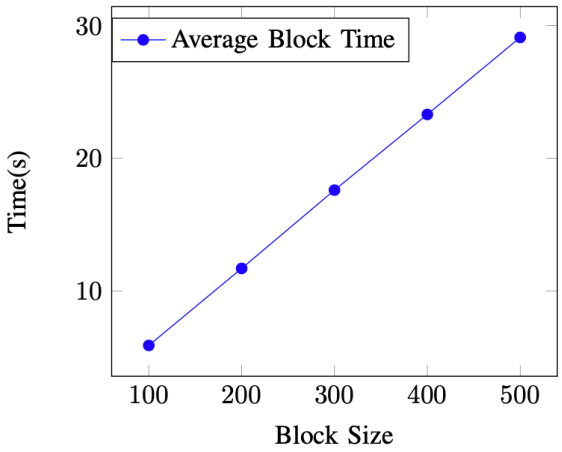

Figure 2: Average Block Time as the number of transactions in a single Block is varied

In our final experiment, we measure the consensus time in relation to the block size for each block. We observed that when the block size is relatively small around 100 transactions in each block, consensus takes about 2 seconds. As we increase the block size, so does the consensus time increase as well.

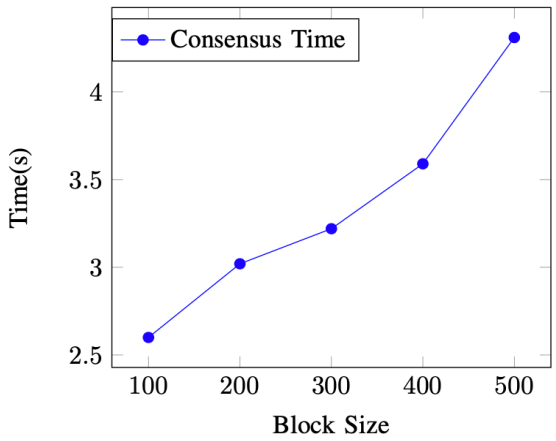

Figure 3: Consensus Time as the number of transactions in a single Block is varied

8. SECURITY ANALYSIS

In our reputation consensus mechanism, the right to generate a new block is reserved to the leader *L*k for a round and the consensus group members for that round, in each round, only the highest ranking nodes are selected to be members of the consensus group. Unlike PoW where an adversary can attack the system only by having sufficient compute power. In addition, the leader election is based on random selection and there's no way for an adversary to deterministically predict the outcome of the selection process. As it takes time for reputation values to grow, an adversary will need to spend a lot of time doing honest work before it can be added as a consensus group member. For cases where an adversary becomes the leader for a round, all consensus group members still need to vote as regards the Block that will eventually be appended to the chain. Only when an adversary controls a significant number of members in the consensus group, at least two-thirds can the security of our approach be tampered with.

8.1. Selfish Mining Attacks

Selfish Mining attacks [26], [27], [28] is a mining strategy where a group of miners collude to exert power over the entire blockchain in order to increase their revenue. In selfish mining attacks, two groups exist side-by-side: an honest group of miners following the standard protocol and a colluding group that follows the selfish mining strategy. The selfish miners mine blocks while keeping them secret, they continue this process until the fork created from the main chain is longer than the main chain. In our approach, since blocks are not mined based solely on the compute power a node possesses or a group of nodes collectively possess, this kind of attack is impossible. Furthermore, there is no way for a node to know the nodes that will be involved in the consensus for a round or which node will be selected as the leader for that round.

8.2. Eclipse Attack

An eclipse attack [29] happens whenever a node in the network is occluded from the rest of the network. Most of the external contact for that node is controlled by the malicious node that launched the attack. This attack is a serious threat to any blockchain. In the case of our approach, the effect of this type of attack is only noticeable if an attacker is able to simultaneously isolate multiple consensus group members which is highly unlikely.

8.3. Flash Attacks

Flash attacks [30] happen whenever an attacker can pur- chase or rent compute power for a short period with the intention of using this compute power to its advantage. This type of attack is only feasible with network types like PoW that require the use of compute power. In our approach, an attacker with a sufficiently large amount of compute power cannot simply launch an attack on the basis of its compute power.

9. CONCLUSIONS

In this work, we proposed a reputation-based consensus mechanism for distributed ledger systems. The consensus scheme uses a social choice function where the weight of nodes that are responsible for consensus is equivalent to the reputation value for that node. In addition, approach uses the liquid rank algorithm where the reputation of a node is calculated by blending the normalized ratings by other nodes in the network for a given period with the reputation values of the nodes giving the ratings. Finally, we built an experimental prototype to show the potential of this approach. We remark that there are several other parts of this system which can be improved and are left to future work.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support and generosity of The Stream Data Analytics and Artificial Intelligence of the Novosibirsk State University without which the present research could not have been carried out.

REFERENCES

[1] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanara- man. (2016) Blockchain technology: Beyond bitcoin. [On- line]. Available: https://j2-capital.com/wp-content/uploads/2017/11/AIR- 2016-Blockchain.pdf?forcedefault=true.

[2] A. Baliga, "Understanding blockchain consensus models," 2017.

[3] G.-T. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain," *J. Inf. Process. Syst.*, vol. 14, pp. 101–128, 2018.

[4] Quantaloop.io. (2020) Types of consensus algorithms in blockchain. [Online]. Available: https://quantaloop.io/proof-of-work-vs-proof-of-stake-101.

[5] F. Hendrikx, K. Bubendorfer, and R. Chard, "Reputation systems: A survey and taxonomy," *Journal of Parallel and Distributed Computing*, vol. 75, pp. 184–197, 2015.

[6] A. Kolonin, B. Goertzel, D. Duong, and M. Ikle, "A reputation system for artificial societies," *arXiv preprint arXiv:1806.07342*, 2018.

[7] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," in *Concurrency: the Works of Leslie Lamport*, 2019, pp. 203–226.

[8] V. Gramoli, "From blockchain consensus back to byzantine consensus," *Future Generation Computer Systems*, vol. 107, pp. 760–769, 2020.

[9] S. Azouvi, P. McCorry, and S. Meiklejohn, "Betting on blockchain consensus with fantomette," *arXiv preprint arXiv:1805.06786*, 2018.

[10] M. S. Ferdous, M. J. M. Chowdhury, M. A. Hoque, and A. Colman, "Blockchain consensuses algorithms: A survey," *arXiv preprint arXiv:2001.07091*, 2020.

[11] L. M. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2018, pp. 1545–1550.

[12] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.

[13] A. Gervais, G. O. Karame, K. Wu¨st, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 3–16.

[14] L. C. Freeman, "Centrality in social networks conceptual clarification," *Social networks*, vol. 1, no. 3, pp. 215–239, 1978.

[15] L. Kleinrock, R. Ostrovsky, and V. Zikas, "Proof-of-reputation blockchain with nakamoto fallback," in *International Conference on Cryptology in India*. Springer, 2020, pp. 16–38.

[16] J. Horton and J. Golden,"Reputation Inflation An Online Marketplace," *New York I*, vol. 1, 2015.

[17] G. Swamynathan, K. C. Almeroth, and B. Y. Zhao, "The design of a reliable reputation system," *Electronic Commerce Research*, vol. 10, no. 3, pp. 239–270, 2010.

[18] K. Hoffman, D. Zage, and C. Nita-Rotaru, "A survey of attack and defense techniques for reputation systems," *ACM Computing Surveys (CSUR)*, vol. 42, no. 1, pp. 1–31, 2009.

[19] M. Gupta, P. Judge, and M. Ammar, "A reputation system for peer- to-peer networks," in *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, 2003, pp. 144–152.

[20] F. Gai, B. Wang, W. Deng, and W. Peng, "Proof of reputation: A reputation-based consensus protocol for peer-to-peer network," in *International Conference on Database Systems for Advanced Applications*. Springer, 2018, pp. 666–681.

[21] J. Yu, D. Kozhaya, J. Decouchant, and P. Esteves-Verissimo,"Repucoin: Your reputation is your power," *IEEE Transactions on Computers*, vol. 68, no. 8, pp. 1225–1237, 2019.

[22] M. T. de Oliveira, L. H. Reis, D. S. Medeiros, R. C. Carrano, S. D. Olabarriaga, and D. M. Mattos, "Blockchain reputation-based consensus: A scalable and resilient mechanism for distributed mistrusting applications," *Computer Networks*, vol. 179, p. 107367, 2020.

[23] P. Berman, J. A. Garay, K. J. Perry *et al.*, "Towards optimal distributed consensus," in *FOCS*, vol. 89. Citeseer, 1989, pp. 410–415.

[24] L. Xiong and L. Liu, "Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843–857, 2004.

[25] A. Kolonin and S. SingularityNET, "Reputation systems for human- computer environments," *Complexity, Informatics and Cybernetics*, 2019.

[26] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *International conference on financial cryptography and data security*. Springer, 2014, pp. 436–454.

[27] C. Grunspan and R. Pe´rez-Marco, "On profitability of selfish mining," *arXiv preprint arXiv:1805.08281*, 2018.

[28] K. A. Negy, P. R. Rizun, and E. G. Sirer, "Selfish mining re-examined," in *International Conference on Financial Cryptography and Data Security*. Springer, 2020, pp. 61–78.

[29] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *24th {USENIX} Security Symposium ({USENIX} Security 15)*, 2015, pp. 129–144.

[30] J. Bonneau, "Why Buy When You Can Rent?" in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 19–26.

AUTHORS

Oladotun Aluko received his BSc(2017) in Computer Science and Engineering from Obafemi Awolowo University, Nigeria. He is currently working on his MSc in Big Data Analytics and Artificial Intelligence at Novosibirsk State University, Novosibirsk, Russia. His research interests Distributed Computing, Blockchain Technology, Machine Learning and Cloud Databases.

Anton Kolonin received his Ph.D in 1998 after he independently developed a software-algorithmic complex for processing geophysical data, introduced into production in many CIS countries. He has also participated as a leader or lead architect in many projects to develop algorithms and software, including those related to the use of AI, including the recognition of static text, moving objects, music, extracting information from texts and identifying events on financial markets – in Russian and foreign companies. Since 2017, he is also a software architect for AI and blockchain in the Singularity NET project, leading work care of unsupervised language learning and reputation systems.