UOS: Distributed ledger data management protocol.

September 25, 2018

Abstract

The concept of a cryptographically protected and distributed transaction ledger has demonstrated its efficiency in a series of projects. The fourth industrial revolution sets a new level of standards for decentralized protocols. The UOS project's approach to the distributed ledger is filtered through the lens of the management of an ever-increasing sea of data produced by a digital society. The protocol architecture aims to correspond with this management goal, using the Delegated Proof of Importance (DPoI) consensus algorithm. DPoI is a high-performance energy effective, network growth inducing algorithm that rewards network participants for the system's economy enhancing operations. Delegated Proof of Importance is an upgrade to the existing blockchain solutions that integrate the concepts of the Delegated Proof of Stake (DPoS) and the Network Theory. The system protocol is designed in accordance with business and end-user requirements such as privacy, transparency, and smooth volatility, which is achieved due to adaptive emissions proportionate to both network activity growth and the volume of the network itself, according to Metcalfe's law [1].

1 Introduction

Recent years have marked an unprecedented growth in the interest in blockchain technologies. So far, this interest has primarily been seen in the form of distributed payment networks. These networks are decentralized and enable fast low-cost transactions avoiding middlemen. Although the general economic pros and cons of blockchain-based networks are subject to rigorous examination, within this paper, we would like to consider the technical aspects of blockchain consensus algorithms.

The blockchain consensus algorithm is a mechanism that allows the network nodes to reach consensus about the contents of a distributed ledger. The consensus algorithm is the very basis of any blockchain network, which also noticeably determines its technical characteristics.

1.1 Previous Work

The problem of a distributed consensus for networks with potentially fraudulent participants (the Byzantine Generals' Problem) was stated long before the creation of blockchain, in 1982. [2] Since then, an array of different solutions have been developed [3]. However, the first solution that did not rely on a trusted third party, was the Proof-of-Work (PoW) algorithm, described in Satoshi Nakamoto's article. [4] Despite its advantages, PoW inherently has a number of shortcomings, namely scalability and performance problems [5], security [6], progressive centralization of the networks around the largest mining pools [8], and most importantly, the need to use vast volumes of real resources, such as electricity and computing power to generate every block [9].

By now, the computing resources needed for hashing blocks in Bitcoin, are tremendous and far exceed the computing power of the world's greatest supercomputers. Energy use for mining is comparable to the power consumption of some countries and it continues to grow [7]. In 2012, in order to mitigate these shortcomings, the PPCoin (now known as PeerCoin) cryptocurrency became the first to utilize an alternative consensus algorithm, Proof-of-Stake (PoS) [10]. In PoS consensus networks, the probability of creating a new block depends on the volume of tokens in a participant's account. PoS also turns out to have several drawbacks and in its current state, according to a number of experts, cannot serve as an adequate replacement to PoW [11], [12]. One of the major weaknesses of PoS is that it additionally motivates users to concentrate all funds in one place or with one user, which leads to centralization of the network. The next iteration of PoS was introduced as the Delegated Proof-of-Stake (DPoS) [13]. Here network members are divided into two groups: members who delegate the authority to create blocks and validate transactions, and validators (producers). This partition provides better scalability and efficiency. Nevertheless, in DPoS there still remains the problem of motivation for a participant to use their assets actively instead of accumulating them, and that has a negative impact on the growth and governability of the network. The Proof-of-Importance consensus algorithm (PoI), was first introduced in the NEM cryptocurrency [14]. PoI incentivizes network participants' activity. The major departure from PoS is that block generation probability and reward distribution depends not only

on the volume of a user's deposits but also on the participant's activity rate and reputation. Thus, the algorithm motivates users to be more active by participating in more transactions and contributing to the network growth. Despite all its merits, POI has some shortcomings in efficiency.

1.2 Project goal

The major goal of UOS is to create a distributed ledger data management protocol with effective power redistribution within the system, which motivates users to participate actively in network development and prevents centralization. Modern blockchain solutions have problems with scalability, security and efficiency, and to solve those problems, UOS Protocol introduces the DPoI (Delegated Proof of Importance) Consensus Algorithm. This consensus algorithm combines the advantages of DPoS and PoI and implements a new feature of delegating validation rights to a limited number of accounts in order to achieve high levels of efficiency and scalability within the network, accounting for transaction activity of the protocol members, and leading to its further development.

2 UOS Protocol Consensus Algorithm Operating Principle

The UOS consensus algorithm (Delegated Proof-of-Importance, DPoI) is based on the EOS core consensus algorithm. In addition to the client's stake amount, our algorithm also considers the client's transactional activity. In UOS Protocol, all of the participants have the option of delegating the right to validate blocks to a limited number of accounts. To prevent activity imitation or fraud between several affiliated accounts, the transaction graph is partitioned into clusters utilizing the SCAN algorithm [15]. In general, the working principle of the UOS Protocol Consensus Algorithm can be explained as follows.

2.1 Importance index Calculation

Importance index, which is used in voting, and could be interpreted as an importance rating of an account i is calculated as follows:

$$r_i = (1 - \omega_a - \omega_s)v_i + \omega_a\pi_i + \omega_s\sigma_i$$

Here v_i is the stake volume index, π_i is the finance activity index calculated with the NCDawareRank algorithm, σ_i if the social network activity index, ω_a and ω_s is the weight coefficients.

Stake volume index is based on the amount of tokens owned by the account, and represents balance relation to the total amount of tokens in the system. Thus, an account with non-zero balance has non-zero importance index. Activity index depends on transaction history of the account. Activity index is calculated only for accounts with the balance exceeding the A_0 threshold. This value is not fixed and is determined by committee members. When calculating the account activity index only transactions with the amount of tokens higher than T_0 are taken into account. That value is also determined by the committee. Hidden transactions are not included. Ancivity index depends on transactions, creation time of which lays in some time interval. The duration of this interval is W blocks. Contribution of every transaction decreases exponentially.

2.2 NCDAwareRank Algorithm

Activity Index is calculated according to the NCDawareRank algorithm using the following recurrent relation:

$$\boldsymbol{\pi}^{(i+1)} = (\eta \boldsymbol{O} + \mu \boldsymbol{M} + (1 - \eta - \mu)\boldsymbol{E})\boldsymbol{\pi}^{(i)}$$

Here $\pi^{(i)}$ is a vector of account importance indexes values. The vector is normalized, i.e. the sum of its elements is 1. $\boldsymbol{O}$ is the outlink matrix, $\boldsymbol{M}$ is the interlevel proximity matrix. See the definitions of the matrices below. η and μ are weight coefficients that determine contributions of the $\boldsymbol{O}$ and $\boldsymbol{M}$ matrices. Their sum must be less than 1. $\boldsymbol{E}$ is the teleportation matrix added to ensure the series is convergent. This is defined as follows:

$$\boldsymbol{E} = \frac{1}{N}\boldsymbol{e}$$

where N is the number of accounts and $\boldsymbol{e}$ is the matrix in which all the elements are equal to 1, the calculation continues until for some i the following condition is fulfilled:

$$norm(\boldsymbol{\pi}^{(i+1)} - \boldsymbol{\pi}^{(i)}) < \varepsilon$$

Here *norm()* is the vector norm defined as the sum of its elements, ε is the predetermined calculation accuracy. As an initial approximation, π_0, a vector with all the elements equal to $\frac{1}{N}$ can be used.

2.3 Outlink matrix calculation

The outlink matrix O is calculated as follows: First the weight matrix is calculated:

$$w_{ij} = \sum_{k|i\to j, h_k \geq H_0 \wedge h_k \leq H_0+W} \theta(a_k - T_0)\theta(s_i - A_0)\theta(s_j - A_0) a_k \exp\left(lnK[\frac{h_k}{D}]\right)$$

where a_k is the sum of transaction k, h_k is the transaction depth (the block order number from the current point, also known as a block height), K and D are transaction contribution decrease parameters, that define how much the contribution of each transaction decreases over time. The purpose of these parameters is that over every D number of blocks, created after the given transaction, the transaction contribution decreases by $w' = Kw$. The sum is taken over all the transactions of a deposit from account i to account j, depth of which lays between H_0 and H_0+W. H_0 and W are parameters, its values are equal $H_0 = 2419200$ and $W = 1000$ currently in the testnet. So, only transactions from time gap, duration of which is W blocks, contribute to activity index.

$$\hat{o}_{ij} = \begin{cases} w_{ji} - w_{ij} & \text{if } w_{ji} - w_{ij} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Then the matrix that was obtained is normalized so that the sum of elements in every column is equal to 1.

$$o_{ij} = \begin{cases} \frac{\hat{o}_{ij}}{\sum\limits_k \hat{o}_{kj}} & \text{if } \sum\limits_k \hat{o}_{kj} > 0, \\ 0 & \text{otherwise} \end{cases}$$

2.4 Interlevel proximity matrix calculation

Let W is a set of all the accounts taken into the Importance Index calculation; W is a set divided into disjoint subsets A_i called NCD-blocks (Nearly Completely Departed). See the SCAN algorithm description for details. For the given "account u", "G_u" is the set of all the accounts, for which the according member of outlink matrix is greater than zero: $o_{uv} > 0$. Then the set of proximal "accounts u" is defined as follows:

$$\chi_u = \bigcup_{v \in \{u\} \cup G_u} A_{(v)}$$

The interlevel proximity matrix is defined as follows:

$$M_{vu} = \begin{cases} \frac{1}{N_u |A_{(v)}|} & \text{if } v \in \chi_u \ , \\ 0 & \text{otherwise.} \end{cases}$$

Where N_u is the number of NCD-blocks in χ_u.

2.5 SCAN algorithm based graph partitioning into clusters

An indirected graph $W = \{V, E\}$ has every vertex representing a client, and every edge representing a non-zero element of the outlink matrix. The structure of the vertex v is the set of all the adjacent vertices:

$$\Gamma(v) = \{w \in V | (v, w) \in E\} \cup \{v\}$$

The structural similarity of two vertices can be defined as follows:

$$\sigma(v, w) = \frac{|\Gamma(v) \cap \Gamma(w)|}{\sqrt{|\Gamma(v)||\Gamma(w)|}}$$

The vertex ε-neighborhood is a set of vertices for which

$$N_\varepsilon(v) = \{w \in \Gamma(v) | \sigma(v, w) \geq \varepsilon\}$$

The *CORE* is a vertex for which the number of elements in the ε-neighborhood is more than μ.

$$CORE_{\varepsilon,\mu}(v) \Leftrightarrow |N_\varepsilon(v)| \geq \mu$$

Vertex w is directly structurally reachable from vertex v if

$$DirREACH(v, w) \Leftrightarrow CORE_{\varepsilon,\mu}(v) \vee w \in N_\varepsilon(v)$$

Vertex w is structurally reachable from vertex v if

$$REACH(v, w) \Leftrightarrow \exists v_1, ...v_n \in V \forall i \in \{1, ...n-1\} DirREACH(v_i, v_{i+1})$$

Vertex v is structurally connected with vertex w if

$$CONNECT(v, w) \Leftrightarrow \exists u \in V REACH(u, v) \vee REACH(u, w)$$

A cluster is a subset of vertices structurally connected to each other. It is possible to show that every vertex can only belong to one cluster. A vertex can also belong to no cluster; in this case it can either be a hub if there are vertices belonging to two different clusters in its environment, otherwise it is an independent vertex.

2.6 Social network activity index

UOS blockchain is integrated with social network. Every account may have social relations with any other account and may have some content belonging to it. Accounts and content objects with relations between them form a graph. Every account and content object obtains rate depending on involving into social relations. We denote this rate as social network activity index. It is contrubute to importance index.

Every type of node has its own social network activity index. We denote as $\boldsymbol{\sigma_A}$ the vector of account indices and as $\boldsymbol{\sigma_C}$ the vector of content indices. Both vectors are normalized to 1:

$$\sum_i \sigma_{Ai} = 1, \sum_i \sigma_{Ci} = 1$$

There are many kinds of relations between nodes. Every kind of relation has several properties:

1. Type of the source node (account or content);
2. Type of the target node (account or content);
3. Direct weight;
4. Reverse weight;
5. Height (age of relation, measured in blocks);
6. Flag "decayable" that controls the dependence of weights on time.

So, a relation can connect either nodes of the same type or nodes of different types. A relation also can be uni-directional or bi-directional.

There are following kinds of relations:

1. Ownership: bi-directional relation bounding an account and a content item;
2. Like: uni-directional relation from an account to a content item;

3. Dislike: uni-directional relation from an account to a content item;
4. Follow: uni-directional relation between two accounts;
5. Trust: uni-directional relation between two accounts.
6. Comment: uni-directional relation between two content items.
7. Membership: bi-directional relation between a company and an account.

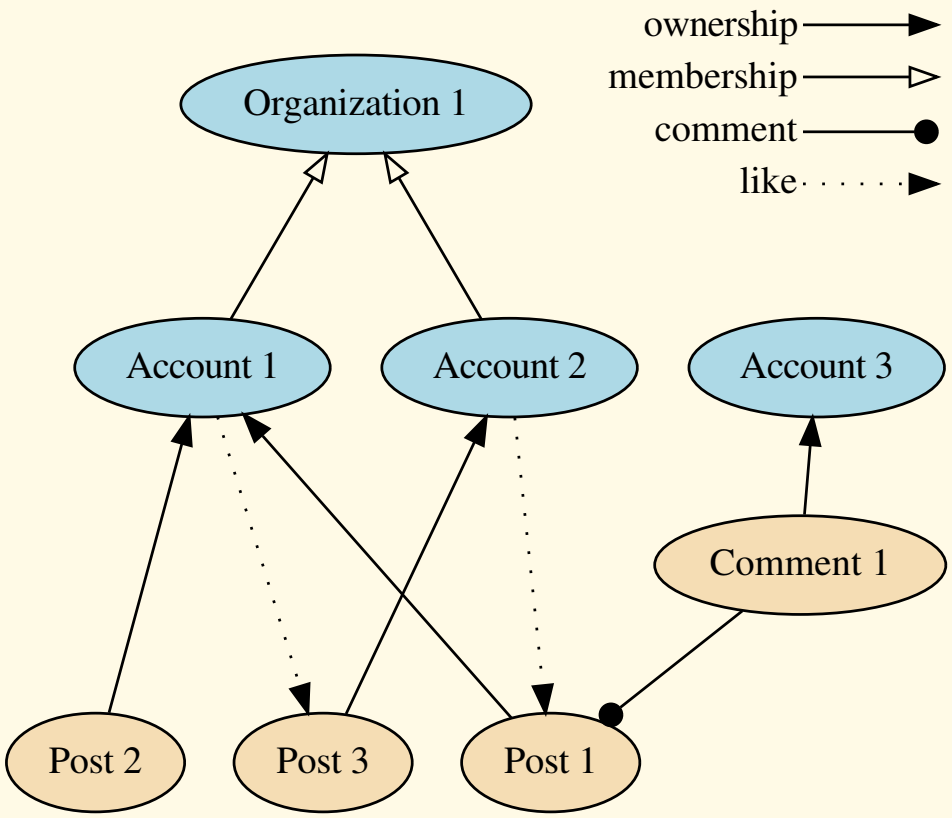

Figure 1: Social network graph example

Following recurrent equation is used for social network activity index calculation:

$$\boldsymbol{\sigma}_A^{(n+1)} = \mu(\boldsymbol{o}_{AA}\boldsymbol{\sigma}_A^{(n)} + \boldsymbol{o}_{AC}\boldsymbol{\sigma}_C^{(n)}) + (1-\mu)(\boldsymbol{T}_{AA}\boldsymbol{\sigma}_A^{(n)} + \boldsymbol{T}_{AC}\boldsymbol{\sigma}_C^{(n)})$$

$$\boldsymbol{\sigma}_C^{(n+1)} = \mu(\boldsymbol{o}_{CA}\boldsymbol{\sigma}_A^{(n)} + \boldsymbol{o}_{CC}\boldsymbol{\sigma}_C^{(n)}) + (1-\mu)(\boldsymbol{T}_{CA}\boldsymbol{\sigma}_A^{(n)} + \boldsymbol{T}_{CC}\boldsymbol{\sigma}_C^{(n)})$$

Where $\boldsymbol{o}_{XX}$ is outlink matrices, $\boldsymbol{T}_{XX}$ is teleportation matrices, μ is numeric koefficient.

In order to construct outlink matrices, we shoud former constrruct weight matrices:

$$w_{XYij} = \sum_{k|(X,i)\to(Y,j)} \alpha_k W_k^R + \sum_{k|(Y,j)\to(X,i)} \alpha_k W_k^D$$

where W_k^D and W_k^R are direct and reverse weights of kth relation. The summation is over all relations connecting a node i of type X and a node j of type Y for reverse weights (and vice versa for direct weights).

α_k is decay multiplier for relation k.

$$\alpha_k = \begin{cases} \exp\left(lnK[\frac{h_k}{D}]\right) & \text{if flag "decayable" is set} \\ 1 & \text{otherwise} \end{cases}$$

where K and D are global parameters, and h_k is height of the kth relation.

Further, outlink matrices can be calculated, based on weight matrix. Weight matrix can contain negative elements, what contradicts the condition of algorytm convergence. So we provide all elements to be not negative.

$$\hat{o}_{XYij} = w_{XYij} + b_{XYj}$$

where b_{XYj} is

$$b_{XYj} = \begin{cases} |\min_k w_{XYkj}| & \text{if } \min_k w_{XYkj} < 0, \\ 0 & \text{otherwise} \end{cases}$$

Further, we normalize columns to 1:

$$o_{XYij} = \begin{cases} \frac{\hat{o}_{XYij}}{\sum_k \hat{o}_{XYkj}} & \text{if } \sum_k \hat{o}_{XYkj} > 0, \\ 0 & \text{otherwise} \end{cases}$$

2.7 The use of importance index in the system

Importance index is used for two purposes:

First, it defines the amount of new tokens which every account receives in case the emission is positive.

Second, importance index defines the account's weight during the voting. The voting allows the delegation of certain powers within the system to a limited number of the accounts.

Producers who own nodes that produce and verify blocks are selected by voting.

Members of the committee are also selected by voting. The committee can vote on blockchain settings, such as the fee amount for a transaction, producer node reward and so on.

3 The Protocol Token

The amount of transactions processed directly depends on the available computing power provided by network members. To allocate the resources efficiently and to avoid spam attacks, a fee in the protocol core cryptographic token is levied on the operations within the UOS network. The protocol allows a user to make token transfer transactions between the network members and launch system smart contracts, e.g. multi-signature, account registration, user tokens creation and so on.

4 Emission

Emission amount at launch is 1,000,000,000 protocol tokens, distributed to the original network accounts to start the protocol. The UOS project implements adaptive emission. The emission volume is calculated regularly once, in a certain time interval, $t_0, t_1, ...t_i$, where $t_{i+1} = t_i + T$. The volume of emission depends on the network activity growth in the preceding time period T.

4.1 Calculating network activity for a period of time

To begin, we calculate the matrix of weights according to the formula:

$$w_{ij}(t_n) = \sum_{k|i \to j, t_k \in [t_{n-1}, t_n]} a_k$$

Here a_k is the amount in k-th transaction, t_k the time at which k-th transaction was created. Summation is performed for all the transactions transferring any amount from account i to account j and created at the time frame from t_{n-1} to t_n.

In fact, each matrix element w_{ij} represents a weight of connection between account i and account j in a given time frame. Next we need to calculate the matrix of connections l:

$$l_{ij}(t_n) = \begin{cases} 1 & \text{if } w_{ji}(t_n) - w_{ij}(t_n) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

We calculate the activity in a given time frame as:

$$A(t_n) = \sum_{i,j} l_{ij}(t_n)$$

In this way, activity is calculated as a number of connections between active accounts in a set timeframe.

4.2 Emission value calculation

Emission value depends on network activity growth.

E_T value is defined as follows, and it is the target value of emission. It defines the upper bound of the aggregate amount of the emission, that is achievable with the following activity value A:

$$\Delta A(t_n) = A(t_n) - A_{max}(t_{n-1})$$

$$E_T(t_n) = \begin{cases} E_T(t_{n-1}) + K_E \Delta A(t_n), & \text{when } \Delta A(t_n) > 0, \\ E_T(t_{n-1}) & \text{otherwise} \end{cases}$$

Here K_E is a coefficient which defines the maximum value of the emission with activity increased by 1. $A_{max}(t_{n-1})$ is the previous maximum value since the system launch:

$$A_{max}(t_{n-1}) = max\Big(A(t_i), t_i \in [t_0, t_{n-1}]\Big)$$

Emission value, which is issued at a certain time t_n, is defined by formula:

$$E(t_n) = \lambda S(t_{n-1}) f\Big(\kappa \frac{E_T(t_n) - E_S(t_{n-1})}{\lambda S(t_{n-1})}\Big)$$

Here λ is the marginal growth of the token amount in the system S per one emission. It is defined through L, which specifies the marginal growth S in a year, expressed as a percentage:

$$\lambda = (1 + \frac{L}{100})^{1/N} - 1$$

Here N is the number of emission issues per year.

$f(x)$ - sigmoidal function (Fig 2). In the present implementation of the algorithm a hyperbolic tangent is used as this function.

κ is a coefficient between 0 to 1 and it defines the speed at which a full emission approaches the target emission E_T if the activity level remains the same over the long term.

Initial values of both E_T and E_S are zero:

$$E_T(t_0) = 0, E_S(t_0) = 0$$

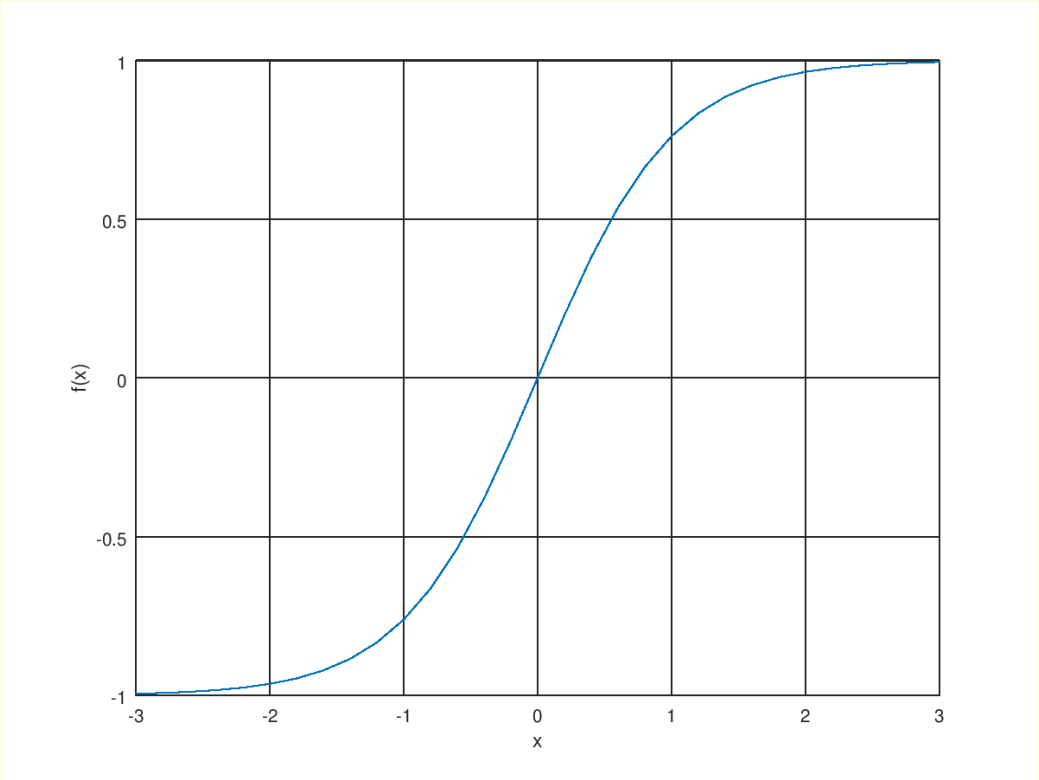

Figure 2: Sigmoidal function

References

[1] Metcalfe, B. (2013). Metcalfe's law after 40 years of ethernet. Computer, 46(12), 26-31. URL: http://ieeexplore.ieee.org/abstract/document/6636305/

[2] Lamport, L., Shostak, R., Pease, M. (1982). The Byzantine generals problem. ACM Transactions on Programming Languages and Systems

(TOPLAS), 4(3), 382-401. URL: https://www.microsoft.com/en-us/research/uploads/prod/2016/12/The-Byzantine-Generals-Problem.pdf

[3] Castro, M., Liskov, B. (2002). Practical Byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems (TOCS), 20(4), 398-461. URL: https://dl.acm.org/citation.cfm?doid=571637.571640

[4] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. URL: https://bitcoin.org/bitcoin.pdf

[5] Croman, K. et al. (2016). On scaling decentralized blockchains. In International Conference on Financial Cryptography and Data Security (pp. 106-125). Springer, Berlin, Heidelberg. URL: http://www.comp.nus.edu.sg/ prateeks/papers/Bitcoin-scaling.pdf

[6] Eyal, I., Sirer, E. G. (2014). Majority is not enough: Bitcoin mining is vulnerable. In International conference on financial cryptography and data security (pp. 436-454). Springer, Berlin, Heidelberg. URL: https://arxiv.org/pdf/1311.0243.pdf

[7] Bitcoin Energy Consumption Index. digiconomist.net. URL: https://digiconomist.net/bitcoin-energy-consumption

[8] Buterin, V. (2014). Mining Pool Centralization at Crisis Levels. URL: https://bitcoinmagazine.com/articles/mining-pool-centralization-crisis-levels-1389302892/

[9] Bentov, I., Gabizon, A., Mizrahi, A. (2016). Cryptocurrencies without proof of work. In International Conference on Financial Cryptography and Data Security (pp. 142-157). Springer, Berlin, Heidelberg. URL: https://link.springer.com/chapter/10.1007/978-3-662-53357-4_10/

[10] King, S., Nadal, S. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. URL: https://peercoin.net/assets/paper/peercoin-paper.pdf

[11] Demeester, T. (2017). Critique of Buterin's A Proof of Stake Design Philosophy. URL: https://medium.com/@tuurdemeester/critique-of-buterins-a-proof-of-stake-design-philosophy-49fc9ebb36c6

[12] Poelstra, A. (2014). Distributed consensus from proof of stake is impossible. URL: https://download.wpsoftware.net/bitcoin/old-pos.pdf

[13] Dantheman. (2017). DPOS Consensus Algorithm - The Missing White Paper. URL: https://steemit.com/dpos/@dantheman/dpos-consensus-algorithm-this-missing-white-paper

[14] NEM Technical Reference. Version 1.2.1. February 23, 2018 URL: https://nem.io/wp-content/themes/nem/files/NEM_techRef.pdf

[15] Xiaowei Xu et al. (2007). SCAN: A Structural Clustering Algorithm for Networks. URL: http://www1.se.cuhk.edu.hk/ hcheng/seg5010/slides/p824-xu.pdf

5 Appendix 1 Glossary of Terms

UOS consensus algorithm (Delegated Proof-of-Importance, DPoI)

The consensus algorithm, which is based on a calculation of the importance index of an account, which in turn depends on Stake Volume Index and Activity index

Importance index

The importance index of an account is calculated as a function of Stake Volume Index and Activity index

Account

An entity represented by a tuple of pairs of keys (public + private), which is registered in a blockchain by an individual.

Producer

Accounts with the right to verify blocks. They must have a node.

Node

A P2P network node, which performs all the calculations in blockchain. A node belonging to a producer account (producer node), produces blocks.

Contents