

Studying the Compounding Effect: The Role of Proof-of-Stake Parameters on Wealth Distribution

Alberto Leporati

University of Milan-Bicocca, Department of Informatics, Systems and Communication, Edificio U14 (ABACUS), Viale Sarca 336, 20126 Milano, Italy

Abstract

The Proof-of-Stake (PoS) consensus algorithm has been criticized, in the literature and in several cryptocurrencies communities, due to the so-called *compounding effect*: who is richer has more tokens to put in stake, hence higher probability of being selected as a block validator and obtain the corresponding rewards, thus becoming even richer. In this paper we present a PoS simulator written in the R language, that allows to tweak several parameters of the consensus algorithm and observe how the distribution of tokens among the users evolves over time. This tool can be used to investigate which combinations of parameter values allow to obtain a “fair” consensus algorithm, in which no one gets richer or poorer by the mere act of validating blocks.

Keywords

Blockchain, Proof-of-Stake, Compounding effect, Wealth distribution, Tokenomics

1. Introduction


The advent of Bitcoin [1] has given rise to an increasing interest in blockchains and distributed ledger technologies (DLTs), attracting many scientists, programmers, and business investors. Since then, many types of blockchains and DLTs have been proposed, both permissionless and permissioned, based on several types of consensus algorithms, among which we can find Proof-of-Work (PoW), Proof-of-Stake (PoS), Delegated Proof-Of-Stake (DPoS), Practical Byzantine Fault Tolerance (PBFT), Proof-of-Burn (PoB), Proof-of-Capacity, and Proof of Elapsed Time (PoET).


In the first years of DLTs, questions of technological nature received the most attention; questions about economics, cryptocurrencies distributions and tokenomics have been addressed much less thoroughly. Some papers in the literature address the issue of (lack of) decentralization in blockchain governance, implicitly assuming that who is richer has more power in making decisions about which transactions and blocks should be validated. Albeit decentralization and wealth distribution among the users of a blockchain are somehow related, the two phenomena do not necessarily coincide. In fact, it is commonly believed that monetary policy concerns cryptocurrency distribution, whereas decentralization is merely a technical (infrastructural) matter. However, monetary policy is not the only important factor for wealth distribution: even

5th Distributed Ledger Technology Workshop (DLT 2023), May 25-26, 2023, Bologna, Italy

✉ alberto.leporati@unimib.it (A. Leporati)

ORCID 0000-0002-8105-4371 (A. Leporati)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

technology solutions, like consensus mechanisms, might influence it. Hence, to understand the implications of wealth distribution each different type of consensus algorithm must be analyzed separately. As an example, a comparison of Proof-of-Work (PoW) and Proof-of-Stake (PoS) consensus mechanisms is very informative. In PoW, newly created units of currency are rewarded to the specialized users, called *miners*, who have access to efficient and powerful hardware. PoW miners might hold a large number of cryptocurrency units; however, a large portion of mined rewards must be sold to cover expenses like electricity bills, rent, and amortization costs of mining rigs. In PoS systems, instead, new tokens are rewarded to *stakers* who hold a large number of cryptocurrency units. Unlike PoW miners, PoS stakers do not experience high costs and are incentivized not to sell their rewards as doing so increases their revenue in the future. This phenomenon, known as *compounding effect*, illustrates that even supposedly monetary-agnostic technology solutions might influence tokenomics.

The topics of wealth distribution, decentralization, and blockchain governance, have become particularly relevant and very much discussed when Ethereum [2] announced their intention to switch from PoW to PoS, which they did on September 15, 2022, in the event known as “The Merge” [3]. To avoid the compounding effect of PoS – whereby the richest get even richer – and also to mitigate the negative externalities posed by Maximal Extractable Value (MEV) strategies – that include, omit, or reorder transactions when making a new block, with the aim of producing as much additional profit as possible – the community has proposed *Flashbots* [4], a (quite elaborate) infrastructure running on top of Ethereum’s blockchain. Flashbots, as well as other recent similar projects, provides an off-chain marketplace to build and propose the most profitable blocks to the validators; however, this off-chain mechanism introduces some degree of opacity in the consensus mechanism, which is against the transparency and fairness principles that drive permissionless blockchains. Further details on how the Flashbots architecture works, and a preliminary analysis of how the rewards have been distributed since its birth, can be found in [5].

In this paper we partially address questions about wealth distribution in blockchains that make use of the Proof-of-Stake family of consensus algorithms. PoS was initially designed to improve the energy consumption derived from PoW [6]. Since its first implementation, PoS has evolved and many researchers have been discussing different approaches, such as *Chain-based PoS*, *Nominated PoS (NPoS)*, *BFT-based PoS*, *Delegated proof of stake (DPoS)*, and *Liquid proof of stake (LPoS)*. For a description on how these algorithms work, we refer the reader to [7, 8, 9, 10]. Even the exact definition of *stake* varies among different implementations: for instance, some cryptocurrencies use the concept of *coin age*, the product of the number of tokens with the amount of time that a single user has held them, rather than merely the number of tokens, to define a validator’s stake [11]. In order to make our study more general, we will not focus our attention on a specific variant of PoS consensus algorithm. Instead, we have developed a PoS simulator whose behavior depends upon several parameters, and may be adapted to simulate any specific PoS-based algorithm. By tweaking these parameters, we can observe how the distribution of cryptocurrency tokens evolves over time. This tool can be used to investigate which combinations of parameter values allow to obtain a “fair” distribution of wealth, in which no one gets richer or poorer by the mere act of validating blocks.

The rest of this paper is structured as followed. In Section 2 we recall some works from the literature, that investigate on the distribution of wealth among blockchain validators. In

Section 3 we describe the operation of our PoS simulator. In Section 4 we show some examples of simulations that can be performed with it, and we discuss the outputs of such simulations. Finally, in Section 5 we draw some conclusions and delineate some directions for future research.

2. Some Related Works

In this section we recall some works published in the literature, that are somehow related with the topic under study. However, as it will be apparent in a moment, our perspective is a bit different, and we believe that our approach may be of some interest to design and/or test the behavior of PoS-based consensus algorithms.

Some scientific studies about the distribution of wealth among the top richest users of PoW and PoS-based blockchains have been performed [12, 13, 14, 15]. For example, [16, 17, 18] showed that the distributions of the top richest balances might be modeled with Zipf's law. Additionally, the Gini coefficients were computed for each user to measure wealth inequality. Moreover, in [16] the authors showed that the wealth of top Bitcoin holders grows faster than the wealth of low balance accounts; this phenomenon is well known as *preferential attachment*, and it plays an important role in the formation of the wealth distribution.

In [19], the authors analyze the distribution of the top richest accounts in cryptocurrencies like Bitcoin, Ethereum, and selected ERC20 tokens. Their analysis involves the data sets snapshotted at different dates with a given time interval. These data sets are used to measure different statistical and concentration metrics – Shannon entropy, Gini index, and Nakamoto coefficient – and analyze their evolution over time, trying to answer the following research question: *Are there any quantitative differences between top account balances held in cryptocurrency “coins” with respect to those held in other types of “tokens”?* It was observed that tokens are, in general, much more centralized than coins, with higher Gini coefficients and smaller Nakamoto coefficients.

All these researches focus on the top richest accounts, and hence might be of particular interest to DLTs where a group of top cryptocurrency holders fulfills a special role. Examples include Decentralized Autonomous Organizations (DAOs) – in which a committee of top token holders is responsible for DAO governance or treasury management – and Delegated Proof-of-Stake (DPoS) blockchains – where a relatively small committee of block validators issue ledger updates or distributed random number generators based on the threshold signature scheme. Since these kinds of analyses require to download and process large amounts of data, they necessarily limit their scope to the top richest users.

Other works focused on the (de)centralization of blockchains, intended as the number of players controlling them. In [20], the authors provided their analysis using three different metrics (Gini coefficient, Shannon entropy, and Nakamoto coefficient) and their evolution over time. It was found that the degree of decentralization in Bitcoin is higher and more volatile, while the degree of decentralization in Ethereum (when still adopting the PoW consensus mechanism) is smaller and more stable. Jensen et al. [21] analyzed decentralization of governance token distribution in four decentralized finance (DeFi) applications on the Ethereum blockchain using Gini and Nakamoto coefficients. Their results indicated that the token distributions for all four DeFi applications are characterized by high Gini coefficients. Similar methods were used in [18], where PoW and PoS cryptocurrencies were compared, analyzing the decentralization of

Bitcoin and Steem [22] using Shannon entropy.

Other papers deal with the centralization/decentralization of PoS-based blockchains. For example, in [19] the authors analyze the time-dependent statistical properties of top cryptocurrency holders for 14 different distributed ledger projects. The provided metrics include approximated Zipf coefficient, Shannon entropy, Gini coefficient, and Nakamoto coefficient. It is thus shown that there are quantitative differences of centralization levels between coins (cryptocurrencies operating on their own independent network) and tokens (which operate on top of a smart contract platform).

However, it is important to note that (de)centralization and wealth distribution may be in some cases related, but they are indeed different phenomena. Further, the data about wealth distribution usually presented in the literature do not represent the wealth of individual cryptocurrency owners but rather the wealth distribution among the cryptocurrency wallets. Apart from the difficulty of establishing the owner of a wallet, a user may be in possession of multiple wallets. Clearly, all these hindrances make it difficult to interpret the results of the analyses.

In this paper we take a different approach with respect to the above cited papers. Instead of analyzing existing data about cryptocurrencies or tokens distributions in blockchains, we study under which conditions a PoS-based consensus algorithm allows to obtain a *fair* wealth distribution over time. By *fair* we mean that who is richer has more possibilities to be chosen to be a validator, but in the long run their wealth does not increase (nor decrease) due to the mere activity of validation. Stated otherwise, the validation activity *alone* should not increase nor decrease anyone's amount of cryptocurrency. To do so, we do not look only at the top 30-50-100 richest cryptocurrency holders, but we consider the distribution of wealth among all the users of the blockchain – to be precise, all users who aspire to be selected as block validators. While the Gini coefficient can be considered a centralization measure, we will use it instead as an indicator of wealth distribution among a population (the blockchain users), as is done in economics studies. Our aim is to help researchers analyze the behavior of existing implementations of PoS consensus algorithms, and the designers of PoS-based consensus algorithms in finding the values of parameters that make the protocol fair and sustainable in the long term. We do so by proposing a PoS simulator, that allows one to tweak several parameters of the consensus algorithm. Starting from an initial token supply, the simulator computes the evolution of wealth distribution over time, measuring its fairness by means of Gini coefficient. We believe that fairness is a necessary condition for the consensus protocol to be sustainable over time. In fact, a protocol that concentrates wealth in the hands of few makes a permissionless blockchains a centralized system, controlled by an oligarchy. This entails that users may no longer trust the system, and therefore abandon it. So, in our opinion, fairness implies sustainability in the long run. Let us note that some authors have adopted a more extreme point of view about PoS: in [23], for example, it is stated that “Proof-of-stake is introducing a set of significant new flaws in both monetary and governance models. Such systems are plutocratic, oligopolistic, and permissioned”. Even without being so extreme, it is true that PoS essentially means “proof of wealth”, since blockchain protocol's rules, upgrades, and changes are directly linked to its participants' stake (that is, their wealth). Other authors have proposed significant modifications to the PoS protocol, to make it more democratic and sustainable [24, 25].

Parameter Name	Meaning
numberOfPeers	The number of participants in the blockchain. More precisely, the number of participants that aim to be selected as validators
numberOfCorruptedPeers	The number of peers that are corrupted, that is, that will be fined because they do not validate correctly the block
numberOfValidators	The number of peers that are chosen to validate a block
minNumberOfTokensPerPeer	The minimum number of tokens assigned to each peer during the distribution of the initial token supply
maxNumberOfTokensPerPeer	The maximum number of tokens assigned to each peer during the distribution of the initial token supply
stakeablePercentage	The percentage of tokens in the current supply of the peers, that can be put into the stake
numberOfRewardTokens	The number of tokens given as a reward to the validators that correctly validate the current block
percentageOfPenalty	The percentage of tokens removed from the amount of tokens staked by the corrupted validators
numberOfIterations	The number of iterations to be simulated, that corresponds to the number of blocks validated

Table 1

The set of parameters currently managed by the PoS simulator, and their definition

3. A Proof-Of-Stake Simulator

In order to address the research problem illustrated in the previous section, we have developed a simulator in the R language [26], that exploits the DescTools package [27] to compute the *Gini coefficient*:

$$G = \frac{1}{2N} \sum_{i=1}^N \sum_{j=1}^N |x_i - x_j|$$

where N is the number of elements in the population (in our case, the number of blockchain users), and x_i , for $1 \leq i \leq N$, is the monetary value associated to the i -th element (in our case, the number of cryptocurrency coins available to the i -th user). The Gini coefficient is an inequality measure widely used in economics and social statistics. For example, it is used to measure the inequality of incomes – or of wealth – among the citizens of a country. It takes values from 0, which corresponds to a complete decentralization (that is, a fair distribution) of wealth, to 1, that corresponds to absolute centralization. Anywhere below 0.3 is considered strongly egalitarian; anything above 0.5 is considered dangerous and divisive. More in general, the Gini coefficient is a measure of the inequality of a statistical distribution.

The simulator, whose source code is available at [28], is not intended as a complete solution but rather as a generic skeleton to be customized according to the precise implementation of the consensus algorithm to be analyzed. In this view, in the next section we provide some examples of analysis performed on a hypothetical implementation of PoS. The simulator allows to track the distribution of tokens (intended here as cryptocurrency coins) over time, for a given PoS consensus algorithm, under a specified choice of parameters. The parameters that can be set are indicated in Table 1.

Algorithm 1 Pseudo-code of the simulated hypothetical PoS implementation

```
1: Number the peers from 1 to numberOfPeers
2: corruptedPeers  $\leftarrow$  random subset of peers of size numberOfCorruptedPeers
3: tokenDistribution  $\leftarrow$  random assignment, to each peer, of a number of tokens in the range
   [minNumberOfTokensPerPeer ... maxNumberOfTokensPerPeer]
4: Sort tokenDistribution in non-decreasing order
5: Print the value of all parameters
6: Print and plot the initial tokenDistribution
7: for iteration  $\leftarrow$  1 to numberOfIterations do
8:   for each peer  $i$  do
9:      $\triangleright$  Compute the number of tokens that the  $i$ -th peer can put in stake
10:    stakeableTokens[ $i$ ]  $\leftarrow$   $\lfloor (\text{stakeablePercentage}/100) * \text{tokenDistribution}[i] \rfloor$ 
11:  end for
12:  stakeableTotal  $\leftarrow$   $\sum_{i=1}^{\text{numberOfPeers}} \text{stakeableTokens}[i]$ 
13:  Define stake as an array of numberOfPeers elements, all initialized to 0
14:   $\triangleright$  Determine the set of validators
15:  validators  $\leftarrow$   $\emptyset$ 
16:   $i \leftarrow 1$ 
17:  while  $i \leq$  numberOfValidators do
18:     $r \leftarrow$  random number in the range [1 ... stakeableTotal]
19:     $\triangleright$  Determine which peer becomes a validator
20:     $j \leftarrow$  the smallest index such that  $\sum_{k=1}^j \text{stakeableTokens}[k] > r$ 
21:    if stake[ $j$ ] = 0 then  $\triangleright$  If the  $j$ -th peer was not previously selected as validator
22:      validators  $\leftarrow$  validators  $\cup$  { $j$ }  $\triangleright$  Add it to the set of validators
23:      stake[ $j$ ]  $\leftarrow$  stakeableTokens[ $j$ ]  $\triangleright$  Put its stakeable tokens in the stake
24:       $i \leftarrow i + 1$   $\triangleright$  Proceed with the choice of the next validator
25:    end if
26:  end while
27:   $\triangleright$  Determine the set of corrupted validators
28:  corruptedValidators  $\leftarrow$  validators  $\cap$  corruptedPeers
29:   $\triangleright$  Remove staked tokens from the token distribution
30:  For each peer  $i$ , let tokenDistribution[ $i$ ]  $\leftarrow$  tokenDistribution[ $i$ ] - stake[ $i$ ]
31:   $\triangleright$  Add rewards to honest validators, and apply penalty to corrupted validators
32:  for  $i \leftarrow 1$  to numberOfPeers do
33:    if the  $i$ -th peer is a honest validator then
34:      stake[ $i$ ]  $\leftarrow$  stake[ $i$ ] + numberOfRewardTokens
35:    end if
36:    if the  $i$ -th peer is a corrupted validator then
37:      stake[ $i$ ]  $\leftarrow$   $\lfloor \text{stake}[i] * \text{percentageOfPenalty}/100 \rfloor$ 
38:    end if
39:  end for
40:   $\triangleright$  Update token distribution
41:  For each peer  $i$ , let tokenDistribution[ $i$ ]  $\leftarrow$  tokenDistribution[ $i$ ] + stake[ $i$ ]
42: end for
43: Print and plot the final tokenDistribution
```

The current version of the simulator (April 2023) operates as illustrated in the pseudo-code of Algorithm 1. In the first two lines, the set of corrupted peers is randomly chosen among the peers, which are numbered from 1 to `numberOfPeers`. These corrupted peers will be the peers that fail to correctly validate the block (if selected as validators), for example because their software has been infected by a malware, or because their computer is controlled by a malicious hacker. In our hypothetical implementation of PoS these corrupted peers will incur in a penalty, that is, a percentage of the tokens they have put into the stake will be removed from their wallet. At line 3, the initial supply of tokens is distributed: each peer will receive a number of tokens randomly sampled in a uniform way in the interval `[minNumberOfTokensPerPeer..maxNumberOfTokensPerPeer]`. In order to simplify visualization of the initial supply in the form of an histogram, at line 4 the sequence of token amounts is sorted in a non-decreasing order. The histogram is then plotted (line 6), and the values assigned to the aforementioned parameters are printed as a reference (lines 5–6).

At this point the main loop (lines 7–35), which consists of `numberOfIterations` iterations, starts. In each iteration, the simulator first computes (through the for loop at lines 8–10) the number of tokens that each peer can put into the stake. This number is computed as the `stakeablePercentage` of tokens possessed by the peer, rounded to the smallest integer (line 9). Then the total number of stakeable tokens is computed (line 11) so that in lines 13–23 the set of validators can be determined, with a probability which is proportional to the number of tokens that each peer is willing to put into the stake. At line 12, the stake is defined as an array of `numberOfPeers` elements, all initialized to 0; the value `stake[i]` will be the number of tokens put into the stake by the i -th peer. The selection of each validator occurs as follows. First, at line 16 an integer number r is randomly picked from the range `[1..stakeableTotal]`, where `stakeableTotal` is the total number of tokens that can be put into the stake – that is, the total number of tokens that all peers are willing to put into the stake. Think at the range `[1..stakeableTotal]` as a linear segment, composed by the concatenation of the segments whose lengths are given by the number of tokens that the corresponding peers put into the stake; the value r determines one of these smaller segments, chosen at random in proportion to its length. This segment is found at line 17, by computing the smallest index j such that $\sum_{k=1}^j \text{stakeableTokens}[k] > r$. If the j -th peer was not previously selected as validator (line 18), then it is added to the set of validators (line 19), and the tokens it is willing to put into the stake are actually moved to the stake (line 20). Otherwise, a new iteration of the while loop at lines 15–23 is executed with the same value of variable i , so that a new value of r is randomly chosen and the selection process for the i -th validator is repeated. In total, `numberOfValidators` validators are chosen. At line 24, the set of validators thus found is intersected with the set of corrupted peers – that were identified at line 2 – to determine the set of corrupted validators. Finally, the token distribution is updated as follows. First, the tokens put into the stake are subtracted from the current token distribution (line 25); then, the stake is modified by adding the rewards to the honest validators (line 28), and applying the fines (in terms of `percentageOfPenalty`) to the corrupted validators (line 31). Finally, the token distribution is updated by adding the new values from the stake (line 34), and a new iteration of the outer loop can start. When all iterations have been performed, an histogram of the final token distribution is plotted.

Notice that a number of simplifying assumptions have been made in our simulator:

- The set of corrupted peers is identified once for all at the beginning of execution (line 2 of Algorithm 1); these are the peers that will be punished in case they are selected to validate a block. Of course it is debatable whether a peer will always behave dishonestly; for example, its behavior may temporarily run out of control because of a malware infection, which is later removed. A more general scheme would thus be to allow the set of validators vary over time, according to some defined (programmable) strategy.
- Even the number of peers is fixed during the simulation, which may not be realistic.
- Also the fact that all peers have the same probability of behaving dishonestly is debatable: on one side who is richer will probably be more prone to hacker attacks, but on the other side they will also have stronger security controls over their hardware and software, thus making the probability of a successful attack low.
- The initial distribution of tokens among the peers is made by sampling in a uniform way in the range [minNumberOfTokensPerPeer..maxNumberOfTokensPerPeer]; more sophisticated initial token supplies could be considered.
- The percentage of tokens that each peer is willing to put into the stake is the same for all peers; it is set by assigning a value – between 0 and 100 – to the variable stakeablePercentage, at the beginning of the simulation. The same applies to the percentage of tokens which are taken from corrupted validators: such a percentage is fixed at the beginning of the simulation, by assigning a value between 0 and 100 to variable percentageOfPenalty.
- A fixed reward, indicated by the value assigned to variable numberOfRewardTokens, is payed to honest validators. More sophisticated reward mechanisms, that may possibly yield to different rewards for different validators, can be conceived.
- Only the Gini coefficient is computed. Other indicators, such as the approximated Zipf coefficient, Shannon entropy, and Nakamoto coefficient, may be of interest.

During the execution, the simulator can print several kinds of information. For example, it can print (and plot) the initial token supply, as well as the token distribution, the stake, the set of chosen validators, and the set of corrupted validators at each iteration. The information currently printed about the token distribution is the following: total number of tokens, average number of tokens possessed by the peers, standard deviation of the distribution, and Gini coefficient; optionally, the amount of tokens assigned to each peer can also be printed. At the end of the simulation, information about the final token distribution is printed and plotted by default. All this information can also be saved to files for further processing.

4. Some Examples of Simulation

In this section we illustrate some examples of execution of our simulator, under two different choices for the values of parameters listed in Table 1. As stated in previous sections, we have not focused on a particular real implementation of PoS. Instead, in this paper we assume that the PoS-based consensus algorithm operates as follows. There is a list of peers, users of the

Parameter Name	1 st experiment	2 nd experiment
numberOfPeers	1000	1000
numberOfCorruptedPeers	10	400
numberOfValidators	20	100
minNumberOfTokensPerPeer	1	1
maxNumberOfTokensPerPeer	1000	1000
stakeablePercentage	50%	50%
numberOfRewardTokens	10	1
percentageOfPenalty	50%	50%
numberOfIterations	100	1000

Table 2
Values assigned to the parameters for the described experiments

blockchain that aim to be selected as validators, which are numbered from 1 to numberOfPeers. This list is static, that is, it does not vary over time; moreover, we disregard other potential users – not willing to become validators – of the system. Among these peers there is a subset of corrupted peers, that also does not change over time. Corrupted means that for some reason, if selected as validators, these peers will fail to correctly validate the current block; in such a case a penalty is applied, removing part (a percentage) of the cryptocurrency tokens they have put into the stake. Each peer receives an initial supply of cryptocurrency tokens, randomly sampled from a fixed range [minNumberOfTokensPerPeer ... maxNumberOfTokensPerPeer]. While this range is fixed, let us observe that by making it large enough it is possible to simulate very unbalanced supplies of tokens among the peers.

The PoS-based protocol runs for a specified number of iterations. At each iteration, every peer is randomly selected to become a validator, with a probability which is proportional to the amount of tokens it is willing to stake; such amount is determined – for every peer – as a fixed percentage of the amount of tokens present in its wallet. The algorithm selects in this way numberOfValidators validators. By intersecting the set of validators with the set of corrupted peers, the set of corrupted validators is found. Finally, the token distribution is updated: each honest validator is returned the tokens it had staked, plus a fixed amount, whereas each corrupted validator is returned a portion of the tokens it had staked, that is, those that remain after applying a penalty to the staked amount. The penalty applied is modeled as a reduction by a fixed percentage, which is assumed to be the same for all corrupted validators.

Table 2 reports the values assigned to the parameters during our simulations.

In our first experiment we performed just 100 iterations – that is, the validation of 100 blocks – with 1000 peers, of which only 10 (corresponding to 1% of the total) are corrupted. The number of validators required to validate a block is 20. Each peer will put 50% of its wallet balance in stake; if it is a honest validator, at the end of the iteration it will receive a reward of 10 tokens, otherwise a penalty of 50% will be applied to the amount of tokens it has put in stake. Figure 1 reports the histograms of the initial (left) and final (middle) token distribution for this experiment. The initial distribution contains 493,913 tokens, that is, an average of 494 tokens per peer. The standard deviation is about 286, and the Gini coefficient is 0.33. So, albeit the richest peer has 1000 times the amount of tokens possessed by the poorest, the

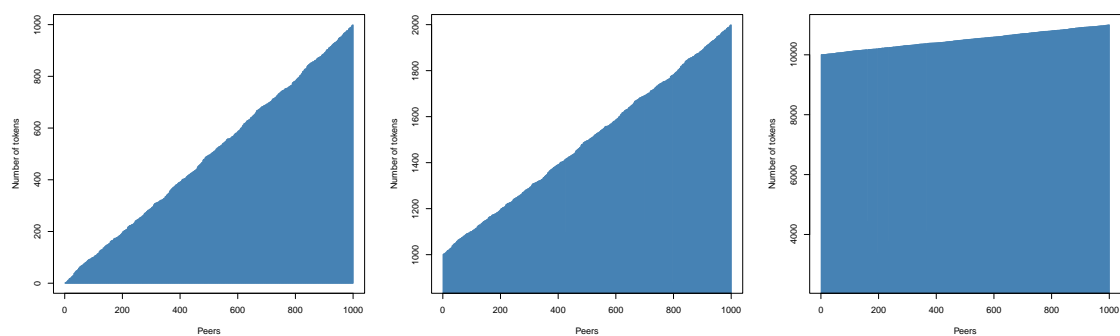


Figure 1: Initial token distribution for the first experiment (left), and token distributions after 100 (middle) and after 1000 (right) iterations.

overall distribution can be considered fair. After 100 iterations, the final distribution contains 1,488,692 tokens (about three times the initial amount), that is, an average of 1489 tokens per peer. The standard deviation is about 288, and the Gini coefficient is 0.11. Thus, albeit the histogram in the middle of Figure 1 would seem to suggest that the algorithm behavior is stable, the Gini coefficient indicates that the wealth distribution has changed, becoming a bit more equidistributed. This is confirmed by letting the simulation run up to 1000 iterations: in this case, the final distribution (see the rightmost histogram of Figure 1) contains 10,436,554 tokens (about 21 times the initial amount), that is, an average of 10437 tokens per peer. The standard deviation is about 723, and the Gini coefficient is 0.02. Clearly, this protocol – with these parameter values, and these reward and penalty policies – is not sustainable in the long term: while every peer becomes richer, the difference between the richest and the poorest tend to diminish over time, with the likely effect that the richer peers will abandon the system as soon as they realize it.

In the second experiment, we tried once again with 1000 peers (see Table 2). This time, however, as many as 400 of them are corrupted (that is, 40% of the total). We set to 100 the number of validators, in the spirit of making block validations more participated, albeit this may pose technical difficulties for the underlying peer-to-peer network communications. In fact, while in theory the size of block producing committee can be unbounded, in practice the procedure of signing new blocks is limited by the bandwidth, as the number of messages exchanged among the committee members grows like a square of its size. As a consequence, in most practical applications the block-producing committee has from 10 to 50 members. Also in this experiment, each peer will put 50% of its wallet balance in stake; if it is a honest validator, at the end of the iteration it will receive a reward of just 1 token, otherwise a penalty of 50% will be applied to the amount of tokens it has put in stake. The rationale behind these choices is to try not to make grow the total amount of tokens in the system, and hence the amount of tokens owned by each peer. The initial distribution is comparable to the one of the previous experiment. It contains 492,279 tokens, that is, an average of 492 tokens per peer; the standard deviation is about 292, and the Gini coefficient is 0.34. Figure 2 reports the histograms of the

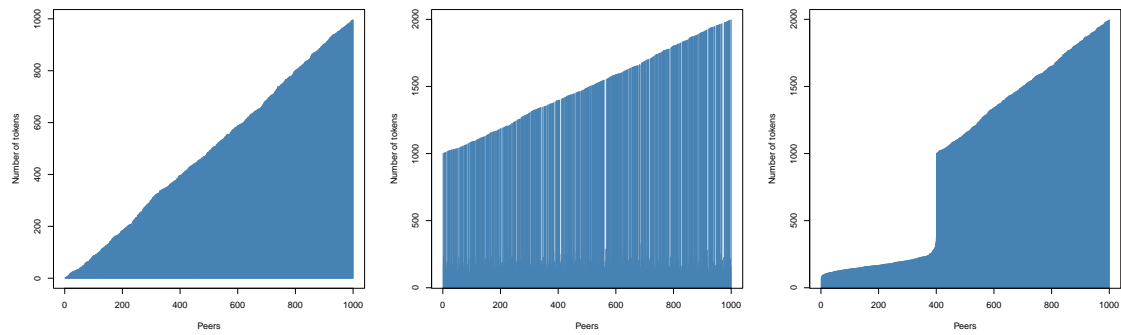


Figure 2: Initial token distribution for the second experiment (left), and final token distribution after 1000 iterations (middle). The rightmost histogram is the same final distribution, sorted by number of tokens in non-decreasing order.

initial (on the left) and final (in the middle) token distribution after 1000 iterations. The final distribution contains 966,737 tokens (about two times the initial amount, and about 65% of the total number of tokens of the second experiment, after the same number of iterations), that is, an average of 967 tokens per peer. This time the standard deviation has raised to about 687, and the Gini coefficient is 0.40, showing that the distribution of tokens has become a little bit less democratic, but quite comparable with the initial distribution. Comparing the two histograms in the middle of Figures 1 and 2, and zooming in the histogram of Figure 2, we can see that the difference among the two is that the second one contains a lot of low bars; these correspond to the corrupted peers, that have been penalized each time they have been selected as validators. This situation is made more evident by sorting the bars in non-decreasing order, obtaining the rightmost histogram of Figure 2. The value 0.40 of the Gini coefficient, compared with the initial value 0.34, seems to indicate that this second version of the PoS algorithm is more stable than the first one, and indeed we could suggest that having a stable Gini coefficient is a necessary – but probably not sufficient – condition for the protocol to be sustainable in the long term. In any case, also this version of the consensus algorithm has some problems: even being rewarded with just 1 token, honest peers tend to become increasingly richer over time; indeed, the only mechanism that drains tokens from the system is the penalization of corrupted peers. Letting these two forces act for enough time, it is very likely that corrupted peers will sooner or later run out of money, whereas the honest peers will incur once again in the original problem of wealth distribution. This indicates that a more elaborated system of interacting forces must be designed; and a corresponding more elaborated simulator must be programmed, to perform tests and experiments. Are perhaps PoS-based blockchains complex systems, whose dynamics and long-term behaviors must be studied using theoretical tools coming from economics and physics, as well as elaborated simulators?

5. Conclusions and Directions for Future Work

In this paper we have described a simulator of PoS-based consensus algorithms. The simulator is not intended to be a complete solution but rather as a generic skeleton to be customized according to the precise implementation of the consensus algorithm to be analyzed. Adopting this point of view, we have provided some examples of analysis performed on a hypothetical implementation of PoS. The simulator allows to tweak several parameters of the consensus algorithm and observe how the distribution of tokens among the users evolves over time. The final aim is to help researchers analyze the behavior of existing implementations of PoS consensus algorithms, and the designers of such algorithms to find the values of parameters that make the protocol fair and sustainable in the long term.

The presented work can be extended in several ways. First of all, we have developed our simulator in R because this language allows to perform statistical computations and operations on vectors in a very simple and natural way. However, being an interpreted language, the performances are not exciting. Even using some specific packages to perform parallel computations, such as `parallel`, one cannot hope to achieve the same performance of compiled languages created specifically for scientific computing, such as Julia [29].

Another possible extension concerns the indices and coefficients used to analyze wealth distributions, as well as the information which is printed at the beginning of the simulation, at the end, and after each iteration. As we have seen, many papers in the literature use also Shannon entropy and Nakamoto coefficient to perform their analyzes; at the moment our simulator only uses the Gini coefficient – we have started from this one because it is widely adopted in economics studies about wealth distributions. Adding the computation of further coefficients is not difficult, and will certainly be done in a future version of the simulator. Similarly, it would be possible to determine, by linear regression, the Zipf's law coefficients that best approximate the wealth distribution under study.

When designing a cryptocurrency, initial supply and subsequent distribution are fundamental problems to tackle and consider. Due to Proof-of-Stake's intrinsic initial supply requirements, blockchain networks implementing PoS as a distributed consensus mechanism present an important pre-mined initial distribution, in terms of coin percentage of the entire network. In this paper we have ignored this aspect, and we have simply assumed that each user initially obtains a number of tokens which is included in a pre-set interval (see line 3 of Algorithm 1). While this means that our simulator is only able to analyze situations where the blockchain has already been running for some time, it is not clear to the author whether the simulator should really consider also the start-up period in which the creators of the blockchain distribute cryptocurrency coins to the prospective users, according to some political, monetary, and marketing strategy.

Finally, while the current paper focused on the PoS simulator and on the reasons that led to its development, a clear direction for future research is to use (more elaborated versions of) it for investigating which combinations of parameters, and which policies – implementing forces that increase or decrease the number of tokens in the system, and their assignment to the peers – make it possible to obtain a variant of PoS that is fair and hence sustainable in the long run. As we said at the end of the previous section, this study will probably involve tools commonly used in the theory of complex systems, and will require the implementation of

much more sophisticated simulators than the one presented in this paper.

References

- [1] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, Cryptography Mailing list at <https://metzdowd.com> (2009).
- [2] G. Wood, Ethereum: A secure decentralised generalised transaction ledger (2014). URL: <https://gavwood.com/paper.pdf>.
- [3] Ethereum.org, The Merge, [tps://ethereum.org/en/upgrades/merge/](https://ethereum.org/en/upgrades/merge/), 2022. Accessed: January 18, 2023.
- [4] Flashbots, <https://www.flashbots.net/>, 2020. Accessed: March 4, 2023.
- [5] D. Mancino, A. Loporati, M. Viviani, D. Giovanni, Exploiting Ethereum after “The Merge”: The Interplay between PoS and MEV Strategies, Submitted to ITASEC 2023 (2023).
- [6] S. N. Sunny King, PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake (2012). URL: <http://www.peercoin.net/>.
- [7] I. Bashir, Blockchain Age Protocols, Apress, Berkeley, CA, 2022, pp. 331–376. doi:10.1007/978-1-4842-8179-6_8.
- [8] I. Bashir, Mastering Blockchain: A deep dive into distributed ledgers, consensus protocols, smart contracts, DApps, cryptocurrencies, Ethereum, and more, 3rd Edition, Packt Publishing, 2020.
- [9] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, E. Dutkiewicz, Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities, IEEE Access 7 (2019) 85727–85745. doi:10.1109/ACCESS.2019.2925010.
- [10] L. Ge, J. Wang, G. Zhang, Survey of Consensus Algorithms for Proof of Stake in Blockchain, Security and Communication Networks 2022 (2022). doi:10.1155/2022/2812526.
- [11] P. Tasca, C. J. Tessone, A Taxonomy of Blockchain Technologies: Principles of Identification and Classification, Ledger 4 (2019). doi:10.5195/ledger.2019.140.
- [12] F. Irresberger, Coin Concentration of Proof-of-Stake Blockchains, SSRN Electronic Journal (2018). doi:10.2139/ssrn.3293694.
- [13] L. Kogan, G. C. Fanti, P. Viswanath, Economics of Proof-of-Stake Payment Systems, MIT Sloan Research Paper No. 5845-19 (2021). doi:10.2139/ssrn.4320274.
- [14] Y. Shifferaw, S. Lemma, Limitations of proof of stake algorithm in blockchain: A review, Zede Journal 39 (2021) 1961–1973.
- [15] N. Dimitri, Monetary Dynamics With Proof of Stake, Frontiers in Blockchain 4 (2021) 443966. doi:10.3389/fbloc.2021.443966.
- [16] D. Kondor, M. Pósfai, I. Csabai, G. Vattay, Do the Rich Get Richer? An Empirical Analysis of the Bitcoin Transaction Network, PLOS ONE 9 (2014) 1–10. doi:10.1371/journal.pone.0086197.
- [17] B. Kuśmierz, S. Müller, A. Capossele, Committee Selection in DAG Distributed Ledgers and Applications, in: K. Arai (Ed.), Intelligent Computing, Springer International Publishing, Cham, 2021, pp. 840–857.
- [18] C. Li, B. Palanisamy, Comparison of Decentralization in DPoS and PoW Blockchains,

- in: Z. Chen, L. Cui, B. Palanisamy, L.-J. Zhang (Eds.), *Blockchain – ICBC 2020*, Springer International Publishing, Cham, 2020, pp. 18–32.
- [19] B. Kusmierz, R. Overko, How centralized is decentralized? Comparison of wealth distribution in coins and tokens, in: *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, IEEE Computer Society, Los Alamitos, CA, USA, 2022, pp. 1–6. doi:10.1109/COINS54846.2022.9854972.
- [20] Q. Lin, C. Li, X. Zhao, X. Chen, Measuring Decentralization in Bitcoin and Ethereum using Multiple Metrics and Granularities, in: *2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW)*, IEEE Computer Society, Los Alamitos, CA, USA, 2021, pp. 80–87. doi:10.1109/ICDEW53142.2021.00022.
- [21] J. R. Jensen, V. von Wachter, O. Ross, How Decentralized is the Governance of Blockchain-based Finance: Empirical Evidence from four Governance Token Distributions, Paper 2102.10096, arXiv.org, 2021.
- [22] Steem blockchain, <https://steem.com/>, 2018. Accessed: March 4, 2023.
- [23] V. Sus, Proof-of-Stake Is a Defective Mechanism, *IACR Cryptol. ePrint Arch. 2022 (2022)* 409. URL: <https://eprint.iacr.org/2022/409.pdf>.
- [24] M. Saad, Z. Qin, K. Ren, D. Nyang, D. Mohaisen, e-PoS: Making Proof-of-Stake Decentralized and Fair, *IEEE Transactions on Parallel and Distributed Systems* 32 (2021) 1961–1973. doi:10.1109/TPDS.2020.3048853.
- [25] A. Li, X. Wei, Z. He, Robust Proof of Stake: A New Consensus Protocol for Sustainable Blockchain Systems, *Sustainability* 12 (2020) 1–15. URL: <https://ideas.repec.org/a/gam/jsusta/v12y2020i7p2824-d340545.html>.
- [26] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2022. URL: <https://www.R-project.org/>.
- [27] S. Andri et mult. al., *DescTools: Tools for Descriptive Statistics*, 2022. URL: <https://cran.r-project.org/package=DescTools>, r package version 0.99.47.
- [28] Leporati, Alberto, *PoS Simulator*, <https://github.com/alepo42/PoS-Simulator>, 2023. Accessed: March 4, 2023.
- [29] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah, *Julia: A fresh approach to numerical computing*, *SIAM review* 59 (2017) 65–98. doi:10.1137/141000671.