5th Distributed Ledger Technology Workshop (DLT 2023)

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

Alberto Leporati

Università di Milano-Bicocca

Dipartimento di Informatica, Sistemistica e Comunicazione

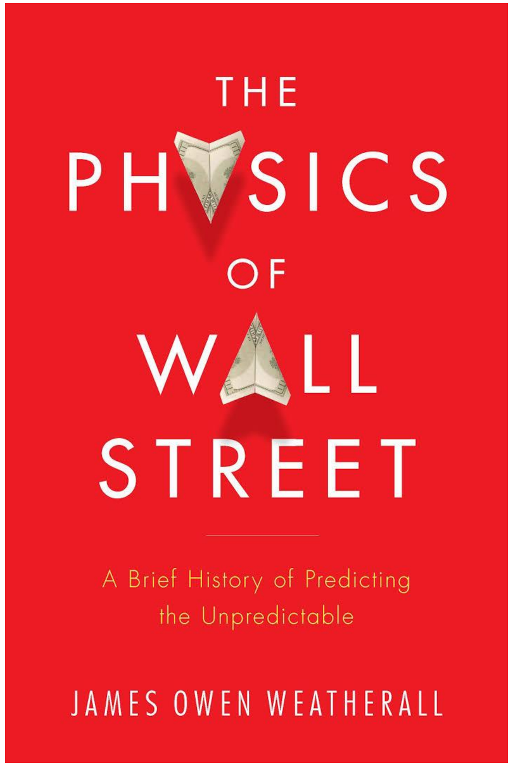
alberto.leporati@unimib.it

Bologna – May 25th, 2023

Background

- A permissionless blockchain that
  - implements a cryptocurrency
  - is used to track cryptocurrency transactions

  can be seen as an economic market, where
  - some cryptocurrency is burned
  - some cryptocurrency is created, usually by minters/validators
- This economic market must be **trusted** and **sustainable** in the long term



- These topics are studied not only in Economics, but also in Mathematics and Physics

Proof-of-Stake

- Several consensus algorithms are used in blockchains, the most famous being
  - Proof-of-Work (PoW)
  - Proof-of-Stake (PoS)
- PoS addresses the energy consumption problem of PoW
- Several versions of PoS have been proposed:
  - «Pure» Pos, Delegated PoS, Chain-based PoS, Nominated PoS, BFT-based PoS, Liquid PoS, …
  - … each with its own governance model

- In Sept. 2022, Ethereum has moved from PoW to PoS, with all the problems related to MEV, frontrunning, offchain block proposals, that introduce opacity in the system

Proof-of-Stake: criticism

- In PoW, *miners* may possess a big amount of cryptocurrency, but they also spend a lot of (fiat) money to update the hardware and pay electricity bills
- No such expenses are associated with PoS: *stakers* put some cryptocurrency in the stake, get the rewards, and are not incentived to spend them
- In PoS, who is rich gets richer, by the compounding effect

IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS

e-PoS: Making Proof-of-Stake Decentralized and Fair

Muhammad Saad, Zhan Qin, Kui Ren, DaeHun Nyang, and David Mohaisen

Coin Concentration of Proof-of-Stake Blockchains

Felix Irresberger*
University of Leeds

December 9, 2018

Proof-of-Stake Is a Defective Mechanism

Vicent Sus
vicent@vicentsus.org

March 24, 2022

Economics of Proof-of-Stake Payment Systems *

Giulia Fanti†, Leonid Kogan‡, Pramod Viswanath§

First draft: November 2018
Latest draft: May 2021

Our question

« Is PoS a **fair** and **sustainable** consensus mechanism? »

- This depends upon how PoS is implemented, not only on monetary policy
- Governing rules depend upon a number of parameters
- **Fair** to us means: no one should get richer or poorer by just validating blocks
- We measure wealth distribution by Gini coefficient
- **Sustainaible** means: users trust the system, hence they do not leave it

- The system must be perceived as **trusted**, not driven by an oligarchy, hence **decentralized** (both in terms of technical infrastructure and **wealth distribution**)

Our goal

- To study how the initial cryptocurrency supply, and the parameters that drive the PoS consensus mechanism, influence the (long term) wealth distribution …
- … by using a simulation approach
- **Note:** we do not focus on a particular implementation of PoS
- This is our first attempt, a more sophisticated simulator is on the way
- Other works in the literature address this problem from a **statistical** point of view (model based on Zipf's law)
    - Instead, we consider the blockchain as a **complex system**, sensitive to the choice of parameter values and the initial state

PoS simulator

- Written in the R language, for simplicity
- Source code available at https://github.com/alepo42/PoS-Simulator
- Just a proof of concept, to test the idea
- More a framework than a ready-to-use simulator

- **Pros:**
  - Vectorial (component-wise) operations
  - Simple management of statistical distribution
  - Simple generation of plots, graphs, etc.
- **Cons:** execution speed!

➡ Limitations on the size of the model, and number of iterations

Parameters

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

Caveats, restrictions

- We simulate a hypothetical, abstract version of PoS
- Fixed number of participants (peers), corrupted peers, and validators
- We simulate a closed system (no interaction with the external environment)
- The initial wealth distribution is chosen uniformly in a fixed range
- The percentage of tokens (coins) that are put in stake is the same for all peers
- … the same goes for the number of coins awarded
- … and the same holds for the percentage to be slashed

The algorithm

Algorithm 1 Pseudo-code of the simulated hypothetical PoS implementation

Number the peers from 1 to numberOfPeers
corruptedPeers $\leftarrow$ random subset of peers of size numberOfCorruptedPeers
tokenDistribution $\leftarrow$ random assignment, to each peer, of a number of tokens in the range [minNumberOfTokensPerPeer ... maxNumberOfTokensPerPeer]
Sort tokenDistribution in non-decreasing order
Print the value of all parameters
Print and plot the initial tokenDistribution
for iteration $\leftarrow$ 1 to numberOfIterations **do**
    for each peer i **do**
        ▷ Compute the number of tokens that the i-th peer can put in stake
        stakeableTokens$[i] \leftarrow \lfloor$(stakeablePercentage$/100)*$tokenDistribution$[i]\rfloor$
    end for
    stakeableTotal $\leftarrow \sum_{i=1}^{\text{numberOfPeers}}$ stakeableTokens$[i]$
    Define stake as an array of numberOfPeers elements, all initialized to 0

The algorithm

```
▷ Determine the set of validators
validators ← ∅
i ← 1
while i ≤ numberOfValidators do
    r ← random number in the range [1 ... stakeableTotal]
    ▷ Determine which peer becomes a validator
    j ← the smallest index such that ∑_{k=1}^{j} stakeableTokens[j] > r
    if stake[j] = 0 then        ▷ If the j-th peer was not previously selected as validator
        validators ← validators ∪{j}                ▷ Add it to the set of validators
        stake[j] ← stakeableTokens[j]              ▷ Put its stakeable tokens in the stake
        i ← i + 1                        ▷ Proceed with the choice of the next validator
    end if
end while
▷ Determine the set of corrupted validators
corruptedValidators ← validators ∩ corruptedPeers
```

The algorithm

```
    ▷ Remove staked tokens from the token distribution
    For each peer i, let tokenDistribution[i] ← tokenDistribution[i] − stake[i]
    ▷ Add rewards to honest validators, and apply penalty to corrupted validators
    for i ← 1 to numberOfPeers do
        if the i-th peer is a honest validator then
            stake[i] ← stake[i] + numberOfRewardTokens
        end if
        if the i-th peer is a corrupted validator then
            stake[i] ← ⌊stake[i] ∗ percentageOfPenalty/100⌋
        end if
    end for
    ▷ Update token distribution
    For each peer i, let tokenDistribution[i] ← tokenDistribution[i] + stake[i]
end for
Print and plot the final tokenDistribution
```

Output produced

- Number of cryptocurrency coins in the system
- Average number of coins per participant (and standard deviation)
- Gini coefficient
- Plot of the coins distribution, possibly sorted in ascending order

---

- By default, this information is produced for the initial and the final distribution
  - It can be produced at any iteration
  - … and the same holds for the list of corrupted peers, chosen validators, and corrupted validators

Gini coefficient

- It can be defined in several ways, for example:

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

where N is the number of individuals in the population, and x_i is the monetary value associated with the i-th individual

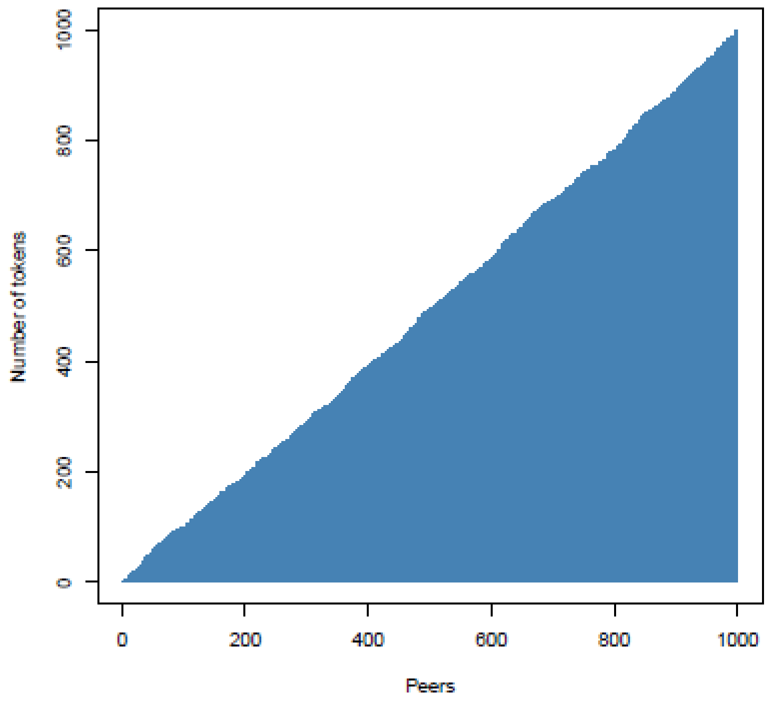
- Invented to investigate and measure wealth/income distribution in populations
- Widely used in Economics and Social Statistics
- It takes values from 0 (complete decentralization) to 1 (absolute centralization)
    - Less than 0.3: egalitarian distribution
    - Greater than 0.5: dangerous and divisive

Examples of simulation

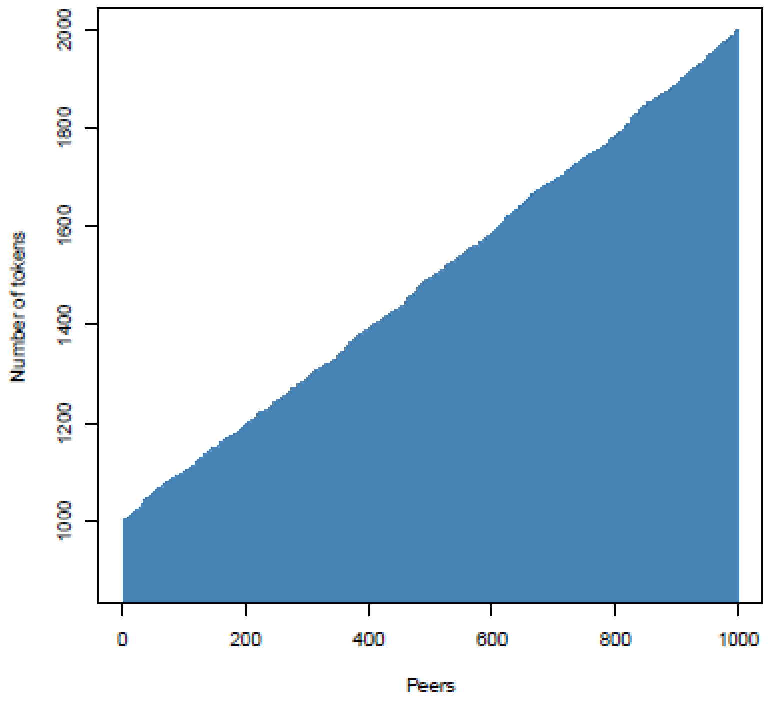
Two simulations, with the following parameters

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

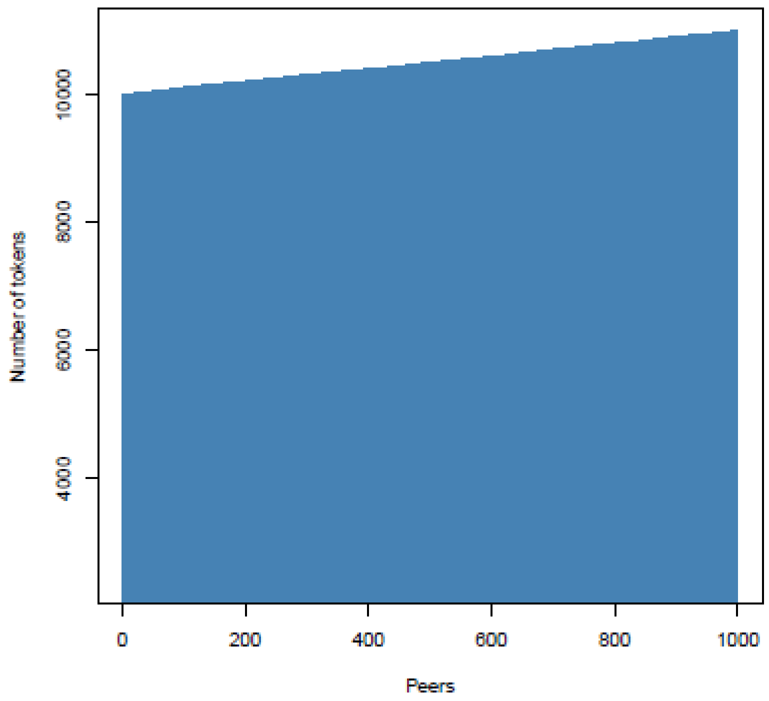
Results of the first simulation



Initial distribution



After 100 iterations



After 1000 iterations

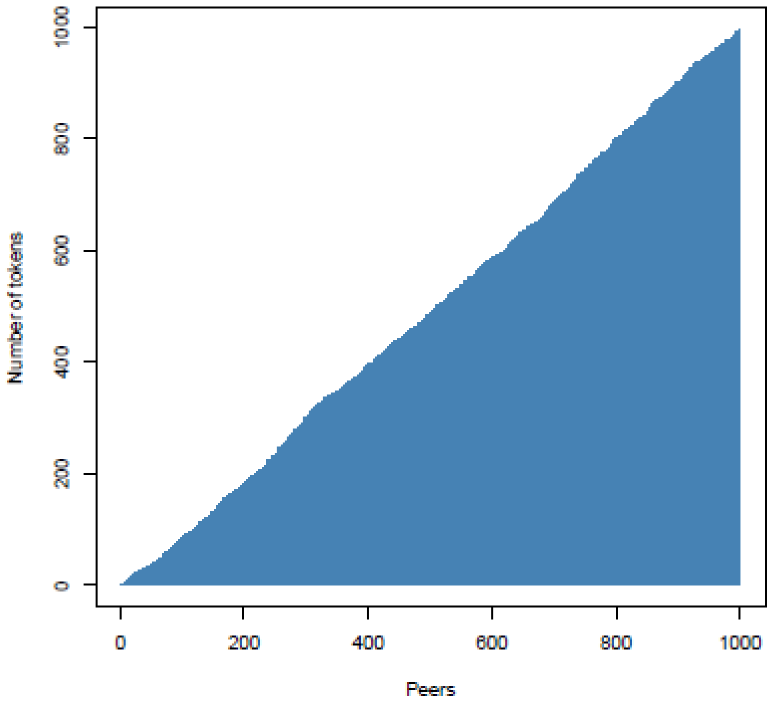
Results of the first simulation

- Initial distribution
  - 493, 913 tokens (average of 494 tokens per peer)
  - Standard deviation: ~ 286
  - Gini coefficient: 0.33

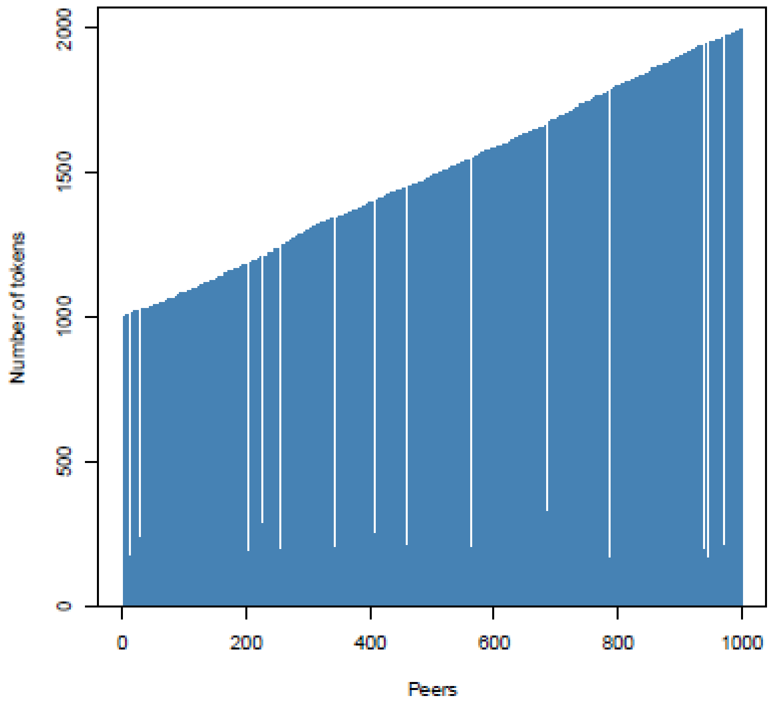
- After 100 iterations
  - 1, 488, 692 tokens (about 3x the initial amount), average of 1489 tokens per peer
  - Standard deviation: ~ 288
  - Gini coefficient: 0.11

- After 100 iterations
  - 10, 436, 554 tokens (about 21x the initial amount), average of 10437 tokens per peer
  - Standard deviation: ~ 723
  - Gini coefficient: 0.02

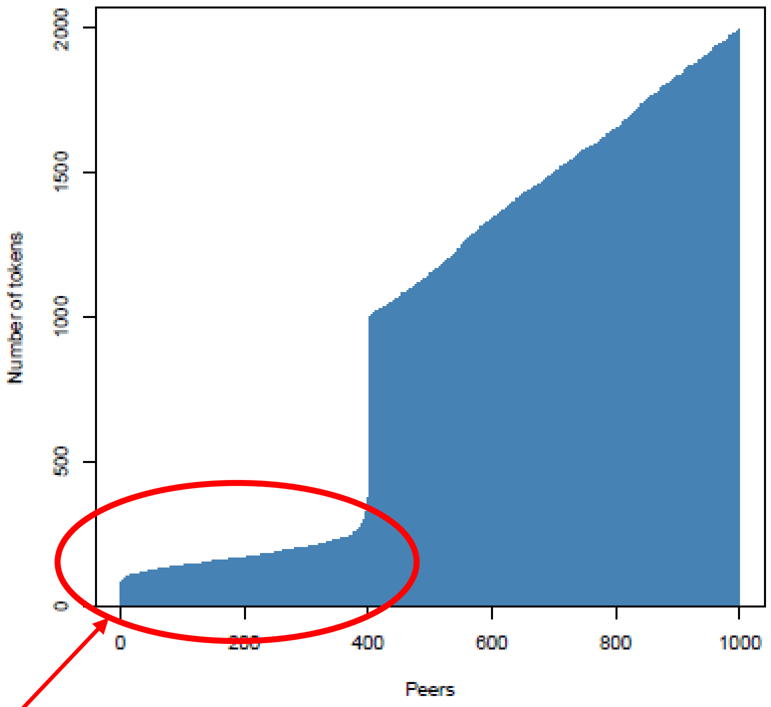
Results of the second simulation



Initial distribution



After 1000 iterations



After 1000 iterations, sorted in ascending order

Corrupted peers

Results of the second simulation

- Initial distribution
  - 492, 279 tokens (average of 492 tokens per peer)
  - Standard deviation: ~ 292
  - Gini coefficient: 0.34

- After 1000 iterations
  - 966, 737 tokens (about 2x the initial amount), average of 967 tokens per peer
  - Standard deviation: ~ 687
  - Gini coefficient: 0.40

Directions for future work

- Re-implement the simulator for speed (parallel implementation in Julia language)
- Allow easier selection of parameters and possible behaviors
- Compute other indexes: Shannon entropy, Nakamoto coefficient
- Compute Zipf’s law parameters
- Improve the output (ex: dynamical plots)
- Test the simulator on a real blockchain, starting from its current state
- Find parameters and behaviors (driving forces) that make a PoS-based blockchain system fair and sustainable in the long term (to design a new PoS-based consensus algorithm)

Thank you
for your attention !

Alberto Leporati

Dipartimento di Informatica
Università di Milano-Bicocca

alberto.leporati@unimib.it