# Staking Pools on Blockchains

No Author Given

No Institute Given

**Abstract.** On several proof-of-stake blockchains, agents engaged in validating transactions can open a pool to which others can delegate their stake in order to earn higher returns. We develop a model of staking pool formation in the presence of malicious agents and establish existence and uniqueness of equilibria. We then identify potential and risk of staking pools. First, allowing for staking pools lowers blockchain security. Yet, honest stake holders obtain higher returns. Second, by choosing welfare optimal distribution rewards, staking pools prevent that malicious agents receive large rewards. Third, when pool owners can freely distribute the returns from validation to delegators, staking pools disrupt blockchain operations, since malicious agents attract most delegators by offering generous returns.

**Keywords:** Delegation · Staking Pool · Blockchain.

## 1 Introduction

From both the financial side and the security side, there are reasons why a proof-of-stake (PoS) blockchain may want to allow the formation of staking pools. With staking pools, agents interested in validating transactions are allowed to open a pool, such that others can delegate their stake for some time to it. For delegating agents who are not interested in validating transactions, this can provide an additional income on their token holdings. In turn, their stakes are blocked and cannot be used for other purposes during the time of commitment. By agents we mean all participants of the decentralized system who own some stake, which usually takes the form of a native token. Agents interested in running a staking pool could earn a higher income from their transaction validation activities.[1]

Ideally, such a staking system makes it more attractive to hold tokens, provides incentives for a sufficient number of agents to run staking pools and act as transaction validators, and increases the share of honest agents, weighted by the stakes they hold, involved in transaction validation. As every staking pool acts as a validator, we occasionally use the word "validator" for a staking pool.

However, malicious agents also run staking pools and may thus enlarge the share of the stake they control in transaction validation.[2] This may undermine

[1] An agent that decides to run a pool and validate transactions is usually referred as a node of a network.

[2] Our assumption that malicious agents always run staking pools is justified, since if they do not, they do not affect blockchain functioning and thus cannot be considered

the security of the blockchain and lead to a collapse of the protocol, as the malicious agents take over.

We explore how such a system can be modeled and designed, so that it operates beneficially for the decentralized consensus mechanism—i.e. by lowering the share of malicious agents who disrupt the validation of transactions—and for the ecosystem as a whole. The model invokes a measure of honest agents who are interested in the returns from holding a stake (of tokens) in a PoS blockchain and thus are also interested in the proper functioning of the blockchain. An agent is honest if s/he is prepared to run the software for validation, as required by the system. Otherwise, honest agents choose actions to maximize their expected returns. The PoS protocol that we consider in this paper perfectly mimics how the proof-of-work (PoW) protocol works. The PoW protocol chooses the next block writer according to who finds the nonce that satisfies certain conditions, that is, the probability that the next writer is chosen proportionally to his/her *hash rate*. Similarly, in the PoS protocol, the next writer of the block is chosen proportionally to his/her stake size. In the PoW protocols, the costs are typically assumed to be different across agents, as they depend on electricity, hardware, and maintenance costs. In the PoS protocol, we have a similar situation, except for electricity costs. Agents have different costs in participating in transaction validation, as availability of appropriate computer software and hardware, speed and bandwidth of the internet, the knowledge of how to run a secure validation node, and opportunity costs to engage in validation activities(that is, costs of not being able to use tokens for other purposes) differ across them. The difference between PoW and PoS protocol pools is the following. In the former, the costs are incurred by all pool members, and therefore the rewards are distributed proportionally. In the latter, the costs are incurred only by the pool runner, while the rest—the delegators—incur no cost. In this paper, we examine the simplest reward distribution, where the pool runner keeps some fraction of the rewards and distributes the rest to the delegators proportionally.

There is a measure of malicious agents who are only interested in disrupting the blockchain, and therefore, their costs are ignored by the designer.

A *Blockchain Designer* aims at maximizing the chance that the blockchain is working (maximizing blockchain security), which will be captured by maximizing the number of honest validators. We will also consider an alternative objective where the Blockchain Designer trades off the probability that the blockchain is running correctly with the costs for all honest agents of validating transactions. This is a standard economic welfare criterion. Two further aspects can be important for a Blockchain Designer: Reducing the rewards for malicious agents, as this decreases their future influence and distributing the rewards to validators as equally as possible.

The Blockchain Designer has two basic options when designing the market for staking pools. First, s/he can fix the return distribution between the pool owner and the pool delegators. We call this “return fixing”. Second, s/he can

malicious. If they delegate their stakes, they act like honest delegators and thus again are not malicious in any way.

allow competition of pool runners regarding how the returns from transaction validation are shared between the pool owner and the pool delegators. This is called “return competition”.

We model the ensuing interaction as a three-stage game. In the first stage, agents decide whether (i) to open a staking pool, (ii) to delegate their stake to some pool or (iii) to abstain from validation activities. Setting up pools for validating transactions is costly, and these costs may differ between agents. In the second stage, either the Blockchain Designer determines the shares uniformly for all running pools (return fixing) or pool owners determine how returns should be shared between pool owners and delegators (return competition). In the third stage, transactions are validated and, depending on the share of stakes controlled by malicious agents, validation either works properly or the blockchain is disrupted.

Our main insights start from the observation that honest agents with high costs to set-up a node as a validator may want to delegate their stake to other pool owners, while honest agents with low costs may want to open their own pool. Malicious agents always open a pool, as this increases their chances to disrupt the blockchain.

We establish existence and uniqueness of equilibria of the stake pool formation game with fixed return distribution between pool owners and delegators and show that they are of the threshold type. We show next that there exists a unique sharing rule of the returns from validation between delegators and pool owners that maximizes the probability that the blockchain operates correctly and we do the same for maximizing welfare of honest agents when costs of running staking pools are taken into account in addition to blockchain security.

Subsequently, we identify the potential and risk of staking pools. Staking pools can never increase current blockchain security over a system in which no such pools are allowed. The reason is as follows. Without staking pools, a share of honest agents participates in validating transactions, as the additional reward is higher than the costs. With staking pools and if the returns are shared with the validators, the return to pool owners declines, as the average rewards from validating transactions is given. Hence, less honest agents are willing to open staking pools, so that malicious agents will control a larger share of stakes in validating transactions [3].

Yet, by optimally choosing the distribution of the validation returns to delegators, the allocation of rewards to honest stakes involved in validation increases, which may be beneficial for subsequent blockchain operations. We show how return splitting between pool owners and delegators has to be determined in order to minimize the rewards to malicious agents.

[3] The return splitting cannot be strictly enforced by the protocol (designer) itself, but should rather be a recommendation. Therefore, if some pool owner publicly offers more rewards to delegators than is recommended, it should be a signal that this pool owner is not honest.

When pool owners can freely distribute the returns from validation to delegators, staking pools decrease blockchain security, since malicious agents attract delegators by distributing most of the returns to them.

We stress that our model is designed for many and small staking pools as this scenario is the ultimate goal of decentralization that should be achieved with a blockchain. On current blockchains, we observe also constellations with large staking pools, like LIDO[4] who hold a huge (above 25%) share of stakes on the largest PoS blockchain Ethereum.

The paper is organized as follows. In the next section, we discuss the related literature. In particular, literature motivating the formation of staking pools in PoS blockchains is reviewed. In Section 3, we introduce the model and preliminaries. In Section 4, we analyze the equilibria of the fixed return game. In Section 5, we discuss designs for staking rewards that maximize security. In Section 6, we analyze a return competition game, where pool owners individually decide on the reward sharing scheme. In Section 7, we study an extension of our basic model by considering endogenous rewards. Section 8 concludes.

## 2 Related Literature

*Staking Pools:* Many blockchains have already implemented staking or will implement it in the near future. Such examples include, but are not limited to Cardano ([12]), Solana ([15]), Polkadot ([14]), Tezos ([9]) and Concordium. All these allow staking pools in which agents who do not run their own staking pool will be able to delegate their stake to an existing pool and benefit from rewards. By delegation, agents are indirectly involved in block proposal and validation, via their stake.

[4] study staking pools among honest agents from an interesting mechanism design perspective. Their reward scheme ensures that a desired number of staking pools is achieved while each pool has approximately the same amount of stake and low-cost agents are running the pools. The reward scheme ensures that reporting the true costs is the dominant strategy. Our paper is complementary as we focus on the design of staking pools in the presence of malicious agents who want to disrupt the blockchain and thus have quite different objectives than honest agents. We examine on how such staking pools affect blockchain security, how security risks can be alleviated and how distribution of rewards to malicious agents can be limited. Our mechanism is also simpler to implement, as there is no communication between the designer and pool runners, and therefore, no need for contracting, unlike in [4]. It is also intuitive to interpret for the agents, than the generic mechanisms studied in there.

*Model Assumptions:* [11] study a voting game with two parties where voters vote for one party or abstain. In their model, voters have individual costs that are drawn according to some distribution function. Similarly to our paper, [11] characterize equilibria of their game which take the form of so called "cut-off

[4] `https://lido.fi`

thresholds". They obtain a pair of thresholds, one for each party. Then, a citizen whose cost is below the corresponding threshold will turn out and vote for his/her party. If a citizen's cost is above the threshold, s/he will abstain. Our model works in a similar manner. We characterize threshold equilibria such that if the cost for an individual is below that threshold, s/he will run a staking pool, and delegate or abstain otherwise.

Similar to [11], [5] studies a voting game where citizens' costs are drawn from a uniform distribution. In our paper, we also consider a uniform distribution for costs.

In our paper, we use continuum approach to model the measure of agents, and thus follow the approach in [8] and [10], for instance. More concretely, agents are modeled as infinitely small. The continuum approximates large communities and it proves to be a tractable approach for the staking pool formation game. The continuum model is thus a limiting case where the number of agents becomes large and delegation to staking pools is done uniformly at random. In our model, besides a pool ID (or address), no further information is provided to delegators. From addresses or pool IDs, no information about pool owners can be inferred. Hence, every pool has equal chances to be chosen by agents.[5] In the basic version of the model, delegation entails no cost.

# 3 Model

## 3.1 The General Set-up

There is a continuum of measure $H$ of *honest* agents[6] and there is a continuum of measure $M$ of *malicious* agents. The continua can be represented by intervals on the real line, with length $H$ and $M$, respectively. Working with a continuum of agents models a blockchain with a large number of participants and approximates the corresponding discrete model. While the continuum model is much more tractable and yields simpler expressions, it requires more subtle interpretations, though.

We assume $H > M$, so that, honest agents are in the majority. Each agent (malicious or honest) has one unit of the stake.[7]

Each honest agent is identified by his/her cost level for validation of transactions, respectively, for running a staking pool on the blockchain.[8] Costs are denoted by $c$ and are heterogeneous across honest agents, as they depend on the

---

[5] In practice, delegators may have more information about pool owners, but they remain anonymous. On the Cardano blockchain, for example, agents can find all staking pools on `pool.pm`, which visualizes all staking pools with the pool IDs, the current stake of the pools, the number of delegators and which blocks were produced by which staking pool.

[6] These agents are called "rational" by other authors, e.g. in [3] and [10].

[7] As the total amount of stakes is infinite, all variables which are integrated over the set of agents are averages in the continuum model.

[8] Costs include, for example, the costs for registering, running the software and forwarding messages on transactions.

availability of appropriate computer capital and human capital. Let the random variable $X$ correspond to the costs for honest agents. Specifically, the costs for honest agents are distributed according to the atomless density function $f(c)$ defined on $[0, T)$. Note that the support interval can be $\mathbb{R}_+$, that is, $T$ can be equal to $\infty$. The corresponding cumulative distribution function is denoted by $F(c)$. Malicious agents are of the Byzantine type and do not care about the costs and returns of running a pool. Hence, we set their costs to zero.

There is also a reward $R \in \mathbb{R}_+$, paid for creating the next block[9]. The agents' types are private information. As assumed above all honest and malicious agents have the same amount of stakes, equal to one unit. There is also a *Blockchain Designer*. The blockchain is assumed to be functioning better if more of validators are honest.

### 3.2 Objectives

The Blockchain Designer and the two types of agents have the following general objectives:

- Maximize the chance that the blockchain is working (maximizing blockchain security), which will be captured by a maximization of the number of honest validators (**Blockchain Designer**).
- Maximize expected reward minus cost (**Honest Agents**).
- Maximize the measure of stakes delegated to them (**Malicious Agents**).

We will formally specify the manifestation of these objectives later. As to the Blockchain Designer, we will consider an alternative objective where s/he trades-off the probability that the blockchain is running correctly with the costs for all honest agents to validate transactions. While we consider maximizing blockchain security as the most important objective, arguably one could also consider a standard economic welfare criterion as a guiding principle for the Blockchain Designer.

Two further aspects can be important for a Blockchain Designer. First, s/he may aim at minimizing the rewards received by malicious agents, as this decreases their future influence. Second, the Blockchain Designer may want to distribute rewards to validators as equally as possible, which is an original motivation of staking. We will discuss to which extent these aspects materialize when we present our results.

### 3.3 Staking Pool Formation Game

We consider the following game, which consists of three stages:

Stage 1: Agents decide either to form a staking pool or not (both honest and malicious). Agents who decide to become a pool owner obtain an identification number, denoted by $i$.

[9] In some contexts it is called mining reward.

Stage 2: Agents who did not register for a staking pool decide whether to delegate their stake to some staking pool or to remain idle.
Stage 3: The blockchain runs, validation takes place (or not), and rewards are distributed.

If an agent $i$ forms a staking pool, we denote by $s_i$ the amount of stakes s/he is receiving. We also denote by $P$ the measure of honest agents who form a staking pool. $D$ and $I$ denote the measure of honest agents who delegate their stakes or stay idle, respectively. We have $H = P + D + I$.

Our main assumption for this game is that delegators distribute themselves evenly across all possible pools. The rationale is that the type of a pool owner is private information, and for delegators, pool owners are all alike. Hence, invoking measure consistency, this assumption implies

$$s_i = \frac{D}{P + M}, \forall i.$$

We note that all pools obtain the same amount of delegated stakes and thus we write $s$ for $s_i$ in the following. The total size of the pool—stakes of the pool owner and delegated stakes—is then $s + 1$. We note that $(s + 1)(P + M) + I = D + P + M + I = M + H$.

The payoffs are determined as follows: The next validator[10] is chosen among the available pools proportionally to the pool size. This particular rule is already implemented in major PoS protocols (e.g., see [12]), as it perfectly replicates the PoW protocol, where probability for the next leader (in our case next validator who writes the block) is proportional to computing powers. Such a system has the advantage that splitting and pooling of the stakeholders does not increase the chances to be chosen as a next validator as the expected return is unaffected by such strategies.

The reward for the next block is given by an amount $R$. Since all pools have the same size, the return distribution is a uniform distribution with density $\frac{1}{P+M}$. Hence, the individual reward a pool expects to receive is $r = \frac{R}{P+M}$. Since we have a continuum model, we note that both the individual return for an individual and the cost of running a pool have zero weight in the average return $R$ and the average amount of costs, respectively. Yet for an agent, only the individual returns and costs matter.

The blockchain designer sets a parameter, denoted by $\lambda$, $0 < \lambda \leq 1$, which determines how rewards have to split between pool owners and delegators. To sum up, the individual expected rewards are as follows:

- A pool receives $r = \frac{R}{P+M}$.
- The pool owner obtains $\lambda \cdot r$.
- An individual delegator obtains $\frac{(1-\lambda)\cdot r}{s}$. The total amount given to the delegators in a pool is $(1 - \lambda) \cdot r$.
- An idle agent obtains 0.

[10] In blockchains, transaction validation is done by creating a new block.

We measure the probability that the blockchain operates correctly by a function

$$P_c(\cdot) : [0, T] \to [0, 1],$$

that depends on the share of honest agents running staking pools, that is (weakly) increasing as a function of this share, and which may reach probability one if a sufficient share of honest agents is participating in validating transactions. Later in this paper, we will study two different versions of this probability function and reward schemes that depend on it. In the basic version of the staking pool formation game, we assume that the returns are paid, no matter whether the blockchain operates correctly or not. The motivation for this assumption is as follows: Whether or not the blockchain operates correctly may not be immediately detected or agents maybe able to sell their rewards immediately after writing the next block. Hence, agents involved in validating transactions aim at maximizing the immediate returns from these activities in such cases. The Blockchain Designer is, of course, interested in how well the blockchain is functioning. In Section 7, however, we make rewards dependent on the operation of blockchains, that is, rewards are only distributed if the blockchain operates correctly.

The main design parameter $\lambda \in [0, 1]$ is a non-negative real number set by the Blockchain Designer. A game with a payoff structure as above, together with $(H, M, R, F, \lambda)$, is called a "staking pool formation game" and is denoted by $\mathcal{G}$.

# 4 Equilibrium Analysis

In this section, we analyze the equilibria of a staking pool formation game.

## 4.1 Equilibrium Concept

In a staking game with non-zero reward $R$, an honest agent will never stay idle. The reason is that since delegation is free of cost and delegators earn some reward, the expected payoff is positive for a delegator, whereas the payoff for staying idle is zero. Only in the case $\lambda = 1$, agents would be indifferent between delegation and staying idle. Hence, delegation weakly dominates staying idle and so $H = P + D$.

In Section **??**, we will consider the case where delegation comes at a small fixed cost and so, staying idle will not be dominated in general. For tractability we assume a tie-breaking rule. First, if an honest agent is indifferent between delegating and staying idle, s/he will choose to delegate. This implies that when there are no delegation costs, no agent remains idle.

We now proceed by focusing on equilibria of the threshold type. In particular, we solve for the threshold equilibrium by looking at the agent with a specific cost level $c^*$ at which this agent is indifferent between running a pool and delegating, i.e., the expected utility from delegating is equal to the expected utility from running a pool.

An important remark is in order. If a threshold equilibrium exists, it is unique. If a threshold equilibrium does not exist, we end up in a corner solution—either no honest agent will run a pool or all honest agents will run a pool, and thus there will be no delegation.

We introduce the following definition:

**Definition 1 (Threshold Equilibrium).** *A cost level $c^* > 0$ is called a "threshold equilibrium" if an agent with cost $c^*$ is indifferent between running a staking pool and delegating. Furthermore, all agents incurring a cost that is lower than $c^*$ will run a staking pool, and all agents with a cost greater than $c^*$ will delegate.*

In the threshold equilibrium we have $P = F(c^*)H$ and $D = (1 - F(c^*))H$.

### 4.2 Equilibrium Characterization

In the following we are looking for the equilibria, in which at least some fraction of honest agents decide to run own pools. We characterize the equilibria of the staking game:

**Theorem 1.** *There exists a unique threshold equilibrium to the game $\mathcal{G}$ if and only if*

$$\lambda > \frac{M}{H+M}. \tag{1}$$

*Proof.* First, we have to set up the indifference condition for an honest agent. That is, we have to equate the expected utility from being a delegator with the expected utility from running a pool. More precisely, the expected utility from being a delegator is

$$\frac{(1-\lambda)r}{s},$$

which is the share $(1-\lambda)$ of the reward $r$, divided by the number of delegators, for the particular pool. Similarly, the expected utility from running a pool is

$$\lambda r - c,$$

which is the share $\lambda$ of the reward $r$, minus the cost $c$ of the particular pool owner. In the equilibrium point $c^*$, an honest agent is indifferent between delegating and running a pool, that is, $c^*$ solves the indifference equation:

$$\frac{(1-\lambda)r}{s} = \lambda r - c^*. \tag{2}$$

From the equilibrium definition, we know that $P = F(c^*)H$ and $D = (1 - F(c^*))H$. Plugging in these values in (2) and simplifying, we obtain:

$$\frac{(1-\lambda)R}{(1-F(c^*))H} = \lambda \frac{R}{F(c^*)H + M} - c^*. \tag{3}$$

We reorder equation (3) as follows:

$$c^* = \frac{\lambda R}{F(c^*)H + M} - \frac{(1-\lambda)R}{(1-F(c^*))H}. \tag{4}$$

We note that the left hand side (LHS) of equation (4) is obviously increasing in $c^*$, while the right hand side (RHS) is decreasing in $c^*$. Indeed, the derivative of the RHS with respect to $c^*$ is

$$-\frac{\lambda R F'(c^*)H}{(F(c^*)H + M)^2} - \frac{(1-\lambda)RF'(c^*)}{(1-F(c^*))^2 H} < 0.$$

Since the LHS of (4) is increasing and is equal 0 for $c^* = 0$, and the RHS is decreasing in $c^*$, the necessary condition to have a solution to the equation is that the RHS is positive for $c^* = 0$. That is, we have the condition

$$\frac{\lambda R}{F(c^*)H + M} - \frac{(1-\lambda)R}{(1-F(c^*))H} > 0,$$

which is, for $c^* = 0$, equivalent to the condition in the theorem,

$$\lambda > \frac{M}{H+M}.$$

The indifference condition of the equilibrium of equation (4) to have an internal solution is obtained by taking $c^* = T$. In this case, the LHS has to be larger than the RHS, which always holds for $\lambda < 1$, as the RHS is equal to $-\infty$.

To establish uniqueness, suppose that $\lambda > \frac{M}{H+M}$. As shown above, if we focus on threshold equilibria, there exists a unique equilibrium characterized with the cost level $c^*$. Suppose that an equilibrium exists which is not of the threshold type. Without loss of generality, assume two cost levels $c_1$ and $c_2$ with $c_2 > c_1$, with the following property. A agent with cost $c_2$ will run a pool, while a agent with $c_1$ will delegate. Hence, for the first agent, it must hold that

$$\frac{\lambda R}{F(c^*)H + M} - c_2 > \frac{(1-\lambda)R}{(1-F(c^*))H},$$

while for the second agent, we must have the opposite inequality, that is,

$$\frac{\lambda R}{F(c^*)H + M} - c_1 < \frac{(1-\lambda)R}{(1-F(c^*))H}.$$

Together, this implies that

$$c_2 < \frac{\lambda R}{F(c^*)H + M} - \frac{(1-\lambda)R}{(1-F(c^*))H} < c_1,$$

which contradicts the assumption $c_2 > c_1$. Hence, any equilibrium will be of threshold type.

Since all equilibria are of the threshold-type, we simply refer to threshold-type equilibria as "equilibria".

The interpretation of the lower bound condition on $\lambda$ in the theorem is straightforward. To have a positive measure of pools owned by honest agents, the share of the reward for the pool owners should be higher than the share of malicious agents in the whole system. As long as $\lambda$ satisfies this condition, we have a unique equilibrium of the game $\mathcal{G}$. From the proof of Theorem 1, it is straightforward to see that if $\lambda = \frac{M}{H+M}$, then the only solution to the indifference condition is $c^* = 0$ and hence, all honest agents will delegate and malicious agents control all stakes. If $\lambda < \frac{M}{H+M}$, then there exists no equilibrium solution, where a positive measure of honest agents run pools.

## 5 Maximal Blockchain Security

In this section, we use the framework developed in the previous sections to design blockchains that maximize security.

We obtain the equilibrium solution of (3) for a given $\lambda$ by simply solving for $c^*$. We denote it by $c^*(\lambda)$. The inverse function is denoted by $\lambda(c^*)$, and can be trivially found from (3). Namely,

$$\lambda(c^*) = \frac{c^* + \frac{R}{(1-F(c^*))H}}{\frac{R}{F(c^*)H+M} + \frac{R}{(1-F(c^*))H}}. \tag{5}$$

$\lambda$ is a designer's variable.

We assume in this section that the probability that the blockchain operates correctly is given by:

$$P_c(c^*) := \frac{P(c^*)}{P(c^*) + M}.$$

That is, the probability that the next block consists of correct transactions is equal to a share of honest pool runners. This is a simple formulation, reflecting that the next block writer is chosen uniformly at random. However, our analysis holds qualitatively for any probability function that is increasing in the share of honest agents and may reach 1 if a sufficient share of honest agents is achieved.

The first goal of the designer is to maximize the share of honest agents running pools, that is, to maximize $P(c^*)$. The probability that the blockchain is run correctly depends on $P$, which is increasing in $c^*$. Increasing $\lambda$ has two effects on the honest agents' decision. First, it motivates an agent to run a pool, as a greater share of the rewards is allocated to the owner of the pool. On the other hand, since a higher $\lambda$ motivates many agents to run pools, there are many pools, and therefore, lower chances for each of them to win the reward. However, we obtain the following result:

**Proposition 1.** *The fraction of honest agents running a pool is maximized for* $\lambda = 1$.

*Proof.* Note that the RHS of (4) is increasing in $\lambda$. By increasing $\lambda$, we have to increase $c^*$ to have equality, as the RHS is decreasing in $c^*$. That is, if $\lambda_1 \leq \lambda_2$, then $c^*(\lambda_1) \leq c^*(\lambda_2)$. Taking the maximum value $\lambda = 1$ transforms equation (4) into

$$c^* = \frac{R}{F(c^*)H + M}. \tag{6}$$

The solution to this equation maximizes the share of honest validators.

We note that by setting $\lambda = 1$, in the threshold equilibrium of game $\mathcal{G}$, we do replicate the levels of honest and malicious stakes involved in transaction validation which would arise in the simple game without delegation and no staking pools. In this game, instead, honest agents are allowed to either validate transactions or abstain. In such a game, the indifference condition of the threshold equilibrium corresponds to $c^* = r$, equivalent to $\lambda = 1$ in the pool formation game. Yet, with $\lambda = 1$ and pool formation, all returns from validation are channelled to staking pool owners while delegators receive nothing. This is a concern for the future evolution of the blockchain since stake holding may be more and more concentrated on pool owners.

Note that the solution $\lambda = 1$ does not maximize social welfare, and does not distribute the rewards on honest agents that have high costs of running a pool either.

## 6 Return Competition

In this section, we reconsider the staking pool formation game. Instead of the Blockchain Designer, we allow that pool runners choose both their own levels of rewards and the rewards they want to distribute to delegators. A pool owner $i$ sets his/her own $\lambda_i$. This game is a variant of the game $\mathcal{G}$ studied so far in the paper.

In the first part of this section, we allow free return competition, that is, pool owner $i$ can choose any $\lambda_i \in [0, 1]$. We denote this game by $\mathcal{G}_0$. Hence, the game unfolds as follows:

Stage 1: Agents (both honest and malicious) either decide to form a staking pool or not. Agents who decide to become a pool owner obtain an identification number, denoted by $i$, and set $\lambda_i$.
Stage 2: Agents who did not register for a staking pool decide whether to delegate their stake to some staking pool or to remain idle.
Stage 3: The blockchain runs, validation takes place (or not), and rewards are distributed.

We obtain the following result:

**Proposition 2.** *In any equilibrium of the game $\mathcal{G}_0$, malicious agents control all stakes involved in transaction validation and the blockchain is disrupted.*

*Proof.* Suppose that there exists an equilibrium in which honest agents run staking pools. Since running such staking pools is costly and there is zero measure of honest agents with zero costs, such an equilibrium necessarily must involve that the minimal value of all offered values $\lambda_i$ by staking pool owners, denoted by $\hat{\lambda}$, must be positive. Otherwise, honest agents are better off by delegating their stakes. Note that the last statement holds, since an individual honest agent has no influence on the probability that the blockchain operates correctly by his/her decision whether to run a staking pool or to delegate,

However, every malicious agent has an incentive to deviate and to set a lower value of $\hat{\lambda}$ for his/her own staking pool in order to attract more delegators, thereby making staking pools for honest agents unattractive. Hence, all honest agents delegate. This is a contradiction that honest agents run staking pools. Hence, in any equilibrium, malicious agents control all stakes involved in transaction validation and the blockchain is disrupted..

In the second part of this section, agents are only allowed to choose their corresponding $\lambda_i$ from the interval $[\bar{\lambda}, 1]$, where $\bar{\lambda} > \frac{M}{H+M}$. We denote this game by $\mathcal{G}_{\bar{\lambda}}$. Invoking standard Bertrand competition logic, we obtain the following result in this case:

**Proposition 3.** *The equilibrium of the extended game $\mathcal{G}_{\bar{\lambda}}$ is the same as the equilibrium of game $\mathcal{G}$.*

*Proof.* First, we note that in the equilibrium, all malicious agents choose the lowest possible level $\bar{\lambda}$. If honest agents choose any $\lambda$ that is strictly larger than $\bar{\lambda}$, then nothing is delegated to them. Therefore, they also choose the same $\bar{\lambda}$.

That is, imposing a lower bound on $\lambda$ also guarantees the same upper bound. This adds to the robustness of the result obtained in Theorem 1 and offers a way to implement the equilibrium solution. On a practical side, the blockchain system does not need to force the agents to have the same level of rewards. Rather, they reach it through rational play. Note that setting any lower bound $\bar{\lambda} \leq \frac{M}{H+M}$ would result in the same (bad) equilibrium obtained in Proposition 2.

## 7 Endogenous Rewards

Throughout the paper, we assumed that rewards for writing the next block are exogenously given and are equal to a constant number $R$, no matter what fraction of honest agents runs pools and participates in the validation. In this section, we assume that rewards are realized only if the blockchain functions correctly, with probability one. This probability is calculated by the following formula:

$$P_c(c^*) := \min\left[1, \left(\frac{P}{P+M} + \theta\right)\right].$$

Here $\theta$ is a real number in $[0, \frac{1}{2}]$ that describes the tolerance of a system regarding the share of malicious agents it can handle without compromising network security.

We say that full network security is achieved when

$$\frac{P}{P+M} \geq 1-\theta.$$

Typically, for example in Byzantine-fault-tolerant protocols, $\theta$ is about $\frac{1}{3}$ in many consensus protocols (see [13], [1], [6] [2] and [7]). Thus, in that case, if the fraction of staking pools run by honest agents is at least $\frac{2}{3}$, then full security is achieved.

Requiring the probability of the blockchain security to surpass the threshold $1-\theta$ imposes a threshold on the cost of pool running in the equilibrium. We denote the corresponding game by $\mathcal{G}^e$ and the cost threshold by $c^\theta$. It is defined by the following equation:

$$c^\theta := \inf_c \frac{F(c)H}{F(c)H+M} \geq 1-\theta. \tag{7}$$

In this setting, to have a unique threshold equilibrium, we need

$$c^\theta < \frac{\lambda R}{F(c^\theta)H+M} - \frac{(1-\lambda)R}{(1-F(c^\theta))H}. \tag{8}$$

We obtain a result similar to the one of Theorem 1:

**Theorem 2.** *There exists a unique threshold equilibrium $c^* > c^\theta$ to the game $\mathcal{G}^e$ if and only if*

$$\lambda > \frac{\frac{c^\theta}{R}(F(c^\theta)H+M)(1-F(c^\theta))H + F(c^\theta)H+M}{H+M}. \tag{9}$$

The proof is analogous to the proof of Theorem 1.

The results from the Section 6 on return competition are also translated directly in this setting. In particular, if the agents are allowed to set any $\lambda \in [0,1]$ as their own pool return, the blockchain security fails. We denote this game by $\mathcal{G}_0^e$. Formally, we obtain the following result:

**Proposition 4.** *In any equilibrium of the game $\mathcal{G}_0^e$ no honest agent runs a pool and the blockchain is disrupted.*

The proof is analogous of the proof of Proposition 2 and exploits the fact that individual honest agents have zero measure. Therefore, any unilateral deviation by an agent does not affect the probability that the blockchain operates correctly.

## 8 Conclusion

In this paper, we initiated the study of the formation of staking pools from game-theoretic and mechanism design perspectives. Our insights can help to design reward distribution rules that improve the blockchain security and fairness of reward distribution. This study might open many further research avenues on how staking pools can be designed optimally for blockchains. For instance, one might introduce quantity constraints such as a leverage constraint on staking pools which limits the share of delegators towards the pool owner. Whether such a constraint further improves the security of the blockchain is left for future research. One might also allow the history of staking and running staking pools to play a role in dynamic versions of the game and thus, an agent's reputation to behave honestly may be taken into account in such staking pool formation games. Future research can also extend our model to arbitrary distribution of stakes across agents.

## References


1. Abd-El-Malek, M., Ganger, G.R., Goodson, G.R., Reiter, M.K., Wylie, J.J.: Fault-scalable Byzantine fault-tolerant services. ACM SIGOPS Operating Systems Review **39**(5), 59–74 (oct 2005). https://doi.org/10.1145/1095809.1095817, https://doi.org/10.1145/1095809.10gaagzavne?,2min,:D,2min, romelzegaqbjgasaibreba,2min,meargawuxebemzadebaadamianitqo95817
2. Aiyer, A.S., Alvisi, L., Clement, A., Dahlin, M., Martin, J., Porth, C.: BAR fault tolerance for cooperative services. In: Herbert, A., Birman, K.P. (eds.) Proceedings of the 20th ACM Symposium on Operating Systems Principles 2005, SOSP 2005, Brighton, UK, October 23-26, 2005. pp. 45–58. ACM (2005). https://doi.org/10.1145/1095810.1095816, https://doi.org/10.1145/1095810.1095816
3. Amoussou-Guenou, Y., Biais, B., Potop-Butucaru, M., Tucci-Piergiovanni, S.: Rational vs Byzantine players in consensus-based blockchains. In: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems. p. 43–51. AAMAS '20, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2020)
4. Brünjes, L., Kiayias, A., Koutsoupias, E., Stouka, A.P.: Reward sharing schemes for stake pools. In: 2020 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 256–275 (2020). https://doi.org/10.1109/EuroSP48549.2020.00024
5. Castanheira, M.: Victory margins and the paradox of voting. European Journal of Political Economy **19**(4), 817–841 (2003). https://doi.org/https://doi.org/10.1016/S0176-2680(03)00038-7, https://www.sciencedirect.com/science/article/pii/S0176268003000387
6. David, B., Gaži, P., Kiayias, A., Russell, A.: Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake protocol. Cryptology ePrint Archive, Report 2017/573 (2017), https://ia.cr/2017/573
7. Dinsdale-Young, T., Magri, B., Matt, C., Nielsen, J.B., Tschudi, D.: Afgjort: A partially synchronous finality layer for blockchains. Cryptology ePrint Archive, Report 2019/504 (2019), https://ia.cr/2019/504
8. Gersbach, H.: Democratic mechanisms. Journal of the European Economic Association **7**(6), 1436–1469 (2009), http://www.jstor.org/stable/40601208
9. Goodman, L.: Tezos — A self-amending crypto-ledger white paper. https://tezos.com/whitepaper.pdf (2014)
10. Halaburda, H., He, Z., Li, J.: An economic model of consensus on distributed ledgers. Working Paper 29515, National Bureau of Economic Research (November 2021). https://doi.org/10.3386/w29515, http://www.nber.org/papers/w29515

11. Herrera, H., Morelli, M., Palfrey, T.: Turnout and power sharing. Economic Journal **124**(574), F131–F162 (02 2014). https://doi.org/10.1111/ecoj.12116, https://doi.org/10.1111/ecoj.12116
12. Kiayias, A., Russell, A., David, B., Oliynykov, R.: Ouroboros: A provably secure proof-of-stake blockchain protocol. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10401, pp. 357–388. Springer (2017). https://doi.org/10.1007/978-3-319-63688-7_12, https://doi.org/10.1007/978-3-319-63688-7_12
13. Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. ACM Transactions on Programming Languages and Systems **4**(3), 382–401 (jul 1982). https://doi.org/10.1145/357172.357176, https://doi.org/10.1145/357172.357176
14. Wood, G.: Polkadot: Vision for a heterogeneous multi-chain framework. https://polkadot.network/PolkaDotPaper.pdf (2016)
15. Yakovenko, A.: Solana: A new architecture for a high performance blockchain v0.8.13. https://solana.com/solana-whitepaper.pdf (2017)