Compounding of Wealth in Proof-of-Stake Cryptocurrencies

Giulia Fanti[‡], Leonid Kogan[⋆], Sewoong Oh[†], Kathleen Ruan[‡], Pramod Viswanath[†], Gerui Wang[†]

[‡]Carnegie Mellon University, [⋆]Massachusetts Institute of Technology, [†]University of Illinois Urbana-Champaign

Abstract. Proof-of-stake (PoS) is a promising approach for designing efficient blockchains, where block proposers are randomly chosen with probability proportional to their stake. A primary concern in PoS systems is the "rich getting richer" effect, whereby wealthier nodes are more likely to get elected, and hence reap the block reward, making them even wealthier. In this paper, we introduce the notion of equitability, which quantifies how much a proposer can amplify her stake compared to her initial investment. Even with everyone following protocol (i.e., honest behavior), we show that existing methods of allocating block rewards lead to poor equitability, as does initializing systems with small stake pools and/or large rewards relative to the stake pool. We identify a *geometric* reward function, which we prove is maximally equitable over all choices of reward functions under honest behavior and bound the deviation for strategic actions; the proofs involve the study of optimization problems and stochastic dominances of Pólya urn processes. These results allow us to provide a systematic framework to choose the parameters of a practical incentive system for PoS cryptocurrencies.

1 Introduction

A central problem in blockchain systems is that of block proposal: how to choose which block should be appended to the global blockchain next. Many blockchains use a proposal mechanism by which one node is randomly selected as leader (or *block proposer*). This leader gets to propose the next block in exchange for a token reward—typically a combination of transaction fees and a freshly-minted *block reward*, which is chosen by the system designers. Early cryptocurrencies, including Bitcoin, mainly used a leader election mechanism called *proof of work* (PoW). Under PoW, all nodes execute a computational puzzle. The node who solves the puzzle first is elected leader. PoW is quite robust to security threats, but also energy-inefficient, consuming more energy than developed nations [1].

An appealing alternative to PoW is called *proof-of-stake* (PoS). In PoS, proposers are not chosen according to their computational power, but according to

Email: gfanti@andrew.cmu.edu, lkogan2@mit.edu, swoh@illinois.edu, kruan@andrew.cmu.edu, pramodv@illinois.edu, geruiw2@illinois.edu

the stake they hold in the cryptocurrency. For example, if Alice has 30% of the tokens, she is selected as the next proposer with probability 0.3. Although the idea of PoS is both natural and energy-efficient, the research community is still grappling with how to design a PoS system that provides security while also incentivizing nodes to act as network validators. Part of incentivizing validators is simply providing enough reward (in expectation) to compensate their resource usage. However, it is also important to ensure that validators are treated fairly compared to their peers. In other words, they cannot only be compensated adequately on average; the variance also matters.

This observation is complicated in PoS systems by a key issue that does not arise in PoW systems: *compounding*. Compounding means that whenever a node (Alice) earns a proposal reward, that reward is added to her account, which increases her chances of being elected leader in the future, and increases her chances of reaping even more rewards. This leads to a rich-get-richer effect, causing dramatic concentration of wealth.

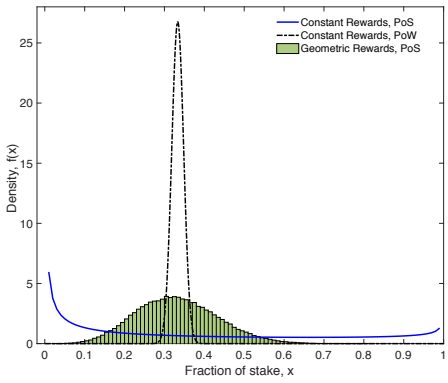


Fig. 1: Fractional stake distribution of a party that starts with 1/3 of the stake in a system initialized with Bitcoin's financial parameters. Results of geometric reward PoS and constant reward PoW are shown after $T = 1,000$ blocks.

To see this, consider what would happen if Bitcoin were a PoS system. Bitcoin started with an initial stake pool of 50 BTC, and the block reward was fixed at 50 BTC/block for several years. Under these conditions, suppose a party A starts with $\frac{1}{3}$ of the stake. Using a basic PoS model described in Section 2, A's stake would evolve according to a standard Pólya urn process [14], converging almost surely to a random variable with distribution Beta$(\frac{1}{3}, \frac{2}{3})$ [17], (blue solid line in Figure 1). In this example, compounding gives A a high probability of accumulating a stake fraction near 0 or 1. This is highly undesirable because the proposal incentive mechanism should not unduly amplify or shrink one party's fraction of stake. Notice that this is not caused by an adversarial or strategic behavior, but by the *randomness* in the PoS protocol, combined with compounding.

In PoW, on the other hand, the analogue would be for party A to hold 1/3 of the computational power. In that case, A's stake after T blocks would be instead binomially distributed with mean $50\,T/3$ (black dashed line in Figure 1). Notice that the binomial (PoW) stake distribution concentrates around 1/3 as $T \to \infty$, so if A contributes 1/3 of stake at the beginning, she also reaps

1/3 of the rewards in the long term.[1] Among randomized protocols that choose proposers independently at each time slot, the binomial distribution is the best we can hope for; it represents the setting where party A wins each block with probability equal to its initial stake. A natural question is whether we can achieve this PoW baseline distribution in a PoS system with compounding.

We study this question from the perspective of the block reward function. Most cryptocurrencies today use a *constant block reward* function like Bitcoin's, which remains fixed over a long timespan (e.g., years). We ask how a PoS system's choice of block reward function can affect concentration of wealth, and whether one can achieve the PoW baseline stake distribution simply by changing the block reward function. This paper has five main contributions:
(1) We define the *equitability* of a block reward function, which intuitively captures how much the fraction of total stake belonging to a node can grow or shrink (under that block reward function), compared to the node's initial investment.
(2) We introduce an alternative block reward function called the *geometric reward function*, whose rewards increase geometrically over time. We show that it is the most equitable PoS block reward function, by showing that it is the unique solution to an optimization problem on the second moment of a time-varying urn process; this optimization may be of independent interest. We note that despite optimizing equitability, geometric rewards do not achieve the PoW baseline stake distribution—this is the *inherent* price we pay for the efficiency of PoS compared to PoW. The green histogram in Figure 1 illustrates the empirical, simulated stake distribution when geometric rewards are used for 1 000 blocks, with total rewards as in the PoW example ($50 \times 1\,000$ units).
(3) Borrowing ideas from mining pools in PoW systems, a natural strategy is for participants in a PoS system to form stake pools. We quantify the exact gains of stake pool formation in terms of equitability, which proves that participating in a stake pool can significantly reduce the compounding effect of a PoS system.
(4) We study the effects of strategic behavior (e.g. selfish mining) on the rich-get-richer phenomenon. We find that in general, compounding can exacerbate the efficacy of strategic behavior compared to PoW systems. However, these effects can be partially mitigated by carefully choosing the amount of block reward dispensed over some time period relative to the initial stake pool size.
(5) Our analyses of the equitability of various reward functions provide guidelines for choosing system parameters—including the initial token pool size and the total rewards to dispense in a given time interval—to ensure equitability. We show that cryptocurrencies that start with large initial stake pools (relative to the block rewards being disseminated) can mitigate the concentration of wealth, both for constant and geometric reward schemes.

The remainder of this paper is organized as follows. In Section 2, we present our model. In Section 3, we study equitability under honest behavior. We use Section 4 to study the effects of strategic behavior on equitability.

[1] Compounding can also happen in PoW if miners use their profits to purchase more mining equipment. However, this feedback loop is much slower and less direct than PoS compounding, so we approximate PoW by a system with no compounding.

1.1 Related work

The compounding of wealth in PoS systems has been widely discussed in forum and blog posts [25,20,30], with recent work on *stake-bleeding attacks* exploiting exactly this property [11]. In this work, we quantify concentration of wealth through a new metric called equitability, which enables us to mathematically compare PoS to PoW, and different block reward schemes. As we discuss in Section 2, equitability is closely tied to the variance of a block reward scheme. Thus far, researchers and practitioners have reduced variance in block rewards through two main approaches: pooling resources (e.g., mining or stake pools) and proposing new protocols for disseminating block rewards.

Resource pooling is common in cryptocurrencies, e.g. in mining pools [28,10]. In PoS systems, the analogous concept is stake pooling, where nodes aggregate their stake under a single node; block rewards are shared across the pool. In Section 3, we show that the proposed geometric reward function is still the most equitable even if some parties are forming stake pools. Recent work by Brunjes *et al.* also studies stake pools and how to incentivize their formation through the design of reward mechanisms [6]. Our work differs in that we aim to optimize equitability, whereas [6] aims to incentivize the formation of a target number of mining pools. Also, [6] does not consider the effects of compounding in PoS.

A second variance reduction approach changes the block reward allocation protocol; our work falls in this category. Two examples are Fruitchains [23], which spread block rewards evenly across a sequence of block proposers, and Ouroboros [16], which rewards nodes for being part of a block formation committee, even if they do not contribute to block proposal. Both of these approaches were proposed in order to provide incentive-compatibility for block proposers; they do not explicitly aim to reduce the variance of rewards. However, they implicitly reduce variance by spreading rewards across multiple nodes, thereby preventing the randomized accumulation of wealth. In our work, instead of changing how block rewards are disseminated, we change the block reward function itself.

2 Models and Notation

We provide a probabilistic model for the evolution of the stakes under a PoS system, and introduce a measure of fairness, we call *equitability*. We begin with a model of a chain-based proof-of-stake system with m parties: $\mathcal{A} = \{A_1, \ldots, A_m\}$. We assume that all parties keep all of their stake in the *proposal stake pool*, which is a pool of tokens that is used to choose the next proposer. We consider a discrete-time system, $n = 1, 2, \ldots, T$, where each time slot corresponds to the addition of one block to the blockchain. In reality, new blocks may not arrive at perfectly-synchronized time intervals, but we index the system by block arrivals. For any integer x, we use the notation $[x] := \{1, 2, \ldots, x\}$. For all $i \in [m]$, let $S_{A_i}(n)$ denote the total stake held by party A_i in the proposal stake pool at time n. We let $S(n) = \sum_{i=1}^{m} S_{A_i}(n)$ denote the total stake in the proposer stake

pool at time n, and $v_{A_i}(n)$ denotes the *fractional stake* of node A_i at time n:

$$v_{A_i}(n) = \frac{S_{A_i}(n)}{S(n)}.$$

For simplicity, we normalize the initial stake pool size to $S(0) = 1$; this is without loss of generality as the random process is homogeneous in scaling both the rewards and the initial stake by a constant. Each party starts with $S_{A_i}(0) = v_{A_i}(0)$ fraction of the original stake. At each time $n \in [T]$, the system chooses a proposer node $W(n) \in \mathcal{A}$ so that

$$W(n) = \begin{cases} A_1 & w.p. \quad v_{A_1}(n) \\ \dots & \\ A_m & w.p. \quad v_{A_m}(n). \end{cases} \tag{1}$$

Upon being selected as a proposer, $W(n)$ appends a *block*, or set of transactions, to the *blockchain*, which is a sequential list of blocks held by all nodes in the system. As compensation for this service, $W(n)$ receives a *block reward* of $r(n)$ stake, which is immediately added to its allocation in the proposer pool. I.e.,

$$S_{W(n)}(n+1) = S_{W(n)}(n) + r(n).$$

The reward $r(n)$ is freshly-minted, so it increases the total token pool size. We assume the total reward dispensed in time period T is fixed, such that $\sum_{n=1}^{T} r(n) = R$.

Modeling Assumptions. Our model implicitly makes several assumptions, such as a single proposer per time slot. Many cryptocurrencies have proposer election protocols that allow more than one proposer to be chosen per time slot (Bitcoin [21], PoSv3 [9], Snow White [5]). If two proposers are elected at time n, for example, then each can append its block to one block at height $n-1$; here the *height* of a block is its index in the blockchain. However, in these systems, only one leader can win the block reward since only one fork of the blockchain is ultimately adopted. Assuming the winner is chosen uniformly at random from the set of selected proposers, the dynamics of our Markov process remain unchanged.

Some cryptocurrencies (e.g., Qtum, Particl) choose proposer(s) as a function of the time slot *and* the preceding block. This does not affect our results in the honest setting (for the same reason as above), but it does increase the efficacy of strategic behavior like grinding [32] and selfish mining [10]. We discuss these implications in Section 4. Although we do not consider BFT-based PoS protocols in this paper [31,12], such protocols provide robustness to strategic behavior by forcing consensus on each block. Such protocols may also provide robustness to compounding, since block rewards can be shared among many nodes.

We have also assumed in this work that users instantly re-invest rewards into the proposer stake pool, for two reasons. (1) In PoS systems where users explicitly deposit stake, existing implementations automatically deposit rewards back

into the stake pool. For example, the reference implementation of Casper the Friendly Finality Gadget (a PoS finalization mechanism proposed for Ethereum) automatically re-allocates all rewards back into the deposited stake pool [26]. (2) In other PoS systems, the stake pool is simply the set of all stake in the system, and is not separate from the pool of tokens used for transactions [9]. Hence as soon as a proposer earns a reward, that reward is used to calculate the next proposer (modulo some maturity period); the user is not actively re-investing block rewards—it just happens naturally. In practice, there may be a delay (maturity period) before the reward is counted; we do not model this effect.

Block reward choices. Many cryptocurrencies use Bitcoin's block reward schedule, which fixes the total supply of coins at about 21 million coins, and halves the reward every 210,000 blocks (≈ 4 years) [2]. Figure 2 illustrates this; if we let T_i and R_i denote the ith block interval and total reward, respectively, we can take $T_i = 210,000$ blocks, and $R_i = 50 \cdot \frac{1}{2^{i-1}} \cdot 210,000$. Several systems have adopted similar block rewards that are constant over long periods of time (e.g., Ethereum [3], ZCash [13], Dash [8], Particl [15]).

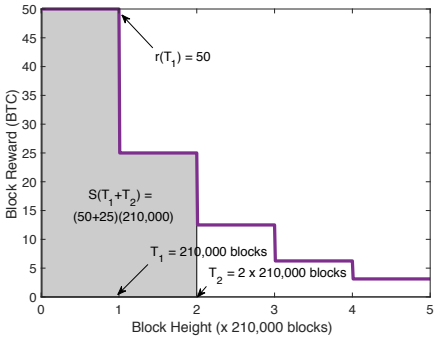


Fig. 2: Bitcoin block rewards as a function of block height. The area of the shaded region gives the total stake after $T_1 + T_2$ time.

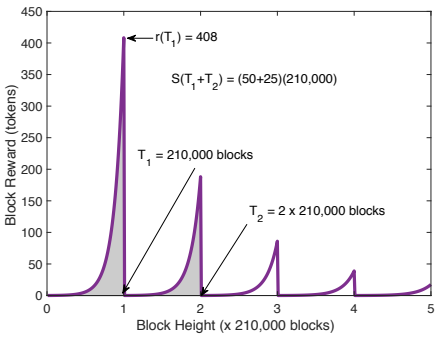


Fig. 3: Geometric block rewards as a function of block height, using Bitcoin-based T_i and R_i values from Figure 2.

In this paper, we revisit the question of how to choose $r(n)$. A key observation is that $r(n)$ must compensate nodes for the cost of proposing blocks. Many cryptocurrencies implicitly adopt the following maxim:

On short timescales, each block should yield the same block reward.

Notice that this maxim does not specify whether the value of a block reward is measured in tokens or in fiat. As illustrated earlier, most cryptocurrencies today measure value in tokens. We call this approach the *constant block reward*:

$$r_c(n) := \frac{R}{T}. \tag{2}$$

A natural alternative is to measure the block reward's value in fiat currency. This depends on the cryptocurrency's valuation over time interval $[T]$; if we assume it to be constant, then the resulting reward function should give a constant fraction of the *total* stake at each time slot. We call this the *geometric* reward:

$$r_g(n) := (1+R)^{\frac{n}{T}} - (1+R)^{\frac{n-1}{T}}. \tag{3}$$

Figure 3 shows geometric block rewards as a function of time if we use the same T_i's and R_i's as in Figure 2, reflecting Bitcoin's block reward schedule.

Equitability. To compare reward functions, we define a metric called equitability. Consider the stochastic dynamic of the fractional stake of a party A that starts with $v_A(0)$ fraction of the initial total stake of $S(0) = 1$. We denote the fractional stake at time n by $v_{A,r}(n)$, to make the dependence on the reward function explicit. A straw-man metric for measuring fairness is the expected fractional stake at time T: i.e., if A contributes 10% of the proposal stake pool at the beginning of the time, then A should reap 10% of the total disseminated rewards on average. This metric is poor because PoS systems elect a proposer (in Eq (1)) with probability proportional to the fractional stake; this ensures that each party's expected fractional reward is equal to its initial stake fraction, for any block reward function. That is, $\forall n \in [T]$, $\mathbb{E}[v_{A,r}(n)] = v_A(0)$. This follows from the law of total expectation and the fact that

$$\begin{aligned}&\mathbb{E}[v_{A,r}(n) \,|\, v_{A,r}(n-1) = v] \\ &\quad = v\frac{v\,S(n-1) + r(n-1)}{S(n)} + \big(1-v\big)\frac{v\,S(n-1)}{S(n)} = v.\end{aligned}$$

Although all reward functions yield the same expected fractional stake, the choice of reward function can nonetheless dramatically change the distribution of the final stake, as seen in Figure 1. We therefore instead propose using the *variance* of the final fractional stake, $\mathrm{Var}(v_{A,r}(T))$, as an equitability metric. Intuitively, smaller variance implies less uncertainty and higher equitability:

Definition 1. *For a positive vector $\boldsymbol{\varepsilon} \in \mathbb{R}^m$, we say a reward function $r : [T] \to \mathbb{R}^+$ over T time steps is* $\boldsymbol{\varepsilon}$-equitable *for $\boldsymbol{\varepsilon} = [\varepsilon_1, \ldots, \varepsilon_m]$ where $\varepsilon_i > 0$, if*

$$\frac{\mathrm{Var}(v_{A_i,r}(T))}{v_{A_i}(0)(1 - v_{A_i}(0))} \;\leq\; \varepsilon_i \tag{4}$$

for all $i \in [m]$. For two reward functions $r_1 : [T] \to \mathbb{R}^+$ and $r_2 : [T] \to \mathbb{R}^+$ with the same total reward, $\sum_{n=1}^{T} r_1(n) = \sum_{n=1}^{T} r_2(n)$, we say r_1 is more equitable *than r_2 for player $i \in [m]$ if*

$$\mathrm{Var}\big(v_{A_i,r_1}(T)\big) \;\leq\; \mathrm{Var}\big(v_{A_i,r_2}(T)\big)\,, \tag{5}$$

when both random processes start with the same initial fraction of $v_{A_i}(0)$.

The normalization in Eq. (4) ensures the left-hand side is at most one, as we show in Remark 1. It also cancels out the dependence on the initial fraction $v_A(0)$ such that the left-hand side only depends on the reward function r and the time T, as shown in Lemma 1.

Remark 1. When starting with an initial fractional stake $v_A(0)$, the maximum achievable variance is

$$\sup_{T\in\mathbb{Z}^+} \sup_{r} \mathrm{Var}(v_{A,r}(T)) \;=\; v_A(0)(1-v_A(0))\,, \tag{6}$$

where the supremum is taken over all positive integers T and reward function $r : [T] \to \mathbb{R}^+$. *(Proof in Appendix C.1)*

From the analysis of a time-dependent Pólya's urn model, we know the variance satisfies the following formula (see proof in Appendix C.2 and also [24]).

Lemma 1. *Let* $e^{\theta_n} \triangleq S(n)/S(n-1)$*, then*

$$\mathrm{Var}(v_{A,r}(T)) = \big(v_{A,r}(0) - v_{A,r}(0)^2\big)\Big(1 - \frac{S(0)^2}{S(T)^2}\prod_{n=1}^{T}(2e^{\theta_n}-1)\Big)\,. \tag{7}$$

(Proof in Appendix C.2)

Although Definition 1 applies to an arbitrary number of parties, Lemma 1 implies that it is sufficient to consider a single party's stake. More precisely:

Remark 2. If reward function $r : [T] \to \mathbb{R}^+$ over T time steps is $\boldsymbol{\varepsilon}$*-equitable* for vector $\boldsymbol{\varepsilon} = [\varepsilon_1, \ldots, \varepsilon_m]$ where $\varepsilon_i > 0$, then r is also $\tilde{\boldsymbol{\varepsilon}}$-equitable, where

$$\tilde{\boldsymbol{\varepsilon}} \triangleq \mathbf{1} \cdot \min_{i\in[m]} \varepsilon_i,$$

with $\mathbf{1}$ denoting the vector of all ones.

As such, the remainder of this paper will study equitability from the perspective of a single (arbitrary) party A. We will also describe reward functions as ε-equitable as shorthand for $\boldsymbol{\varepsilon}$-equitable, where $\boldsymbol{\varepsilon} = \mathbf{1} \cdot \varepsilon$. Note that even if the total reward R is fixed, equitability can differ dramatically across reward functions. In the example of Figure 1, the constant reward function is 0.5-equitable. On the other hand, the geometric rewards of (3) have a smaller chance of losing all its fractional stake (i.e. $v_{A,r_g}(T) \approx 0$) or taking over the whole stake (i.e. $v_{A,r_g}(T) \approx 1$). It is 0.05-equitable in this example.

3 Equitability under Honest Behavior

In this section, we analyze the equitability of different block reward functions, assuming that every party is honest, i.e. follows protocol, and the PoS system is *closed*, so no stake is removed or added to the proposal stake pool over a fixed

time period T. Each party's stake changes only because of the block rewards it earns and compounding effects. We discuss the effects of strategic behavior in Section 4, and open systems in Appendix A.3.

The metric of equitability leads to a core optimization problem for PoS system designers: given a fixed total reward R to be dispensed, how do we distribute it over the time T to achieve the highest equitability? Perhaps surprisingly, we show that this optimization has a simple, closed-form solution.

Theorem 1. *For all $R \in \mathbb{R}^+$ and $T \in \mathbb{Z}^+$, the geometric reward r_g defined in* (3) *is the most equitable among functions that dispense R tokens over time T, jointly over all parties A_i, for $i \in [m]$.* (Proof in Appendix C.3)

Intuitively, geometric rewards optimize equitability because they dispense small rewards in the beginning when the stake pool is small, so a single block reward cannot substantially change the stake distribution. The rewards subsequently grow proportionally to the size of the total stake pool, so the effect of a single block remains bounded throughout the time period. We emphasize that the geometric reward function does not depend on the initial stake of the party A, and hence is universally most equitable for all parties in the system simultaneously.

Composition. The geometric reward function does not only optimize equitability for a single time interval. Consider a sequence $(T_1, R_1), \ldots, (T_k, R_k)$ of checkpoints, where T_i is increasing in i, and R_i denotes the amount of reward to be disbursed between time $T_{i-1} + 1$ and T_i (inclusive). These checkpoints could represent target inflation rates on a monthly or yearly basis, for instance. A natural question is how to choose a block reward function that optimizes equitability over all the checkpoints jointly. The solution is to iteratively and independently apply geometric rewards over each time interval, giving a block reward function like the one shown in Figure 3.

Theorem 2. *Consider a sequence of checkpoints $\{(T_i, R_i)\}_{i\in[k]}$. Let $\tilde{R}_j := \sum_{i=1}^{j} R_i$. The most equitable reward function is*

$$r(n) = (1 + \tilde{R}_{i-1}) \left(\left(\frac{1 + \tilde{R}_i}{1 + \tilde{R}_{i-1}} \right)^{\frac{n - T_{i-1}}{T_i - T_{i-1}}} - \left(\frac{1 + \tilde{R}_i}{1 + \tilde{R}_{i-1}} \right)^{\frac{n - 1 - T_{i-1}}{T_i - T_{i-1}}} \right) \tag{8}$$

for $n \in [T_{i-1} + 1, T_i]$. (Proof in Appendix C.4)

When there is only one checkpoint, Theorem 2 simplifies to Theorem 1. This implies that checkpoints can be chosen *adaptively*, i.e., they do not need to be fixed upfront to optimize equitability. Because of composition, we assume a single checkpoint for the remainder of this paper. In practice, the abrupt change in geometric block rewards at a checkpoint (Figure 3) may lead to miner/validator attrition [4]. Liquidity limits may slow down this attrition, but cannot stop it [5].

One option is that a PoS system need not choose its block reward function based on equitability alone; it could also consider smoothness and/or monotonicity constraints. Another is that PoS blockchains could use geometric rewards only for the first epoch (when compounding poses the greatest risk), and then transition to a smoother block reward schedule of their choosing. We leave such exploration to future work.

Stake Pools. Participants also have the freedom to form stake pools, as explored in [28,10,6]. We show that stake pools reduce the variances of the fractional stake of all pool members, and quantify this gain. Consider a single party that owns $v_A(0)$ fraction of the stake at time $t = 0$. We know from Lemma 1 that the variance at time T is $\mathrm{Var}(v_{A,r}(T)) = \big(v_A(0) - v_A(0)^2\big)\Big(1 - \frac{S(0)^2}{S(T)^2}\prod_{n=1}^{T}(2e^{\theta_n} - 1)\Big)$. Consider a case where the same party now participates in a stake pool, where the pool P has $v_P(0)$ of the initial stake (including the contribution from party A), and every time the stake pool is awarded a reward for block proposal, the reward is evenly shared among the participants of the pool according to their stakes. The stake of party A under this pooling is denoted by $v_{\tilde{A}}(T)$, and it follows from Lemma 1 immediately that

$$\begin{aligned}\mathrm{Var}(v_{\tilde{A},r}(T)) &= \Big(\frac{v_A(0)}{v_P(0)}\Big)^2\big(v_P(0) - v_P(0)^2\big)\Big(1 - \frac{S(0)^2}{S(T)^2}\prod_{n=1}^{T}(2e^{\theta_n} - 1)\Big) \\ &= \frac{1 - v_P(0)}{v_P(0)}\frac{v_A(0)}{1 - v_A(0)}\mathrm{Var}(v_{A,r}(T)) \ . \end{aligned} \tag{9}$$

Thus party A's variance reduces by a factor of $(v_P(0)/v_A(0))((1 - v_A(0))/(1 - v_P(0)))$ by joining a stake pool of size $v_P(0)$. Note that the variance is monotonically decreasing under stake pooling. In practice, stake pools can organically form as long as this gain in equitability exceeds the cost of pool formation. Applying the definition 1 to a single party A, an ε-equitable party A will achieve $\varepsilon\frac{v_A(0)(1-v_P(0))}{v_P(0)(1-v_A(0))}$-equitability by forming a stake pool. Further, geometric rewards are still the most equitable reward function in the presence of stake pools. This follows from the fact that the effect of pooling is isolated from the effect of the choice of the reward function in Eq. (9),

Practical parameter selection. The equitability of a system is determined by four factors: the number of block proposals T, choice of reward function r, initial stake of a party $v_A(0)$, and the total reward R. We saw that geometric rewards optimize equitability; in this section, we study its dependence on T, $S(0)$, and R. Recall that without loss of generality, we normalized the initial stake $S(0)$ to be one. For general choices of $S(0)$, the total reward R should be rescaled by $1/S(0)$. The evolution of the fractional stakes is exactly the same for one system with $S(0) = 2$ and $R = 200$ and another with $S(0) = 1$ and $R = 100$. We assume here that the system designer can choose the total reward R, either

by setting the initial stake size $S(0)$ and/or the total reward during T. We study how equitability trades off with the total reward R for different choices of the reward function.

Geometric rewards. For $r_g(n)$, we have $e^{\theta_n} = (1+R)^{1/T}$. It follows from Lemma 1 that

$$\frac{\mathrm{Var}(v_{A,r_g}(T))}{v_A(0)-v_A(0)^2} \;=\; 1 - \frac{(2(1+R)^{1/T}-1)^T}{(1+R)^2}\;, \tag{10}$$

When R is fixed and we increase T, we can distribute small amounts of rewards across T and achieve vanishing variance. On the other hand, if R increases much faster than T, then we are giving out increasing amounts of rewards per time slot and the uncertainty grows. This follows from the above variance formula, which we make precise in the following.

Remark 3. For a closed PoS system with a total reward $R(T)$ chosen as a function of T and a geometric reward function $r_g(n) = (1+R(T))^{n/T} - (1+R(T))^{(n-1)/T}$, it is sufficient and necessary to set

$$R(T) \;=\; \left(\left(\frac{1}{1-\sqrt{\frac{\log(1/(1-\varepsilon))}{T}}} \right)^T - 1 \right) \big(1+o(1)\big)\;, \tag{11}$$

in order to ensure ε-equitability asymptotically, i.e. $\lim_{T\to\infty} \frac{Var(v_{A,r_g}(T))}{v_A(0)(1-v_A(0))} = \varepsilon$.

Remark 3 follows from substituting the choice of $R(T)$ in the variance in Eq. (10), which gives

$$\begin{aligned}\lim_{T\to\infty} \frac{\mathrm{Var}(v_{A,r_g}(T))}{v_A(0)-v_A(0)^2} &= \lim_{T\to\infty} 1 - \Big(1 - \frac{\log(1/(1-\varepsilon))}{T}\Big)^T (1+o(1)).\\ &= \varepsilon\;,\end{aligned} \tag{12}$$

The limiting variance is monotonically non-decreasing in R and non-increasing in T, as expected from our intuition. For example, if R is fixed, one can have the initial stake $S(0)$ as small as $\exp(-\sqrt{T}/(\log T))$ and still achieve a vanishing variance. As the geometric reward function achieves the smallest variance (Theorem 1), the above $R(T)$ is the largest reward that can be dispensed while achieving a desired normalized variance of ε in time T (with initial stake of one). This scales as $R(T) \simeq (1+1/\sqrt{T})^T \simeq e^{\sqrt{T}}$. We need more initial stake or less total reward, if we choose to use other reward functions.

Constant rewards. In comparison, consider the constant reward function of Eq. (2). As $e^{\theta_n} = (1+nR/T)/(1+(n-1)R/T)$, it follows from Lemma 1 that

$$\begin{aligned}\frac{\mathrm{Var}(v_{A,r_c}(T))}{v_A(0)-v_A(0)^2} &= 1 - \frac{1+R+\frac{R}{T}}{1+R+\frac{R}{T}+\frac{R^2}{T}}\\ &= \frac{R^2}{(T+R)(1+R)}\;.\end{aligned} \tag{13}$$

Again, this is monotonically non-decreasing in R and non-increasing in T, as expected. The following condition immediately follows from Eq. (13).

Remark 4. For a closed PoS system with a total reward $R(T)$ chosen as a function of T and a constant reward function $r_c(n) = R(T)/T$, it is sufficient and necessary to set

$$R(T) \;=\; \frac{\varepsilon\, T}{1-\varepsilon}\,(1+o(1))\;, \tag{14}$$

in order to ensure ε-equitability asymptotically as T grows.

By choosing a constant reward function, the cost we pay is in the size of the total reward, which can now only increase as $O(T)$. Compared to $R(T) \simeq e^{\sqrt{T}}$ of the geometric reward, there is a significant gap. Similarly, in terms of how small initial stake can be with fixed total reward R, constant reward requires at least $S(0) \simeq R/T$. This trend gets even more extreme for a decreasing reward function, which we describe in Appendix A.2.

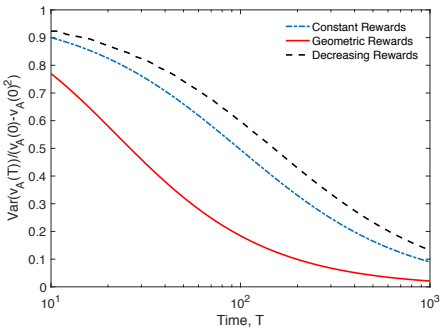


Fig. 4: Normalized variance after dispensing $R = 10$ tokens over T blocks, under different reward schemes.

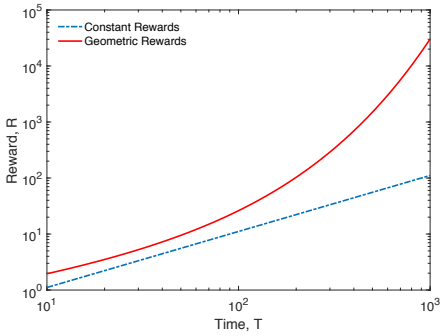


Fig. 5: Amount of reward that can be dispensed over T blocks while guaranteeing a normalized variance of at most $\varepsilon = 0.1$.

Comparison of Rewards. For $S(0) = 1$ and $R = 10$, Figure 4 illustrates the normalized variance of the three reward functions as a function of T, the total number of blocks. As expected, variance decays with T and geometric rewards exhibit the lowest normalized variance. Similarly, for a fixed desired (normalized) variance level of $\varepsilon = 0.1$, Figure 5 shows how the total reward grows as a function of time T. Notice that under constant rewards, the reward allocation grows linearly in T, whereas geometric rewards grow subexponentially while still satisfying the same equitability constraint. These observations add nuance to the ongoing conversation about how to initialize PoS cryptocurrencies. A recent lawsuit against Ripple highlighted that the large initial stake pool could

put disproportionate power in the hands of the system designers [29]. While Ripple itself is not PoS, our results suggest that in standard PoS systems, a large initial stake pool can actually help to ensure equitability.

4 Strategic Behavior

In practice, proposers can behave strategically to maximize their rewards (e.g., *selfish mining* [10,27,22]). In selfish mining, miners who discover blocks do not immediately publish them, but build a private *side chain* of blocks. By eventually releasing a side chain that is longer than the main chain, the adversary can invalidate honest blocks. This gives the adversary a greater fraction of main chain blocks and wastes honest parties' effort. In this section, we show that such strategic attacks are exacerbated by the compounding effects of PoS, and geometric rewards do not provide adequate protection.

Modeling the space of strategic behaviors in PoS requires more nuance than the corresponding problem in PoW [27]. We include a full model in Appendix B.1, which includes all the notation required to prove the theorems in this section. Due to space limitations, we summarize the model here. We consider two parties: A, which is adversarial, and H, which is honest. At any time, both parties can see the main chain, which is built upon by the honest party. We denote the length of this chain at time t by ℓ_t. In parallel, A can maintain as many private side chains as it wants, as long as the sequence of block proposers in each side chain respects the global leader election sequence. Since each block is associated with a time slot, A must have been the elected leader for each block in a side chain. I.e., if a side chain block is associated with time slot n, then $W(n) = A$.

At each time slot, the adversary has three options: (1) It can *wait*, or continue to build upon its side chains without releasing them. (2) It can *match* the main chain by releasing enough blocks from a single side chain to equal the height of the main chain, ℓ_t. After a match, there will be two publicly-visible chains of length ℓ_t in the system; we assume the honest party adopts the adversarial side chain with probability γ, a parameter that captures the adversary's connectivity. (3) The adversary can *override* the main chain by releasing a side chain up of length $\ell_t + 1$. If the side chain is longer than $\ell_t + 1$, it only releases the first $\ell_t + 1$ blocks. Since the released side chain is longer than the main chain, it is always adopted by the honest party. Given this action space, the adversary's goal is to maximize the fraction of main chain blocks that belong to the adversary.

4.1 Strategic selfish mining

We show that adversarial gains from strategic behavior are exacerbated by compounding. In practice, the adversary needs a strategy that balances the gains of keeping a long side chain to potentially overtake a long main chain, with the loss in intermediate leader elections due to withheld rewards. We propose a family of schemes called *Match-Override-k* (MO-k). Under MO-k, the adversary only keeps side chains whose tip is at most k blocks ahead of the main chain. The

strategy is as follows: Every time a new honest block is generated, it is appended to the main chain. Next, if there is a side chain that (a) is longer than ℓ_t, and (b) does not already include the entire honest chain, the adversary *matches* the main chain. Now there are two chains of equal length in the system; with probability γ, the newly-released side chain becomes the new main chain. Otherwise, the previous honest main chain continues to be the main chain, and the failed side chain is discarded. If there is no such side chain to match, then the adversary *waits*. Any side chains shorter than ℓ_t are discarded.

Every time a new adversarial block is generated, the adversary appends it to every side chain she is managing currently. She also starts a new side chain branching from the tip of the main chain, if there is not a side chain there already. The adversary now checks every side chain. If there is a side chain that branches at the tip of the main chain and is at least k blocks ahead of the main chain, the adversary *overrides* with this side chain, thereby incrementing the main chain length by one. Otherwise, the main chain remains as is, and the adversary *waits*.

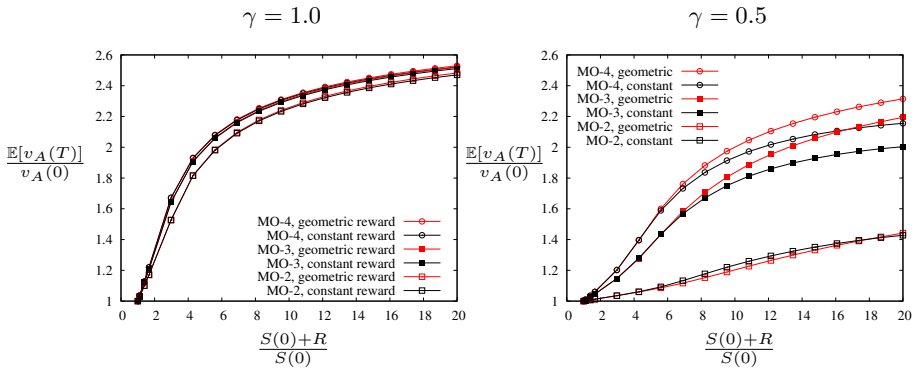


Fig. 6: Average fractional stake of an adversary can increase significantly as the total reward R increases. We fix initial fraction $v_A(0) = 1/3$, $S(0) = 1$, and $T = 10,000$ time steps, and show for two values of network connectivity of the adversary $\gamma \in \{0.5, 1.0\}$ and varying total reward R.

Figure 6 simulates how much the adversary can gain in average fractional stake by using MO-k strategies. As the total reward R increases, the relative fractional stake approaches 3, which is the maximum achievable value, since the expected fractional stake is normalized by $v_A(0) = 1/3$. The simulations were run for $T = 10,000$ time steps, with $S(0) = 1$. When the adversary is well-connected, i.e., $\gamma = 1.0$, such attacks are effective even with short side chains, such as $k = 3$ or 4. Further, there is no distinguishable difference in the reward function used. On the other hand, when the adversary has 0.5 probability of matching honest chains, $\gamma = 0.5$, it is more effective to keep longer side chains. Figure 6 demonstrates dramatic gains in fractional stake due to strategic behavior. A natural question is how large these gains can be. The following theorem gives an

upper bound on stake amplification due to strategic behavior. Given the time-varying nature of the underlying random process and the optimization over a large space of strategic actions, the proof is mathematically sophisticated. This proof, discussed in Appendix C.10 and B.2, involves stochastic dominance results of time-varying Pólya urn processes, and may be of independent interest.

Theorem 3. *Let $v_A(t)$ denote the fractional stake of the adversary under selfish mining (mathematically defined in Appendix B.1), when the total initial stake is $S(0)$, initial fractional stake of the adversary is $v_A(0)$, and the total reward dispensed over time T is $R = cT$. If $R \leq S(0)(1 - v_A(0))$, then*

$$\mathbb{E}[v_A(T)] \;=\; (1+\eta)\, v_A(0)\;, \tag{15}$$

where $\eta \triangleq R/(S(0)+c)$. (Proof in Appendix C.10)

We find empirically in Appendix B.2 that this upper bound is tight when $\frac{S(0)+R}{S(0)}$ is small. Under the assumption that R is less than the stake of the honest party, the gain of strategic behavior over honest behavior is bounded by $\mathbb{E}[v_A(T)] - \mathbb{E}[v_A(0)] \leq \eta\, v_A(0)$, since under honest behavior the mean fractional stake is $v_A(0)$ for all t. This implies that having a small initial stake $S(0)$ relative to the total reward R makes the system vulnerable to strategic behavior. This justifies the common practice of starting a PoS system with large initial stake.

5 Conclusion

This work measures the concentration of wealth in PoS systems, showing that existing block reward functions (e.g., constant, decreasing rewards) have poor equitability. We introduce a maximally-equitable geometric reward function. The negative effects of compounding can be further mitigated by choosing the total block rewards for each epoch to be small compared to the initial stake pool size.

Several open questions remain. First, our results do not account for proposers add or removing stake during an epoch. Another challenge, discussed in Section A.3, is that geometric rewards may not be desirable in practice because of the sharp changes in block rewards between epochs. A natural solution is to impose smoothness constraints on the class of reward functions—an interesting direction for future work. Finally, although strategic players are not specific to PoS systems, we show that geometric rewards alone do not protect against them. Designing incentive-compatible consensus protocols is a major open question.

References

1. Bitcoin energy consumption index, 2018. https://digiconomist.net/BITCOIN-ENERGY-CONSUMPTION.
2. Controlled supply. bitcoinwiki, 2018. `https://en.bitcoin.it/wiki/Controlled_supply#cite_note-2`.

3. Mining. Ethereum Wiki, 2018. https://github.com/ethereum/wiki/wiki/Mining.
4. Bambrough, B. A bitcoin halvening is two years away – here's what'll happen to the bitcoin price. *Forbes* (May 2018).
5. Bentov, I., Pass, R., and Shi, E. Snow white: Provably secure proofs of stake. *IACR Cryptology ePrint Archive 2016* (2016), 919.
6. Brünjes, L., Kiayias, A., Koutsoupias, E., and Stouka, A.-P. Reward sharing schemes for stake pools. *arXiv preprint arXiv:1807.11218* (2018).
7. Buterin, V., and Griffith, V. Casper the friendly finality gadget. *arXiv preprint arXiv:1710.09437* (2017).
8. Duffield, E., and Diaz, D. Dash: A privacycentric cryptocurrency. *Self-published* (2015).
9. Earls, J. The missing explanation of proof of stake version 3, 2017. http://earlz.net/view/2017/07/27/1904/the-missing-explanation-of-proof-of-stake-version.
10. Eyal, I., and Sirer, E. G. Majority is not enough: Bitcoin mining is vulnerable. *Communications of the ACM 61*, 7 (2018), 95–102.
11. Gaži, P., Kiayias, A., and Russell, A. Stake-bleeding attacks on proof-of-stake blockchains. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)* (2018), IEEE, pp. 85–92.
12. Gilad, Y., Hemo, R., Micali, S., Vlachos, G., and Zeldovich, N. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles* (2017), ACM, pp. 51–68.
13. Hopwood, D., Bowe, S., Hornby, T., and Wilcox, N. Zcash protocol specification. Tech. rep., Technical report, 2016–1.10. Zerocoin Electric Coin Company, 2016.
14. Johnson, N. L., and Kotz, S. *Urn models and their application: an approach to modern discrete probability theory*, vol. 77. Wiley New York, 1977.
15. Kaiser, I. A decentralized private marketplace: Draft 0.1.
16. Kiayias, A., Russell, A., David, B., and Oliynykov, R. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Annual International Cryptology Conference* (2017), Springer, pp. 357–388.
17. Mahmoud, H. *Pólya urn models*. Chapman and Hall/CRC, 2008.
18. Micali, S. Algorand: the efficient and democratic ledger. *CoRR, abs/1607.01341* (2016).
19. Miller, A., Möser, M., Lee, K., and Narayanan, A. An empirical analysis of linkability in the monero blockchain. *arXiv preprint 1704* (2017).
20. moh_man. How does pos stake concept deal with rich becoming richer issue? Reddit, 2017. https://www.reddit.com/r/ethereum/comments/6x0xv8/how_does_pos_stake_concept_deal_with_rich/.
21. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system.
22. Nayak, K., Kumar, S., Miller, A., and Shi, E. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on* (2016), IEEE, pp. 305–320.
23. Pass, R., and Shi, E. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing* (2017), ACM, pp. 315–324.
24. Pemantle, R. A time-dependent version of pólya's urn. *Journal of Theoretical Probability 3*, 4 (1990), 627–637.
25. Rammeloo, G. The economics of the proof of stake consensus algorithm. Medium, 2017. https://medium.com/@gertrammeloo/the-economics-of-the-proof-of-stake-consensus-algorithm-e28adf63e9db.

26. Ryan, D., and Liang, C.-C. Hybrid casper ffg, 2017. https://github.com/ethereum/EIPs/blob/master/EIPS/eip-1011.md.
27. Sapirshtein, A., Sompolinsky, Y., and Zohar, A. Optimal selfish mining strategies in bitcoin. In *International Conference on Financial Cryptography and Data Security* (2016), Springer, pp. 515–532.
28. Schrijvers, O., Bonneau, J., Boneh, D., and Roughgarden, T. Incentive compatibility of bitcoin mining pool reward functions. In *International Conference on Financial Cryptography and Data Security* (2016), Springer, pp. 477–498.
29. Taylor-Copeland, J. *Coffey vs. Ripple class action complaint.* 2018.
30. Trustnodes.com. ?proof of work is the rich get richer squared? says vitalik buterin. Trustnodes, 2018. https://www.trustnodes.com/2018/07/10/proof-work-rich-get-richer-squared-says-vitalik-buterin.
31. Vukolić, M. The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *International Workshop on Open Problems in Network Security* (2015), Springer, pp. 112–125.
32. Wiki, E. Proof of stake faqs. https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQs.

Appendix

A Additional Discussions

A.1 Modeling Assumptions

Our model implicitly makes several assumptions. The first is that we assign a single leader (proposer) per time slot. Many cryptocurrencies have leader election protocols that allow more than one proposer to be chosen per time slot (e.g., Bitcoin, PoSv3, Snow White). If two leaders are elected at time n, for example, then each leader can append its block to one block at height $n-1$; here the *height* of a block is its index in the blockchain. However, in these systems, only one leader can win the block reward since only one fork of the blockchain ultimately gets adopted. Assuming the final winner is chosen uniformly at random from the set of selected leaders, the dynamics of our Markov process remain unperturbed.

Other cryptocurrencies (e.g., Qtum, Particl) choose the next proposer(s) as a function of the time slot *and* the preceding block. Again, this can lead to multiple proposers per time slot. This does not affect our results in the honest setting (for the same reason as above), but it does impact strategic behavior. In most blockchain systems, honest proposers always build on the head of the blockchain. However, in systems where the proposer's identity depends on the previous block, a strategic node can increase its chance of being a leader by appending to a block that is not at the head of the blockchain. If done repeatedly, the strategic player may eventually produce a chain that is longer than the honest chain (causing the honest nodes to switch over), which also contains mostly blocks belonging to the strategic player. This increases the player's reward, and is called a *grinding attack*. Such PoS systems are more vulnerable to strategic behavior than the system we analyze, where proposer election is a function of only the time slot. Despite this, we find that our model is drastically vulnerable to strategic behavior. Hence, the problem can only be worse in blockchains that use block contents to choose the next leader. We discuss the implications of this in Section A.3.

We have also assumed that users always re-invest their rewards into the proposer stake pool. We maintain that this is a reasonable assumption for two reasons: (1) In PoS systems where users explicitly deposit stake, existing implementations automatically deposit rewards back into the stake pool. For example, the reference implementation of Casper the Friendly Finality Gadget (a PoS finalization mechanism proposed for Ethereum) automatically re-allocates all rewards back into the deposited stake pool [26]. (2) In other PoS systems, the stake pool is simply the set of all stake in the system, and is not separate from the pool of tokens used for transactions [9]. Hence as soon as a proposer earns a reward, that reward is used to calculate the next proposer (modulo some maturity period); the user is not actively re-investing block rewards—it just happens naturally.

Finally, we have chosen not to explicitly model node unavailability, e.g. due to hardware or network failures; in our context, node unavailability means that

a selected proposer may forfeit its chance to propose, even though it was chosen. Assuming such node failures occur i.i.d. across draws from the proposer pool, such events do not alter our model dynamics. If a proposer is offline, the selection process is simply re-run; the slot in question is given to the next node, which is again chosen proportionally to the stake allocation in the proposer pool.

A.2 Decreasing reward function

Some cryptocurrencies use continuously *decreasing* reward functions. For instance, Monero dispenses block rewards as per

$$r_M(n) \;=\; \max\left(c_0\,,\,\frac{\lfloor (M - S(n))/c_1 \rfloor}{c_2}\right)\,, \tag{16}$$

for some constants c_0, c_1, c_2, and M. In practice, $c_0 = 0.6$, $c_1 = 2^{19}$, $c_2 = 10^{12}$ and $M = 2^{64}$ [19]. Monero itself is not a PoS cryptocurrency, but if this decreasing reward were applied to the PoS setting, it would have even higher variance than constant rewards. Consider a simpler choices of the constants such that

$$r_M(n) \;=\; \alpha(M - S(n-1) - S(0))\,, \tag{17}$$

for all $n \in [T]$, for a choice of $\alpha = 1/T$ and $M = R(T)/(1-(1-\alpha)^T)$. Recall that $S(n-1) - S(0) = \sum_{i=1}^{n-1} r_M(i)$. As we assume $S(0) = 1$, it follows after some calculations that $r_M(n) = \alpha(1-\alpha)^{n-1}M$ and $S(n) = (1-(1-\alpha)^n)M + 1$. It follows from Lemma 1 that

$$\frac{\mathrm{Var}(v_{A,r_M}(T))}{(v_A(0) - v_A(0)^2)} \;=\; 1 - \frac{S(0)^2}{S(T)^2}\prod_{n=1}^{T}\frac{1 + M(1-(1-\alpha)^{n-1}(1-2\alpha))}{1 + M(1-(1-\alpha)^{n-1})}\,. \tag{18}$$

A.3 Practical Considerations

There are three main issues that relate to actually building a chain-based PoS system with geometric rewards. The first is how to choose the relevant parameters T and R, which has been discussed at length in Section 3. The second is how to deal with changing stake fractions that arise due to user-initiated transactions, e.g., selling their stake. The third discusses how to handle strategic behavior by block proposers in practice.

Dynamic Proposer Stake One challenge in the analysis of PoS systems is the fact that stake can move rapidly between parties, e.g. if nodes choose to sell their stake. Computing the objective function in the optimization of equitability is tedious when accounting for the dynamic addition and removal of stake, and it is not clear that geometric rewards are robust to rapid stake transactions. However, in practice, PoS systems often restrict the timescale over which stake can be added or removed, precisely to add robustness. For example, Casper FFG

constrains users to keep their stake in a validation pool for at least 4 months in order to participate [7,26].

In our system, an analogous stability constraint would be to impose that stake ratios should not change during each time interval of T blocks. If this constraint is met, then geometric rewards can be recalculated at each block interval T to account for dynamically changing Theorem 2 implies that this strategy optimizes the overall equitability of the reward scheme, even if the stake transactions are not known *a priori*. Moreover, if we choose T on the order of days as suggested in Section 3, this constraint is relatively mild from a user's perspective. It is important to note that users need not explicitly deposit their funds into a common pot in order to enforce the proposed stability constraints. This can be enforced implicitly by programming the selection mechanism to only consider stake that has been associated with the same public key for some minimum time interval. Such a strategy has been suggested in several proposed PoS systems, including Ouroboros [16], Algorand [18], Casper [7,26].

Control selfish mining Strategic behavior is a significant concern in PoW cryptocurrencies [10,22,27], and even more so in PoS systems. In Section 4 we demonstrate the efficacy of a strategic attack through which a rational user can artificially boost her proportion of the block rewards. In a sense, the results from Section 4 are negative. Choosing a small reward (with respect to the initial fraction) at each time step does not fully solve the problem, and there may be economic reasons to give out larger block rewards within a given time period.

Ultimately, we expect that this problem cannot be solved solely by changing the block reward function. Rather, it may be more effective to control the *effects* of strategic behavior than to identify a scheme under which strategic behavior is equivalent to honest behavior. For instance, the proposer selection protocol could choose only proposers whose fraction of proposed blocks in the last K blocks is commensurate with the proposer's stake (within some statistical error). Such a policy would detect nodes who produce more than their fair share of blocks, and limit their ability to propose more blocks.

B Strategic Behavior

B.1 Model

We restrict ourselves in this section to two parties: A, which is adversarial, and H, which is honest. Note that this is without loss of generality, as H represent the collective set of multiple honest parties as their behavior is independent of how many parties are involved in H. The adversarial party A can also represent the collective set of multiple adversarial parties, as having a single adversary A is the worst case when all adversaries are colluding. Throughout this section, we use the terms *adversarial* and *strategic* interchangeably.

Since A does not always publish its blocks on schedule, we distinguish the notion of a block slot (indexed by $n \in [T]$) and wall-clock time (indexed by

$t \in [T]$). It will still be the case that each *block slot* n has a single leader $W(n)$—in practice, this is determined by a distributed protocol—and a new block slot leader is elected at every tick of the wall clock (i.e., at a given time t, $W(n)$ is only defined for $n \leq t$). However, due to strategic behavior (i.e., the adversary can withhold its own blocks and override honest ones), it can happen that no block occupies slot n, even at time $t \geq n$; moreover, the occupancy of block slot n can change over time. Thus, unlike our previous setting, if we wait T time slots, the resulting chain may have fewer than T blocks. This is consistent with the adversarial model considered in PoS systems (e.g., Ouroboros [16]) that elect a single leader per block slot. Other PoS systems, like PoSv3 [9], choose an independent leader to succeed each block; such a PoS model can lead to even worse attacks, which we do not consider in this work.

The honest party and the adversary have two different views of the blockchain, illustrated in Figure 7. Both honest and adversarial parties see the *main chain* B_t; we let $B_t(n)$ denote the block (i.e., leader) of the nth slot, as perceived by the honest nodes at time t. If a block slot n does not have an associated block at time t (either because the nth block was withheld or overridden, or because $n > t$), we say that $B_t(n) = \emptyset$. Notice that due to adversarial manipulations, it is possible for $B_t(n) = \emptyset$ and $B_{t-1}(n) \neq \emptyset$, and vice versa.

In addition to the main chain, the adversary maintains arbitrarily many private *side chains*, $\tilde{B}_t^1, \ldots, \tilde{B}_t^s$, where s denotes the number of side chains. The blocks in each side chain must respect the global leader sequence $W(n)$. An adversary can choose at any time to publish a side chain, but we also assume that the adversary's attacks are *covert*: it never publishes a side chain that conclusively proves that it is keeping side chains. For example, if the main chain contains a block B created by the adversary for block slot n, the adversary will never publish a side chain containing block $\tilde{B} \neq B$, where $\tilde{B}$ is also associated with block slot n.

Each side chain $\tilde{B}_t^i$ with $i \in [s]$ overlaps with the honest chain in at least one block (the genesis block), and may diverge from the main chain after some $f_t^i \in \mathbb{N}_+$ (Figure 7). That is,

$$f_t^i := \max\{n \in \mathbb{N}_+ \,:\, B_t(n) = \tilde{B}_t^i(n)\}.$$

Different side chains can also share blocks; in reality, the union of side chains is a tree. However, for simplicity of notation, we consider each path from the genesis block to a leaf of this forest as a separate side chain, instead of considering side trees. We use ℓ_t and $\tilde{\ell}_t^i$ to denote the chain length of B_t and $\tilde{B}_t^i$, respectively, at time t:

$$\ell_t = |\{n \in [T] \,:\, B_t(n) \neq \emptyset\}| \quad , \quad \text{and} \quad \tilde{\ell}_t^i = |\{n \in [T] \,:\, \tilde{B}_t^i(n) \neq \emptyset\}|,$$

and we use the heights h_t and $\tilde{h}_t^i$ to denote the block indices of the ℓ_tth and $\tilde{\ell}_t^i$th blocks, respectively:

$$h_t = \max\{n \in [T] \,:\, B_t(n) \neq \emptyset\} \quad , \quad \text{and} \quad \tilde{h}_t^i = \max\{n \in [T] \,:\, \tilde{B}_t^i(n) \neq \emptyset\}.$$

If $f_t^i = h_t$, then the adversary is building its ith side chain from the tip of the current main chain.

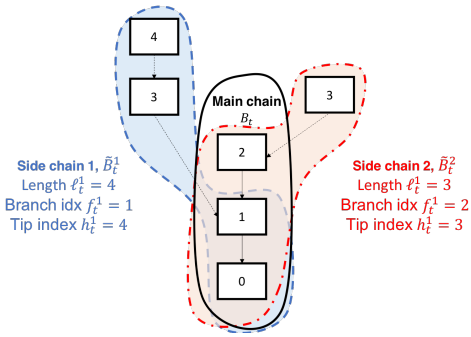


Fig. 7: In PoS, the adversary can keep arbitrarily many side chains at negligible cost, and release (part of) a side chain whenever it chooses.

State space. The state space for the system consists of three pieces of data: (1) The current time $t \in [T]$; (2) The main chain B_t; and (3) The set of all side chains $\{\tilde{B}_t^i\}_{i\in[s]}$. Notice in particular that the set of side chains grows exponentially in t. In practice, most systems prevent the main chain from being overtaken by a longer side chain that branches more than Δ blocks prior to h_t; this is called a *long-range attack*. Hence we can upper bound the size of the side chain set by imposing the condition that for all $i \in [s]$, $h_t - f_t^i \leq \Delta$. Regardless, the size of the state space is considerably larger than it is in prior work on selfish mining in PoW [27], where the computational cost of creating a block forces the adversary to keep a single side chain.

Objective. The adversary A's goal is to maximize its fraction of the total stake in the main chain by the end of the experiment,

$$v_A(t) \quad = \quad \frac{|\{n \in [T] \,:\, (W(n) = A) \wedge (B_T(n) \neq \emptyset)\}|}{\ell_T}.$$

This objective is closely related to the metric of prior work [27], except for the finite time duration.

Strategy space. The adversary has two primary mechanisms for achieving its objective: choosing where to append its blocks, and choosing when to release a side chain. If the honest party H is elected at time t, by the protocol, it always builds on the longest chain visible to it; since we assume small enough network latency, H appends to block $B_{t-1}(h_{t-1})$. However, if A is elected at time t, A can append to *any* known block in $B_{t-1} \cup \{\tilde{B}_{t-1}^i\}_{i\in[s]}$. The system must allow such a behavior for robustness reasons: even an honest proposer may not have received a block $B_{t-1}(h_{t-1})$ or its predecessors due to network latency.

The adversary can also choose when to release blocks. In our model, H always releases its block immediately when elected. However, an adversarial proposer elected at time t can choose to release its block at any time $\geq t$; it can also choose not to release a given block. Late block announcements are also tolerated because of network latency; it is impossible to distinguish between a node that releases their blocks late and a node whose blocks arrive late because of a poor network connection.

Notice that if A is elected at time t and chooses to withhold its block, the system advances to time $t + 1$ without appending A's block to the main chain.

This means that the next proposer $W(t+1)$ is selected based on the stake ratios at time $t-1$. So the adversary may have incurred a selfish mining gain from withholding its block, but it lost the opportunity to compound the t^{th} block reward. This tradeoff is the main difference between our analysis and prior work on selfish mining attacks in PoW systems.

Drawing from [10,27], at each time slot t, the adversary has three classes of actions available to it: match, override, and wait.
(1) The adversary **matches** by choosing a side chain $\tilde{B}_t^i$ and releasing the first h_t blocks. This means the released chain has the same height as the honest chain. In accordance with [10,27], we assume that after a match, the honest chain will choose to build on the adversarial chain with probability γ, which captures how connected the adversarial party is to the rest of the nodes.
(2) The adversary **overrides** by choosing a side chain $\tilde{B}_t^i$ and releasing the first $h = h_t + 1$ blocks. The released chain becomes the new honest chain.
(3) If the adversary chooses to **wait**, it does not publish anything, and continues to build on all of its side chains.

Unlike [10,27], we do not explicitly include an action wherein the adversary adopts the main chain. Because our model allows the adversary to keep an unbounded number of side chains, adopting the main chain is always a suboptimal strategy; it forces the adversary to throw away chains that could eventually overtake the main chain. The primary nuance in the adversary's strategy is choosing *when* to match or override (rather than waiting), and *which* side chain to choose. Identifying an optimal mining strategy through MDP solvers as in [27] is computationally intractable due to the substantially larger state space in this PoS problem.

B.2 Upper Bounds

We begin by showing upper bounds on the adversary's ability to amplify its expected fractional stake by using strategic behavior. We assume a constant reward function where a reward of c is dispensed to a proposer whose block is appended to the main chain. We begin with an upper bound on $v_A(t)$, the fraction of stake that can be achieved by the adversary.

Always-Match-1 (AM-1): To show our upper bound, we analyze a random process called always-match-1 (AM-1). AM-1 is an urn process with state

$$\mathbf{X}(t) = \begin{bmatrix} X_A(t) \\ X_H(t) \end{bmatrix}, \qquad \mathbf{X}(0) = \begin{bmatrix} S_A(0) \\ S_H(0) \end{bmatrix},$$

where as before, $S_A(t)$ denotes the number of tokens held by party A at time t. X_H and X_A can be thought of as the honest and adversarial stake, respectively; compared to S_A and S_H, they evolve under different dynamics, which are described below. We let $v_A(t) := \frac{X_A(t)}{X_A(t)+X_H(t)}$ denote the fraction of the urn occupied by X_A at time t. At each tick of a discrete clock, the state is updated

as follows:

$$\mathbf{X}(t+1) = \begin{cases} \begin{bmatrix} X_A(t) \\ X_H(t)+c \end{bmatrix} & \text{w.p.} \quad 1-v_A(t) \\ \\ \begin{bmatrix} X_A(t)+c \\ \max\{X_H(t)-c,0\} \end{bmatrix} & \text{w.p.} \quad v_A(t). \end{cases} \tag{19}$$

Intuitively, if the honest X_H wins a given draw, then the honest pool gains c unit of reward. If the adversarial pool X_A instead wins a given draw, it negates c honest units, and adds c units to the adversarial pool. The following theorem shows that AM-1 gives a *universal* upper bound on $v_A(t)$ under any arbitrary strategic behavior by the adversary. We refer to Appendix C.5 for a proof.

Theorem 4. *Under the constant reward function, for any adversarial strategy resulting in a stake fraction time series $v_A(t)$, the AM-1 random process $\tilde{v}_A(t)$ stochastically dominates $v_A(t)$, i.e. $\mathbb{P}(v_A(t) \le a) \ge \mathbb{P}(\tilde{v}_A(t) \le a)$ for all $a \in [0,1]$ and any $t \in \mathbb{Z}^+$.* (Proof in Appendix C.5)

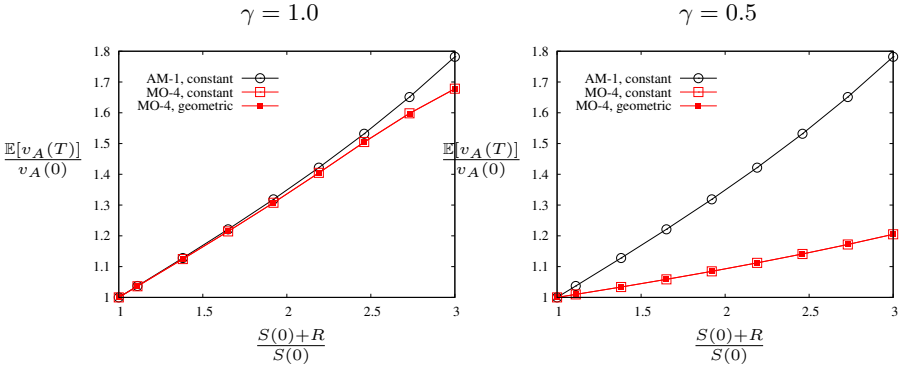


Fig. 8: AM-1 urn process provides an upper bound on the compounding effect of any adversarial strategy. We fix initial fraction $v_A(0) = 1/3$, $S(0) = 1$, and $T = 10,000$ time steps, and show for two values of network connectivity of the adversary $\gamma \in \{0.5, 1.0\}$ defined in the strategy space subsection of Section **??** and varying total reward R.

Figure 8 (left) shows that for small values of the total reward $R \le 2S(0)$ and when adversaries are well connected to the honest nodes ($\gamma = 1$), the AM-1 upper bound is quite close to an achievable strategy of MO-4. The right panel show that when the adversaries are less connected ($\gamma = 0.5$), then the strategic behavior takes over less stake. We analyze an upper bound (inspired by AM-1), which reveals that a PoS system is less vulnerable against strategic attacks when initial stake $S(0)$ is larger.

Analytical upper bound. We introduce and analyze a new random process called always-match-2 (AM-2), which is an upper bound on AM-1, but has the merit that the expected fractional stake is tractable in a closed form.

Always-Match-2 (AM-2): Similar to AM-1, AM-2 is an urn process with state

$$\mathbf{X}(t) = \begin{bmatrix} X_A(t) \\ X_H(t) \end{bmatrix}, \qquad \mathbf{X}(0) = \begin{bmatrix} S_A(0) \\ S_H(0) \end{bmatrix},$$

where as before, $S_A(t)$ denotes the number of tokens held by party A at time t. At each tick of a discrete clock, the state is updated as follows:

$$\mathbf{X}(t+1) = \begin{cases} \begin{bmatrix} X_A(t) \\ X_H(t) + c \end{bmatrix} & \text{w.p.} \quad 1 - v_A(t) \\ \\ \begin{bmatrix} X_A(t) + 2c \\ \max\{X_H(t) - c, 0\} \end{bmatrix} & \text{w.p.} \quad v_A(t). \end{cases} \tag{20}$$

The addition of $2c$ units of adversarial reward keeps the total change in urn size constant across time steps, which simplifies the analysis of this urn process. The following theorem shows that AM-2 gives an upper bound on the AM-1 process. We refer to Appendix C.6 for a proof.

Theorem 5. *Under the constant reward function, the AM-2 process stochastically dominates the AM-1 process.* (Proof in Appendix C.6)

We are interested in how much an adversary can gain by acting strategically. The above theorem provides a tool for characterizing an upper bound on any strategies, by analyzing AM-2. Recall Theorem 3 from Section 4, which makes this formal:

Theorem 3. *Let $v_A(t)$ denote the fractional stake of the adversary under selfish mining (mathematically defined in Appendix B.1), when the total initial stake is $S(0)$, initial fractional stake of the adversary is $v_A(0)$, and the total reward dispensed over time T is $R = cT$. If $R \leq S(0)(1 - v_A(0))$, then*

$$\mathbb{E}[v_A(T)] \;=\; (1+\eta)\, v_A(0)\,, \tag{15}$$

where $\eta \triangleq R/(S(0)+c)$. (Proof in Appendix C.10)

This analysis allows us to quantify the price of compounding under adversarial strategy. When there is no compounding effect, either under a PoW system or because the rewards are not automatically appended to the stake, an upper bound on adversarial strategy we consider in this paper has been analyzed in [27]. Translating the bound into the same notations as in Theorem 3, we get that when there is no compounding, an adversary's fractional stake is bounded by

$$v_A(T) \;\leq\; \Big(1 + \frac{v_A(0)\,\eta}{1 + (1 - v_A(0))\,\eta}\Big)\, v_A(0) + O\Big(\frac{\log T}{T}\Big)\,,$$

with high probability. For large enough T, with high probability. Compared to Eq. (15), when η is large, compounding allows the adversary's gain to grow linearly in η whereas the adversary's gain is a constant in η with no compounding. This shows that strategic parties can gain significantly over honest parties, under PoS systems with compounding effects.

C Proofs

C.1 Proof of Remark 1

We first prove the converse, $\mathrm{Var}(v_{A,r}(T)) \leq v_A(0)(1-v_A(0))$ for all T and r. This follows from the fact that $\mathbb{E}[v_{A,r}(T)] = v_A(0)$, and $v_{A,r}(T)$ is bounded below by zero and above by one. Maximum variance is achieved when all probability mass is concentrated on the boundary of zero and one.

We prove the achievability, by constructing a simple constant reward function, with total reward $R = T^2$ is increasing super-linearly in T. From the variance computation of a constant reward function in Eq. (13), it follows that $\lim_{T\to\infty} \mathrm{Var}(v_{A,r_c}(T)) = v_A(0)(1 - v_A(0))$.

C.2 Proof of Lemma 1

Let $e^{\theta_n} \triangleq S(n)/S(n-1)$ and $r(n) = S(n+1) - S(n)$, then

$$\begin{aligned}
\mathbb{E}[v_{A,r}(n+1)^2 | v_{A,r}(n)] &= v_{A,r}(n)\Big(\frac{S(n)v_{A,r}(n) + r(n)}{S(n+1)}\Big)^2 + (1 - v_{A,r}(n))\Big(\frac{S(n)\, v_{A,r}(n)}{S(n+1)}\Big)^2 \\
&= \frac{(S(n)^2 + 2r(n)S(n))v_{A,r}(n)^2 + r(n)^2 v_{A,r}(n)}{S(n+1)^2} \\
&= (2e^{-\theta_{n+1}} - e^{-2\theta_{n+1}})v_{A,r}(n)^2 + (e^{-\theta_{n+1}} - 1)^2 v_{A,r}(n) \,.
\end{aligned}$$

It follows that

$$\begin{aligned}
\mathbb{E}[v_{A,r}(T)^2] - \mathbb{E}[v_{A,r}(T)] &= (2e^{\theta_T} - e^{2\theta_T})\big(\mathbb{E}[v_{A,r}(T-1)^2] - \mathbb{E}[v_{A,r}(T-1)]\big) \\
&= \big(\mathbb{E}[v_{A,r}(0)^2] - \mathbb{E}[v_{A,r}(0)]\big) \prod_{n=1}^{T} (2e^{-\theta_n} - e^{-2\theta_n}) \,.
\end{aligned}$$

Hence,

$$\begin{aligned}
\mathrm{Var}(v_{A,r}(T)) &= \big(\mathbb{E}[v_{A,r}(0)] - \mathbb{E}[v_{A,r}(0)^2]\big)\Big(1 - \prod_{n=1}^{T} (2e^{-\theta_n} - e^{-2\theta_n})\Big) \\
&= \big(v_{A,r}(0) - v_{A,r}(0)^2\big)\Big(1 - \prod_{n=1}^{T} e^{-2\theta_n} \prod_{n=1}^{T} (2e^{\theta_n} - 1)\Big) \\
&= \big(v_{A,r}(0) - v_{A,r}(0)^2\big)\Big(1 - \frac{S(0)^2}{S(T)^2} \prod_{n=1}^{T} (2e^{\theta_n} - 1)\Big) \,.
\end{aligned}$$

C.3 Proof of Theorem 1

Lemma 1 and Remark 2 imply that in order to show joint optimality over all parties, it is sufficient to show that for an arbitrary party A,

$$\mathrm{Var}\big(v_{A,r_g}(T)\big) \;\leq\; \mathrm{Var}\big(v_{A,r}(T)\big)\;, \tag{21}$$

for all $r \in \mathbb{R}^T$ such that $\sum_{n=1}^{T} r(n) = R$ and $r(n) \geq 0$ for all $n \in [T]$. To this end, we prove that r_g is a unique optimal solution to the following optimization problem:

$$\begin{aligned} \text{minimize}_{r\in\mathbb{R}^T}\ & \mathrm{Var}(v_{A,r}(T)) \\ \text{s.t.}\ & \sum_{n\in[T]} r(n) = R \\ & r(n) \geq 0\ ,\ \forall n \in [T]. \end{aligned} \tag{22}$$

Using Lemma 1, we have an explicit expression for $\mathrm{Var}(v_{A,r}(T))$. After some affine transformation and taking the logarithmic function of the objective, we get an equivalent optimization of

$$\text{maximize}_{\theta\in\mathbb{R}^T}\ \sum_{n=1}^{T} \log(2e^{\theta_n} - 1) \tag{23}$$

$$\text{s.t.}\ \sum_{n\in[T]} \theta_n = \log(1+R)\ ,$$

$$\theta_n \geq 0, \forall n \in [T]. \tag{24}$$

This is a concave maximization on a (rescaled) simplex. Writing out the KKT conditions with KKT multipliers λ and $\{\lambda_n\}_{n=1}^{T}$, we get $\forall n \in [T]$:

$$\frac{2e^{\theta_n}}{2e^{\theta_n}-1} - \lambda_n - \lambda = 0 \tag{25}$$

$$\lambda_n \geq 0 \tag{26}$$

$$\theta_n \lambda_n = 0 \tag{27}$$

Among these solutions, we show that $\theta^* = ((\log(1+R))/T)\,\mathbb{1}$ is the unique optimal solution, where $\mathbb{1}$ is a vector of all ones. Consider a solution of the KKT conditions that is not θ^*. Then, we can strictly improve the objective by the following operation. Let $i, j \in [T]$ denote two coordinates such that $\theta_i = 0$ and $\theta_j \neq 0$. Then, we can create $\tilde{\theta}$ by mixing θ_i and θ_j, such that $\tilde{\theta}_n = \theta_n$ for all $n \neq i, j$ and $\tilde{\theta}_i = \tilde{\theta}_j = (1/2)\theta_j$. We claim that $\tilde{\theta}$ achieves a smaller objective function as $\log(2e^{\theta_j} - 1) < 2\log(2e^{\theta_j/2} - 1)$. This follows from Jensen's inequality and strict concavity of the objective function. Hence, θ^* is the only fixed point of the KKT conditions that cannot be improved upon.

In terms of the reward function, this translates into $S(n)/S(n-1) = (1+R)^{1/T}$ and $r(n) = (1+R)^{n/T} - (1+R)^{(n-1)/T}$.

C.4 Proof of Theorem 2

By the same logic as the proof of Theorem 1, the optimization problem of interest can be written as

$$\begin{aligned} \text{maximize}_{\theta\in\mathbb{R}^{T_k}} \quad & \sum_{n=1}^{T_k} \log(2e^{\theta_n}-1) \\ \text{s.t.} \quad & \sum_{n=T_{i-1}+1}^{T_i} \theta_n = \log\left(\frac{1+\tilde{R}_i}{1+\tilde{R}_{i-1}}\right), \forall i \in [k] \\ & \theta_n \geq 0, \forall n \in [T_k] \,, \end{aligned} \tag{28}$$

where recall that $\theta_n = \frac{S(n)}{S(n-1)}$, and we define $T_0 := 0$. Notice that this optimization problem is separable over the variables in different time intervals, so we can separately solve k optimization problems, each of the form

$$\begin{aligned} \text{maximize}_{\theta\in\mathbb{R}^{T_i-T_{i-1}}} \quad & \sum_{n=T_{i-1}+1}^{T_i} \log(2e^{\theta_n}-1) \\ \text{s.t.} \quad & \sum_{n=T_{i-1}+1}^{T_i} \theta_n = \log\left(\frac{1+\tilde{R}_i}{1+\tilde{R}_{i-1}}\right) \\ & \theta_n \geq 0, \forall n \in [T_{i-1}+1, T_i] \end{aligned}$$

for each $i \in [k]$. Using the same KKT conditions as in Theorem 1, we get that $\theta_n^* = \frac{1}{T_i - T_{i-1}} \log(\frac{1+\tilde{R}_i}{1+\tilde{R}_{i-1}})$, which in turn implies that for $n \in [T_{i-1}+1, T_i]$,

$$S(n) = (1+\tilde{R}_{i-1}) \left(\frac{1+\tilde{R}_i}{1+\tilde{R}_{i-1}}\right)^{(n-T_{i-1})/(T_i-T_{i-1})}$$

and

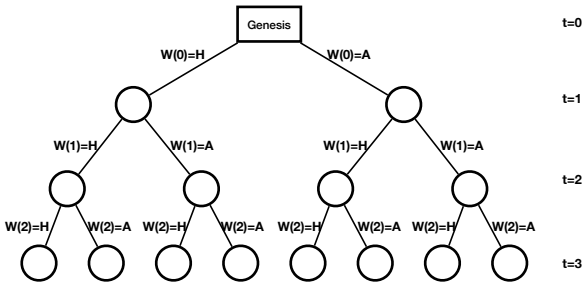
$$r(n) = (1+\tilde{R}_{i-1}) \left(\left(\frac{1+\tilde{R}_i}{1+\tilde{R}_{i-1}}\right)^{\frac{n-T_{i-1}}{T_i-T_{i-1}}} - \left(\frac{1+\tilde{R}_i}{1+\tilde{R}_{i-1}}\right)^{\frac{n-1-T_{i-1}}{T_i-T_{i-1}}} \right).$$

C.5 Proof of Theorem 4

We first represent the standard Pòlya's urn process using a binary tree of the state evolution according to who won at each time step. Recall that the winner is assigned according to

$$W(t) = \begin{cases} H & \text{with probability } 1 - v_A(t) \,, \\ A & \text{with probability } v_A(t) \,, \end{cases} \,,$$

which determines who gets the reward. We need the following notations for the proof. We denote the outcome of the random winner drawings as $W(0:2) =$



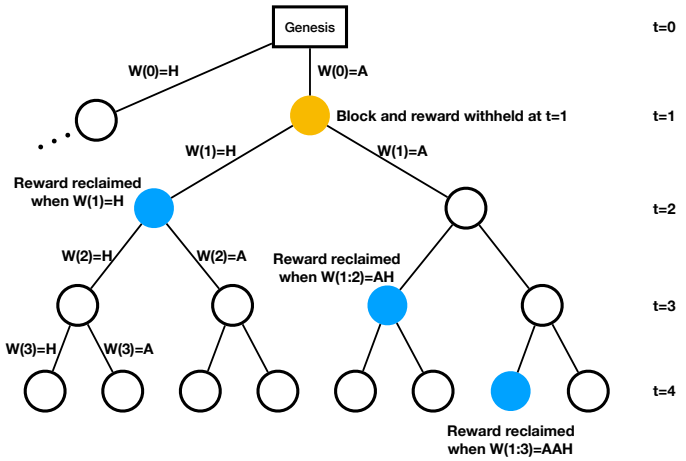
AAH if the winner at time 0 is the adversary (meaning that the adversary was elected the leader and an adversarial block is generated), at time $t = 1$ is the adversary, and at $t = 2$ is the honest party. Under this event, we denote the factional stake of the adversary by $\tilde{x}(AAH)$, and the total stake by $S(AAH)$. We use the notion of the standard Pòlya's urn process to denote the process with constant reward c at each time.

A strategic behavior consists of union of the following actions. When elected a leader at a certain time t, say $t = 1$ in the figure below, the adversary may decide to withhold its currently generated block and also the reward. This withheld reward (and the block) is awarded when the adversary either matches or overrides (based on the instance of the future winner elections). If *matched*, the reclaimed block also takes away one of the honest blocks (and the corresponding reward). If *overridden*, the reclaimed block may or may not take away one of the honest blocks.

We represent the strategy of the adversary on a *single* withheld block (generated at time $t = 1$), using the following binary decision tree. We consider a binary decision tree of height T, as follows (e.g. $T = 4$). We can encode any strategy of the adversary on when to reclaim the withheld reward on the binary tree. We are hiding the part of the tree branching from $W(0) = H$ as it is not affected by the current adversary we are considering.

For example, the adversary withholds a block if he is elected a leader at time $t = 1$, which is encoded as the orange node. The adversary might choose to reclaim the reward (by publishing the side-chain that includes the withheld block of interest) if the next winner is an honest one ($W(1) = H$), if next two winners are adversarial and honest in that order ($W(1 : 2) = AH$), and if next three winners are adversarial, adversarial, and honest in that order ($W(1 : 3) = AAH$). These are encoded on a binary tree as shown above in blue nodes. Any binary tree where a single path from an orange node (block withheld) to a leaf only contains a single blue node (block reclaimed) is a valid strategy, as a withheld reward can only be claimed once. We do not explicitly encode wether a honest block is taken away when adversarial block is reclaimed, as it does not change the proof as we will show.

In the above example, the orange node at the event $W(0) = A$ encodes the strategy that a unit c reward is withheld if the adversary wins at time $T = 0$.



Hence, the resulting stake at that node is $\tilde{x}(A) = \tilde{X}(0)$ as no reward is claimed, and $S(A) = S(0)$. the blue node at the event AH denotes that the adversary reclaims the withheld reward if the next winner is an honest party. Under this event of AH, the resulting stake at that node AH is $\tilde{x}(AH) = \tilde{X}(A) + (c/S(A))$ as c unit reward is given to the adversary and c unit reward is taken from the honest party, and and the total stake $S(AH) = S(A)$ remains unchanged.

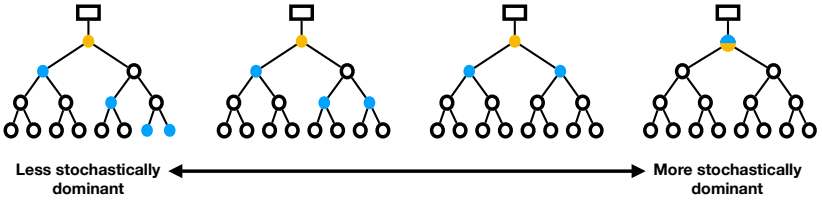
The next lemma provides a set of operations on the colored binary tree we can perform, in order to turn it into a more stochastically dominant process. We give a proof in Appendix C.7.

Lemma 2. *Given a representation of a random process with an adversarial strategy as a colored binary tree, the following operation results in a new random process that stochastically dominates the old one:*

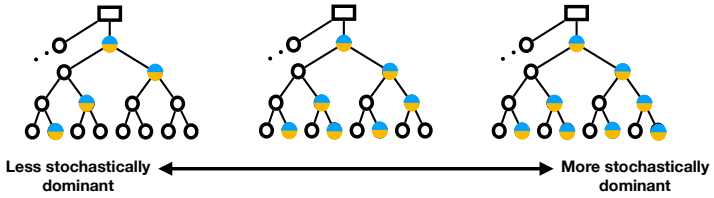
A.1. convert a white leaf node to a blue leaf node;
A.2. convert two blue leaf nodes who are siblings into one blue parent node with two white offsprings;
A.3. convert two blue sibling nodes into one blue parent node with two white offsprings; and
A.4. convert two blue offsprings of an orange node into one parent node that is blue and orange with two white offsprings.

Note that a blue and orange node denotes the combination of a orange node and a blue node, where one unit reward given to the adversary and one unit reward taken away from the honest party. Applying the above operations in the order of $A.1$, $A.2$, $A.3$, and $A.4$, we have the following random urn process that stochastically dominates any adversarial behavior with a *single* reward withheld.

From the preservation of stochastic dominance by the standard Pòlya's urn process (as shown in Lemma 4), we can now convert a white node in which an



adversary is a winner into a blue and orange node, from top to bottom. The resulting process is exactly the AM-1 process, finishing the proof of stochastic dominance when only a *single* reward is withheld at time $t = 1$.



When a single reward is withheld at time $t > 1$, we need a preservation of stochastic dominance for AM-1. The following lemma justifies conversion of a white node into a blue and orange node when the descendent nodes follow AM-1. We provide a proof in Appendix C.9.

Lemma 3 (Preservation of stochastic dominance of AM-1 process). *Consider two AM-1 processes with the same initial total stake $S(0)$. One process has a random initial fractional stake $v_A(0)$, which is stochastically dominated by that of the other process $v'_A(0)$. Then, the final fractional stake preserves the dominance, i.e. $v_A(T)$ is stochastically dominated by $v'_A(T)$.*

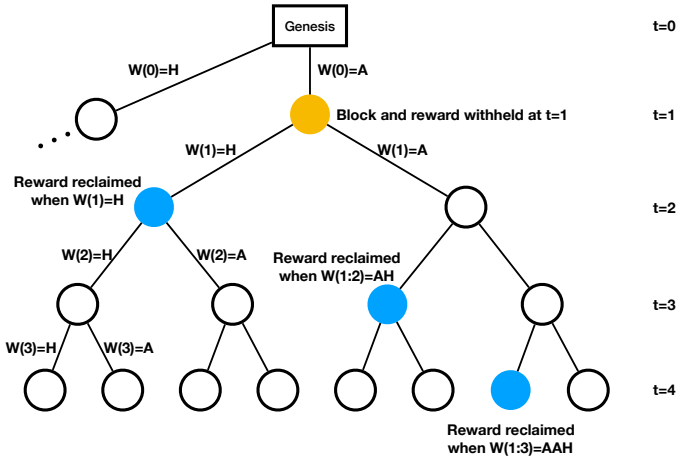
In general, a strategic behavior consists of multiple rewards withheld at multiple nodes in the binary tree. Each withheld reward will have some strategy for being reclaimed in the future. Lemmas 4 and 3 ensure that the above argument for converting such a strategy into AM-1 process still holds when multiple rewards are withheld. This finishes the proof of the claim that AM-1 stochastically dominates any adversarial strategy.

C.6 Proof of Theorem 5

The fact that AM-2 further stochastically dominates AM-1 follows immediately from Lemma 3 and converting a binary tree representation of AM-1 to that of AM-2 from top to bottom. We omit this part of the proof as it is straightforward.

C.7 Proof of Lemma 2

We use the following example of a adversarial strategy for illustration of the proof:



A.1. **Convert a white leaf node to a blue leaf node.** As this change only affects one sample path, only one instance is affected, say $T = 4$ and $W(0:T-1) = AAAA$. The probability of this outcome does not change, but only the fractional stake corresponding to this outcome changes from $\tilde{x}(AAAA)$ to $\tilde{x}'(AAAA) = \tilde{x}(AAAA) + (c/S(AAAA))$. As $c/S(AAAA) > 0$, the process after the changing the white leaf node into a blue one is strictly stochastically dominant. After this conversion, the node $AAAA$ is now blue, in which case we apply the next operation.

A.2. **Convert two blue leaf nodes who are siblings into one blue parent node with two white offsprings.** This change affects two sample paths, say $T = 4$, $W(0:T-1) = AAAA$ and $W(0:T-1) = AAAH$. The fractional stakes do not change, but the probabilities do. $\mathbb{P}(AAAH|AAA) = 1 - \tilde{x}(AAA)$ changes to $\mathbb{P}'(AAAH|AAA) = 1 - \tilde{x}(AAA) - (c/S(AAA))$. $\mathbb{P}(AAAA|AAA) = \tilde{x}(AAA)$ changes to $\mathbb{P}'(AAAA|AAA) = \tilde{x}(AAA) + (c/S(AAA))$. Given that $\tilde{x}(AAAH) \leq \tilde{x}(AAAA)$ and $c/S(AAA) > 0$, the resulting process is stochastically dominant. After this conversion, the node AAA is now blue, in which case we apply the next operation.

A.3. **Convert two blue sibling nodes into one blue parent node with two white offsprings.** This change affects many sample paths, all descendants of a single node (parent of the two blue nodes of interest), say node AA, and node AAA and node AAH are blue nodes. We know from rule *A*.2. that conditioned on the event $W(0:1) = AA$, $\tilde{x}'(2)$ stochastically dominates $\tilde{x}(2)$. The rest of

the process follows the standard Pòlya's urn process. Hence, the following lemma implies the desired claim. We provide a proof of this lemma in Appendix C.8.

Lemma 4 (Preservation of stochastic dominance of the standard Pòlya's urn process). *Consider two standard Pòlya's urn processes with the same initial total stake $S(0)$ and the same constant reward c. One process has a random initial fractional stake $v_A(0)$, which is stochastically dominated by that of the other process $v'_A(0)$. Then, the final fractional stake preserves the dominance, i.e. $v_A(T)$ is stochastically dominated by $v'_A(T)$.*

After this conversion, the node AA is now blue, in which case we apply the next operation.

A.4. **Convert two blue offsprings of an orange node into one parent node that is blue and orange with two white offsprings.** Note that the stakes at time $t = 2$ remain unchanged by the conversion, i.e. $\tilde{x}(AA) = \tilde{x}'(AA)$, and $\tilde{x}(AH) = \tilde{x}'(AH)$. Only the corresponding probability of events change. Node A is orange with, say $\tilde{x}(A) = v_A(t = 0)$, as the reward is withheld at time $t = 1$. Hence, $\mathbb{P}(AA|A) = v_A(0)$, whereas $\mathbb{P}'(AA|A) = v_A(0) + (c/S(A))$, after the conversion. It follows from the fact that $c/S(A) > 0$ and $\tilde{X}(AA) > \tilde{X}(AH)$ (and also Lemma 4) that the conversion results in a process that is stochastically dominant.

C.8 Proof of Lemma 4

We prove it by a recursion. Consider the following representation of the standard Pòlya's urn process. $v_A(t)$ denotes the fractional stake of party A at time t, that starts as $v_A(0)$. $S(t) = S(0) + c\,t$ denotes the total stake. First, we claim that if $v_A(0) < v'_A(0)$ deterministically, then $v_A(1) \overset{\mathcal{D}}{\leq} v'_A(1)$. This follows from the fact that

$$v_A(1) = \begin{cases} \frac{v_A(0)S(0)+c}{S(0)+c} & \text{w. p. } v_A(0)\ , \\ \frac{v_A(0)S(0)}{S(0)+c} & \text{w. p. } 1 - v_A(0)\ , \end{cases} \quad , \quad \text{and} \quad v'_A(1) = \begin{cases} \frac{v'_A(0)S(0)+c}{S(0)+c} & \text{w. p. } v'_A(0)\ , \\ \frac{v'_A(0)S(0)}{S(0)+c} & \text{w. p. } 1 - v'_A(0)\ , \end{cases} \tag{29}$$

For these two valued discrete random variable, not only are those two values both larger for the latter process, but also the probability mass for the larger of those two values are also higher for the latter process. Hence, $v_A(1) \overset{\mathcal{D}}{\leq} v'_A(1)$. Note that assuming stochastic dominance of $v_A(0) \overset{\mathcal{D}}{\leq} v'_A(0)$ leads to the same conclusion. Hence, we can recursively apply the above result to prove the desired lemma.

C.9 Proof of Lemma 3

Consider the following representation of the AM-1 process. Let $v_A(t)$ denote the fractional stake of party A at time t, that starts as $v_A(0)$. Let $S(t)$ denote the

total stake at time t, that starts with $S(0)$. First, we claim that if $v_A(0) < v'_A(0)$ deterministically, then $v_A(1) \overset{\mathcal{D}}{\leq} v'_A(1)$. This follows from the fact that

$$v_A(1) = \begin{cases} \frac{v_A(0)S(0)+c}{S(0)} & \text{w. p. } v_A(0)\ , \\ \frac{v_A(0)S(0)}{S(0)+c} & \text{w. p. } 1-v_A(0)\ , \end{cases}, \quad \text{and} \quad v'_A(1) = \begin{cases} \frac{v'_A(0)S(0)+c}{S(0)} & \text{w. p. } v'_A(0)\ , \\ \frac{v'_A(0)S(0)}{S(0)+c} & \text{w. p. } 1-v'_A(0)\ , \end{cases} \tag{30}$$

The rest of the proof follows similarly as in the proof of Lemma 4 in Section C.8.

C.10 Proof of Theorem 3

This proof is best understood by reading Appendix B.2, which contains a number of related lemmas. As $R \leq S(0)(1 - v_A(0))$, $x_H(0)$ will always be non-negative. Then,

$$\begin{aligned} \mathbb{E}[v_A(t+1)|v_A(t)] &= v_A(t)\frac{v_A(t)S(t)+2c}{S(t+1)} + (1-v_A(t))\frac{v_A(t)S(t)}{S(t+1)} \\ &= v_A(t)\frac{S(t+2)}{S(t+1)}\ . \end{aligned}$$

It follows that

$$\begin{aligned} \mathbb{E}[v_A(T)] &= \mathbb{E}[v_A(T-1)]\frac{S(T+1)}{S(T)} \\ &= v_A(0)\frac{S(T+1)}{S(1)} \\ &= v_A(0)\Big(1+\frac{cT}{S(0)+c}\Big)\ . \end{aligned}$$