The Last Survivor of PoS Pools: Staker's Dilemma

Yuming Huang
National University of Singapore
huangyuming@u.nus.edu

Jing Tang*
The Hong Kong Uni. of Sci. and Tech.
jingtang@ust.hk

Qianhao Cong
National University of Singapore
cong_qianhao@u.nus.edu

Richard T. B. Ma
National University of Singapore
tbma@comp.nus.edu.sg

Lei Chen
The Hong Kong Uni. of Sci. and Tech.
leichen@cse.ust.hk

Yeow Meng Chee
National University of Singapore
ymchee@nus.edu.sg

ABSTRACT

In blockchains using the Proof-of-Work (PoW) consensus mechanism, a mining pool is a joint group of miners who combine their computational resources and share the generated revenue. Similarly, when the Proof-of-Stake (PoS) consensus mechanism is adopted, the staking pool imitates the design of the mining pool by aggregating the stakes. However, in PoW blockchains, the pooling approach has been criticized to be vulnerable to the *block withholding (BWH)* attack. BWH attackers may steal the dividends from victims by pretending to work but making invalid contributions to the victim pools. It is well known that BWH attackers against PoW face the *miner's dilemma*. To our knowledge, despite the popularity of PoS, we are the first to study the pool BWH attack against PoS. Surprisingly, we find that, for a network only consisting of one attacker pool and one victim pool, the attacker will eventually manipulate the network while the victim will vanish by losing the stake ratio gradually. Moreover, in a more realistic scenario with multiple BWH attacker pools and one solo staker who does not join any pools, we show that only one lucky attacker and the solo staker will survive, whereas all the other pools will vanish gradually, revealing the *staker's dilemma*. These findings indicate that compared with PoW, the BWH attack against PoS has a much severer impact, e.g., producing a giant pool that destroys the decentralization of blockchains. Our analysis is confirmed with extensive experiments on real blockchain systems and numerical simulations.

PVLDB Reference Format:
Yuming Huang, Jing Tang, Qianhao Cong, Richard T. B. Ma, Lei Chen, and Yeow Meng Chee. The Last Survivor of PoS Pools: Staker's Dilemma. PVLDB, 16(1): XXX-XXX, 2023.
doi:XX.XX/XXX.XX

1 INTRODUCTION

1.1 Background

Blockchain is a decentralized database that contains all historical and current transaction data inside its system. Every blockchain node stores a copy of the blockchain, in which transactions are recorded inside sequentially linked blocks. New transactions are confirmed when they are packed into a newly generated block. The block generating process is referred to as *mining* and a network node who generates blocks is called a *miner*. To successfully mine a block, miners must satisfy specific conditions, e.g., *Proof-of-Work (PoW)* [36, 50] or *Proof-of-Stake (PoS)* [66]. Essentially, a valid PoW requires a miner to solve a cryptographic puzzle, e.g., the hash of PoW contains more than a certain number of leading zeros. Miners spend computational power to solve this puzzle and the miner who successfully proposes a valid proof will be rewarded some cryptocurrencies as *block reward*. On the other hand, PoS blockchains inherit most components of PoW blockchains but change the proposer selection algorithm. Instead of competing for computational power, PoS *stakers* propose blocks relying on the number of cryptocurrencies they possess. The staker who possesses more stakes will be more likely to generate a block and receive the block reward. Currently, PoW and PoS are the most popular consensus algorithms which are adopted by more than 80% public blockchain systems [33, 65, 74].

With more participants joining in, the difficulty of generating a block increases rapidly to keep the average time between blocks steady (e.g., 10 minutes per block in Bitcoin). For instance, in Bitcoin, each mining machine may have to compute for centuries to create one block. To reduce mining variance, miners form *mining pools* [63, 64], where all members aggregate computational power and share the revenue whenever anyone of them proposes a block. Typically, a mining pool appoints a *pool manager* who will record the computational power contributed by each participant and distribute block rewards accordingly. Similarly, PoS stakers imitate the design of PoW mining pools and then invent *decentralized staking pools* [30, 70]. Basically, the staking pool is a smart contract that allows reward distribution through an automatically executed program. Stakers deposit their cryptocurrency inside the contract and share the staking reward. Up to now, the majority of both computational power in PoW networks and staking power in PoS networks are contributed by pools [61].

However, in PoW blockchains, the pooling approach has been criticized to be vulnerable to the *block withholding (BWH)* attack [25, 46, 55, 58]. That is, malicious pools may infiltrate part of their hash power into the other victim pools such that they pretend to work for the victim pools to share a proportion of rewards from the victims but never contribute valid blocks. Although a malicious pool may lose some reward because of the invalid computational power in the victim pools, the reward obtained from the private mining not only offsets its loss but also increases the overall income of the attacker [46]. Such an attack destroys the fairness of the blockchain incentive design, where malicious attackers earn unfairly higher revenue compared to other honest participants. Even worse, since

*Corresponding author: Jing Tang.

Proceedings of the VLDB Endowment, Vol. 16, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

stakes serve as competing resources in PoS blockchains, the attacker against PoS will achieve an even higher revenue in the future leveraging the unfair revenue earned in the history [35]. Eventually, this unfair revenue will further harm the decentralization of network and threat the data reliability and integrity of blockchain if the majority of stakes are controlled by malicious stakers.

1.2 Contributions

We note that there are quite a few studies [1, 8, 25, 46] focusing on the BWH attack against PoW blockchains, showing severe economic impact. However, despite the popularity of PoS, we are the first to investigate the BWH attack among PoS staking pools. In PoW, the block selection depends on the effective computational power controlled by miners, which is independent of the previous mining history. Consequently, the strategy of BWH attackers is static [25, 46], regardless of the mining outcome. As an example, suppose that the attacker pool controls 70% hash power and the victim pool has the remaining 30% computational resource. The attacker always infiltrates 35% hash power into the victim pool so that it maximizes its expected reward to be ~79%. However, in PoS, stakes serve as competing resources, where attacker's revenue not only depends on the initial investment but also relies on the historical mining revenue. As a result, a PoS attacker may dynamically adjust his strategy according to the mining history to optimize its revenue. As an example, suppose again that the attacker has 70% stakes and the victim possesses 30% wealth initially. At the early stage, the attacker infiltrates 35% wealth into the victim pool. Benefiting from the attack, the attacker may control a stake ratio of 79% later so that infiltrating 35% staking power into the victim pool will bring an expected reward of 85%. On the contrary, i.e., infiltrating 39.5% stakes allows the attacker to receive 88% of total revenue in the future rounds. Therefore, the attacker will adjust the strategy and infiltrate more stakes against the victim.

To cope with the dynamics of participants' wealth, we propose a stochastic process to model the evolution of stakers for the BWH attack against PoS. Specifically, the outcome of reward allocation relies on both the stake distribution and attackers' strategies, which in turn adjusts the stake distribution directly and affects attackers' decisions indirectly to optimize their future revenue. In particular, we consider that the BWH attacker adopts a myopic strategy [60] that maximizes the expected revenue in the next block.

We start out with a simple scenario where there are only two staking pools, including one attacker and one victim, competing stakes in the network. Unlike the BWH attack against PoW where the attacker infiltrates a *fixed* portion of its hash power and obtains a *fixed* extra revenue, we find that the BWH attacker against PoS gradually accumulates its stakes which further increases its return in the future rounds. Eventually, the attacker will control 100% stakes almost surely to manipulate the entire network.

Next, we address the situation where two staking pools infiltrate each other. The stake distribution will affect pools' infiltration actions, and on the other hand will be adjusted by the outcome of game competition. As a result, a stochastic game [59] involves. We show again that one player will eventually monopolize all shares whereas the other player will vanish almost surely. The staking pool with more stakes has a higher chance to become the ultimate winner and the poorer pool is more likely to lose everything.

Finally, we discuss a general case with multiple attacker pools and a solo staker[1] who operates the mining program honestly and does not join any pools. Interestingly, we find that all attackers intend to infiltrate each other and every attacker achieves less reward than they would have if they all behave honestly. In addition, except for the solo staker, every attacker gradually loses its stake ratio during the process, with the result that only one attacker and the solo starker will survive, whereas the other attackers, unfortunately, will all vanish. This finding, complementing the famous *miner's dilemma* [25] for PoW, reveals the *staker's dilemma* for PoS.

In summary, we are the first to thoroughly analyze the BWH attack against PoS and have made the following contributions.

(1) We propose a stochastic process to model the evolution of wealth possessed by stakers for three BWH attack scenarios.
(2) In a simple scenario where one attacker infiltrates staking power into one victim pool, we show that the attacker will finally control 100% share of stakes whereas the victim will lose all shares gradually.
(3) In a game scenario where two pools infiltrate each other, we show that one pool will manipulate all wealth.
(4) In a general scenario with multiple pools and a solo staker, we find that only one pool (e.g., the largest pool very likely) and the solo staker will survive from the attack while all other pools will vanish, indicating the staker's dilemma.
(5) We carry out extensive experiments on real blockchain systems and numerical simulations to validate our analysis. Experimental results confirm our theoretical findings and provide insights into other PoS attacks.

According our findings, to avoid wealth loss with respect to the BWH attack against PoS, stakers are willing to join big pools so that the decentralization of blockchains is damaged. Therefore, more efforts are required to resist such a attack due to its severe damage.

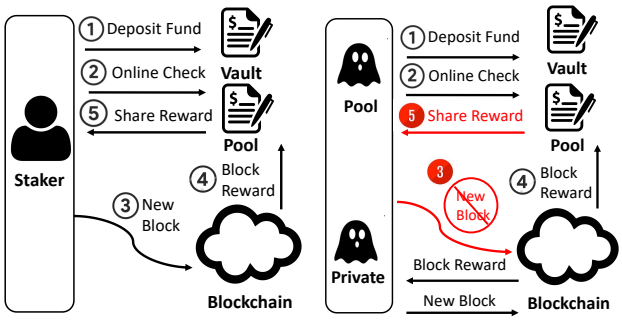
1.3 Organizations

Section 2 introduces the detailed design of pools and the BWH attack against PoS protocols. Section 3.1 gives the notations and assumptions. Section 4 studies a simple case where one attacker pool infiltrates one victim pool. Section 5 explores the game of the BWH attack between two pools. Section 6 investigates a more realistic case with multiple pools and a solo staker. Section 7 verifies our analysis through extensive experiments. Section 8 discusses the lessons learned from the BWH attack against PoS. Section 9 reviews related work. Finally, Section 10 concludes the paper.

2 PRELIMINARIES

Although Bitcoin and other blockchain systems are believed to be secured if the majority of miners behave honestly, current research [25, 32, 46] shows that blockchain systems are vulnerable to several attacks that allow attackers to steal additional rewards from victims. In this section, we first introduce the staking pool design in PoS blockchains. Later, we show the vulnerabilities of PoS, through a comparison with PoW, to the block withholding (BWH) attack.

[1]We use a single solo staker to represent a group of solo stakers, since the effect on each individual solo staker is additive.



(a) Decentralized Staking Pool (b) BWH Attack Against PoS

Figure 1: Block withholding attack against pools.

2.1 Decentralized PoS Staking Pool

The blockchain incentives motivate new stakers joining in the network, which increases the difficulty of generating a block. As a result, a personal staker may spend years to create one block. Naturally, stakers wish to reduce the variance of income by joining pools. On the other hand, stakers do not trust a third party middleman who manages the cryptocurrency on behalf of them in case of losing all of their money. To resolve trust issues, stakers invented *decentralized staking pools* [68, 70]. The working process of staking pools can be found in Figure 1(a). Typically, a staking pool is an automatically executed program running on a smart contract. Specifically, a staking pool contains two components: a vault contract and a pool contract. The vault contract safely keeps the assets of clients and the pool contract receives and distributes the staking rewards according to the contribution of stakers. To join staking pools, every staker deposits its cryptocurrency into a personal vault contract as a collateral asset. Meanwhile, stakers are also required to call a pool contract to show that it is staying online and running the staking program. The pool contract records real-time online status and the collateral asset of each participant, based on which the pool contract can calculate the contribution of everyone. During the staking process, the vault contract operates as a personal representative proposer to create blocks. If a staker is lucky enough to be selected to append a new block, the vault contract receives the staking reward on behalf of the staker and automatically transfers this reward to the pool contract. Finally, the pool contract redistributes staking rewards to everyone online according to the contribution recorded. Currently, the decentralized staking pool mechanism gradually replaces the centralized staking pools. It has been applied by the advanced blockchain systems like ETH 2.0 [31] and Rocket Pool [70–72].

2.2 Block Withholding Attack

In PoW blockchains, the pooling approach has been criticized to be vulnerable to the *block withholding (BWH)* attack [25, 46, 55, 58]. In what follows, we explain the vulnerability of PoS staking pools to the BWH attack. Before elaborating the details, we first introduce the BWH attack against PoW, which shares similar behaviors with the BWH attack against PoS to some extent.

2.2.1 BWH Attack Against PoW. Malicious miners may pretend to work for a pool by submitting the *Partial-Proof-of-Work (PPoW)* to receive the mining dividends but withhold valid blocks (i.e., PoW) they found. Specifically, an attacker joins a victim mining pool using part of its mining power and lets the remaining hash power operate privately. If the attacker finds a valid block for the pool, it discards this block rather than broadcasting it, invalidating the revenue that belonged to the victim pool. Unfortunately, the pool manager cannot detect the malicious behavior and the attacker also receives its dividend share from the manager. Luu et al. [46] confirmed the profitability of the classical BWH attack and showed that the attacker earns extra revenue if it carefully selects an appropriate proportion of hash power for infiltrating. The reason is that the infiltrating power reduces the global mining competition so that the attacker can obtain more rewards from the private mining.

Furthermore, Eyal [25] demonstrated that public mining pools may also launch pool BWH attacks against each other. That is, mining pools can infiltrate mining power towards other pools, just as the miner performs the classical BWH attack. As an example, pool A infiltrates pool B by assigning some of its miners calculating tasks for the work of pool B. Everytime the infiltrating miners submit a PPoW or PoW to pool A, pool A only relays the PPoW to pool B but discards the PoW. Since pool B cannot differentiate the malicious behavior of pool A, pool A will receive dividends from pool B by submitting PPoW. Eventually, both the infiltrating miners and honest miners of pool A will share the revenue evenly. Eyal [25] discovered an interesting "miner's dilemma" with similar properties to the famous "prisoner's dilemma". Specifically, Eyal [25] argued that, under certain conditions, every mining pool may attack the other pools and all of them earn less than what they would obtain if everyone had mined honestly, where the loss goes to the pockets of the solo miners who do not join any pools.

2.2.2 BWH Attack Against PoS. Similar to the PoW mining pool, the PoS staking pool is also vulnerable to the BWH attack. Although the staking pool periodically requests proofs to ensure the stakers staying online, the contract still cannot distinguish whether a staker is proposing blocks honestly or discarding the valid block. In fact, attackers may try to pretend to be an honest staker staying online but discard the valid block they found. Figure 1(b) gives an illustration of the BWH attack against the PoS staking pools. Initially, the attacker splits the staking power into two parts such that it operates an honest staking software using the private staking power and conducts the BWH attack using the pool infiltrating staking power. Then, the attacker registers an account from the victim pool and deposits infiltrating stakes into the vault contract. During the staking process, it pretends to work for the victim pool by showing that it is staying online. Whenever it finds a block using the infiltrating stakes, it discards the block. Although the attacker never contributes valid blocks, the staking pool will still share dividends to the attacker. On the other hand, the attacker can obtain more rewards from the private staking power by reducing the global effective stakes. As a result, the attacker can achieve more rewards than it deserves by conducting the BWH attack against PoS pools. Moreover, similar to the pool BWH attack [25], PoS staking pools can also infiltrate staking power into other PoS pools to steal revenue. In this paper, to generalize the analysis, we focus on the pool BWH attack against PoS.[2]

[2]We shall discuss the vulnerability of various types of staking pools to the pool BWH attack in Section 8.1.

3 MODEL AND POOL GAME

3.1 Notations and Definitions

For the sake of simplicity, we consider a permissionless PoS blockchain using Nakamoto's consensus. We consider a network with a total of m decentralized public staking pools, including the malicious pools that perform BWH attacks and honest pools, and a solo staker, where the solo staker in this paper represents a group of solo stakers. We first study two special cases with $m = 2$, including a scenario with one attacker and one victim in Section 4 and a game between two attackers in Section 5. To better capture reality, we further elaborate on a game when multiple pools conduct the BWH attack against one another together with a solo staker in Section 6.

Given the current block height of n, denote by $z_i(n)$ the total stake ratio of members *loyal* to pool i, including stakers that are either mining honestly in pool i or used for infiltrating pool j. For notational convenience, let $z_0(n)$ represent the stake ratio of the solo staker. Then, $\sum_{i=0}^{m} z_i(n) = 1$. We consider that the proposer reward in each block remains the same during the mining process and is normalized to w on the basis of initial stakes in the genesis block, i.e., w equals the actual block reward divided by the total amount of initial stakes. Note that the proposer reward of blockchain may dynamically change during the mining process. As an example, the halving period of Bitcoin is around 210,000 blocks, which approximates to 4 years in reality. Thus, the reward between two halving events remains relatively stable. In addition, we also consider that the pool propagates the valid block and earns the reward instantly, and redistributes the block reward to the pool participants simultaneously. To explain, although participants in the staking pool may withdraw reward manually, we assume all participants claim dividends instantly once the reward becomes available so that they can re-invest and obtain more passive income. Let $R_i(n+1)$ be the factual proportion of the $(n+1)$-th block reward credited to the stakers loyal to pool i. Then, we can use $R_i(n)$ to bridge the connection between $z_i(n)$ and $z_i(n+1)$, i.e.,

$$(1 + (n+1)w) \cdot z_i(n+1) = (1 + nw) \cdot z_i(n) + w \cdot R_i(n+1), \quad (1)$$

where $(1 + nw) \cdot z_i(n)$ is the total amount of stakes in the network with a block height of n and $w \cdot R_i(n+1)$ is the amount of stakes credited to the stakers loyal to pool i with respect to the $(n+1)$-th block reward, both normalized by the amount of stakes in the genesis block. Note that $R_i(n+1)$ is a random variable depending on the outcome of proposer selection, producing a stochastic process for $z_i(n+1)$.

To characterize the BWH attack, denote by $x_{i,j}(n)$ the amount of staking ratio pool i infiltrating pool j, where $\sum_{j \neq i} x_{i,j}(n) \leq z_i(n)$. We consider that the strategy of attackers is myopic, i.e., maximizing the pool's expected revenue in the next block. To explain, the research in behavioral finance shows that investment managers exhibit myopic loss aversion [23], which implies that the myopic strategy optimizing the short-term reward is usually preferable. Moreover, in Section 8.3, we compare the short-term myopic strategy and the long-term optimal strategy, and find that the two strategies are identical with respect to the BWH attack. In what follows, we formally define the myopic strategy. Let $\mathbf{z}(n) = \big(z_0(n), z_1(n), z_2(n), \ldots, z_m(n)\big)$ be the vector of stake ratios possessed by participants and $\mathbf{x}_i(n) = \big(x_{i,1}, x_{i,2}, \ldots, x_{i,m}\big)$ be the vector of infiltration strategies by pool i where $x_{i,i} = 0$ for notational simplicity. Furthermore, let $\mathbf{X}(n) = \big(\mathbf{x}_1(n), \mathbf{x}_2(n), \ldots, \mathbf{x}_m(n)\big)$ be the infiltration strategies of all pools. Then, we can rewrite $R_i(n+1)$ as a function of $\mathbf{z}(n)$ and $\mathbf{X}(n)$, i.e.,

$$R_i(n+1) := R_i\left(\mathbf{z}(n), \mathbf{X}(n)\right).$$

Therefore, the myopic strategy can be formally defined as follows.

Definition 3.1 (Myopic Strategy). Given the current block height of n with a stake distribution $\mathbf{z}(n)$, the myopic strategy of pool i is an infiltrating vector $\mathbf{x}_i^*(n)$ that maximizes its expected revenue $\mathbb{E}[R_i(n+1)]$ of block $(n+1)$, i.e.,

$$\mathbf{x}_i^*(n) := \arg\max_{\mathbf{x}_i(n)} \mathbb{E}\Big[R_i\big(\mathbf{z}(n), \mathbf{X}(n)\big)\Big]. \quad (2)$$

According to Definition 3.1, it is trivial to see that finding the myopic strategy for every pool i simultaneously is actually a Nash equilibrium, which is referred to as $\mathbf{X}^*(n)$.

Finally, we remark that in this paper, we suppose that attackers do not perform other malicious actions except the BWH attack. In fact, blockchain systems are shown to be vulnerable to tens of attacks, such as selfish mining [26, 27, 42], briberty attack [48], nothing at stake attack [39]. Due to the scope of this paper, we consider that the attackers only conduct the BWH attack and the honest stakers passively participate in the competition without performing any additional actions.

3.2 General Analysis

In particular, given the current block height of n with a known stake distribution $\mathbf{z}(n)$, the pool game in PoS is equivalent to that in PoW with a hash distribution $\mathbf{z}(n)$. However, in PoW, the selection of block proposers relies on the computational power, which is independent of the mining history, i.e., $\mathbf{z}(n)$ being constant, resulting in static strategies for BWH attackers. On the other hand, in PoS, the selection of block proposers relies not only on the initial investment but also on the previous mining outcome, as the rewards will change the stake distribution among stakers, i.e., $\mathbf{z}(n)$ being a random variable of a stochastic process. Consequently, the analysis of the BWH attack against PoS is more challenging than that against PoW, since the latter is just one step in the Markov chain of the former and the former involves a complex analysis of stochastic process.

Leveraging the results for PoW, we first characterize the reward ratio $R_i(n+1)$ of each pool i given the stake ratio vector $\mathbf{z}(n)$ and the infiltrating matrix $\mathbf{X}(n)$. Note that the reward credited to the members loyal to each pool contains not only the direct incentives from the block reward by selecting the pool as the proposer but also the dividends from infiltrated pools. In particular, for pool i, $R_i(n+1)$ can be obtained as follows.

$$R_i(n+1) = z_i(n) \cdot \frac{\mathbb{1}_{Y(n+1)=i} + \sum_{j=1}^{m} \left(x_{i,j}(n) \cdot \frac{R_j(n+1)}{z_j(n)}\right)}{z_i(n) + \sum_{j=1}^{m} x_{j,i}(n)}, \quad (3)$$

where $x_{i,i}(n) = 0$ for each i since a pool never infiltrate to himself, $Y(n+1)$ is the random variable indicating the pool who proposes the $(n+1)$-th block, and $\mathbb{1}_{Y(n+1)=i}$ is an indicator function with a value of 1 if pool i proposes the $(n+1)$-th block and otherwise 0. To explain, the stakers loyal to pool share a fraction of $\frac{z_i(n)}{z_i(n)+\sum_{j=1}^{m} x_{j,i}(n)}$

reward delegated to pool i when pool i is infiltrated by other pools with a total of $\sum_{j=1}^{m} x_{j,i}(n)$ stake ratio. Meanwhile, $\mathbb{1}_{Y(n+1)=i}$ is the direct staking reward delegated to pool i while $x_{i,j}(n) \cdot \frac{R_j(n+1)}{z_j(n)}$ is the dividend from pool j by infiltrating $x_{i,j}$ stake ratio into pool j with a revenue density of $\frac{R_j(n+1)}{z_j(n)}$. One can verify that (3) also applies to the solo staker considering $x_{0,i} = x_{i,0} = 0$ for every pool i such that $R_0(n+1) = \mathbb{1}_{Y(n+1)=0}$.

In what follows, we derive a close form solution to $R_i(n+1)$ based on (3). Denote by $r_i(n+1)$ the revenue density of pool i, i.e.,

$$r_i(n+1) := \frac{R_i(n+1)}{z_i(n)}.$$

Based on (3), define the revenue density vector $\mathbf{r}$, direct staking reward vector $\mathbf{s}$, and infiltration relation matrix $\mathbf{G}$ as follows[3]

$$\mathbf{r} := \big(r_1(n+1), \ldots, r_m(n+1)\big)^{\mathrm{T}}, \tag{4}$$

$$\mathbf{s} := \left(\frac{\mathbb{1}_{Y(n+1)=1}}{z_1(n) + \sum_{j=1}^{m} x_{j,1}(n)}, \ldots, \frac{\mathbb{1}_{Y(n+1)=m}}{z_m(n) + \sum_{j=1}^{m} x_{j,m}(n)}\right)^{\mathrm{T}}, \tag{5}$$

$$\mathbf{G} := \left(\frac{x_{i,j}(n)}{z_i(n) + \sum_{k=1}^{m} x_{k,i}(n)}\right)_{i,j}. \tag{6}$$

Then, we have

$$\mathbf{r} = \mathbf{s} + \mathbf{G}\mathbf{r}.$$

Since each pool cannot infiltrate stake more than it possesses, i.e., $\sum_{j=1}^{m} \mathbf{G}_{i,j} \le 1$, $(\mathbf{I} - \mathbf{G})$ is a diagonally dominant matrix, where $\mathbf{I}$ is an $m \times m$ identity matrix. By the Levy-Desplanques theorem [9], $(\mathbf{I} - \mathbf{G})$ is nonsingular, i.e., invertable. Finally, we can get that

$$\mathbf{r} = (\mathbf{I} - \mathbf{G})^{-1}\mathbf{s}. \tag{7}$$

In addition, as can be seen, the total stakes hidden by the withholding behavior equals $\sum_{j=1}^{m}\sum_{k=1}^{m} x_{j,k}(n)$. Then, the probability that pool i proposes the $(n+1)$-th block is given by

$$\Pr[Y(n+1) = i] = \frac{z_i(n) - \sum_{j=1}^{m} x_{i,j}(n)}{1 - \sum_{j=1}^{m}\sum_{k=1}^{m} x_{j,k}(n)}. \tag{8}$$

Note that the above equation also applies to the solo staker considering $x_{0,j}(n) = 0$ for every pool j. Recall that in Definition 3.1, the myopic strategy of pool i is to optimize its expected revenue $\mathbb{E}[R_i(n+1)]$, which is equivalent to maximize its expected revenue density $\mathbb{E}[r_i(n+1)]$. Taking expectation for (7) yields

$$\bar{\mathbf{r}} = (\mathbf{I} - \mathbf{G})^{-1}\bar{\mathbf{s}}, \tag{9}$$

where

$$\bar{\mathbf{r}}_i := \bar{r}_i(n+1) = \mathbb{E}[r_i(n+1)], \tag{10}$$

$$\begin{aligned}\bar{\mathbf{s}}_i &:= \mathbb{E}\left[\frac{\mathbb{1}_{Y(n+1)=i}}{z_i(n) + \sum_{j=1}^{m} x_{j,i}(n)}\right] \\ &= \frac{z_i(n) - \sum_{j=1}^{m} x_{i,j}(n)}{\left(z_i(n) + \sum_{j=1}^{m} x_{j,i}(n)\right)\left(1 - \sum_{j=1}^{m}\sum_{k=1}^{m} x_{j,k}(n)\right)}.\end{aligned} \tag{11}$$

Again, the above equation also applies to the solo staker considering $x_{0,i} = x_{i,0} = 0$ for every pool i. Since the infiltration among public pools hides part of current staking power and reduces competition, the solo staker benefits from the withholding behavior. In particular, the expected revenue density $\bar{r}_0(n+1)$ of the solo staker increases from 1 (where no attack occurs) to

$$\bar{r}_0(n+1) = \frac{1}{1 - \sum_{i=1}^{m}\sum_{j=1}^{m} x_{i,j}(n)}.$$

This indicates that the expected income of the solo staker is higher than the reward it deserves.

[3] We omit the fixed parameter n when it is explicitly given without confusion.

4 ONE ATTACKER

We first consider a simple scenario consisting of two pools competing in the network, including a BWH attacker pool and a victim pool. The attacker registers a staking account from the victim pool and becomes one of the victim pool' clients. We characterize the myopic strategy of the attacker and show the stationary wealth distribution leveraging the techniques of stochastic approximation [53].

4.1 Property of Myopic Strategy

Without loss of generality, suppose that pool 1 is the attacker and pool 2 is the victim with $z_1(n) + z_2(n) = 1$. Then, given the infiltrating stake ratio $x_{1,2}(n)$, by the definitions of $\bar{\mathbf{s}}$ in (11) and $\mathbf{G}$ in (6), we have

$$\bar{\mathbf{s}} = \left(\frac{z_1(n) - x_{1,2}(n)}{z_1(n)\cdot(1 - x_{1,2}(n))}, \frac{z_2(n)}{(z_2(n) + x_{1,2}(n))\cdot(1 - x_{1,2}(n))}\right)^{\mathrm{T}},$$

$$\mathbf{G} = \begin{pmatrix} 0 & \frac{x_{1,2}(n)}{z_1(n)} \\ 0 & 0 \end{pmatrix}.$$

Thus, it is easy to get that

$$(\mathbf{I} - \mathbf{G}) = \begin{pmatrix} 1 & -\frac{x_{1,2}(n)}{z_1(n)} \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad (\mathbf{I} - \mathbf{G})^{-1} = \begin{pmatrix} 1 & \frac{x_{1,2}(n)}{z_1(n)} \\ 0 & 1 \end{pmatrix}.$$

According to (9), we have

$$\begin{aligned}\bar{r}_1(n+1) &= \frac{z_1(n)z_2(n) + z_1(n)x_{1,2}(n) - x_{1,2}^2(n)}{z_1(n)\cdot(z_2(n) + x_{1,2}(n))\cdot(1 - x_{1,2}(n))} \\ &= \frac{1}{z_1(n)} - \frac{z_2^2(n)}{z_1(n)\cdot\big(z_2(n) + z_1(n)x_{1,2}(n) - x_{1,2}^2(n)\big)}.\end{aligned}$$

It is trivial to see that $\bar{r}_1(n+1)$ achieves the maximum at $x_{1,2}^* = \frac{z_1(n)}{2}$, which is the myopic strategy for pool 1 according to (2). Thus, to maximize the expected revenue myopically, the attacker always infiltrates 50% of its shares into the victim pool whereas leaving the remaining 50% shares mining honestly. For such a case, we have $\bar{r}_1^*(n+1) = 1 + \frac{z_1(n)z_2(n)}{(2 - z_1(n))^2} > 1$, indicating that the attacker will obtain higher reward than it deserves by conducting the BWH attack, where the extra revenue is stolen from the victim.

Moreover, by the definition of $\mathbf{s}$ in (5), we have

$$\mathbf{s} = \left(\frac{\mathbb{1}_{Y(n+1)=1}}{z_1(n)}, \frac{\mathbb{1}_{Y(n+1)=2}}{1 - z_1(n)/2}\right)^{\mathrm{T}},$$

where $\Pr[Y(n+1) = 1] = \frac{z_1(n)}{2 - z_1(n)}$ and $\Pr[Y(n+1) = 2] = \frac{2 - 2z_1(n)}{2 - z_1(n)}$. Putting it into (7) yields

$$r_1(n+1) = \frac{\mathbb{1}_{Y(n+1)=1}}{z_1(n)} + \frac{\mathbb{1}_{Y(n+1)=2}}{2 - z_1(n)}, \text{ and } r_2(n+1) = \frac{\mathbb{1}_{Y(n+1)=2}}{1 - z_1(n)/2}.$$

Finally, combining with (1), we have

$$z_1(n+1) = \frac{z_1(n)\cdot\big(1+nw+w\cdot r_1(n+1)\big)}{1+(n+1)w},$$
$$z_2(n+1) = \frac{z_2(n)\cdot\big(1+nw+w\cdot r_2(n+1)\big)}{1+(n+1)w}.$$

Now, the evolution of $z_1(n)$ and $z_2(n)$ is explicitly devised via a Markov chain.

4.2 Stationary Wealth Distribution

The above analysis indicates that in expectation, the share of the attacker will be accumulated gradually while the share of the victim will be diluted. In what follows, we derive the stationary wealth distribution for the two pools, utilizing the techniques of stochastic approximation [53]. We first introduce some useful definitions and lemmas of stochastic approximation.

Definition 4.1 (Stochastic Approximation [53]). A stochastic approximation algorithm $\{Z_n\}$ is a stochastic process taking value in $[0,1]$, adapted to the filtration $\mathcal{F}(n)$, that satisfies,

$$Z_{n+1} - Z_n = \gamma_{n+1}\big(f(Z_n) + U_{n+1}\big), \tag{12}$$

where $\gamma_n, U_n \in \mathcal{F}_n$, $f\colon [0,1] \mapsto \mathbb{R}$ and the following conditions hold almost surely

(i) $c_l/n \le \gamma_n \le c_u/n$,
(ii) $|U_n| \le K_u$,
(iii) $|f(Z_n)| \le K_f$, and
(iv) $|\mathbb{E}[\gamma_{n+1}U_{n+1} \mid \mathcal{F}_n]| \le K_e\gamma_n^2$,

where c_l, c_u, K_u, K_f, K_e are finite positive real numbers.

The stochastic approximation algorithm was originally invented by Robbins and Monro [54] in 1951 to solve numerical solutions of $f(x) = 0$. Specifically, stochastic approximation is a random walk like a stochastic process, where Z_n denotes a process sequence starting with an initial value Z_0, γ_n is a moving step size gradually decreasing its value with round n, U_n is a random noise decreasing along with n and finally converges to 0, and $f(\cdot)$ serves as a drift function guiding the updates of Z_n. As an example, if $f(Z_n) > 0$, Z_{n+1} is more likely to increase, while if $f(Z_n) < 0$, Z_{n+1} is more likely to decrease its value.

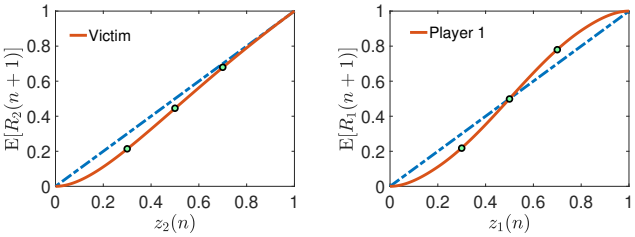
In what follows, we construct an stochastic approximation algorithm for $\{z_i(n)\}$. In fact, the evolution of $z_i(n)$ in (1) can be rewritten in the form of (12) in stochastic approximation. That is,

$$z_i(n+1) - z_i(n) = \frac{w}{1+(n+1)w}\cdot\Big(R_i(n+1) - z_i(n)\Big). \tag{13}$$

In particular, for pool i, let

$$Z_n := z_i(n),$$
$$\gamma_{n+1} := \frac{w}{1+(n+1)w},$$
$$f(Z_n) := \mathbb{E}\big[R_i(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\big] - z_i(n),$$
$$U_{n+1} := R_i(n+1) - \mathbb{E}\big[R_i(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\big].$$

Then, (13) is transformed to (12). We show that such a transformation satisfy all the conditions (i)–(iv) in Definition 4.1. Specifically, observing that $\gamma(n) = \frac{w}{1+nw} \in [\frac{w}{(1+w)n}, \frac{1}{n}]$, we set $c_l = \frac{w}{1+w}$ and $c_u = 1$ satisfying condition (i). For condition (ii), we know that $|U(n)| \le 1$ and thus we set $K_u = 1$. For condition (iii), according to our analysis in Section 4.1, we can get that



(a) Single Pool Attack **(b) Two Pool Game**

Figure 2: Expected reward of player.

$$f(Z_n) = \begin{cases} \frac{z_1^2(n)z_2(n)}{(2-z_1(n))^2}, & \text{if } Z_n = z_1(n), \\ -\frac{z_1^2(n)z_2(n)}{(2-z_1(n))^2}, & \text{if } Z_n = z_2(n). \end{cases} \tag{14}$$

It is easy to check that $|f(Z_n)| \le 1$ and hence we set $K_f = 1$. Finally, for condition (iv), we find that $\mathbb{E}[\gamma_{n+1}U_{n+1} \mid \mathcal{F}_n] = 0$ and thus K_e can be any finite positive real number. Therefore, $\{z_i(n)\}$ is a stochastic approximation algorithm for both pool 1 and pool 2.

Furthermore, intuitively, as an stochastic approximation algorithm, Z_n will finally converge to one of the zero points of $f(Z_n)$ [53]. That is, by (14), Z_n converges to 0 or 1. Note that Z_n may not converge to every zero point of $f(Z_n)$. Specifically, if a zero point q is a stable zero point, Z_n converges to q when $n \to \infty$ with a positive probability. Otherwise, if q is an unstable zero point, Z_n converges to q with zero probability. Using stochastic approximation, we show that $\{z_1(n)\}$ (resp. $\{z_2(n)\}$) will finally converge to 1 (resp. 0) almost surely. The formal proof is non-trivial and can be found in our technical report [6].

Theorem 4.2. *When $z_1(0) + z_2(0) = 1$, the classical BWH attack ends with the fact that,*

$$\lim_{n\to\infty} \Pr[z_1(n) = 1] = 1, \text{ and } \lim_{n\to\infty} \Pr[z_2(n) = 0] = 1.$$

Theorem 4.2 shows that eventually the attacker will manipulate the entire network and the victim will lose all its shares. Figure 2(a) illustrates the expected return of the victim with different skate ratio controlled. It can be seen that no matter how much stake it possesses, the expected revenue of the victim is always lower than the its shares. As a result, the victim's shares will gradually be diluted and finally will converge to 0. Moreover, we observe that the gap between the expected revenue and the stake ratio affects the moving speed. Specifically, if the victim controls more stakes, i.e., $z_2(0) = 70\%$ initially, its shares will converge to 0 in a much longer time than that if $z_2(0)$ initial with 30%.

5 GAME BETWEEN TWO POOLS

Next, we move to a more complicated setting where two decentralized staking pools compete and infiltrate each other. Similar to the one attacker scenario in Section 4, we analyze the myopic strategies of both attackers under the Nash equilibrium and further provider the stationary wealth distribution on the basis of stochastic approximation.

5.1 A Unique Equilibrium

Consider two pools with $z_1(n)$ and $z_2(n)$ stake ratios, respectively, such that $z_1(n) + z_2(n) = 1$. Meanwhile, pool 1 infiltrates pool 2 with $x_{1,2}(n)$ stake ratio while pool 2 infiltrates pool 1 with $x_{2,1}(n)$ stake ratio. For notational convenience, we simply write $y(n)$ as y when the content is clear, where y can be z_1, z_2, $x_{1,2}$, or $x_{2,1}$. Again, using $\bar{\mathbf{s}}$ in (11) and $\mathbf{G}$ in (6), we have

$$\bar{\mathbf{s}} = \left(\frac{z_1 - x_{1,2}}{(z_1 + x_{2,1}) \cdot (1 - x_{1,2} - x_{2,1})}, \frac{z_2 - x_{2,1}}{(z_2 + x_{1,2}) \cdot (1 - x_{1,2} - x_{2,1})}\right)^{\mathrm{T}},$$

$$\mathbf{G} = \begin{pmatrix} 0 & \frac{x_{1,2}}{z_1 + x_{2,1}} \\ \frac{x_{2,1}}{z_2 + x_{1,2}} & 0 \end{pmatrix}.$$

Thus, we can get that

$$(\mathbf{I} - \mathbf{G})^{-1} = \begin{pmatrix} \frac{(z_1 + x_{2,1})(z_2 + x_{1,2})}{z_1 z_2 + z_1 x_{1,2} + z_2 x_{2,1}} & \frac{x_{1,2}(z_2 + x_{1,2})}{z_1 z_2 + z_1 x_{1,2} + z_2 x_{2,1}} \\ \frac{x_{2,1}(z_1 + x_{2,1})}{z_1 z_2 + z_1 x_{1,2} + z_2 x_{2,1}} & \frac{(z_1 + x_{2,1})(z_2 + x_{1,2})}{z_1 z_2 + z_1 x_{1,2} + z_2 x_{2,1}} \end{pmatrix}.$$

According to (9), we have

$$\bar{r}_1(n+1) = \frac{z_1 z_2 + z_1 x_{1,2} - x_{1,2}^2 - x_{1,2} x_{2,1}}{(1 - x_{1,2} - x_{2,1})(z_1 z_2 + z_1 x_{1,2} + z_2 x_{2,1})}, \tag{15}$$

$$\bar{r}_2(n+1) = \frac{z_1 z_2 + z_2 x_{2,1} - x_{2,1}^2 - x_{1,2} x_{2,1}}{(1 - x_{1,2} - x_{2,1})(z_1 z_2 + z_1 x_{1,2} + z_2 x_{2,1})}. \tag{16}$$

This result is also obtained for the BWH attack against PoW by Eyal [25]. In PoW, Alkalay-Houlihan and Shah [1] showed that there is a unique Nash equilibrium $(x_{1,2}^*, x_{2,1}^*)$ when pool 1 chooses $x_{1,2}$ to maximize $\bar{r}_1(n+1)$ in (15) and pool 2 chooses $x_{2,1}$ to maximize $\bar{r}_2(n+1)$ in (16), i.e.,

$$(x_{1,2}^*, x_{2,1}^*) = \begin{cases} \left(0, \frac{z_2}{2}\right), & \text{if } z_1 \le \frac{1}{5}, \\ \left(\frac{z_1}{2}, 0\right), & \text{if } z_1 \ge \frac{4}{5}, \\ \left(\frac{\sqrt{z_1 z_2}\left(2\sqrt{z_1} - \sqrt{z_2}\right)}{\sqrt{z_1} + \sqrt{z_2}}, \frac{\sqrt{z_1 z_2}\left(2\sqrt{z_2} - \sqrt{z_1}\right)}{\sqrt{z_1} + \sqrt{z_2}}\right), & \text{otherwise.} \end{cases}$$

We can directly apply this result to the BWH attack against PoS such that $(x_{1,2}^*, x_{2,1}^*)$ is the myopic strategy for pool 1 and pool 2. In particular, both pools conduct BWH attacks against each other if $|z_1 - z_2| < 60\%$, while if a pool endures a great disadvantage in assets, i.e., with stake ratio no more than 20%, it will give up attacking.

Moreover, by the definition of $\mathbf{s}$ in (5), we have

$$\mathbf{s} = \left(\frac{\mathbb{1}_{Y(n+1)=1}}{z_1 + x_{2,1}^*}, \frac{\mathbb{1}_{Y(n+1)=2}}{z_2 + x_{1,2}^*}\right)^{\mathrm{T}},$$

where $\Pr[Y(n+1) = 1] = \frac{z_1 - x_{1,2}^*}{1 - x_{1,2}^* - x_{2,1}^*}$ and $\Pr[Y(n+1) = 2] = \frac{z_2 - x_{2,1}^*}{1 - x_{1,2}^* - x_{2,1}^*}$. Putting it into (7) yields

$$r_1(n+1) = \frac{(z_2 + x_{1,2}^*) \cdot \mathbb{1}_{Y(n+1)=1} + x_{1,2}^* \cdot \mathbb{1}_{Y(n+1)=2}}{z_1 z_2 + z_1 x_{1,2}^* + z_2 x_{2,1}^*},$$

$$r_2(n+1) = \frac{x_{2,1}^* \cdot \mathbb{1}_{Y(n+1)=1} + (z_1 + x_{2,1}^*) \cdot \mathbb{1}_{Y(n+1)=2}}{z_1 z_2 + z_1 x_{1,2}^* + z_2 x_{2,1}^*}.$$

Finally, combining with (1), we can explicitly obtain the evolution of $z_1(n)$ and $z_2(n)$ via a Markov chain.

5.2 Stationary Wealth Distribution

Next, we analyze the stationary wealth distribution under the BWH game between two players. Analogous to the analysis of one attacker in Section 4.2, $\{z_i(n)\}$ is also a stochastic approximation algorithm for each pool i. Due to space limitations, the detailed derivation of stochastic approximation is provided in our technical report. In the following theorem, we show that only one of the two pools will survive eventually and the other will vanish. Interested readers are referred to our technical report [6] for the proof.

Theorem 5.1. *For two pools with any initial stake ratios such that $z_1(0) + z_2(0) = 1$, the block withholding game ends with the fact that for each pool $i = 1, 2$,*

$$\lim_{n \to \infty} \Pr[z_i(n) \in \{0, 1\}] = 1.$$

Figure 2(b) shows the average reward of the next block obtained by pool 1 under different stake ratio. Specifically, if pool 1 possesses less than 1/2 stakes, it achieves a reward in a single block lower than its shares and thus its shares will be diluted during the staking process. On the contrary, if pool i possesses more than 1/2 stakes, it is wins higher reward than its proportion of wealth, making its shares grow until it monopolizes almost all of stakes. Moreover, we observe that the gap between expected revenue approximates to its shares when its stake ratio approaches 50%. This indicates that both pools will conduct long term competition until the tie breaks. Once one pool wins sufficient advantage (i.e., with more than 60% stake ratio), its advantages will accumulate much faster.

6 GAME AMONG MULTIPLE POOLS

Finally, we consider a general network with m decentralized public staking pools and a solo staker. Again, we analyze the myopic strategy of each attacker and show the stationary wealth distribution.

6.1 Uniqueness of Nash Equilibrium

In general, finding a Nash equilibrium, i.e., myopic strategy, for such a network is not easy. At first glance, we may first compute $\bar{\mathbf{r}}$ via (9). Then, for each pool i, take the gradient of $\bar{\mathbf{r}}_i$ with respect to $\mathbf{x}_i$, referred to as $\nabla_{\mathbf{x}_i} \bar{\mathbf{r}}_i$. As a result, letting $\nabla_{\mathbf{x}_i} \bar{\mathbf{r}}_i = \mathbf{0}$ for each i yields the Nash equilibrium $\mathbf{X}^*$. However, such an approach may produce wrong results, since there might exist some $x_{i,j}^*$ with negative value that is not a feasible solution. For instance, consider the two-attackers scenario in Section 5.1. When $z_1 < \frac{1}{5}$ (resp. $z_1 > \frac{4}{5}$), one can verify that $x_{1,2}^* = \frac{\sqrt{z_1 z_2}(2\sqrt{z_1} - \sqrt{z_2})}{\sqrt{z_1} + \sqrt{z_2}}$ (resp. $x_{2,1}^* = \frac{\sqrt{z_1 z_2}(2\sqrt{z_2} - \sqrt{z_1})}{\sqrt{z_1} + \sqrt{z_2}}$) derived from such an approach is negative, which is infeasible. To tackle this issue, we propose a gradient descent algorithm[4] to derive the numerical solution of a Nash equilibrium. Specifically, we slightly adjust $x_{i,j}$ with a small value ϵ and then compute the new $\bar{\mathbf{r}}_i'$ via (9) using $\mathbf{z}$ and the adjusted $\mathbf{X}'$ so that $(\bar{\mathbf{r}}_i' - \bar{\mathbf{r}}_i)/\epsilon$ approximates the partial derivative with respect to $x_{i,j}$. Thus, we update the value of $x_{i,j}$ to $x_{i,j} + \alpha \cdot (\bar{\mathbf{r}}_i' - \bar{\mathbf{r}}_i)/\epsilon$, where α is the learning rate. To ensure feasibility, we set $x_{i,j}$ to be 0 if it is negative and to be z_i if $x_{i,j} > z_i$. Interestingly, we find that an (almost) identical Nash equilibrium is returned regardless of the initialization of $\mathbf{X}$ when we repeat the experiments. This suggests that there might exist a

[4] We present the algorithm details in our technical report.

unique Nash equilibrium that can be empirically obtained via our gradient descent algorithm.

After obtaining $\mathbf{X}^*$, by (5)–(7), we can get $\mathbf{r}$. Combining with (1) yields $\mathbf{z}(n+1)$, which is a random variable vector with respect to the probability distribution $\Pr[Y(n+1) = i]$ given in (8).

6.2 Stationary Wealth Distribution

In the following, we analyze the stationary distribution of $\mathbf{z}(n)$ when $n \to \infty$. We first give some useful facts to conclude the stationary states.

Fact 1: Any state with BWH attacks is not stationary.

If any pool conducts the BWH attack and hides some staking power, the total network competition reduces. As a result, the solo staker always benefits from the BWH attack, i.e., the expected revenue density is larger than one, resulting in further update of $\mathbf{z}(n)$. This indicates that $\mathbf{z}(n)$ does not converge.

Fact 2: Any state with at least 2 pools is not stationary.

If such a state is stationary, according Fact 1, we know that no pool conducts the attack. However, according to our analysis in Section 4, pool 1 infiltrating pool 2 can increase its revenue. As a consequence, no-one-attacks is not a Nash equilibrium, which is not stationary obviously.

Fact 3: $\mathbf{z}(n)$ will eventually converge.

Generally, $\{\mathbf{z}(n)\}$ is a high dimensional stochastic approximation algorithm [10]. Such a stochastic process will either converge to some stationary points or trap into a circle so that it never converge [10]. However, jumping among a circle of states cannot happen for $\{\mathbf{z}(n)\}$, since the solo attacker keeps accumulating extra wealth. Therefore, $\mathbf{z}(n)$ will converge.

Putting it together, we know that $\mathbf{z}(n)$ will converge to a stationary state (by Fact 3) where at most one pool survives together with the solo staker (by Fact 2).

Lemma 6.1 (Staker's Dilemma). *The stake's dilemma ends with the fact that the solo staker and at most one pool will survive while the other pools will finally vanish.*

The *Staker's dilemma* in PoS compliments the famous *miner's dilemma* [25] in PoW. In miner's dilemma, Eyal [25] shows that PoW attackers lose a static proportion of reward compared to they would have if they all had behaved honestly. In the staker's dilemma, we show that all attackers will vanish except for one lucky survivor, which brings severer damage.

7 EXPERIMENTS

In this section, we evaluate the effects of the BWH attack in PoS systems with both system experiments and numerical simulations. Moreover, we also examine the effects of the BWH attack in PoW systems as a comparison.

7.1 Experimental Setup

We use NXT client and Ethereum as the representatives of PoS and PoW blockchains, respectively. In particular, we select NXT client evaluation version v1.12.2 [37] and Go Ethereum client v1.16 [24]. Note that the original NXT client suffers from fairness issue in block proposer selection [35]. To ensure incentive fairness, we deploy the improved lottery algorithm [35] where the winning probability is proportional to the staking power.

In real system experiments, current PoW mining pools and PoS staking pools are closed-source business projects, due to which we cannot evaluate the real pool system. Therefore, we simulate their core reward distribution process. In particular, the staking pool manager program distributes the revenue delegated to it for every 50 blocks based on the proportion of current deposit of each staker. Meanwhile, pools also update their strategies for every 50 blocks and keep the strategies unchanged until next update.

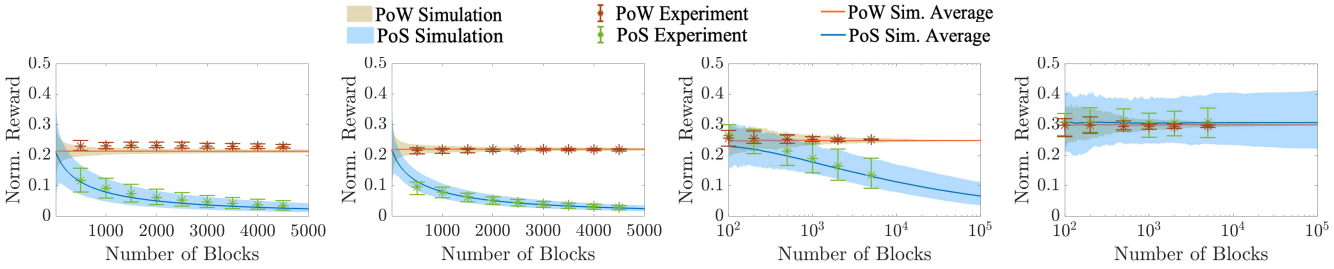
All the above blockchain clients and pool management programs are conducted on Amazon Web Service (AWS). Specifically, we deploy PoS protocols and staking pool programs on c5a.large instances with a 2 cores AMD EPYC 7002 CPU and 4GB RAM. Since PoW systems require more computational resources, PoW mining experiments are deployed on c5a.8xlarge instances each with a 32 cores AMD EPYC 7002 CPU and 64GB RAM. In each experiment, we build a private network connecting all stakers (resp. miners) and each staker (resp. miner) is deployed on an individual AWS instance. We repeat each experiment 100 times for PoS and 10 times for PoW, respectively, and report the statistical results. Numercial simulations are also carried out to supplement the system experiments especially when the latter are constrained by either computational resources or mining time. We repeat numerical simulation 1,000 times for both PoS and PoW.

7.2 BWH Attack Against a Pool

We study the effect of the BWH attack on the profitability of a typical sized pool. Currently, the most popular mining pool on Ethereum manages approximately 30% hash power [61]. Thus, we focus on pool 2 who controls 30% stakes (or hash rates) initially attacked by other BWH pools. In Figure 3, we measure the evolution of the accumulated staking reward proportion for pool 2 in PoW and PoS under four BWH scenarios with a normalized reward $w = 0.01$ of each block with respect to the total amount of initial stakes in the genesis block. In the figure, the bar reports the results of system experiments, in which the central star indicates the sample average and the top and bottom points show the 5-th and 95-th percentiles of the experimental results, respectively. Similarly, the solid line denotes the sample average of numerical simulations, and the upper and lower edges of the shadow area indicate the 5-th and 95-th percentiles of simulations, respectively.

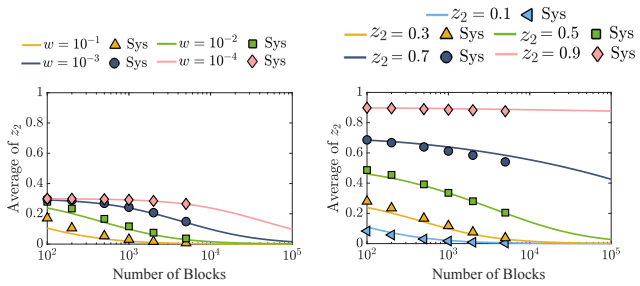
Figure 3(a) shows the result for the one attacker scenario, where pool 2 is the victim and the attacker possesses 0.7 stake/hash ratio initially. As can be seen, in PoW, the average accumulated reward of pool 2 always remains at 0.2, which is 33.3% lower than its initial hash power of 0.3, and the accumulated reward converges to the average when time evolves. However, unlike PoW, the reward of pool 2 in PoS shows a continuously downward trend. Specifically, the reward of pool 2 also initiates at 0.2, but later the reward gradually reduces along with the BWH attack. After 5,000 blocks, the average accumulated reward of pool 2 approximates to 0, which confirms our asymptotic result in Theorem 4.2 that the victim will finally vanish.

Figure 3(b) shows the result when two pools infiltrate each other, where pool 1 controls 0.7 stake/hash ratio. We observe that the



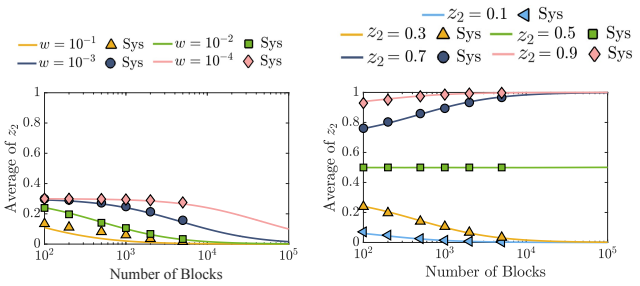
(a) One Attacker (b) Two Attackers (c) Two Attackers + Solo Staker (d) Five Attackers + Solo Staker

Figure 3: Evolution of accumulated reward proportion for a pool with $z(0) = 0.3$ under $w = 0.01$.



(a) Under different block reward (b) Under different wealth

Figure 4: Stakes possessed by victim under one attacker.



(a) Under different block reward (b) Under different wealth

Figure 5: Stakes possessed by player 2 under two attackers.

reward of pool 2 under this case shows a similar trend to that in Figure 3(a). In fact, when pool 2 controls competition resources significantly less than the other attacker, e.g., $z_2 \leq 0.3$, its average reward is almost the same no matter whether pool 2 conducts the BWH attack or not. To explain, as shown in Figure 2(a) and Figure 2(b), pool 2's action makes negligible effect on the expected reward when $z_2 \leq 0.3$. Meanwhile, we also find that pool 2 gradually loses the reward which approaches 0 after 5,000 blocks. This phenomenon verifies our analysis in Theorem 5.1.

Figure 3(c) shows the result for a network with two attackers and one solo staker/miner, where pool 1 controls 0.5 stake/hash ratio and the solo staker/miner controls the remaining 0.2 stake/hash ratio. We observe that in both PoW and PoS, the reward loss of pool 2 is smaller than those in the above two cases where pool 1 controls a higher stake/hash ratio of 0.7. Specifically, the accumulated reward proportion of pool 2 is around 0.25 when $n = 100$ and 0.16 even after $n = 5{,}000$. The reason is that when pool 1 controls less resources, the damage of the BWH attack against pool 2 will be relieved. However, it is just a matter of time that pool 2 will vanish, e.g., its reward reducing to 0.07 when $n = 10^5$, which confirms Lemma 6.1.

Figure 3(d) shows the result for a more realistic situation with five attackers and one solo staker/miner, where each the other pool controls 0.125 stake/hash ratio and the solo staker/miner controls the remaining 0.2 stake/hash ratio. Different from previous results, we find that the average reward proportion of pool 2 maintains around 0.3 which equals its initial shares. The reason is that in such a case, pool 2 is the largest pool among the five attackers. As a result, pool 2 becomes the last survivor in addition to the solo staker/miner, while the other four pools will vanish eventually. This result suggests that joining a large pool not only reduces variance but also protects wealth under BWH attacks.

7.3 One Attacker: More Experiments

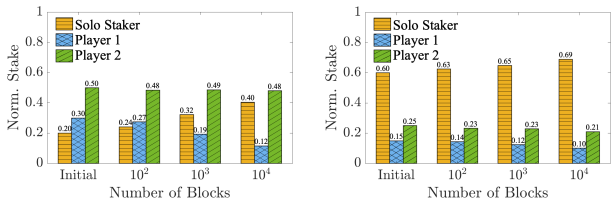
In what follows, we further study the effect of several parameters for the one attacker scenario, including the block reward w and the initial stake ratio possessed by the victim pool 2. Figure 4 compares the stake ratio possessed by the victim pool 2 along with the total number of blocks competed under different parameter settings. The line and marker represent the average of simulation and system results, respectively.

7.3.1 Impact of Block Reward. Figure 4(a) shows the evolution of the victim's stake ratio $z_2(n)$ with $z_2(0) = 0.3$ under different block reward $w \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. We observe that $z_2(n)$ in all settings gradually converges to 0, and decreases faster when the block reward w is larger. This is because if more reward stakes are releasing in each block, the attacker can stole more skates from the victim to accumulate the attacker's advantage quicker.

7.3.2 Impact of Wealth. Figure 4(b) shows the evolution of the victim's stake ratio $z_2(n)$ with $w = 10^{-2}$ under different initial stake ratio $z_2(0) \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. It can be seen that the average shares of the victim in all settings reduces with the number of blocks competed, and decreases faster when the initial stake ratio $z_2(0)$ is smaller. To explain, as shown in Figure 2, the damage is severer when the victim pool size is smaller.

7.4 Two Attackers: More Experiments

For a network consisting of two attackers, we further study the effect of three important parameters, including the block reward w, the initial stake ratio $z_2(0)$ of pool 2, and the initial stake ratio $z_0(0)$ of the solo staker.



(a) $z_0(0) = 0.2$ **for solo staker** **(b)** $z_0(0) = 0.6$ **for solo staker**

Figure 6: Stakes possessed by two pools and solo staker.

7.4.1 Impact of Block Reward and Wealth. Figure 5(a) compares the effect of block reward under two attackers, where the result is similar to that under one attacker. Moreover, Figure 5(b) compares the share of pool 2 under different wealth settings with $z_2(0) \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Interestingly, we observe that $z_2(n)$ converges to 0 if $z_2(0) < 0.5$ and converges to 1 if $z_2(0) > 0.5$. It implies the fact that, the bigger staking pool will finally monopoly all shares and the smaller pool will loses its share in the staking competition. Note that when $z_2(0) = 0.5$, checking the detailed results, we find that $z_2(n)$ will converge to 0 and 1 a fifty-fifty chance, resulting in an average value of 0.5.

7.4.2 Impact of Solo Staker. Figures 6(a) and 6(b) compare the stake ratios possessed by two pools and the solo staker. In Figure 6(a), we set $z_0(0) = 0.2$, $z_1(0) = 0.3$ and $z_2(0) = 0.5$. In Figure 6(b), the solo staker controls more power, i.e., $z_0(0) = 0.6$, and the two pools scale down their stake ratios to $z_1(0) = 0.15$, $z_2(0) = 0.25$, respectively. We observe from Figure 6(a) that pool 1 loses 60% of its shares in the competition while pool 2 only loses 4% of its wealth in the staker's dilemma, indicating that the richer player has more advantages over the poorer player. Furthermore, we also find that, if the solo staker controls less stakes (i.e., $z_0(0) = 0.2$ compared to $z_0(0) = 0.6$), the stake ratio of the solo staker increases faster, e.g., from 0.2 (resp. 0.6) to 0.40 (resp. 0.69) after 10^4 blocks with an increase of 100% (resp. 15%). This is because if attackers 1 and 2 control more stake, they will infiltrate more stakes. For such a case, the solo staker can earn more extra reward.

7.5 Multi-Pool Game: More Experiments

Finally, we study the general network with multiple pools and on solo staker, with the focus of the impact of the number of pools and the wealth distribution.

7.5.1 Impact of Player number. Figure 8 presents the wealth distribution of all players, including the solo staker. We set $z_0(0) = 0.2$ for the solo staker and equally allocate the remaining stakes to m pools. Interestingly, we observe that pool i loses 32.4% shares decreasing from 0.4 to 0.27 when $m = 2$, and only loses 12.4% stakes dropping from 0.08 to 0.06 when $m = 10$ after $n = 10^4$ blocks. In fact, BWH attackers prefer infiltrating less stakes if they control less stakes so that every player loses less. As an example, pool 1 infiltrates pool 2 with a stake ratio of 0.12 when $m = 2$ while the infiltrating stake ratio becomes 7×10^{-4} when $m = 10$. Thus, the total infiltrating stake ratio of all attackers is 0.24 when $m = 2$ and 0.056 when $m = 10$, respectively. As a result, every attacker infiltrates less stakes and steal less reward from other pools which lightens the staker's dilemma, and leaves less benefits for the solo staker.

7.5.2 Impact of Stake Distribution. Figure 7 compares the stakes of players under different initial wealth distributions. Specifically, we set $z_0(0) = 0.2$ for the solo staker and allocate the remaining stakes to 5 pools in three ways, i.e., equally, arithmetically and geometrically. That is, in equal allocation, each pool share 0.16 stake ratio. In arithmetical and geometrically allocations, five pools share the remaining wealth in a proportion of $\{1 : 2 : 3 : 4 : 5\}$ and $\{1 : 2 : 4 : 8 : 16\}$, respectively. We observe that the solo staker obtains 0.32 stakes, 0.36 stakes and 0.38 stakes in equal, arithmetical and geometrical allocations, respectively, after $n = 10^4$ blocks are completed. This indicates that a greater wealth gap among attackers can increase the revenue of the solo staker. Moreover, in geometrical allocation, we also find that pool 1 achieves 0.42 stakes, which is slightly higher than its initial stakes 0.41. It implies that the BWH attacker may achieve higher reward than it deserves under a condition that it has a significant advantage over other attackers.

8 DISCUSSIONS

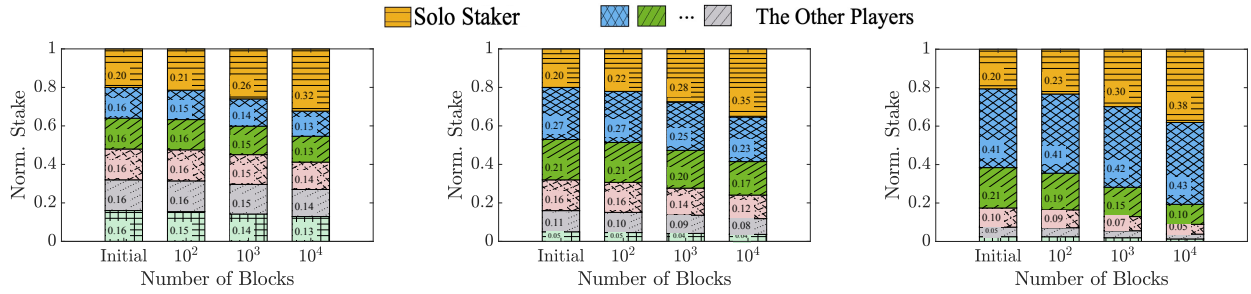
8.1 Staking Pools in Practice

In the following, we discuss three practical categories of staking pools that are vulnerable to the variants of the BWH attack.

Centralized Staking Pool. A centralized staking pool allows multiple stakers to combine their staking power under the management of a custodian entity. Typically, the stake holders who decide to join a pool must deposit their coins into a third party account. Then, the pool manager unites staking power in process of verifying new blocks and receiving staking rewards. However, pool managers may misconduct for more revenue since the operations of manager are not publicly verifiable. For example, the pool manager may outsource clients' stakes to decentralized pools and conduct the classical BWH attack acting as a giant staker. On the other hand, centralized pools have been critiqued that clients take the risk of losing funds as the pool manager may disappear with money.

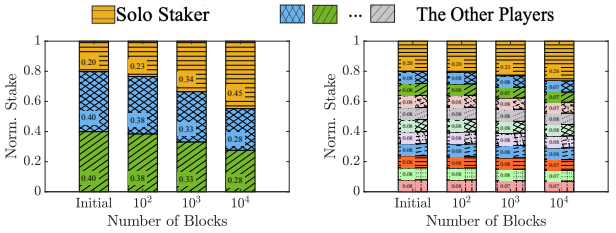
Decentralized Staking Pool. A decentralized staking pool allows clients aggregate their staking power without sending funds to a third party, which enables a more secure model. In a nutshell, the client locks its stakes into a vault smart contract exchanging for a staking power token. The staking power token represents the proportion of the stake owned to each client. After that, the client operates an individual blockchain verification program on its own computer, binding its staking power token. At the same time, the program must periodically prove to the contract that it is staying online and contributing blocks. Once a client of the pool submits a valid block, the smart contract will automatically receive the reward and distribute to every pool member. Apparently, decentralized pools are vulnerable to the BWH attack since attackers may pretend to work for a pool by showing staying online but in fact withholds the valid block.

Heterogeneous Staking Pool. In decentralized staking pools, users are required to operate staking software, which is a tedious workload for most ordinary users. To attract more users, the heterogeneous staking pool combines the centralized and decentralized staking pools, achieving the advantages of both. Heterogeneous



(a) Equally Distribute (b) Arithmetically Distribute (c) Geometrically Distribute

Figure 7: Evolution of Stakes under different wealth distribution.



(a) Two Players + Private Stakers (b) Ten Players + Private Staker

Figure 8: Stakes under different number of Players

pools provide a staking proxy service. Specifically, the personal staker deposits cryptocurrency to proxy managers who manage the fund on behalf of themselves, just like the centralized staking pool. Meanwhile, both the proxy pool and the decentralized staking pool will share the reward to reduce staking variance. In addition, the proxy manager as a third party can infiltrate the stakes deposited into other decentralized pools, i.e., launching the BWH attack. In practice, blockchain pools like Rocket Pool [69] combine both centralized and decentralized pooling services for which their product satisfies different needs of clients.

8.2 Insights into Other PoS Attacks

While many researches analyzed the payout of miners in PoW attacks, rare work studies the long term payout of PoS attackers. In this paper, we show that the attacker in PoS may manipulate the entire network, unlike the BWH attacker in PoW obtaining a fixed extra revenue. The lessons learned here can provide insights into other attacks against PoS where the manipulation may happen.

Selfish Mining Attack. While the classical PoW blockchain assumes that miners publish blocks immediately once found, Eyal and Sirer [26] pointed out that participants who connect to the rest of the network extremely well (e.g., no network latency) may temporarily withhold their blocks to eliminate the hash power of other competitors and obtain more revenue, namely selfish mining. Such a attack also exists in the longest-chain based PoS blockchains [11, 29] such as NXT [65], Qtum [67] and Blackcoin [62]. Again, in PoW, selfish miners earn a *constant* extra revenue in expectation. On the contrary, in PoS, selfish stakers may accumulate advantages in a stochastic process like BWH attackers to monopolize the network.

Nothing at Stake Attack. In the original PoS design, stakers regard the longest chain as the valid chain and stop mining on the outdated forks. However, since mining on PoS forks consumes nothing, malicious stakers may maintain not only the current longest chain but also some (even all) outdated chains simultaneously [39]. If a staker is lucky enough, a outdated chain may occasionally grow quickly to replace the current longest chain so that this staker can receive a big profit. Such an attack is known as the nothing at stake attack. Similarly, malicious attackers may gradually accumulate more shares and even monopolize all shares if they conduct the nothing-at-stake attack for a long time.

8.3 Myopic Strategy and Optimal Strategy

In this paper, we consider that BWH attackers take the myopic strategy that maximizes its reward of the next block. We carry out numeric simulations to empirically compare the short term myopic strategy with the long term optimal strategy that maximize its total revenue in a long run, e.g., of the next 10 blocks. For simplicity, we look at the one attacker scenario in Section 4 where the attacker always infiltrates the victim pool with half of its stakes using the myopic strategy. To get the optimal strategy, we conduct a brute-force search, considering not only the strategy space but also the random outcome of proposer selection. To ensure the search space countable, the continuous decision space is discretized into 100 points uniformly. Interestingly, the empirically results show that the attacker using the optimal strategy also infiltrates half of its stakes, regardless the outcome of proposer selection, which is exactly the myopic strategy. Intuitively, if an attacker wins more expected reward in the next block, it is more likely that the attacker can accumulate more advantages to obtain higher extra revenue in the following new blocks.

9 RELATED WORK

Incentive Attacks. Data integrity and immutability of permissionless blockchain rely on decentralized governance. Malicious attackers may accumulate more reward than they deserve and further cause transaction rollback and data tempering. The BWH attack has been regarded as an important incentive risks by cryptocurrency community since it is raised. In 2013, Mattie [47] posted the first BWH attack report on the Bitcoin forum. Mattie [47] found that PoW miners may send the valid block they found to a privately solo mining client, rather than reporting to the mining pool they serve. Later, Luu et al. [46] formally analyzed the profitability of BWH, and revealed that BWH attackers always achieve a positive payoff by conducting this attack. Eyal [25] modeled the miner's dilemma

such that mining pools infiltrates computational power into each other but the solo miner is final beneficiary. Alkalay-Houlihan and Shah [1] further studied how much computational resource would be wasted at the equilibrium point of the miner's dilemma. All the aforementioned work focused on the BWH attack against PoW. In this paper, we study the BWH attack against PoS by modeling a stochastic process. We show that in PoS, attackers not only enjoy a higher reward but also accumulate advantage during the attack process and finally create monopolization. Surprisingly, our asymptotic analysis reveals that only one pool will finally survive together the solo staker while other pools will vanish, which is obviously much severer than that in PoW where the pools attacked by BWH just lose a fixed proportion of revenue.

In addition to the BWH attack, ather blockchain attacks on incentives have also attracted broader interests from researchers. For example, recent work [26, 27, 44, 57] argued that attackers may temporarily withhold their valid block to eradicate the computation efforts of the other competitors, i.e., selfish mining. McCorry et al. [48] and Gao et al. [32] revealed that malicious miners may bribe other miners so that they consciously create forks in blockchain for higher profit, i.e., briberty attack. Other blockchain attacks are also explored in terms of transaction sequencing algorithm [77, 78], price volatility [16, 40], and miner's movement [43, 73].

Blockchain-as-a-Database. Blockchain has inspired many innovative applications as a decentralized database. Refiner [76] applied an incentive design to motivate data sharing and training in federated learning. BFT-Store [52] enhanced storage scability by leveraging erasure coding with the Byzantine fault tolerance protocol. P^2B-Trace utilized blockchain and zero-knowledge proof on a privacy-preserving COVID-19 contact tracing initiative. CAPER [2] adopted a directed cyclic graph consensus protocol to support confidential transactions and cross-application transactions. CALYPSO [41] applied threshold cryptography on access control to provide privacy-preserving data management without relying on a trusted third party. In addition, some benchmark evaluations compared the difference between permissioned and permissionless blockchains [18, 19, 56], the efficiency of PoW hash functions [28], and the type of transaction failures [13]. Our work evaluates the BWH attack against the PoS consensus algorithm, which clearly suggests that the BWH attack may destroy the data integrity and immutability of decentralized databases as stakers/miners are willing to join a monopolistic pool to protect their wealth.

Transaction Processing. There is a trend of integrating database design into high performance blockchain systems [14]. ResilientDB [34] ultilized a topological-aware consensus algorithm to achieve excellent scalability. Babylon [51] reused PoW mining power to ensure the data immutability of PoS blockchains and protect PoS blockchains from the nothing-at-stake attack and long history attack. Buchnik and Friedman [12] proposed FireLedger, a high throughput blockchain based on a new communication frugal optimistic permissioned protocol. Xu et al. [75] invented Slimchain, a novel blockchain that scales transactions through off-chain storage and parallel processing. Amiri et al. [3] introduced SharPer, a decentralized flattened protocol, to establish cross-shard consensus which also allows high throughput and low latency. Complementing the research of transaction processing, we focus on the security perspective. The insights revealed in the vulnerabilities are helpful for guiding the design of high performance blockchain.

Pólya Urn Processes. The evolution of stakes under the BWH attack has a close relation to the (nonlinear) generalized Pólya urn process [5, 15, 45, 53]. To study the stationary distribution of Pólya urn process, various approaches are invented, including martingales [7, 21], stochastic approximation [38, 53, 54], exponential embedding [22, 49] and brownian motion embedding [15]. In this paper, we utilize the stochastic approximation [53] to derive the stationary distribution of players' wealth under the BWH attack. We show that the evolution of stake shares will finally stop at one of the stable points of the revenue function so that every player achieves a fixed incentive. Our analysis surprisingly shows that, in addition to the solo staker who does not involve any attack, only one attacker will survive while all other competitors will lose all of their wealth. We believe that such an approach can be applied to various attacks against PoS protocol, e.g., selfish mining and nothing-at-stake attack.

Stochastic Game. The BWH attack among staking pools is a stochastic game [59]. Shapley [59] described a discrete time Markov decision process controlled by a group of players. The action of each player depends on the previous state, the random movement of state and the actions chosen by players. In general, equilibrium points in stochastic game are difficult to compute. Sobel [60], and Dirven and Vrieze [20] showed that if some conditions are satisfied, stochastic games exist the myopic equilibrium that is equivalent to the static Nash equilibrium, which can be analyzed numerically and, sometimes, qualitatively. Several applications are proved satisfying the conditions like capital accumulation [4] and advertising [17]. In this paper, we also use a myopic game to characterize the BWH attack against PoS, based on which we derive the players' myopic strategy and stationary wealth distribution of players.

10 CONCLUSION

In this paper, we study the pool BWH attack against PoS protocols under three scenarios, including one attacker, game between two tools and generally among multiple pools. We model the evolution of each player's wealth via a stochastic process and asymptotically analyze the stationary wealth distribution. Through rigorous theoretical analysis and extensive empirical evaluations using both real blockchain systems and numerical simulations, we find that (i) for a network with one attacker and one victim, the attacker will dominate the whole network while the victim pool will vanish no matter how much stake ratio it possesses initially; (ii) for a network with two pools attacking each other, one pool (i.e., the larger one very likely) will monopolize the network and the other will vanish; and (iii) for general case with multiple pools and a solo staker, only one BWH pool (i.e., the largest one very likely) and the solo staker will survive while the other pools will lose all of their wealth, revealing the staker's dilemma. Unfortunately, these results encourage stakers to join big pools to reduce income variance and protect wealth, damaging the decentralization spirit of blockchains and even triggering severe attacks like the 51% attack. Therefore, preventing the BWH attack is essential for PoW blockchains.

REFERENCES

[1] Colleen Alkalay-Houlihan and Nisarg Shah. 2019. The pure price of anarchy of pool block withholding attacks in bitcoin mining. In *Proc. AAAI*, Vol. 33. 1724–1731.

[2] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2019. Caper: A Cross-Application Permissioned Blockchain. *Proc. VLDB Endowment* 12, 11 (2019), 1385–1398.

[3] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. 2021. Sharper: Sharding permissioned blockchains over network clusters. In *Proc. ACM ICDE*. 76–88.

[4] Kenneth J Arrow. 1962. Optimal capital adjustment. *Studies in Applied Probability and Management Science* (1962), 1–17.

[5] W Brian Arthur, Yu M Ermoliev, and Yu M Kaniovski. 1987. Non-linear Urn Processes: Asymptotic Behavior and Applications.

[6] Anonymous Author. 2023. The Last Survivor of PoS Pools: Staker's Dilemma. https://github.com/Yumingg/For_VLDB_Review/blob/main/BWH_Dilemma.pdf

[7] Kazuoki Azuma. 1967. Weighted Sums of Certain Dependent Random Variables. *Tohoku Mathematical Journal, Second Series* 19, 3 (1967), 357–367.

[8] Samiran Bag, Sushmita Ruj, and Kouichi Sakurai. 2016. Bitcoin block withholding attack: Analysis and mitigation. *IEEE Transactions on Information Forensics and Security* 12, 8 (2016), 1967–1978.

[9] Duane W Bailey and Douglas E Crabtree. 1969. Bounds for determinants. *Linear Algebra Appl.* 2, 3 (1969), 303–309.

[10] Julius R Blum. 1954. Multidimensional stochastic approximation methods. *Ann. Mathematical Statistics* (1954), 737–744.

[11] Jonah Brown-Cohen, Arvind Narayanan, Alexandros Psomas, and S Matthew Weinberg. 2019. Formal barriers to longest-chain proof-of-stake protocols. In *Proc. ACM EC*. 459–473.

[12] Yehonatan Buchnik and Roy Friedman. 2020. FireLedger: A High Throughput Blockchain Consensus Protocol. *Proc. VLDB Endowment* 13, 9 (2020), 1525–1539.

[13] Jeeta Ann Chacko, Ruben Mayer, and Hans-Arno Jacobsen. 2021. Why do my blockchain transactions fail? a study of hyperledger fabric. In *Proc. ACM SIGMOD*. 221–234.

[14] Sara Cohen, Adam Rosenthal, and Aviv Zohar. 2020. Reasoning about the Future in Blockchain Databases. In *Proc. IEEE ICDE*. 1930–1933.

[15] Andrea Collevecchio, Codina Cotar, and Marco LiCalzi. 2013. On a Preferential Attachment and Generalized Pólya's Urn Model. *Ann. Applied Probability* 23, 3 (2013), 1219–1253.

[16] Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. 2020. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *Proc. IEEE S&P*. 910–927.

[17] Phoebus J Dhrymes. 1962. On optimal advertising capital and research expenditures under dynamic conditions. *Economica* 29, 115 (1962), 275–279.

[18] Tien Tuan Anh Dinh, Rui Liu, Meihui Zhang, Gang Chen, Beng Chin Ooi, and Ji Wang. 2018. Untangling Blockchain: A Data Processing View of Blockchain Systems. *IEEE TKDE* 30, 7 (2018), 1366–1385.

[19] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. 2017. Blockbench: A Framework for Analyzing Private Blockchains. In *Proc. ACM SIGMOD*. 1085–1100.

[20] CAJM Dirven and OJ Vrieze. 1986. Advertising models, stochastic games and myopic strategies. *Operations research* 34, 4 (1986), 645–649.

[21] Joseph Leo Doob. 1953. *Stochastic Processes*. Vol. 101. New York Wiley.

[22] Eleni Drinea, Alan Frieze, and Michael Mitzenmacher. 2002. Balls and Bins Models with Feedback. In *Proc. ACM SODA*. 308–315.

[23] Kristoffer W Eriksen and Ola Kvaløy. 2010. Myopic investment management. *Review of Finance* 14, 3 (2010), 521–542.

[24] Ethereum. 2021. Geth v1.9.11. https://github.com/ethereum/go-ethereum

[25] Ittay Eyal. 2015. The Miner's Dilemma. In *Proc. IEEE S&P*. 89–103.

[26] Ittay Eyal and Emin Gün Sirer. 2018. Majority Is Not Enough: Bitcoin Mining Is Vulnerable. *Commun. ACM* 61, 7 (2018), 95–102.

[27] Chen Feng and Jianyu Niu. 2019. Selfish Mining in Ethereum. In *Proc. IEEE ICDCS*. 1306–1316.

[28] Zonghao Feng and Qiong Luo. 2020. Evaluating Memory-Hard Proof-of-Work Algorithms on Three Processors. *Proc. VLDB Endowment* 13, 6 (2020), 898–911.

[29] Matheus VX Ferreira and S Matthew Weinberg. 2021. Proof-of-Stake Mining Games with Perfect Randomness. In *Proc. ACM EC*. 433–453.

[30] Ethereum Foundation. 2020. Github: Ethereum 2.0 Specifications. https://github.com/ethereum/eth2.0-specs

[31] ETH Foundation. 2022. ETH 2.0 Beacon Chain Spec. https://github.com/ethereum/consensus-specs/blob/v0.1/specs/core/0_beacon-chain.md

[32] Shang Gao, Zecheng Li, Zhe Peng, and Bin Xiao. 2019. Power Adjusting and Bribery Racing: Novel Mining Attacks in the Bitcoin System. In *Proc. ACM CCS*. 833–850.

[33] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. 2017. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In *Proc. ACM SOSP*. 51–68.

[34] Suyash Gupta, Sajjad Rahnama, Jelle Hellings, and Mohammad Sadoghi. 2020. ResilientDB: Global Scale Resilient Blockchain Fabric. *Proc. VLDB Endowment* 13, 6 (2020), 868–883.

[35] Yuming Huang, Jing Tang, Qianhao Cong, Andrew Lim, and Jianliang Xu. 2021. Do the Rich Get Richer? Fairness Analysis for Blockchain Incentives. In *Proc. ACM SIGMOD*. 790–803.

[36] Markus Jakobsson and Ari Juels. 1999. Proofs of Work and Bread Pudding Protocols. (1999), 258–272.

[37] Jelurida. 2021. NXT Evaluation Toolkit v1.12.2. https://bitbucket.org/Jelurida/nxt-clone-starter/src/master/

[38] Yu Kaniovski and Georg Pflug. 1995. Non-standard Limit Theorems for Urn Models and Stochastic Approximation Procedures. *Comm. Statistics* 11, 1 (1995), 79–102.

[39] Sanket Kanjalkar, Joseph Kuo, Yunqi Li, and Andrew Miller. 2019. Short paper: I can't believe it's not stake! resource exhaustion attacks on PoS. In *Proc. ACM FC*. Springer, 62–69.

[40] Ariah Klages-Mundt, Dominik Harz, Lewis Gudgeon, Jun-You Liu, and Andreea Minca. 2020. Stablecoins 2.0: Economic foundations and risk-based models. In *Proc. ACM AFT*. 59–79.

[41] Eleftherios Kokoris-Kogias, Enis Ceyhun Alp, Linus Gasser, Philipp Jovanovic, Ewa Syta, and Bryan Ford. 2021. CALYPSO: Private data management for decentralized ledgers. *Proc. VLDB Endowment* (2021).

[42] Yujin Kwon, Dohyun Kim, Yunmok Son, Eugene Vasserman, and Yongdae Kim. 2017. Be Selfish and Avoid Dilemmas: Fork After Withholding (FAW) Attacks on Bitcoin. In *Proc. ACM CCS*. 195–209.

[43] Yujin Kwon, Hyoungshick Kim, Jinwoo Shin, and Yongdae Kim. 2019. Bitcoin vs. Bitcoin Cash: Coexistence or Downfall of Bitcoin Cash? *Proc. IEEE S&P* (2019), 935–951.

[44] Yujin Kwon, Jian Liu, Minjeong Kim, Dawn Song, and Yongdae Kim. 2019. Impossibility of Full Decentralization in Permissionless Blockchains. In *Proc. ACM AFT*. 110–123.

[45] Sophie Laruelle and Gilles Pagès. 2019. Nonlinear Randomized Urn Models: A Stochastic Approximation Viewpoint. *Electronic Journal of Probability* 24, 98 (2019), 1–47.

[46] Loi Luu, Ratul Saha, Inian Parameshwaran, Prateek Saxena, and Aquinas Hobor. 2015. On Power Splitting Games in Distributed Computation: The Case of Bitcoin Pooled Mining. In *Proc. IEEE CSF*. 397–411.

[47] Joe Mattie. 2013. A block withholding miner. https://bitcointalk.org/index.php?topic=267181.msg2860365#msg2860365

[48] Patrick McCorry, Alexander Hicks, and Sarah Meiklejohn. 2018. Smart Contracts for Bribing Miners. In *Proc. FC*. 3–18.

[49] Michael Mitzenmacher, Roberto Oliveira, and Joel Spencer. 2004. A Scaling Result for Explosive Processes. *Electronic Journal of Combinatorics* 11, 1 (2004), 1–14.

[50] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System.

[51] Tas Ertem Nusret, David Tse, Fisher Yu, and Sreeram Kannan. 2022. Babylon: Reusing Bitcoin Mining to Enhance Proof-of-Stake Security. arXiv preprint, https://ui.adsabs.harvard.edu/abs/2022arXiv220107946N.

[52] Xiaodong Qi, Zhao Zhang, Cheqing Jin, and Aoying Zhou. 2020. BFT-Store: Storage Partition for Permissioned Blockchain via Erasure Coding. In *Proc. IEEE ICDE*. 1926–1929.

[53] Henrik Renlund. 2010. Generalized Pólya Urns via Stochastic Approximation. arXiv preprint, https://arxiv.org/abs/1002.3716.

[54] Herbert Robbins and Sutton Monro. 1951. A Stochastic Approximation Method. *Ann. Mathematical Statistics* 22, 3 (1951), 400–407.

[55] Meni Rosenfeld. 2011. Analysis of bitcoin pooled mining reward systems. *arXiv preprint arXiv:1112.4980* (2011).

[56] Pingcheng Ruan, Tien Tuan Anh Dinh, Dumitrel Loghin, Meihui Zhang, Gang Chen, Qian Lin, and Beng Chin Ooi. 2021. Blockchains vs. Distributed Databases: Dichotomy and Fusion. In *Proc. ACM SIGMOD*. 1504–1517.

[57] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. 2016. Optimal Selfish Mining Strategies in Bitcoin. In *Proc. FC*. 515–532.

[58] David Schwartz. 2018. *How is block-solution-withholding a threat to mining pools?* https://bitcoin.stackexchange.com/questions/1338/how-is-block-solution-withholding-a-threat-to-mining-pools

[59] Lloyd S Shapley. 1953. Stochastic games. *PNAS* 39, 10 (1953), 1095–1100.

[60] Matthew J Sobel. 1981. Myopic solutions of Markov decision processes and stochastic games. *Operations Research* 29, 5 (1981), 995–1009.

[61] Mining Pool Stats. 2019. *Mining Pool Stats*. https://miningpoolstats.stream/

[62] Blackcoin Team. 2020. *Blackcoin Cryptocurrency*. https://blackcoin.org/

[63] BTC.com Team. 2022. BTC.com Mining Pool. https://pool.btc.com/

[64] F2Pool Team. 2022. F2Pool Mining Pool. https://www.f2pool.com//

[65] Jelurida Team. 2021. *NXT Cryptocurrency*. https://www.jelurida.com/nxt

[66] Peercoin Team. 2020. *Peercoin Cryptocurrency*. https://www.peercoin.net/

[67] Qtum Team. 2021. *Qtum Cryptocurrency*. https://qtum.org

[68] Qtum Team. 2022. Qtum Super Pool. https://blog.qtum.org/super-staker-pool-6f3cf2e00e15

[69] Rocket Team. 2021. Rocket Pool Homepage. https://rocketpool.net/

[70] Rocket Team. 2022. Rocket Pool Spec. https://docs.rocketpool.net/guides/

[71] Rocket Team. 2022. Rocket Pool Staking Protocol Part 1. https://medium.com/rocket-pool/rocket-pool-staking-protocol-part-2-e0d346911fe1
[72] Rocket Team. 2022. Rocket Pool Staking Protocol Part 2. https://medium.com/rocket-pool/rocket-pool-staking-protocol-part-1-8be4859e5fbd
[73] Itay Tsabary and Ittay Eyal. 2018. The Gap Game. In *Proc. ACM CCS*. 713–728.
[74] Gavin Wood. 2014. Ethereum: A Secure Decentralised Generalised Transaction Ledger. *Ethereum Project Yellow Paper* 151 (2014), 1–32.
[75] Cheng Xu, Ce Zhang, Jianliang Xu, and Jian Pei. 2021. SlimChain: scaling blockchain transactions through off-chain storage and parallel processing. *Proc. VLDB Endowment* 14, 11 (2021), 2314–2326.
[76] Zhebin Zhang, Dajie Dong, Yuhang Ma, Yilong Ying, Dawei Jiang, Ke Chen, Lidan Shou, and Gang Chen. 2021. Refiner: a reliable incentive-driven federated learning system powered by blockchain. *Proc. VLDB Endowment* 14, 12 (2021), 2659–2662.
[77] Liyi Zhou, Kaihua Qin, Antoine Cully, Benjamin Livshits, and Arthur Gervais. 2021. On the just-in-time discovery of profit-generating transactions in defi protocols. *arXiv preprint arXiv:2103.02228* (2021).
[78] Liyi Zhou, Kaihua Qin, Christof Ferreira Torres, Duc V Le, and Arthur Gervais. 2021. High-frequency trading on decentralized on-chain exchanges. In *Proc. IEEE S&P*. 428–445.

A PROOFS

A.1 Proof of Theorem 4.2

Recall that in stochastic approximation, Z_n will converge to one of the zero points but may not converge to every zero point. In this following, we first introduce some useful lemmas for characterizing the zero points.

Lemma A.1 (Convergence to Zero Points [53]). *If f is continuous then $\lim_{n\to\infty} Z_n$ exists almost surely and is in $Q_f = \{x\colon f(x) = 0\}$.*

Lemma A.2 (Convergence of Stable Zero Points [53]). *Suppose $q \in Q_f$ is a stable zero point, i.e.,, $f(x)(x-q) < 0$ whenever $x \neq q$ is close to q. If every neighborhood of q is attainable then $\Pr[Z_n \to q] > 0$.*

The above lemma characterizes the convergence of a stable zero point in stochastic approximation. As an example, any point $x < q$ (resp. $x > q$) is close to q, the inequality ensures $f(x) > 0$ (resp. $f(x) < 0$), meaning that $f(x)$ gradually increases (resp. decreases) the value of x closer to $x = q$. Thus, x converges to q with positive probability.

Lemma A.3 (Non-convergence of Unstable Zero Points [53]). *Suppose $q \in Q_f$ is an unstable point such that $f(x)(x-q) \geq 0$ whenever $x \neq q$ in a neighborhood of q, and that $\mathbb{E}[U_{n+1}^2 \mid \mathcal{F}_n] \geq K_L$ holds, for some $K_L > 0$. Then, $\Pr[Z_n \to q] = 0$.*

Oppositely, if q is an unstable zero point, any point $x < q$ (resp. $x > q$) further reduces (resp. increases) its value being repulsed against q by the drift function. Meanwhile, the noise of the process must be positive to ensure that x will leave q even if $x = q$. As a result, x converges to q with zero probability.

Lemma A.4 (Non-convergence of Unstable Boundary Points [53]). *In a stochastic approximation algorithm $\{Z_n\}$, suppose $q \in Q_f$ is an unstable point such that $f(x)(x-q) > 0$, then we have $\Pr[Z_n \to q] = 0$ if there exists $x \neq q$ in the neighborhood of q such that,*

(i) $\mathbb{E}[U_{n+1}^2 \mid \mathcal{F}_n] \leq K_u'|Z_n - q|$,
(ii) $f^2(x) \leq K_f'|x-q|$,
(iii) $k \cdot |Z_k - q| \to \infty$, *as* $k \to \infty$,

where K_f' and K_u' are positive constants.

In addition, if an unstable point locates at the boundary and the noise term vanishes at q. Lemma A.4 provides another sufficient condition of non-convergence.

Now, we are ready to prove Theorem 4.2.

Proof. Without loss of generality, we study the asymptotic convergence of $z_2(n)$, where the convergence of $z_1(n)$ can be directly obtained. Recall that $\{z_2(n)\}$ is a stochastic approximation algorithm by setting

$$Z_n := z_2(n),$$
$$\gamma_{n+1} := \frac{w}{1+(n+1)w},$$
$$f(Z_n) := \mathbb{E}\big[R_2(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\big] - z_2(n) = -\frac{z_2(n)(1-z_2(n))^2}{(1+z_2(n))^2},$$
$$U_{n+1} := R_2(n+1) - \mathbb{E}\big[R_2(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\big].$$

In addition, we observe that $f(Z_n)$ is continuous when $Z_n \in [0,1]$. Thus, by Lemma A.1, the process Z_n finally converges to one of zero points of $f(Z_n)$ almost surely. Let $f(x) = 0$, we know that $f(x)$ has two zero points, i.e., $Q_f = \{0, 1\}$. In the remaining proof, we check the stability of every zero point.

We show that $q = 0$ is a stable zero point. Let $Z_n = x$ locates near 0 and $x > 0$, we always have

$$f(x) \cdot x = -\frac{x(1-x)^2}{(1+x)^2} \cdot x < 0.$$

Moreover, every neighborhood of q is attainable since Z_n can get into the neighborhood after a sufficiently large number of steps with positive probability. This indicates $q = 0$ is stable. By Lemma A.2, Z_n converges to 0 with a positive probability.

Next, we show that $q = 1$ is an unstable boundary point in Lemma A.4. Let x be a point inside a neighborhood of $q = 1$ and close to $q = 1$, we clearly have,

$$f(x) \cdot (x-1) = -\frac{x(1-x)^2}{(1+x)^2} \cdot (x-1) = \frac{x(1-x)^3}{(1+x)^2} \geq 0.$$

Finally, it remains to check condition (i)–(iii) in Lemma A.4. For the condition (i), the noise term has the following relation with $R_2(n)$,

$$\mathbb{E}[U_{n+1}^2 \mid \mathcal{F}_n] = \mathbb{E}[R_2^2(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)] - \mathbb{E}^2[R_2(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)].$$

We know that

$$\mathbb{E}^2[R_2(n+1) \mid \mathcal{F}_n] = \frac{z_2^4(n)}{(1-x_{1,2}^*(n))^2(z_2(n)+x_{1,2}^*(n))^2}.$$

If the attacker pool proposes the $(n+1)$-th block, i.e., $Y(n+1) = 1$, the victim receives no reward; if the victim is selected as the proposer, it gets $\frac{z_2(n)}{z_2(n)+x_{1,2}^*(n)}$ reward and share $\frac{x_{1,2}^*(n)}{z_2(n)+x_{1,2}^*(n)}$ to the attacker. Therefore,

$$\mathbb{E}[R_2^2(n+1) \mid \mathcal{F}_n] = \Pr[Y(n+1) = 2] \cdot \left(\frac{z_2(n)}{z_2(n)+x_{1,2}^*(n)}\right)^2 = \frac{z_2^3(n)}{(1-x_{1,2}^*(n))(z_2(n)+x_{1,2}^*(n))^2}.$$

Then, the noise term has the following relation,

$$\frac{\mathbb{E}[U_{n+1}^2 \mid \mathcal{F}_n]}{|Z_n - 1|} = \frac{8z_2^3(n)}{(1+z_2(n))^4} \leq 8 \triangleq K_u',$$

where $Z_n = z_2(n)$ locates inside a neighborhood of $q = 1$, i.e., $Z_n = z_2(n) \in [1-\varepsilon, 1)$. Moreover, for condition (ii),

$$\frac{f^2(x)}{|x-q|} = \frac{x^2(1-x)^3}{(1+x)^4} \le 1 \triangleq K_f'.$$

Finally, for condition (iii), we show by induction that

$$Z_n \le 1 - \frac{1-Z_0}{(1+Z_0)\sqrt{n}} \triangleq 1 - \frac{c}{\sqrt{n}}, \tag{17}$$

which directly implies that $k \cdot |Z_k - 1| = \frac{(1-Z_0)\sqrt{k}}{1+Z_0} \to \infty$, as $k \to \infty$. In fact, letting $z_2(n) = Z_n$ in (13), we have

$$Z_{n+1} - Z_n \le \frac{1}{n+1} \cdot \left(\frac{Z_n}{Z_n + (1-Z_n)/2} - Z_n \right),$$

since $R_2(n+1) \le Z_n + (1-Z_n)/2$ when pool 1 infiltrates $z_1(n)/2 = (1-Z_n)/2$ stake ratio. Rearranging it yields

$$(n+1)Z_{n+1} \le nZ_n + 2 - \frac{2}{1+Z_n}. \tag{18}$$

Thus, $Z_1 \le 2 - \frac{2}{1+Z_0} = 1 - \frac{1-Z_0}{(1+Z_0)}$. By induction, suppose that (17) holds for n. Then, by (18), we have

$$\begin{aligned}
&(n+1)Z_{n+1} \\
&\le n\left(1 - \frac{c}{\sqrt{n}}\right) + 2 - \frac{2}{1+1-\frac{c}{\sqrt{n}}} \\
&= n + 1 - c \cdot \frac{2n - c\sqrt{n} + 1}{2\sqrt{n} - c} \\
&= n + 1 - c \cdot \left(\sqrt{n+1} + \frac{(\sqrt{n+1} - \sqrt{n})^2 + (\sqrt{n+1} - \sqrt{n})c}{2\sqrt{n} - c} \right) \\
&\le n + 1 - c\sqrt{n+1}.
\end{aligned}$$

Rearranging it yields $Z_{n+1} \le 1 - \frac{c}{\sqrt{n+1}}$, which proves (17) for any integer n. Putting it together, by Lemma A.4, $q = 1$ is an unstable boundary point and Z_n converges to 1 with zero probability.

Since $q = 0$ is the unique stable zero point, by Lemma A.1, $z_2(n)$ converges to 0 almost surely. Meanwhile, $z_1(n) = 1 - z_2(n)$ so that it converges to 1 almost surely. This completes the proof. □

A.2 Proof of Theorem 5.1

Proof. Without loss of generality, we focus on the asymptotic convergence of the wealth managed by player 1, i.e., $z_1(n)$. The wealth of player 2 can be easily derived based on $z_1(n)$. First, we analyze the distribution of reward in each block, which is helpful for the following analysis.

According to our analysis in Section 5.1, we know that

$$R_1(n+1) = z_1 \cdot \frac{(z_2 + x_{1,2}^*) \cdot \mathbb{1}_{Y(n+1)=1} + x_{1,2}^* \cdot \mathbb{1}_{Y(n+1)=2}}{z_1 z_2 + z_1 x_{1,2}^* + z_2 x_{2,1}^*},$$

where $\Pr[Y(n+1)=1] = \frac{z_1 - x_{1,2}^*}{1 - x_{1,2}^* - x_{2,1}^*}$, $\Pr[Y(n+1)=2] = \frac{z_2 - x_{2,1}^*}{1 - x_{1,2}^* - x_{2,1}^*}$, and

$$(x_{1,2}^*, x_{2,1}^*) = \begin{cases} \left(0, \frac{z_2}{2}\right), & \text{if } z_1 \le \frac{1}{5}, \\ \left(\frac{z_1}{2}, 0\right), & \text{if } z_1 \ge \frac{4}{5}, \\ \left(\frac{\sqrt{z_1 z_2}\left(2\sqrt{z_1} - \sqrt{z_2}\right)}{\sqrt{z_1} + \sqrt{z_2}}, \frac{\sqrt{z_1 z_2}\left(2\sqrt{z_2} - \sqrt{z_1}\right)}{\sqrt{z_1} + \sqrt{z_2}} \right), & \text{otherwise.} \end{cases}$$

In addition, let

$$\begin{aligned}
Z_n &:= z_1(n), \\
\gamma_{n+1} &:= \frac{w}{1 + (n+1)w}, \\
f(Z_n) &:= \mathbb{E}\left[R_1(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\right] - z_1(n), \\
U_{n+1} &:= R_1(n+1) - \mathbb{E}\left[R_1(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\right].
\end{aligned}$$

We immediately have

$$Z_{n+1} - Z_n = \gamma(n)\Big(f(Z_n) + U_{n+1}\Big).$$

Now, to ensure $\{Z_n\}$ being a stochastic approximation algorithm, it remains to verify that conditions (i)–(iv) in Definition 4.1 are all satisfied. For condition (i), it is trivial to see that $\gamma_n = \frac{w}{1+(n+1)w} \in [\frac{w}{n(w+1)}, \frac{1}{n}]$, and setting $c_l = \frac{w}{1+w}$ and $c_u = 1$ meets the requirement. For condition (ii), we know $|U_n| \le 1$ and thus we set $K_u = 1$. For condition (iii), we can get that

$$f(Z_n) = \begin{cases} -\frac{Z_n(1-Z_n)^2}{(1+Z_n)^2}, & \text{if } Z_n \le \frac{1}{5}, \\ \frac{Z_n^2(1-Z_n)}{(2-Z_n)^2}, & \text{if } Z_n \ge \frac{4}{5}, \\ \frac{Z_n(1-Z_n)(2Z_n-1)}{1-2Z_n(1-Z_n)+\sqrt{Z_n(1-Zn)}}, & \text{otherwise.} \end{cases}$$

It is easy to check that $|f(Z_n)| \le 1$ and hence we set $K_f = 1$. Finally, for condition (iv), we find that $\mathbb{E}[\gamma_{n+1}U_{n+1} \mid \mathcal{F}_n] = 0$ and thus K_e can be any finite positive real number. Therefore, $\{z_1(n)\}$ is a stochastic approximation algorithm. Moreover, $f(Z_n)$ is a continuous function if $0 \le Z_n \le 1$. By Lemma A.1, Z_n will finally converge to a stable zero point of $f(Z_n)$.

When $Z_n \le \frac{1}{5}$, let $f(Z_n) = -\frac{Z_n(1-Z_n)^2}{(1+Z_n)^2} := 0$. We then have $Z_n = 0$ or $Z_n = 1$. Apparently, $Z_n = 1$ is infeasible. Meanwhile, using the same argument in the proof of Theorem 4.2, we know that $Z_n = 0$ is a table zero point. When $Z_n \ge \frac{4}{5}$, using a symmetric argument, we know that $Z_n = 1$ is a table zero point. Finally, when $\frac{1}{5} < Z_n < \frac{4}{5}$, letting $f(Z_n) := 0$ gives rise to $Z_n = 0$, $Z_n = 1/2$ and $Z_n = 1$. For such a case, both $Z_n = 0$ and $Z_n = 1$ are infeasible. We further show that $Z_n = 1/2$ is an unstable point by Lemma A.3. Let x be a point locate at the neighborhood of $1/2$, i.e., $x \in (1/2-\varepsilon, 1/2+\varepsilon)$. Then, we have

$$f(x) \cdot (x - 1/2) = \frac{2x(1-x)(x-1/2)^2}{1 - 2x(1-x) + \sqrt{x(1-x)}} \ge 0.$$

In addition, we can rewrite $R_1(n+1)$ as

$$R_1(n+1) = z_1 \cdot \frac{z_2 \cdot \mathbb{1}_{Y(n+1)=1} + x_{1,2}^*}{z_1 z_2 + z_1 x_{1,2}^* + z_2 x_{2,1}^*},$$

Thus, we have

$$\begin{aligned}
U_{n+1} &= R_1(n+1) - \mathbb{E}\left[R_1(n+1) \mid \mathcal{F}_n, \mathbf{X}^*(n)\right] \\
&= z_1 z_2 \cdot \frac{\mathbb{1}_{Y(n+1)=1} - \Pr[Y(n+1)=1]}{z_1 z_2 + z_1 x_{1,2}^* + z_2 x_{2,1}^*}
\end{aligned}$$

Then, we can get that

$$\begin{aligned}
\mathbb{E}[U_{n+1}^2 \mid \mathcal{F}_n] &= \frac{\Pr[Y(n+1)=1] \cdot (1 - \Pr[Y(n+1)=1]) \cdot z_1^2 z_2^2}{(z_1 z_2 + z_1 x_{1,2}^* + z_2 x_{2,1}^*)^2} \\
&= \frac{z_1 z_2 \sqrt{z_1 z_2}}{4(1 - \sqrt{z_1 z_2})^2 (\sqrt{z_1} + \sqrt{z_2})^2},
\end{aligned}$$

Algorithm 1: `SearchNE`

Input: Stake ratio vector $\mathbf{z}$, learning rate α, step size ϵ
Output: Nash equilibrium $\mathbf{X}^*$

Initialize a random $\mathbf{X}$;
while $\mathbf{X}$ *does not converge* **do**
  foreach *BWH attacker* i **do**
    foreach *target pool* j **do**
      $x'_{i,j} \leftarrow x_{i,j} + \epsilon$;
      Calculate $\bar{\mathbf{r}}_i$ (resp. $\bar{\mathbf{r}}'_i$) using $\mathbf{z}$ and $\mathbf{X}$ (resp. $\mathbf{X}'$) via (9);
      Update $x_{i,j} \leftarrow x_{i,j} + \alpha \cdot \left(\bar{\mathbf{r}}'_i - \bar{\mathbf{r}}_i\right)/\epsilon$;
      $x_{i,j} \leftarrow \max(0, \min(z_i, x_{i,j}))$;
return $\mathbf{X}^* \leftarrow \mathbf{X}$;

where $\Pr[Y(n+1) = 1] = \frac{\sqrt{z_1}}{\sqrt{z_1}+\sqrt{z_2}}$ and $z_1 z_2 + z_1 x^*_{1,2} + z_2 x^*_{2,1} = 2\sqrt{z_1 z_2}(1 - \sqrt{z_1 z_2})$. Define $t := \sqrt{z_1 z_2}$ such that $t \in (\frac{2}{5}, \frac{1}{2})$ when $z_1 \in (\frac{1}{5}, \frac{4}{5})$. Then, we have

$$\mathbb{E}[U^2_{n+1} \mid \mathcal{F}_n] = \frac{t^3}{4(1-t)^2(1+2t)} > \frac{1}{4(\frac{5}{2}-1)^2(\frac{5}{2}+2)} = \frac{2}{81} \triangleq K_L.$$

By Lemma A.3, Z_n converges to $1/2$ with zero probability since $1/2$ is an unstable zero point. Putting it together, Z_n must converge to either 0 or 1. This completes the proof. □

A.3 Search Algorithm for Nash Equilibrium

Algorithm 1 presents the pseudo-code of our gradient descent algorithm, namely `SearchNE`, to derive the numerical solution of a Nash equilibrium for the general scenario when there are multiple pools and one solo staker.