

# Weighted Voting on the Blockchain: Improving Consensus in Proof of Stake Protocols

Stefanos Leonardos\*, Daniël Reijbergen\*, Georgios Piliouras\*

\*Singapore University of Technology and Design

## Abstract

Proof of Stake (PoS) protocols rely on voting mechanisms to reach consensus on the current state. If an enhanced majority of *staking nodes*, also called validators, agree on a proposed block, then this block is appended to the blockchain. Yet, these protocols remain vulnerable to faults caused by validators who abstain either accidentally or maliciously.

To protect against such faults while retaining the PoS selection and reward allocation schemes, we study *weighted voting* in validator committees. We formalize the block creation process and introduce validators' *voting profiles* which we update by a *multiplicative weights* algorithm relative to validators' voting behavior and aggregate blockchain rewards. Using this framework, we leverage *weighted majority voting rules* that optimize collective decision making to show, both numerically and analytically, that the consensus mechanism is more robust if validators' votes are appropriately scaled. We raise potential issues and limitations of weighted voting in trustless, decentralized networks and relate our results to the design of current PoS protocols.

## Index Terms

Proof of Stake, Consensus, Weighted Voting, Multiplicative Weights Update

## I. INTRODUCTION

In the Nakamoto consensus protocol [41] that underpins the popular Bitcoin cryptocurrency, a single miner claims the right to append the next block to the blockchain by a demonstrating so-called Proof of Work (PoW), i.e., the solution to a moderately hard cryptographic puzzle. The strengths and vulnerabilities of this mechanism are well understood, see [27]–[29] and [26], [32], [51]. Two fundamental properties satisfied by PoW selection are the following: (i) Miners holding  $p\%$  of computational power will create on average  $p\%$  of blocks that will be part of the blockchain assuming that all miners follow the protocol. This property relies on the randomness of the cryptographic puzzle and ensures *fairness* in the allocation of mining rewards [45], [46]. (ii) Miners can be selected as the next block creators only when they actually create and submit the block to the network. This property asserts that the random selection of the next block creator and the creation of the next block occur simultaneously.

Proof of Stake (PoS) or virtual mining protocols [16] constitute alternative selection mechanisms that aim to retain PoW's benefits while improving on its weaknesses [13], [17]. While different PoS protocols propose different schemes [14], [19], [30], [33], [35], [47], in general, the block proposal mechanism is the following: blocks are created by staking nodes, also called *validators* [36], [49], who are granted the right to participate in the block creation process by locking capital in protocol currency, called *stake*. Subsequently, a pseudo-random mechanism selects nodes proportionally to their stake to form committees that will decide on the validity of a block proposed by a selected node, called *block proposer* or *leader*. At the core of the consensus mechanism, the selected validators cast approval or disapproval votes on the proposed block. If an enhanced majority of approval votes, usually  $2/3$  of the committee size [37], is reached, then the proposed block is accepted and appended to the blockchain.

However, maliciously or accidentally abstaining validators can cause consensus failures and stall the blockchain<sup>1</sup>. The reason is, that unlike Nakamoto consensus [56], PoS protocols decouple the selection of block creator(s) from the actual block creation and hence do not satisfy property (ii) above. Since consensus on the “valid” state of the blockchain relies on the voting behavior of all participating validators, this also implies that the actual rate of blocks that a validator gets to create may differ from the times that they get selected in a committee. Hence, while the PoS selection mechanism aims to ensure fair allocation of “mining” rewards in line with PoW's property (i) above, in practice, it may fail to do so. These observations motivate the following question:

*How can we improve the efficiency and robustness of the consensus mechanism, i.e., how can we use information on validators' past voting patterns to enforce that with overwhelming probability the selected validator committees will reach consensus on the “valid” state of the blockchain?*

This work was supported in part by the National Research Foundation (NRF), Prime Minister's Office, Singapore, under its National Cybersecurity R&D Programme (Award No. NRF2016NCR-NCR002-028) and administered by the National Cybersecurity R&D Directorate. Georgios Piliouras acknowledges SUTD grant SRG ESD 2015 097, MOE AcRF Tier 2 Grant 2016-T2-1-170 and NRF 2018 Fellowship NRF-NRFF2018-07.

<sup>1</sup>Malicious abstention refers to non-voting or incorrect voting to attack the protocol, whereas accidental abstention refers to a variety of reasons, such as dropping offline, experiencing network latency, bugs in client updates or being a victim of a censoring/eclipse attack.

The problem of improving consensus has been treated in the context of fixed-size committee voting by a rich stream of literature [12], [42], [43], [53], [58]. The derived solutions depend on quantifying voters' abilities to make correct decisions and apply *weighted voting rules* in which votes are weighted according to voters' profiles. Our goal is to apply these results in the setting of PoS protocols. The link is immediate, since, as remarked in [12], the assumptions of their original model imply both decentralized information processing and limited communication. The additional challenges that have to be treated in the blockchain context, concern the updating of these profiles and protection against their manipulation by adversaries.

To address this problem, we formulate the proper mathematical framework and develop a model to quantify validators' *voting profiles*. The proposed scheme is applied *once* voting committees have formed and does not modify the underlying PoS selection and reward mechanisms. Each staking node (validator) is assigned a score based on their so-far contribution to protocol execution. When selected for a committee, their vote is weighted relative to their profile and consensus is decided according to a *weighted majority rule* that maximizes the expected collective rewards [12], [53]. Finally, based on their vote and the overall consensus outcome, their voting profile is revised according to a fully parametrizable *multiplicative weights update algorithm* [2], [9]. Supported by numerical examples and simulations, our findings demonstrate that weighted voting renders the consensus mechanism more efficient, even if more than  $1/3$  of nodes are not properly voting. In this way, the proposed scheme restores fairness without compromising other PoS features. Additionally, since it does not modify the underlying PoS mechanism, it can be tested, implemented and reverted with minimal cost to existing protocol users.

We raise issues that pertain to weighted voting such as loss of anonymity & centralization and discuss their relevance to protocol design and implementation.

### A. Related Work

Although weighted voting in distributed systems is known to increase efficiency, incentive compatibility [4], [6], and network reliability [8], it also raises additional risks [3], [7], [60]. Similar to *proof of reputation* systems [59], weighted voting deviates from the principle *one node – one vote* and hence, is vulnerable to manipulation by adversaries. Weighted voting becomes particularly relevant in less anonymous [34], private or permissioned blockchains [1], [55], [57], in *delegated* PoS protocols [31], [39] and in PoS protocols with low targeted number of *staking pools* [18]. However, under rather general assumptions it can also be relevant for permissionless Proof of Stake or hybrid Proof of Work and Proof of Stake systems [19], [33]. We provide such a use case in Section V-A.

A profiling system reduces or eliminates the anonymity of staking nodes which is at odds with the design philosophy of permissionless blockchains. However, with blockchain governance yet to be determined, introducing less anonymity may be a desired feature. Recent results support relatively low desired numbers of staking nodes [20] or stake pools [18]. In such schemes, reputation will implicitly or explicitly influence protocol execution. Moreover, stake pools can retain anonymity at the user level, i.e., while the pool itself becomes identifiable by its voting profile, the users remain anonymous. In any case, the introduction of voting profiles and weighted voting seems particularly relevant to permissioned blockchains or delegated PoS mechanisms.

More interesting is the implementation of the weighted voting algorithm in conjunction with state of the art consensus schemes such as the delegated Proof of Stake of EOS.IO [31]. Instead of substituting the underlying consensus mechanism, the weighted voting scheme enhances its functionality and operability by providing an additional layer of defense against accidental or adversarial system failures. In fact, in consensus systems that use a limited number of delegates, weighted voting can prove beneficial in accelerating consensus and enhancing their scalability, liveness and safety. This rests on the fact that assigning and updating profiles to the delegates is much easier than in general permissionless blockchains. However, even in the latter case, the expected latency (and overhead) that will be introduced to the system by the profiling system does not exceed nor significantly increases the computational complexity of updating the validators' stakes.

Finally, claiming that *all* PoS protocols fit under the stylized model – see Section II – that we use to design the proposed weighted voting scheme would be an oversimplification and wrong. Yet, most PoS protocols that we are aware of involve voting mechanisms and hence, may benefit – to a larger or lesser extent – from the present proposal. An incomplete list includes [14], [30], [33], [35], [47]. For more extensive details of PoS protocols, we refer to [10], [17], [22], [24]. Further comparisons to existing proposals and implementation details are given in Section VI

### B. Outline

In Section II, we abstract the PoS consensus mechanism as a voting game. In Section III, we construct the improved voting scheme and illustrate it with examples. Section IV is devoted to the design of the updating algorithm and the derivation of bounds on the validators' faulty behavior that can be tolerated with significantly affecting their profiles. The resulting scheme – weighted voting rule and multiplicative weight updating algorithm – are tested and related in potential use cases in Section V. We conclude the paper with a discussion about limitations and implementation issues in Section VI and a summary of our findings in Section VII.

As an extension of a conference paper [38], the present article contributes both technical and application specific materials that make the paper self-contained and enhance the intuition behind the derived results. In addition to a detailed comparison with related work, cf. Section I-A, technical extensions comprise Lemma 1 and its proof, the proof of Theorem 3, Figure 1 and Remark 4. From an application oriented perspective, the present paper elaborates on the design and optimal parametrization of such voting schemes in practice. This is done in Section IV-A, Section V-A and Section V-B in the context of the state of the art PoS proposals and implementations, such as Ethereum with Casper FFG and EOS.IO. The main focus of the analysis is in the mechanics of committee formation and the risk mitigation that is achieved via the proposed scheme for potential investors. In particular, the new content addresses incentives and adversarial behavior in potential use cases and moves forward to explain the steps for the adoption of the proposed scheme in practice.

## II. THE MODEL

Our terminology is based on the Ethereum 2.0 PoS protocol specification [19]. Yet, our model remains as general as possible in an effort to capture similar voting mechanisms implemented by related PoS platforms.

**Time:** Time is divided into *time slots*  $t \in \{0, 1, 2, \dots\}$  of fixed duration  $d$ . Each time slot is dedicated to the proposal and creation of a new *block*  $B_t$ . Time slot  $t = 0$  is the time of creation of the *genesis block*  $B_0$ .

**Validators:** The main actors in the block proposal and creation mechanism are the *staking nodes*, also called *validators*, denoted by  $i \in I$ , where  $I \subseteq \mathbb{N}$  is the *set* of all validators.  $\mathcal{B}_{i,t}$  will denote the set of blocks for which validator  $i$  is aware of at time  $t \geq 0$ .

**Stake:** The deposit or *stake* is the amount of the underlying cryptocurrency that a potential validator locks as *Proof of Stake* (PoS) to participate in the block creation process. Such deposits may change over time. Accordingly, let  $v_{i,t} \geq 0$  denote the stake of validator  $i \in I$  and  $v_t := \sum_{i \in I} v_{i,t}$  the total stake at time slot  $t \in \mathbb{N}$ . If  $v_{i,t} > 0$ , then validator  $i$  is called *active* at time slot  $t$ . Validators who have withdrawn from  $I$  or who have not entered it yet, can be thought of as validators with stake  $v_{i,t} = 0$ . Thus, although the set of validators is dynamic, we may write  $I$  instead of  $I_t$  to denote the set of validators at any time  $t \geq 0$ .

**Block proposer & committees:** To create blocks and extend the blockchain, active validators are selected *proportionally to their stake* by a pseudo-random mechanism which assigns to each time slot  $t$  a *leader* or *block proposer* and a fixed-sized *committee*  $N_t = \{1, 2, \dots, n\}$  of validators<sup>2</sup>. The block proposer is assigned the task to propose a block  $B_t$  to the committee. In turn, the committee votes on whether the proposed block should be appended to the blockchain or not. This process constitutes the core of the consensus mechanism and will be the focus of the present paper.

**Validators' strategies:** The set of strategies of a validator who has been selected in a committee will be denoted by  $S = \{-1, 1\}$ , where  $-1$  stands for rejecting and  $1$  for approving the proposed block. In particular,  $-1$  also corresponds to *not* casting a vote, either deliberately or accidentally. Accordingly, let  $X_{i,t}$  denote the indicator random variable

$$X_{i,t} = \begin{cases} 1, & \text{if validator } i \text{ voted on the approval of the proposed block } B_t \text{ in time slot } t \\ -1, & \text{otherwise} \end{cases}$$

We will write  $\mathbf{X}_t := (X_{1,t}, X_{2,t}, \dots, X_{n,t})$  to denote the random vector of all indicator random variables *prior to* the validators' voting decisions and  $\mathbf{x}_t := (x_{1,t}, x_{2,t}, \dots, x_{n,t}) \in \{-1, 1\}^n$  to denote the realized *decision* or *action* profile of the committee at time  $t \geq 0$ . Accordingly, we will write  $X := \{-1, 1\}^n$  to denote the set of all possible decision profiles.

**Decision rule:** A *decision rule*  $f : X \rightarrow \{-1, 1\}$ , also called *social choice function* or *aggregation of preferences rule*, is a function that receives as input the action profile  $\mathbf{x}_t$  and outputs a decision in  $\{-1, 1\}$ , where  $-1$  and  $1$  stand for disapproval and approval of the proposed block, respectively. We will focus on (*simple or enhanced*) *majority rules*  $f_q, q \in [0.5, 1]$  defined by

$$f_q(\mathbf{x}_t) := \begin{cases} 1, & \text{if } \sum_{i=1}^n x_{i,t} \geq (2q - 1)n, \\ -1, & \text{otherwise} \end{cases} \quad (1)$$

If at least a fraction  $q \in [0.5, 1]$  of the selected validators approve the proposed block, then this block is appended to the blockchain. Otherwise, the time slot remains empty and the mechanism progresses to the next time slot.

If block proposers follow the protocol and do not behave maliciously, then all proposed blocks are valid. Moreover, if the network is not partitioned and network latency is insignificant (lower than the time slot duration during which votes are expected to appear), then there is no controversy about which blocks are valid, since all validators view essentially the same blockchain (state), i.e.,  $\mathcal{B}_{i,t} := \mathcal{B}$  for all  $i \in I, t \geq 0$ . Under these conditions, the required majority should be reached and valid

<sup>2</sup>The mechanics of the pseudo-random mechanism vary between different protocols. Here, we are not interested in risks associated with manipulating this mechanism and focus on the mechanics of the voting process *once* a random committee has been formed.

blocks should be regularly approved and appended [30], [47]. However, in practice, two main reasons may lead to failures on the consensus mechanism

- adversarial behavior: a malicious node abstains from voting or censors other validators' votes to block the required majority and stall the block creation process.
- accidental behavior: validators drop offline accidentally or due to negligence, they experience bugs on client updates or bad network connectivity, their votes do not propagate through the network in the expected time slot or they are victims themselves of a censoring/eclipse attack.

Our goal is to study how existing results on optimizing aggregation of preferences in committees, in particular the results of [12], [42], [53], can be applied to the PoS blockchain setting and improve the underlying consensus mechanism. The application of these results will be immediate once we have defined the proper framework.

### III. AN IMPROVED VOTING RULE

To optimize the consensus process from an aggregative perspective, we quantify the collective benefits and losses (payoffs) from correct and wrong decisions respectively. This is done in Table I. The benefit from making a correct decision, i.e.,

		Proposed Block $B_t$	
		Valid (1)	Invalid (-1)
Committee	Approve	1	$-\ell_a$
	Reject	$-\ell_r$	1

TABLE I  
COLLECTIVE WELFARE FROM CONSENSUS OUTCOME.

approving a valid block or rejecting an invalid block, is scaled to 1. Here,  $-1$  denotes an invalid and  $1$  a valid block  $B_t$ , cf. Section VI. If a valid block is rejected, then a loss of  $\ell_r > 0$  is incurred which corresponds to the waste of computational resources and the failure of the system to process pending transactions. On the other hand, if validators vote for an invalid block, then  $\ell_a > 0$  represents the losses from validating a conflicting history<sup>3</sup>. Determining the exact values of  $\ell_r$  and  $\ell_a$  is a matter of protocol parametrization. In the present treatment, and without compromising the generality of the results, we will assume that  $\ell_r \ll \ell_a$ , i.e., that accepting invalid blocks has greater repercussions for the system, since such blocks typically stem from malicious – and not accidental – behavior. Finally, let  $\alpha \in (0, 1)$  denote the prior probability that a proposed block is *invalid*, e.g., blocks that an adversarial is trying to create.

Given the above, we seek to maximize the *expected collective welfare*  $E_t$  at time slot  $t > 0$ .  $E_t$  depends on the probabilities of accepting a valid block and of rejecting an invalid block under the decision rule  $f_q$ ,

$$\pi_1(f_q) := P(f_q(\mathbf{X}_t) = 1 \mid B_t = 1)$$

and  $\pi_{-1}(f_q) := P(f_q(\mathbf{X}_t) = -1 \mid B_t = -1)$ , respectively. Using this notation,

$$E_t(f_q) = (1 - \alpha)(1 + \ell_r)\pi_1(f_q) + \alpha(1 + \ell_a)\pi_{-1}(f_q) \quad (2)$$

where we have omitted constant terms. To estimate  $\pi_1(f_q)$  and  $\pi_{-1}(f_q)$ , and hence, to maximize  $E_t$ , we need to reason about the decision rule  $f_q$  and particularly, about the distribution of the decision variables  $X_{i,t}$ . Fortunately, this can be done by retrieving existing information about validators' past votes that have been stored as messages on the blockchain. This is captured by the notion of validators' *voting profiles* that we introduce next.

**Voting profiles:** Each validator  $i \in I$  is assigned a *score*  $p_{i,t} \in [0, 1]$  that corresponds to their *voting profile* at the start of time slot  $t$ . The value  $p_{i,t}$  can be thought of as validator  $i$ 's *decision ability* or *probability* that  $i$  will vote correctly, i.e.,

$$p_{i,t} := P(X_{i,t} = 1 \mid B_t = 1) = P(X_{i,t} = -1 \mid B_t = -1) \quad (3)$$

for  $i \in I, t \geq 0$ . For instance, in its simplest form,  $p_{i,t}$  can be thought of as the fraction of validators  $i$ 's correct votes to the number of slots in  $\{0, 1, \dots, t-1\}$  that  $i$  was selected in a committee. In what follows, we will examine a more elaborate scheme to define the profiles  $p_{i,t}$  that depends on the collective welfare of the consensus outcome, cf. Table I.

**Initializing & suspending profiles:** We will set a newly entering validator  $i$ 's voting profile at  $p_{i,0} := 0.5$  and will require that  $p_{i,t} \in [0.5, 1)$ , for any  $t \geq g$ , where  $g \geq 1$  denotes an initial grace period. If  $p_{i,t} < 0.5$ , for some  $t \geq g$ , then validator  $i$  will be suspended from  $I$ . The reasoning behind these choices is detailed in Section VI.

<sup>3</sup>This may include approval votes for a block that a malicious node is trying to create in order to double-spend or perform some other kind of attack. It may also involve votes that get wasted on blocks that will be subsequently reverted or abandoned.

**Updating scheme:** In general, an *updating scheme* is given by a function  $h : [0, 1] \times \{-1, 1\} \rightarrow [0, 1]$  which revises validator  $i$ 's voting profile after time slot  $t$  based on  $i$ 's prior voting profile  $p_{i,t}$ , the correctness of their voting decision  $x_{i,t}$  and the current state  $\mathcal{B}_t$ , i.e.,

$$p_{i,t+1} = h(p_{i,t}, x_{i,t}, \mathcal{B}_t)$$

for  $t > 0$ . The current state  $\mathcal{B}_t$  may include all relevant information, such as the collective welfare parameters, cf. Table I, the validity of the proposed block and the consensus outcome. If a validator  $i \in I$  has not been selected for slot  $t$ , then simply  $p_{i,t+1} = p_{i,t}$ . A concrete updating scheme that fits in this description is developed in Section III.

Given the introduced validators' voting profiles, we now return to the problem of maximizing the collective welfare  $E_t$ .

**Lemma 1.** For a selected validator committee  $N = \{1, 2, \dots, n\}$  at time slot  $t$ , we may condition on the vector of validators' voting profiles  $\mathbf{p}_t = (p_{1,t}, p_{2,t}, \dots, p_{n,t})$  and write the probability  $\pi_1(f_q | \mathbf{p}_t)$  of approving a correct block with the decision rule  $f_q$  as

$$\pi_1(f_q | \mathbf{p}_t) = \sum_{\mathbf{x}_t: f_q(\mathbf{x}_t)=1} \left( \prod_{i: x_{i,t}=1} p_{i,t} \prod_{j: x_{j,t}=-1} (1 - p_{j,t}) \right) \quad (4)$$

and similarly for  $\pi_{-1}(f_q | \mathbf{p}_t)$ .

*Proof.* This expression for  $\pi_1(f_q | \mathbf{p}_t)$  is derived as follows: The summation ranges over all action profiles  $\mathbf{x}_t$  for which the decision rule  $f_q$  approves the proposed block, i.e.,  $f_q(\mathbf{x}_t) = 1$ . The double product inside the parenthesis is precisely the likelihood of each of these profiles given that validators' decisions are independent. Specifically, the first product ranges over all validators  $i \in I$  who vote correctly, i.e.,  $i : x_{i,t} = 1$ , and multiplies each one's probability of a correct vote, i.e.,  $p_{i,t}$ , and the second ranges over the remaining validators  $j \in I$  who vote incorrectly, i.e.,  $j : x_{j,t} = -1$ , and multiplies each one's probability of voting incorrectly, i.e.,  $(1 - p_{j,t})$ .  $\square$

Given these expressions, the problem of maximizing  $E_t(f_q)$  can now be studied as an instant of the *committee-voting* models in [12] and [42], [53].

**Example 2** (Adjusted from [53]). Consider a committee of 5 validators with voting profiles, i.e., empirical probabilities of voting correctly,  $\mathbf{p} = (0.9, 0.9, 0.6, 0.6, 0.6)$ , as in [53]. In the unweighted case, or equivalently in the case in which all votes are weighted equally, and under the 2/3-majority decision rule  $f_{2/3}$  that is commonly used in consensus protocols [37], [48], the probability of reaching consensus on the correct block is equal to the probability that at least 4 out of the 5 validators vote correctly. This is

$$0.9^2 0.6^3 + 2 \cdot 0.6^3 (0.9) (0.1) + 3 \cdot 0.9^2 0.6^2 (0.4) \approx 0.56$$

which is lower even than the lowest voting profile of 0.6. A naive improvement would be to only consider the vote of the validator with the highest voting profile, i.e., of either the first or the second validator. This would increase the probability of correct voting to 0.9, however at a toll on decentralization.

In the naive improvement of the previous example, the vote of the best validator received a *weight* of 1 and the vote of all others a *weight* of 0. This raises the question of whether we can assign non-trivial *weights* (scale factors) to all validators and still improve the probability of a correct decision. The answer is affirmative and hinges on the notions of *weighted voting* and *weighted majority rules*.

**Weighted majority rule:** For a set of  $n$  validators, let  $\mathbf{w}_t := (w_{1,t}, w_{2,t}, \dots, w_{n,t})$  denote a vector of non-negative *weights* or *scaling factors*, with  $w_t := \sum_{i=1}^n w_{i,t}$ . The *weighted majority rule*,  $f_q(w)$ , or simply  $f_q$ , is defined as

$$f_q(\mathbf{x}_t) := \begin{cases} 1, & \text{if } \sum_{i=1}^n w_{i,t} x_{i,t} \geq (2q - 1) w_t, \\ -1, & \text{otherwise} \end{cases} \quad (5)$$

for  $q \in [0.5, 1]$ . If all votes are equally weighted, i.e., if  $w_{i,t} = 1$  for all  $i = 1, \dots, n, t > 0$ , then (5) reduces to (1).

Using this notation, our goal is to determine the weights  $\mathbf{w}_t$  and the weighted majority rule  $f_{\bar{q}_t}$  – or equivalently the quota  $\bar{q}_t$  – that optimize the collective welfare  $E_t$  given the selected committee  $N_t$  of validators at time slot  $t$ , i.e.,

$$\max_{q, \mathbf{w}_t} \{E_t(f_q, w_t)\} \quad (6)$$

This is the statement of the following Theorem which is due to [12] and in special instances due to [42], [53].

**Theorem 3** (Optimal Weighted Voting Scheme, [12]). Consider a committee  $N_t = \{1, 2, \dots, n\}$  of validators with voting profiles  $\mathbf{p}_t = (p_{1,t}, p_{2,t}, \dots, p_{n,t})$  that have been selected to vote on the proposed block in time slot  $t > 0$ . Then, given  $\alpha$  and the collective welfare parameters  $\ell_a, \ell_r$  in Table I, the decision rule that maximizes the collective welfare, cf. (6), is given by the weighted majority rule  $f_{\bar{q}_t}$ , with quota

$$\bar{q}_t := \frac{1}{2} \left[ 1 - \left( \ln \left( \frac{1 - \alpha}{\alpha} \right) + \ln \left( \frac{1 + \ell_r}{1 + \ell_a} \right) \right) w_t^{-1} \right] \quad (7)$$

and individual weights

$$w_{i,t} := \ln \left( \frac{p_{i,t}}{1 - p_{i,t}} \right), \quad \text{for } i = 1, 2, \dots, n. \quad (8)$$

*Proof.* To solve (6), we will control the decision rule  $f_q$  in (2). Then, the optimal quota and individual weights in (7) and (8) will follow naturally. To see this, we expand (2) using (4) which gives

$$\begin{aligned} E_t(f_q) &= (1 - \alpha)(1 + \ell_r) \pi_1(f_q) + \alpha(1 + \ell_a) \pi_{-1}(f_q) \\ &= (1 - \alpha)(1 + \ell_r) \sum_{\mathbf{x}_t: f_q(\mathbf{x}_t)=1} \left( \prod_{i: x_{i,t}=1} p_{i,t} \prod_{j: x_{j,t}=-1} (1 - p_{j,t}) \right) + \alpha(1 + \ell_a) \sum_{\mathbf{x}_t: f_q(\mathbf{x}_t)=-1} \left( \prod_{i: x_{i,t}=-1} p_{i,t} \prod_{j: x_{j,t}=1} (1 - p_{j,t}) \right) \\ &= \sum_{\mathbf{x}_t: f_q(\mathbf{x}_t)=1} \left( (1 - \alpha)(1 + \ell_r) \prod_{i: x_{i,t}=1} p_{i,t} \prod_{j: x_{j,t}=-1} (1 - p_{j,t}) \right) + \sum_{\mathbf{x}_t: f_q(\mathbf{x}_t)=-1} \left( \alpha(1 + \ell_a) \prod_{i: x_{i,t}=-1} p_{i,t} \prod_{j: x_{j,t}=1} (1 - p_{j,t}) \right) \end{aligned}$$

where we used that  $p_{i,t}$  denotes the probability of a correct decision, cf. (3). Our control variable to optimize the expected collective welfare in the previous equation is the decision rule  $f_q$ . It is immediate, that a general decision rule  $f$  which maximizes  $E_t$  is given by the following condition: for each decision profile  $\mathbf{x}_t \in X = \{-1, 1\}^n$  append every block for which

$$(1 - \alpha)(1 + \ell_r) \prod_{i: x_{i,t}=1} p_{i,t} \prod_{j: x_{j,t}=-1} (1 - p_{j,t}) > \alpha(1 + \ell_a) \prod_{i: x_{i,t}=-1} p_{i,t} \prod_{j: x_{j,t}=1} (1 - p_{j,t}) \quad (9)$$

and reject it otherwise. The rule compares the contribution of the decision profile  $\mathbf{x}_t$  to the expected collective welfare if it is accepted (left hand side) to its contribution if it is rejected (right hand side) and returns the outcome that yields the maximum contribution. This establishes its optimality. It remains to translate this rule into a form decision rule in the form of (5) from which we will be able to deduct the optimal quota and weights. By grouping similar terms, we can rewrite (9) as

$$\frac{1 - \alpha}{\alpha} \cdot \frac{1 + \ell_r}{1 + \ell_a} \prod_{i: x_{i,t}=1} \frac{p_{i,t}}{(1 - p_{i,t})} > \prod_{j: x_{j,t}=-1} \frac{p_{j,t}}{(1 - p_{j,t})}$$

and by taking the natural logarithm

$$\ln \left( \frac{1 - \alpha}{\alpha} \right) + \ln \left( \frac{1 + \ell_r}{1 + \ell_a} \right) + \sum_{i: x_{i,t}=1} \ln \left( \frac{p_{i,t}}{1 - p_{i,t}} \right) > \sum_{j: x_{j,t}=-1} \ln \left( \frac{p_{j,t}}{1 - p_{j,t}} \right)$$

where we used that the function  $\ln$  is monotone increasing. To proceed, we will use the transformations  $y_i := (x_i + 1)/2$ , so that

$$y_i = \begin{cases} 1, & \text{if } x_i = 1, \\ 0, & \text{if } x_i = -1 \end{cases}$$

and  $z_i := (1 - x_i)/2$ , so that

$$z_i = \begin{cases} 1, & \text{if } x_i = -1, \\ 0, & \text{if } x_i = 1 \end{cases}$$

Using this notation, we may rewrite the last inequality as

$$\ln \left( \frac{1 - \alpha}{\alpha} \right) + \ln \left( \frac{1 + \ell_r}{1 + \ell_a} \right) + \sum_{i=1}^n \left[ \ln \left( \frac{p_{i,t}}{1 - p_{i,t}} \right) y_i - \ln \left( \frac{p_{i,t}}{1 - p_{i,t}} \right) z_i \right] > 0$$

Now, since  $y_i - z_i = x_i$ , we obtain that

$$\sum_{i=1}^n \ln \left( \frac{p_{i,t}}{1 - p_{i,t}} \right) x_i > -\ln \left( \frac{1 - \alpha}{\alpha} \right) - \ln \left( \frac{1 + \ell_r}{1 + \ell_a} \right) \quad (10)$$

where the left hand side equals the left hand side of (5) with  $w_{i,t} := \ln\left(\frac{p_{i,t}}{1-p_{i,t}}\right)$ . This establishes (8). To obtain (7), it remains to bring the right hand side of this inequality in the form of (5). This is equivalent to determining  $\bar{q}_t$  so that the following equality holds

$$(2\bar{q}_t - 1) w_t = -\ln\left(\frac{1-\alpha}{\alpha}\right) - \ln\left(\frac{1+\ell_r}{1+\ell_a}\right)$$

Solving for  $\bar{q}_t$  yields (7). □

**Remark 4.** Based on the findings of Theorem 3 about the optimal quota and weights, cf. (7) and (8), we have the following remarks.

- The optimal quota (7) depends on the validators' profiles and hence, it may vary between different time slots  $t > 0$ . Also, it may vary according to the values of the parameters  $\alpha, \ell_r, \ell_a$ . For instance, the selection  $\alpha = 1/2$  neutralizes the bias due to assumptions on the distribution of valid versus invalid blocks whereas the selection  $\ell_r = \ell_a$  neutralizes the bias due to differences in perceived network costs/rewards. In this way, Theorem 3 maximizes the collective welfare – or equivalently, the probability of consensus on the correct decision – by an easily adaptable and *dynamic* decision rule.
- In blockchain applications, it may be desirable to enforce certain restrictions, as for example that the required weighted majority will be no less than  $2/3$  of the total weights or that each individual weight will be no less than some threshold value. As [53, p.332] explains, even in such cases of additional restrictions and/or perturbed assumptions, selecting weights that are proportional (or equal) to the optimal ones (8) will improve the probability of a correct outcome compared to unweighted decision making.
- For any voting profile  $p \in (0, 1)$ , the optimal weight  $w = \ln(p/(1-p))$  is given in Figure 1. For voting profiles  $p < 0.5$ , the

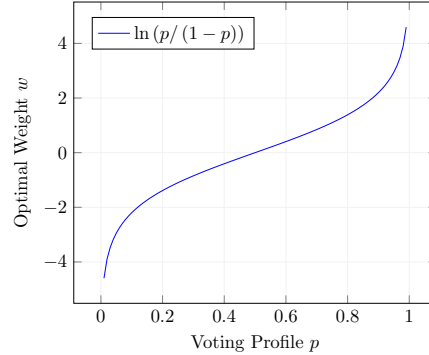


Fig. 1. Optimal weight  $w = \ln(p/(1-p))$  for any possible voting profile  $p \in (0, 1)$ .

weight is negative which motivated our selection of suspending validators with profiles less than 0.5. Intuitively, this means that these validators behave more than 50% of the time not as expected and hence have an adverse impact to the consensus mechanism. Such validators could be even replaced by a random coin (which would be correct 50% of the time). By properly initializing and suspending profiles above the 0.5 threshold, such mathematically trivial situations can be avoided. This is discussed in more detail in Section VI.

- On the other hand, the optimal weights increase steeply as the voting profiles approach 1. For instance, a validator with voting profile  $p_1 = 0.9$  has an optimal weight  $w_1 \approx 2.2$ , whereas a validator with voting profile  $p_2 = 0.99$  has an optimal weight  $w_2 \approx 4.6$ . In concrete terms, this means that a 10% improvement in the validator's voting profile resulted in a more than 100% improvement in the validator's optimal weight. In this way, the weighting mechanism disproportionately punishes even the slightest deviations from the optimal behavior and hence, incentivizes validators to behave correctly as consistently as possible. This feature can be utilized to enforce the socially desired outcome of almost zero failures in the consensus mechanisms.
- Finally, note that for applications, (5) can be simplified as follows. First, by dividing both sides with  $w_t$ , we can normalize the weights to sum up to 1, i.e.  $w'_{i,t} := w_{i,t}/w_t$  for  $i = 1, 2, \dots, n$ . Second, by letting  $y_i = (x_i + 1)/2$  as in the proof of Theorem 3, with  $y_i = 1$  if  $x_i = 1$  and  $y_i = 0$  if  $x_i = -1$ , we can rewrite (5) as

$$\sum_{i=1}^n w'_{i,t} (2y_i - 1) \geq 2\bar{q}_t - 1$$

or equivalently as

$$2 \sum_{i=1}^n w'_{i,t} y_i - \sum_{i=1}^n w'_{i,t} \geq 2\bar{q}_t - 1$$

Since  $\sum_{i=1}^n w'_{i,t} = 1$ , dividing both sides by 2 yields the approval condition

$$\sum_{i=1}^n w'_{i,t} y_i \geq \bar{q}_t \quad (11)$$

which is more handy than (but equivalent to) (5) and can be hence used for applications. This is done in Example 3 and Example 9 below.

**Example 3 (Continued).** Assume for simplicity that  $\alpha = 1/2$  and that  $\ell_r = \ell_a$ . In this case, the optimal rule is simple weighted majority, i.e.,  $\bar{q} = 1/2$ , or by substituting in (5),  $f_{\bar{q}}(\mathbf{x}) = 1$  if  $\sum_{i=1}^n w_i x_i \geq 0$  (dependence on  $t$  is omitted to simplify the notation). Using (8) and normalizing the weights to sum up to 1, we obtain  $\mathbf{w} = (0.392, 0.392, 0.072, 0.072, 0.072)$ . With these choices, the probability of approving a valid block is approximately 0.927 as shown in Table II. The weighting has

Profiles	Weights	Decision profiles $\mathbf{x}$ , with $f_{\bar{q}}(\mathbf{x}) = 1$ .						
0.9	0.392	1	1	1	1	-1	-1	-1
0.9	0.392	1	-1	-1	-1	1	1	1
0.6	0.072	-1,1	1	1	-1,1	1	1	-1,1
0.6	0.072	-1,1	1	-1,1	1	1	-1,1	1
0.6	0.072	-1,1	-1,1	1	1	-1,1	1	1
$\sum_{i=1}^5 w_i x_i$		> 0	> 0	> 0	> 0	> 0	> 0	> 0

TABLE II  
WINNING PROFILES UNDER THE OPTIMAL DECISION SCHEME.

resulted in a voting rule which approves a block if the two high-profile (0.9) validators agree on its validity (first column of decision profiles). The votes of the remaining validators come into play only if these two disagree. In this case, it suffices that a majority (2 out of 3) of the remaining validators approve the block (remaining 6 columns of decision profiles). The probabilities of decision profiles  $\mathbf{x}_t$  with  $f_{\bar{q}_t}(\mathbf{x}_t) = 1$  sum up to approximately 0.927 as claimed

$$0.9^2 + \binom{2}{1} 0.9 \cdot 0.1 \left( \binom{3}{3} 0.6^3 + \binom{3}{2} 0.6^2 \cdot 0.4 \right) \approx 0.927$$

The weights  $\mathbf{v}_t = (1/3, 1/3, 1/9, 1/9, 1/9)$  yield the same result and are in that sense, equivalent to the optimal ones. In fact, there may be several other optimal choices. As mentioned above, if we impose additional restrictions such as a de facto 2/3-weighted majority, the weights given by (8) may not be optimal anymore. However, as [53] remarks, they will still yield an improved probability compared to the unscaled case. In this example, a similar calculation as in Table II shows that the probability of reaching the 2/3-majority is 0.81 with the  $\mathbf{w}_t$  weights and approximately 0.85 with the  $\mathbf{v}_t$  weights.

The *ProcessSlot* procedure in Algorithm 1 executes the weighted voting algorithm in each slot  $t$ . First, the validators and a block proposer are selected according to the underlying PoS protocol (as part of the *PoSGetCommittee* procedure). The committee can be a random sample or deterministic subset taken from the entire validator set, or it can be the validator set in its entirety. For example, in Delegated Proof of Stake as implemented in TRON and EOS.IO, *PoSGetCommittee* would return the set of superdelegates. Next, the block proposed by the block proposer is retrieved and stored in the variable  $B$  – if no valid block has been proposed, then we assume that the value *null* is stored in  $B$ . Once the committee has been formed and a valid block has been proposed, we retrieve for each committee member the block that they voted for using the *CollectVotes* procedure. We then check in *ProcessVotes* whether the validators voted for  $B$  – if so, we record their vote as a 1, and as a  $-1$  otherwise. Next, the weighted voting procedure<sup>4</sup> is applied independent of the underlying PoS protocol – if a sufficient number of committee members voted for the  $B$ , then it is committed and appended to the blockchain. In this way, it contributes towards a more efficient and fair consensus mechanism while remaining decoupled from the PoS selection mechanism. This results in a two-layered scheme that on the one hand improves the efficiency of the consensus mechanism of the PoS protocol and on the other hand can be implemented and reverted with minimal cost to the users.

#### IV. MULTIPLICATIVE WEIGHTS UPDATE ALGORITHM

We now turn to one of the main challenges of implementing the voting profile scheme in the dynamic blockchain setting, which is the *update* of voting profiles after every time slot. The updating scheme may considerably vary depending on the desired result: e.g., in [59], a reputation system is proposed in which reputation increases according to a *sigmoid function* when

<sup>4</sup>The function that determines the optimal quota is given in a general form (line 26) to account for implementations with a rule different from the one proposed in (7), as e.g., a constant 2/3-majority rule.



---

**Algorithm 1** Weighted Voting in Committees

---

```
1: procedure PROCESSSLOT( $t, (p(i))_{i \in I}$ )
2:    $(v_1, \dots, v_n) \leftarrow \text{POSGETCOMMITTEE}(t)$ 
3:    $B \leftarrow \text{POSPROPOSEBLOCK}((v_1, \dots, v_n), t)$ 
4:   if  $B \neq \text{null}$  then
5:      $(x_1, \dots, x_n) \leftarrow \text{COLLECTVOTES}((v_1, \dots, v_n), B)$ 
6:      $(\text{vote}(1), \dots, \text{vote}(n)) \leftarrow \text{PROCESSVOTES}(t)$ 
7:     if  $\text{WEIGHTEDVOTING}((p(v_1), \dots, p(v_n)), (\text{vote}(1), \dots, \text{vote}(n)))$  then
8:        $\text{COMMITBLOCK}(B)$ 
9:   procedure  $\text{PROCESSVOTES}((x_1, \dots, x_n), B)$ 
10:  for  $i \in \{1, \dots, n\}$  do
11:    if  $x_i == B$  then ( $\triangleright$ )  $x_i = i$ 's vote message
12:       $\text{vote}(i) \leftarrow 1$ 
13:    else
14:       $\text{vote}(i) \leftarrow -1$ 
15:  procedure  $\text{WEIGHTEDVOTING}((p_i)_{i \in N}, (\text{vote}(i))_{i \in N})$ 
16:  for  $i \in \{1, \dots, n\}$  do
17:     $w_i \leftarrow \ln(p_i) - \ln(1 - p_i)$ 
18:     $\text{sum} \leftarrow \text{sum} + w_i \cdot \text{vote}(i)$ 
19:   $q = \text{OPTIMALQUOTA}((w_i)_{i \in N}, \alpha, \ell_r, \ell_a)$ 
20:  return  $\text{sum} \geq 2q - 1$ 
```

---

nodes vote correctly and decreases sharply (to 0) after a single violation. While this approach adheres to intuition and comes with certain merits, practical applications may call for more flexibility in the updates.

To develop a parametrizable scheme, we utilize the approach of [2] who generalize the standard *multiplicative weights update* (MWU) algorithm to a non-binary setting in which experts' scores are revised according to the impact of their decision on the social welfare. Using Table I, the corresponding MWU algorithm for the present application is given in Table III. Here,  $\delta > 0$

<b>Proposed Block <math>B_t</math></b>		
	Valid (1)	Invalid (-1)
<b>Committee</b>	Approve $p_{i,t}(1 + \delta)$	Reject $p_{i,t}(1 - \delta)^{\ell_a}$
	Reject $p_{i,t}(1 - \delta)^{\ell_r}$	Approve $p_{i,t}(1 + \delta)$

TABLE III  
MULTIPLICATIVE WEIGHTS UPDATES.

is a (small) number subject to the exact protocol parametrization. Apart from the efficiency properties of the MWU algorithm that are well known, see [2], [9], [44] and references therein, this scheme can leverage the prevailing network conditions and adjust the updates accordingly. This can be realized by replacing  $\delta$  and/or  $\ell_r, \ell_a$  with *sequences of updating parameters*,  $(\delta_t, \ell_{r,t}, \ell_{a,t})_{t > 0}$ . The proposed scheme for updating validators' voting profiles is given for completeness in Algorithm 2. If a new node comes online ( $T$  is the slot when it does), it runs the *MWInitialize* procedure to get the voting profiles consistent with the rest of the network. Once this has been completed, the function *MWUpdate* is executed every slot to update the voting profiles. If a validator's profile drops below 0.5, they are removed from the set – this happens in line 20. Similarly, new validators may be added in each round, depending on the protocol implementation. This processed in line 23. The *min* in line 14 ensures that the voting strength of a single validator cannot go to infinity (after all, if  $p(i)$  approaches 1 then  $w_i$  approaches  $\infty$  as per line 17 in Algorithm 1).

#### A. Tolerance of Validator Faulty Behavior

Before testing the weighted voting and MWU algorithms numerically in the Section V, we study analytically the effect of the updating scheme on the validators' profiles. Due to the parametric nature of the proposed algorithm, the question that naturally arises from the validators' perspective is the following: *what is the threshold of faulty behavior that can be sustained by the algorithm without harming my profile?* In other words, what is the *minimum* percentage of time that a validator should strive to behave properly in order to sustain or improve their voting profile.

To explore this question, we first introduce some notation. Let  $q \in [0, 1]$  denote the percentage of time – measured in time slots, cf. Section II – that a (tagged) validator behaves according to the protocol, i.e., votes as expected by default, and let

---

**Algorithm 2** Validators' Voting Profiles

---

```
1: procedure MWINITIALIZE( $I_0, \delta, \ell_r, \ell_a, T$ )
2:    $I \leftarrow I_0$ 
3:    $t \leftarrow 1$ 
4:   for  $i \in I$  do  $p(i) \leftarrow 0.5$ 
5:   while  $t \leq T$  do
6:      $(I, (p(i))_{i \in I}) \leftarrow \text{MWUPDATE}(I, (p(i))_{i \in I}, t, \delta, \ell_r, \ell_a)$ 
7:      $t \leftarrow t + 1$ 
8: procedure MWUPDATE( $I, (p(i))_{i \in I}, t, \delta, \ell_r, \ell_a$ )
9:    $(v_1, \dots, v_n) \leftarrow \text{POSGETCOMMITTEE}(t)$ 
10:   $B \leftarrow \text{POSPROPOSEBLOCK}((v_1, \dots, v_n), t)$ 
11:   $(x_1, \dots, x_n) \leftarrow \text{COLLECTVOTES}((v_1, \dots, v_n), B)$ 
12:  for  $i \in I$  do
13:    if  $v_i = B$  then
14:       $p(v_i) \leftarrow \min \{1 - \epsilon, p(v_i) (1 + \delta)\}$  ( $\triangleright$ )  $\epsilon = 10^{-5}$ 
15:    else if  $x_i = \text{null}$  then
16:       $p(v_i) \leftarrow p(v_i) (1 - \delta)^{\ell_r}$ 
17:    else
18:       $p(v_i) \leftarrow p(v_i) (1 - \delta)^{\ell_a}$ 
19:    if  $p(v_i) < 0.5$  and  $t \geq g$  then ( $\triangleright$ )  $g$  is the grace period
20:       $I \leftarrow I \setminus \{v_i\}$  ( $\triangleright$ ) suspend validator  $i$ 
21:   $(u_1, \dots, u_k) \leftarrow \text{GETNEWVALIDATORSINSLOT}(t)$ 
22:  for  $i \in \{1, \dots, k\}$  do
23:     $I \leftarrow I \cup \{u_i\}$ 
24:     $p(u_i) \leftarrow 0.5$ 
25:  return  $(I, (p(i))_{i \in I})$ 
```

---

$q_1 \in [0, 1]$  denote the percentage of time that this validator does not vote on valid blocks (e.g., because they accidentally drop offline when it is their turn to vote). Accordingly, it must be that  $q + q_1 \leq 1$ , with  $1 - q - q_1 \in [0, 1]$  denoting the percentage of time that the validator votes on the approval of invalid blocks. Finally, let  $p_0$  denote the starting profile of the validator at the period under scrutiny and for any time frame  $T > 0$ , let  $p_T$  denote the validator's profile at time  $T$ . Our aim is to obtain necessary and sufficient conditions on  $q$  and  $q_1$ , so that the validator will improve or sustain their initial voting profile, i.e.,  $p_T \geq p_0$ , where  $p_T$  depends on the validator's voting behavior via  $q$  and  $q_1$ . Typically, we will be interested in large time periods,  $T \gg 0$ , that properly reflect the validator's voting behavior as expressed by  $q$  and  $q_1$ . However, the result of Theorem 6 remains correct for any  $T > 0$ .

**Theorem 6.** Consider the updating scheme in Table III with parameters  $\delta, \ell_r, \ell_a > 0$  such that  $\ell_r \ll \ell_a$  and  $\delta \in (0, 1)$ , (typically  $\delta$  is a small number close to 0). A validator who follows the protocol  $q \cdot 100\%$  of the time and does not vote on valid blocks  $q_1 \cdot 100\%$  of the time, with  $q, q_1 \in [0, 1]$  and  $q + q_1 \leq 1$ , will improve upon their initial profile if and only if

$$q \geq c_1 (1 - c_2 q_1) \tag{12}$$

where  $c_1 := \left(1 - \frac{\ln(1+\delta)}{\ell_a \ln(1-\delta)}\right)^{-1}$  and  $c_2 := 1 - \ell_r/\ell_a$  are positive constants that depend on the updating parameters  $\delta, \ell_r$  and  $\ell_a$ .

*Proof.* The positivity of  $c_1$  and  $c_2$  follows directly from the assumptions. Indeed,  $\ell_r < \ell_a$  implies that  $\ell_r/\ell_a < 1$  and hence, that  $c_2 > 0$ . Similarly,  $\delta \in (0, 1)$  implies that  $\ln(1-\delta) < 0$  and  $\ln(1+\delta) > 0$  and hence, that  $c_1 > 0$  as the inverse of the sum of two positive terms (in particular,  $c_1 < 1$ ).

Concerning (12), let  $p_0$  denote the validators initial voting profile and let  $p_T$  denote the validator's voting profile after  $T$  time slots. Then, given that  $q$  denotes the fraction of time slots in which the validator voted correctly and  $q_1$  the fraction of time slots in which the validator did not approve a valid block, we have that

$$p_T = (1 + \delta)^{qT} \left( (1 - \delta)^{\ell_r} \right)^{q_1 T} \left( (1 - \delta)^{\ell_a} \right)^{(1 - q - q_1)T} p_0 = \left( (1 + \delta)^q (1 - \delta)^{(\ell_r q_1 + \ell_a (1 - q - q_1))} \right)^T p_0$$

Hence,  $p_T \geq p_0$  if and only if  $\left( (1 + \delta)^q (1 - \delta)^{(\ell_r q_1 + \ell_a (1 - q - q_1))} \right)^T \geq 1$ . Since, the basis term is positive by assumption, we may take the natural logarithm of both sides to obtain the equivalent condition

$$q \ln(1 + \delta) + (\ell_r q_1 + \ell_a (1 - q - q_1)) \ln(1 - \delta) > 0$$

Using that  $(1 + \delta) > 1$  and some standard algebraic operations, we can solve the last inequality for  $q$  to obtain the necessary and sufficient condition

$$q \geq \frac{1}{1 - \frac{\ln(1 + \delta)}{\ell_a \ln(1 - \delta)}} \cdot (1 - q_1 (1 - \ell_r / \ell_a))$$

which concludes the proof.  $\square$

**Remark 7 (Policy Implications).** The result of Theorem 6 can be utilized in the design policy of the updating algorithm to enforce certain requirements in the validators' realized correct behavior,  $q$ . To see this, we first observe that  $c_1$  can be rewritten as

$$c_1 = \ell_a \cdot \left( \ell_a - \frac{\ln(1 + \delta)}{\ln(1 - \delta)} \right)^{-1}$$

Since  $\delta$  is typically a small positive number, e.g.  $\delta = 10^{-3}$  and  $\ell_a$  is a positive number large enough to represent the potential losses from appending an invalid block to the blockchain,  $c_1$  gets arbitrary close to, but strictly less than 1 for increasing values of  $\ell_a$ . Moreover, for large values of  $\ell_a$  and small values of  $\ell_r$ ,  $c_2 := 1 - \ell_r / \ell_a$  is also close to, but strictly less than 1. This implies that  $1 - \varepsilon < c_1 c_2 < 1$  for some relatively small  $\varepsilon > 0$ . Hence, using that  $q + q_1 \leq 1$ , we may rewrite (12) as

$$q + c_1 c_2 q q \geq c_q \iff (1 - c_1 c_2) q + c_1 c_2 (q + q_1) \geq c_1 \implies (1 - c_1 c_2) q + c_1 c_2 \geq c_1 \implies q \geq \frac{c_1 - c_1 c_2}{1 - c_1 c_2}$$

This gives a lower bound on  $q$  that does not depend on  $q_1$ . It is obvious that if the parameters  $\delta, \ell_r, \ell_a$  are selected in a way to push  $c_1$  close to 1, then a validator will have to vote consistently correctly to retain or improve their voting profile. In general, this shows the advantages of the parametric approach and the flexibility that it conveys to the policy makers or consensus designers (in case of public, permissionless blockchains) to enforce desired outcomes or even to dynamically adapt the consensus algorithm in response to prevailing network conditions.

## V. NUMERICAL TESTS & PRACTICAL USE CASES

To visualize the proposed scheme, we study some instantiations of the weighted voting and MWU algorithms. Before going into the details of each case, the following hold in general.

- **Adversarial model:** an adversary blocks  $v\%$  of the votes, either by abstaining (accidentally or intentionally) or by censoring other validators. To demonstrate the efficiency of the model in extreme – or worst case – conditions, we show simulations for  $v = 40\%, 50\%$  and  $60\%$ . For lower values of  $v$ , i.e., for less adversarial power, the results are similar (better) and thus omitted.
- **Parameter choice:**  $\alpha$ , the prior probability that a proposed block is invalid, is set to  $1/2$ . This choice neutralizes the bias due to assumptions on the distribution of valid-invalid blocks in (7) and corresponds to the most general model. To capture that costs from approval of invalid blocks are higher than costs from rejection of valid blocks, we select  $\ell_a = 12 \gg 10^{-2} = \ell_r$ . The resulting graphs (recovery times) are robust in a wide range of these parameters. Yet, very low values of  $\ell_a$  may lead to unwanted behavior: validators who are commonly off-line may still improve their profiles despite not voting on valid blocks. Finally, the updating parameter  $\delta$  has been chosen according to the related literature [2], [9]. Different values of  $\delta$  affect the rate of convergence of the profiles.
- **Simulation environment:** The numerical results have been established in MATLAB R2018b. The simulations validate the algorithms for resuming consensus, cf. Figures 2 and 3 and for updating a validator's profile, cf. Figures 4 and 5. Further issues related to scalability and computational complexity on a proper – or simulated – blockchain are discussed in Section VI. Figure 2 illustrates a scenario with an adversary blocking 40% of the votes. At the start of the attack, all  $n$  validators<sup>5</sup> have a voting profile  $p = 0.9$ . We examine two choices of the updating parameter  $\delta = 10^{-3}$  (red) and  $\delta = 2 \times 10^{-3}$  (blue). In both cases,  $\alpha = 1/2$  and  $\ell_r = 10^{-2}, \ell_a = 12$ . The depicted curves indicate that the weighted approval votes (vertical axis) rise above the  $2/3$  majority threshold<sup>6</sup> for both cases. The pace is different and depends on the selection of  $\delta$ . The sharp bends in both lines correspond to the point in which the scores of voting validators numerically reach 1. After this point, the majority of the voting validators increases at a very slow pace which is a desirable property that allows for a more smooth recovery in case that the abstaining 40% resume voting. Specifically, the exact formula for updating their profiles is

$$p_{i,t+1} = \min \{ 1 - 10^{-15}, \max \{ 0.5, p''_{i,t+1} \} \}$$

<sup>5</sup>The exact number does not change the results. For simplicity, we used  $n = 100$ .

<sup>6</sup>While the optimal quota, cf. (7), remains slightly above  $1/2$ , we consider the  $2/3$ -majority rule which is currently implemented in PoS protocols.

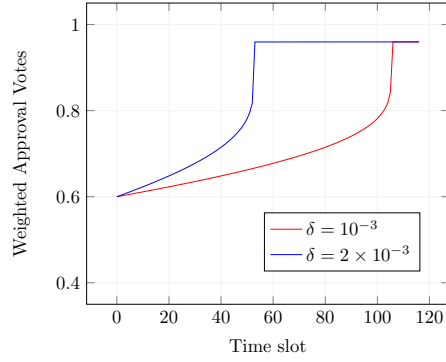


Fig. 2. Time slots to resume consensus on valid blocks with 40% non-voting nodes under mild (red line) and aggressive (blue line) updating parameter  $\delta$ . In both cases,  $\ell_r = 10^{-2}$ .

where  $p''_{i,t+1}$  is the value calculated by Table III. This formula ensures that  $p_{i,t}$  remains in  $[0.5, 1)$ .

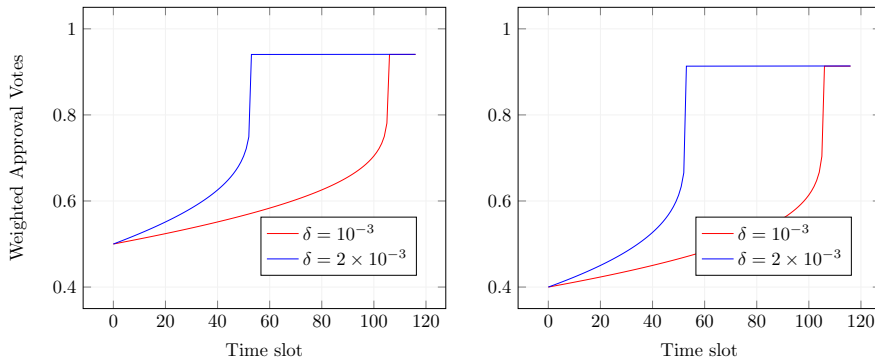


Fig. 3. Time slots to resume consensus on valid blocks with 50% (left panel) and 60% (right panel) non-voting nodes under mild (red line) and aggressive (blue line) updating parameter  $\delta$ . In both cases,  $\ell_r = 10^{-2}$ .

**Remark 8.** To avoid the numerical instability at  $p = 1$  – for which the denominator at  $\ln(1/(1-p))$  becomes equal to 0 – we need to restrict profiles away from 1. Specifically, to implement and test this scheme, we require that voting profiles do not exceed  $1 - \epsilon$  for some very small  $\epsilon > 0$ . However, due to the steep increase of the optimal weights that was mentioned in Remark 4, the actual value of the imposed threshold  $\epsilon$  matters and it should be chosen to be numerically close to 0. Moreover, again due to the disproportional increase of the optimal weights for profiles close to  $1 - \epsilon$ , validators who achieve this threshold (and remain at it by continuing to vote correctly after reaching it), have disproportionately higher power to influence the consensus outcome.

As a comparison, Figure 3 illustrates the process of resuming approval of blocks in the same scenario but with an adversary controlling 50% of the stake (left panel) and 60% of the stake (right panel). The results indicate a very similar recovery pattern, cf. Figure 2, independently of the initial stake of the non-voting validators. The evolution of a validator’s voting profile is illustrated in Figure 4. In the depicted scenario, the validator’s initial profile is 0.9. The validator votes correctly 80% of the time, but drops 10% of the time off-line, and votes on invalid blocks another 10% of the time. Again, we examine two cases for different values of the update parameter,  $\delta = 2 \times 10^{-2}$  (left panel) and  $\delta = 10^{-2}$  (right panel). While the patterns differ, in both depicted panels the voting profile falls due to the regular incorrect votes. We remark, that lower values of  $\ell_a$  would result in upwards sloping curves (not depicted here) implying that a validator could regularly vote incorrectly and still improve their voting profile. Similarly, higher values of  $\delta$  would allow validators to quickly recover their profiles after pitfalls which is an undesirable property. The depicted patterns in the evolution of the voting profile are robust in the choice of the initial score and the value of  $\ell_r$ .

Finally, Figure 5 extends the above scenarios to a period in which the validator resumes proper voting. Specifically, we assume that the validator votes correctly on every block except of occasional time slots – less than 10% of the time – in which they go offline. Again, the two panels correspond to different values of  $\delta$ . In both cases, the pattern is linear (the sharp bends correspond to the occasional drops) with a slope that can be adjusted by the choice of  $\delta$ . In sum, the simulations support the versatility of the proposed scheme and leave the exact parametrization (static or dynamic) subject to each protocol’s implementation and scope.

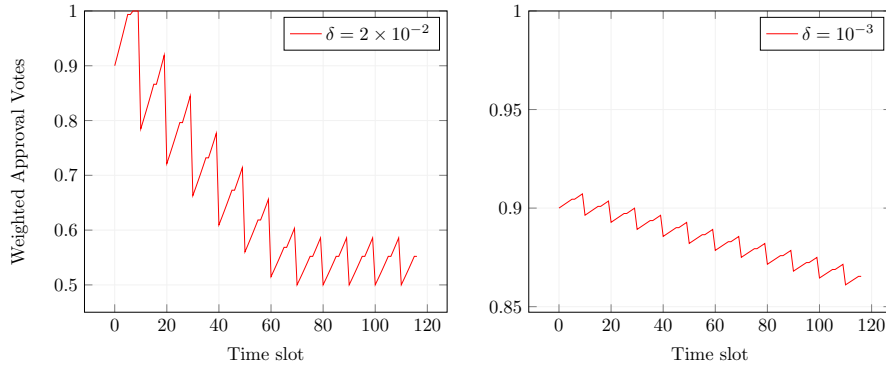


Fig. 4. Evolution of a validator’s voting profile who periodically drops offline and periodically votes on an invalid block for different values of parameter  $\delta$ . In both panels, the validator’s initial score is 0.9,  $\ell_a = 12$  and  $\ell_r = 10^{-2}$ .

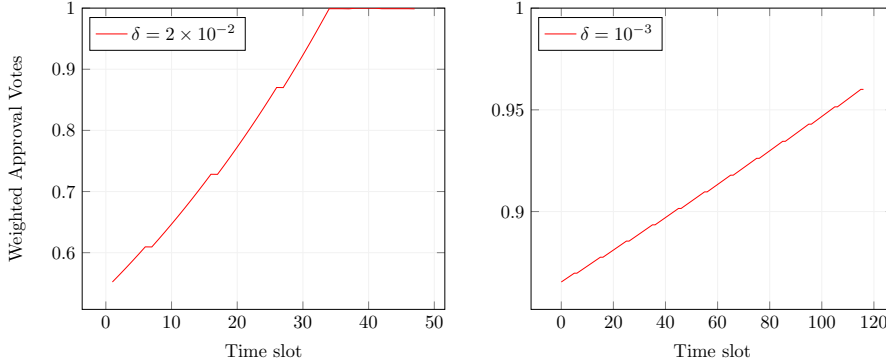


Fig. 5. Recovery of the validator’s voting profile after resuming proper voting (with only occasional offline-drops) in the scenarios of Figure 4. Recovery exhibits linear pattern for both choices of  $\delta$ . It is fast for  $\delta = 2 \times 10^{-2}$  (aggressive adjustment) and slow for  $\delta = 10^{-3}$  (mild adjustments).

### A. Use Case: Ethereum Blockchain with Casper FFG

To further test the proposed scheme in a potential practical scenario, we consider the implementation of weighted voting in Ethereum’s Casper the Friendly Finality Gadget (FFG) as a PoS overlay on top of the PoW main chain [21]. This is a hybrid consensus system in which, roughly, PoS validators – stakers – vote to confirm (validate) blocks that have been proposed by “conventional” PoW miners at regularly-occurring *checkpoints*. To become a validator, a user has to *deposit* a chosen amount of ETH, Ethereum’s native cryptocurrency. If more than two thirds of the validators (where each validator is weighted proportionally to the size of their deposit) vote to approve a given checkpoint block, then it is considered *justified*, and if two checkpoint blocks in a row are justified then the first block is considered *finalized*. Ethereum nodes will never drop finalized blocks during a chain reorganization, so finalization provides an additional layer of security to the users. If more than one third of the validators go offline or are subjected to a network partition or eclipse attack, then checkpoints cannot be finalized. The system eventually recovers in the following way: after each checkpoint, those validators who did not vote are *penalized* which means that their fraction of the total deposit decreases. Eventually, their deposit fraction will fall below one third – hence, the voting validators regain a two-thirds supermajority and the ability to finalize. Of course, it would be harsh on the non-voting validators if their deposits were shrunk quickly during a network partition or eclipse attack, which would by itself disincentivize potential validators. The penalties are therefore initially very mild, but they become harsher over time. This is illustrated in Figure 6.

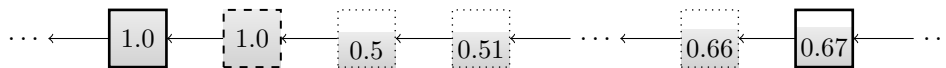


Fig. 6. Illustration of Casper FFG. The boxes represent checkpoint blocks, and each value inside a box represents the fraction of validators that voted for that checkpoint. Thick edges around a box indicate that the checkpoint is finalized, dashed edges indicate mere justification, and a dotted edge a checkpoint that is neither. If a considerable fraction (in this case 50%) of validators suddenly go offline, then finalization stalls. In Casper FFG, the protocol is eventually recovered by gradually slashing the deposits of offline validators. Using weighted voting, this can be achieved without causing financial losses to the validators. For more details of the liveness of the Hybrid Casper proposal see [21].

The main contribution of the currently proposed weighted voting scheme in Ethereum’s Hybrid Casper Protocol setting is

that it enables the protocol to recover without hurting the deposits of the validators. Instead, validators who come back online eventually regain their voting profile without any financial losses. This also allows for faster recovery during network partitions: although network partitions that last for more than a day are extremely rare, a quick recovery that is achieved by slashing the deposits of offline validators may discourage potential validators from depositing.

The practical advantages of weighted voting are achieved without affecting either the underlying PoW blockchain or the PoS checkpoint finalization protocol. Instead, it operates within the selected committees and hence affects in a minimal way any suggested/existing scheme that is functional on the blockchain. This means that this scheme can be launched and reverted with minimal impact on client implementations and updates.

### B. Use Case: Incentives and Risk Mitigation in Proof of Stake Protocols

Several current PoS proposals including Ouroboros [18], [33], Casper FFG [19], [20], and Delegated Proof of Stake (DPOS) as implemented in EOS.IO, suggest the following properties.

**P1:** a limited number of pools or committee members, which is achieved through a minimum deposit for each potential validator.

**P2:** a fixed period (number of protocol epochs) of time for which the staked deposit will remain locked.

As we discuss in the following, weighted voting and updating schemes become particularly relevant under these conditions. Up to now, we have assumed that the PoS reward mechanism is retained, i.e., that validators are rewarded proportionally to their stake. However, if there are large discrepancies between the voting profiles, validators with high voting profiles contribute more to the consensus than validators with lower profiles. This may raise issues about the distribution of staking rewards and the adopted reward scheme.

To deal with such issues in a similar setting of decentralized multi-agent decision-making, [7] and [3] study incentives under the conventional reward scheme that gives to the participating agents their Shapley [52], [54] or Banzhaf values [11], [25]. In blockchain applications, the relevant question to address is whether the decision scheme together with the reward scheme create incentives for validators to merge, split or annex their deposits to larger pools [3]. [7] quantify the potential profit from creating various identities to participate in the weighted voting mechanism and show that given the number  $n$  of participating agents, a Sybil attack cannot earn to a potential adversary more than  $2n/(n+1) \approx 2$  times their initial values. Moreover, both [7] and [3] demonstrate that it is computationally difficult, i.e., NP-hard, to find such profitable manipulations (mergers or splitters).

These findings are promising and suggest that reward schemes that have been developed in the context of traditional coalitional games can be used to incentivize proper behavior in the consensus mechanisms of blockchain protocols. By using properties P1.-P2. as above, we significantly mitigate the related risks. Indeed, setting a number of allowed validating pools (P1) together with a large enough minimum required deposit increase the difficulty for splitters. Yet, taking these precautions to the other extreme takes its toll on decentralization and hence such measurements should be exercised to a limited extent. In any case, the fixed period for which deposits remain locked (P2), seems to significantly reduce the potential for movements in the short run at no significant tradeoffs with other blockchain protocol design principles.

*Eclipse Attacks:* The weighted voting scheme is engineered to accelerate recovery of consensus and prevent the blockchain from stalling when a fraction of validators is off-line or in general does not vote. Validators that continue to vote increase their voting profiles and hence, their power to influence the consensus outcome, whereas the voting profiles of non-voting validators deteriorate. Furthermore each validator's optimal weight depends only on that validator's voting behavior (or profile) and hence, can be computed independently of the committee in which this validator participates. However, this is not true for the *relative power* of each validator in each committee. An example is given below.

**Example 9.** Consider a validator  $i = 1$  with profile  $p_{1,t} = 0.99$  and two different committees:

**Committee in  $t = 1$ :**  $n = 10$  validators with voting profiles,  $\mathbf{p}_1 = (0.99, 0.7, \dots, 0.7)$ ,

**Committee in  $t = 2$ :**  $n = 10$  validators with voting profiles,  $\mathbf{p}_2 = (0.99, 0.95, \dots, 0.95)$ .

As in the numerical tests, we will assume that  $\ell_r = 10^{-2}$ ,  $\ell_a = 12$  and  $\alpha = 1/2$  for both committees. Applying (7) on these values yields the optimal quotas  $\bar{q}_1 = 0.604$  for  $t = 1$  and  $\bar{q}_2 = 0.54$  for  $t = 2$ . The unnormalized optimal weight of validator 1 is equal to  $w_1 = \ln(0.99/(1 - 0.99)) = 4.595$  in both committees. However, their normalized weights  $w'_{i,t}$ ,  $t = 1, 2$  which will be used to assess the block approval condition (11), are equal to  $w'_{1,1} = 0.376$  and  $w'_{1,2} = 0.148$  respectively. This shows that validator 1 is better off in the first committee.

To reach this conclusion, the second committee was selected with much worse – in terms of their voting profiles – validators. However, similar discrepancies in validator 1's optimal normalized weight can be observed even in the presence of only 2 equally good validators.

**Committee in  $t = 3$ :**  $n = 10$  validators with voting profiles,  $\mathbf{p}_1 = (0.99, 0.99, 0.95, 0.7, \dots, 0.7)$ .

In this case, the optimal quota  $\bar{q}_3$  is equal to  $\bar{q}_3 = 0.57$  and validator 1’s optimal normalized weight is equal to  $w'_{1,3} = 0.254$ . Similarly, validator 2’s optimal normalized weight  $w'_{2,3}$  is also equal to 0.254 and validator 3’s optimal normalized weight is equal to 0.163. This highlights the power of these 3 validators to decide the consensus outcome in committee 3.

The previous example reveals two interesting facts. On the one hand, the dynamic adjustments in the optimal quota and validator’s normalized weights create a consensus mechanism with intriguing properties when compared to static alternatives. The versatility of this algorithm under the concurrent theoretical guarantees that it optimizes the probability of a correct decision, or more accurately the expected collective welfare, motivate further research both from theoretical and practical perspectives concerning its adoption.

On the other hand, the dependence of the validators’ *relative power* on the profiles of the other committee members creates an incentive for malicious behavior that should be taken into account. Specifically, since validators’ relative power increases as the profiles of the other committee members decrease, this motivates adversarial validators to perform *eclipse attacks* and prevent other validators from voting properly. However, for such attacks to be profitable, the attackers need to sustain their profile which means that they need to keep voting properly. Moreover, the actual outcome on the overall collective welfare will depend on the trade-off between the size of the attacker’s stake and the size of the eclipsed validator’s stake along with detection and potential penalties on such behaviors. In any case, fixing parameters as in P1-P3 seems to reduce the arsenal of potential attackers but a precise verdict on whether such attacks can be profitable depends on the exact protocol parametrization and should be examined separately in every case of adoption.

## VI. DESIGN & LIMITATIONS

The introduction of validators’ voting profiles and the improvement in the consensus mechanism come with a trade-off in terms of security. Since the system becomes dependent on information that can be retrieved from the blockchain – validators’ votes are stored as messages [20] – this raises new risks on adversarial manipulation of this information. In the current section, we try to address these risks alongside implementation issues and limitations.

*Defense against known attacks:* To defend against consensus level attacks, current PoS proposals leverage the fact that non-voting nodes can be identified and penalized [5], [19]. Yet, these actions are ineffectual against adversaries who can censor other validators or replenish their own stake and withstand the penalties to retain more than the required consensus-quota of the total stake [30], [47]. Although pessimistic, the scenario in which adversaries sustain an attack despite suffering losses on their own stake gains credence in the presence of potential *out-of-protocol* profits. Further, consensus mechanisms that rely on PoS selection are vulnerable to *flash* or *blindsiding* attacks conducted by entering nodes [15], [23] or to accidental faults such as network latency, bad connectivity or simple negligence. Weighted voting provides lines of defense (in an obvious way) against these kinds of attacks or faults while retaining the benefits of the underlying PoS design. In addition, the preserved reliance on PoS for the selection of validators in committees and the allocation of rewards, protects against adversarial nodes that maintain small stake but high voting profile or vice versa.

*Valid-invalid blocks:* A likely contentious assumption of the present model is that proposed blocks can be identified as valid or invalid<sup>7</sup>. In practice, different nodes may have different views of the blockchain and hence perceive proposed block differently. Yet, on closer inspection, this assumption can still be supported in the current framework: if a node is honest but has a (completely) different view of the canonical chain due to (say) poor connection to the network, then their votes do not contribute to extending the canonical chain and indeed can be regarded as incorrect. For instance, in Ethereum, which is the base case for this paper, *valid-invalid* votes are well-defined and identified [49], [50].

*Updating scheme & Computational Complexity:* Dealing with the issue of valid-invalid blocks becomes more relevant in the design of the updating scheme. Clearly, faults that can be identified as deliberate should be treated differently than accidental ones. For instance, a validator who has honestly participated in the protocol for a long period of time and drops offline for a short period of time, should be able to quickly recover their previous voting profile. This motivates updating profiles by two variables  $s_{i,t}$  and  $l_{i,t}$  representing *short-term* and *long-term* voting respectively. In general, the advantages of the here employed generalized MWU algorithm – e.g., convergence rates and tight bounds on its overall performance [2] – can be further exploited alongside stability properties of opinion dynamics in networks [40] to yield more robust results also in the present context. Finally, the computational complexity of initializing, storing and updating validators’ profiles does not exceed the complexity of performing the same functions on validators’ stakes and hence it is not expected to have a significant impact on the overhead of any consensus that keeps track of validators’ stakes. This intuition is even more likely to materialize within the framework of most state of the art proposals, like EOS.IO, Casper and Ouroboros that promote consensus systems with a limited number delegates, staking pools or potential validators.

<sup>7</sup>This is a chicken-egg problem: if we can identify the “canonical” blocks, then we can improve consensus with a scheme as the one proposed here. But in blockchains, the “canonical” blocks are precisely the ones for which consensus was reached.

*Entry & threshold voting profiles:* The levels at which voting profiles are initialized and suspended are crucial, since they can incentivize or prevent *Sybil attacks*. The exact parametrization can be case-dependent, justified by simulations or incorporate prior information for each entering validator, e.g., reputable financial institution versus unknown private staker. The present choice to initialize voting profiles at 0.5 and to require that they remain in  $[0.5, 1)$  for all  $t \geq g > 0$ , where  $g$  denotes a potential initial grace period, is supported by both intuitive and theoretical arguments. From a mathematical perspective, the initial choice of 0.5 represents an uninformative prior on an entering validator's voting profile. Similarly, the reason for the upper bound is purely numerical, namely to avoid the instability in  $\ln(p_{i,t}/(1-p_{i,t}))$  if  $p_{i,t} = 1$ . In contrast, the suspension of validators with voting profile – or probability of making a correct decision – lower than 0.5 is more fundamental. [12] provide a detailed probabilistic argument to explain that such voters are harmful to consensus and their votes should not be considered. Moreover, in the specific context of distributed networks, allowing nodes to participate with scores lower than their initial one triggers *Sybil attacks*, since it motivates switching to new or creating multiple accounts. Finally, suspending validators in terms of their voting behavior relaxes the need for frequent economic penalties [20]. This makes the PoS ecosystem more secure and appealing to investors who would otherwise be concerned of suffering losses due to accidental misbehavior, e.g., dropping off-line or being censored.

*Future implementation:* Currently, the proposed scheme seems more attractive for systems with low numbers of staking nodes: these may be permissioned and delegated PoS blockchains or blockchains with fixed number of stake pools. More closely related to this spirit are the proposals of [1], [19], [33], [39]. In these settings, the computational complexity of implementing a profiling system is not an issue. However, this is also expected to remain true in the general case of permissionless blockchains, since extra data storage is limited to one additional value per validator and computations to update profiles are linear in the size of the committees. Light on this issues will be shed by future implementations on simulated blockchains and if successful, on properly designated testnets of these blockchains. In the core of these studies will be the understanding of issues related to network conditions (latency, scalability), computational complexity to store and retrieve profiles and the testing of different updating schemes in terms of their convergence rates and efficiency bounds.

## VII. SUMMARY & CONCLUSIONS

Existing PoS protocols select staking nodes proportionally to their stake to form block-creating committees. Yet, they do not guarantee that selected committees will create blocks, since consensus may fail due to accidental or adversarial behavior. Thus, the perceived fairness in the distribution of rewards in proportion to the stake of participating nodes is actually violated. Motivated by this observation, we studied *weighted voting* as a way to improve the consensus mechanism. We introduced validators' voting profiles – that quantify the probability that a validator will cast a correct vote based on their so far contribution to the protocol – and defined the proper mathematical framework to apply the results of [12] on optimal decision rules in committee voting. Using the approach of [2], we designed a multiplicative weights algorithm to update individual validator's profiles according to their voting behavior, the consensus outcome and collective blockchain welfare. The result is a two-layered scheme in which selection of nodes and allocation of rewards are performed by the underlying PoS mechanism whereas blocks are decided by a weighted majority voting rule. This scheme improves consensus *within* selected committees by scaling votes according to validators' profiles without interfering with the PoS execution. Hence, it can be tested, implemented and reverted with minimal cost to existing users. On the negative side, the introduction of a profiling scheme in a distributed network raises new risks associated with manipulation of the relevant information. We discussed such risks along with actions that should be considered in the design of future PoS protocols.

## REFERENCES

- [1] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, pages 30:1–30:15, New York, NY, USA, 2018. ACM.
- [2] S. Arora, E. Hazan, and S. Kale. The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. *Theory of Computing*, 8(6):121–164, 2012.
- [3] H. Aziz and M. Paterson. False Name Manipulations in Weighted Voting Games: Splitting, Merging and Annexation. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '09, pages 409–416, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.
- [4] H. Aziz, M. Paterson, and D. Lee. Efficient Algorithm for Designing Weighted Voting Games. 2007 IEEE International Multitopic Conference, pages 1–6, Dec 2007.
- [5] S. Azouvi, P. McCorry, and S. Meiklejohn. Betting on Blockchain Consensus with Fantomette. *preprint arXiv:1805.06786*, 2018.
- [6] Y. Azrieli and S. Kim. Pareto Efficiency and Weighted Majority Rules. *International Economic Review*, 55(4):1067–1088, 2014.
- [7] Y. Bachrach and E. Elkind. Divide and Conquer: False-name Manipulations in Weighted Voting Games. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '08, pages 975–982, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.
- [8] Y. Bachrach, J. Rosenschein, and E. Porat. Power and Stability in Connectivity Games. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '08, pages 999–1006, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.



- [9] J. P. Bailey and G. Piliouras. Multiplicative weights update in zero-sum games. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, EC '18, pages 321–338, New York, NY, USA, 2018. ACM.
- [10] S. Bano, A. Sonnino, M. Al-Bassam, S. Azouvi, P. McCorry, S. Meiklejohn, and G. Danezis. Consensus in the age of blockchains. *preprint arXiv:1711.03936*, 2017.
- [11] J. Banzhaf. Weighted voting doesn't work: A mathematical analysis. *Rutgers Law Review*, 19:317–343, 1965.
- [12] R. C. Ben-Yashar and S. I. Nitzan. The Optimal Decision Rule for Fixed-Size Committees in Dichotomous Choice Situations: The General Result. *International Economic Review*, 38(1):175–186, 1997.
- [13] I. Bentov, A. Gabizon, and A. Mizrahi. Cryptocurrencies without proof of work. In J. Clark, S. Meiklejohn, P. Y. Ryan, D. Wallach, M. Brenner, and K. Rohloff, editors, *Financial Cryptography and Data Security*, pages 142–157, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
- [14] I. Bentov, R. Pass, and E. Shi. Snow white: Provably secure proofs of stake. Cryptology ePrint Archive, Report 2016/919, 2016. <https://eprint.iacr.org/2016/919>.
- [15] J. Bonneau. Why Buy When You Can Rent? *Financial Cryptography and Data Security*, pages 19–26, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
- [16] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. 2015 IEEE Symposium on Security and Privacy, pages 104–121, 2015.
- [17] J. Brown-Cohen, A. Narayanan, A. Psomas, and S. M. Weinberg. Formal barriers to longest-chain proof-of-stake protocols. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, EC '19, pages 459–473, New York, NY, USA, 2019. ACM.
- [18] L. Brünjes, A. Kiayias, E. Koutsoupias, and A.-P. Stouka. Reward Sharing Schemes for Stake Pools. *preprint arXiv:1807.11218*, 2018.
- [19] V. Buterin. Ethereum 2.0 spec – Casper and sharding. Available [online]. [Accessed: 30-10-2018], 2018.
- [20] V. Buterin and V. Griffith. Casper the Friendly Finality Gadget. *preprint arXiv:1710.09437*, 2017.
- [21] V. Buterin, D. Reijnders, S. Leonardos, and G. Piliouras. Incentives in ethereum's hybrid casper protocol. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, ICBC 2019, pages 236–244, May 2019.
- [22] C. Cachin and M. Vukobratović. Blockchain Consensus Protocols in the Wild. *preprint arXiv:1707.01873*, 2017.
- [23] V. Chia, P. Hartel, Q. Hum, S. Ma, G. Piliouras, D. Reijnders, M. Van Staaldouin, and P. Szalachowski. Rethinking blockchain security: Position paper. *Proceedings of IEEE Blockchain 2018*, 2018.
- [24] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, and J. Wang. Untangling Blockchain: A Data Processing View of Blockchain Systems. *IEEE Transactions on Knowledge and Data Engineering*, 30(7):1366–1385, 2018.
- [25] P. Dubey and L. Shapley. Mathematical Properties of the Banzhaf Power Index. *Mathematics of Operations Research*, 4(2):99–131, 1979.
- [26] I. Eyal and E. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In N. Christin and R. Safavi-Naini, editors, *Financial Cryptography and Data Security*, pages 436–454, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.
- [27] J. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. Cryptology ePrint Archive, Report 2014/765, 2014. <https://eprint.iacr.org/2014/765>.
- [28] J. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol with chains of variable difficulty. Cryptology ePrint Archive, Report 2016/1048, 2016. <https://eprint.iacr.org/2016/1048>.
- [29] J. Garay, A. Kiayias, N. Leonardos, and G. Panagiotakos. Bootstrapping the Blockchain, with Applications to Consensus and Fast PKI Setup. Cryptology ePrint Archive, Report 2016/991, 2016. <https://eprint.iacr.org/2016/991>.
- [30] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, pages 51–68, New York, NY, USA, 2017. ACM.
- [31] I. Grigg. EOS – An Introduction. Available [online]. [Accessed: 17-12-2018], 2017.
- [32] A. Kiayias, E. Koutsoupias, M. Kyropoulou, and Y. Tselekounis. Blockchain mining games. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, EC '16, pages 365–382, 2016.
- [33] A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In J. Katz and H. Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 357–388. Springer International Publishing, 2017.
- [34] P. Koshiy, D. Koshiy, and P. McDaniel. An Analysis of Anonymity in Bitcoin Using P2P Network Traffic. In N. Christin and R. Safavi-Naini, editors, *Financial Cryptography and Data Security*, pages 469–485. Springer Berlin Heidelberg, 2014.
- [35] J. Kwon. Tendermint: Consensus without mining. Available [online]. [Accessed: 17-12-2018], 2014.
- [36] J. Kwon. Tendermint Core. Available [online]. [Accessed: 17-12-2018], 2014.
- [37] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.
- [38] S. Leonardos, D. Reijnders, and G. Piliouras. Weighted Voting on the Blockchain: Improving Consensus in Proof of Stake Protocols. 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), pages 376–384, May 2019.
- [39] Lisk. Lisk's Consensus Algorithm. Available [online]. [Accessed: 17-12-2018], 2018.
- [40] T. Mai, I. Panageas, and V. Vazirani. Opinion dynamics in networks: Convergence, stability and lack of explosion. 44th International Colloquium on Automata, Languages, and Programming, ICALP, pages 140:1–140:14, 2017.
- [41] S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. Available: [online]. [Accessed: 14-11-2018], 2008.
- [42] S. Nitzan and J. Paroush. Optimal Decision Rules in Uncertain Dichotomous Choice Situations. *International Economic Review*, 23(2):289–297, 1982.
- [43] S. Nitzan and J. Paroush. Are Qualified Majority Rules Special? *Public Choice*, 42(3):257–272, 1984.
- [44] G. Palaiopoulos, I. Panageas, and G. Piliouras. Multiplicative weights update with constant step-size in congestion games: Convergence, limit cycles and chaos. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5872–5882. Curran Associates, Inc., 2017.
- [45] R. Pass, L. Seeman, and A. Shelat. Analysis of the Blockchain Protocol in Asynchronous Networks. In J.-S. Coron and J.-B. Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, pages 643–673. Springer International Publishing, 2017.
- [46] R. Pass and E. Shi. Fruitchains: A fair blockchain. In *PODC '17*, Proceedings of the ACM Symposium on Principles of Distributed Computing, pages 315–324, New York, NY, USA, 2017. ACM.
- [47] R. Pass and E. Shi. Thunderella: Blockchains with Optimistic Instant Confirmation. In J.-B. Nielsen and V. Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, pages 3–33, 2018.
- [48] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, Apr. 1980.
- [49] D. Ryan. Validator Implementation Guide. Available [online]. [Accessed: 17-12-2018], 2018.
- [50] D. Ryan and C.-C. Liang. Ethereum improvement proposal 1011. Available: [online]. [Accessed: 3-9-2018].
- [51] A. Sapirshstein, Y. Sompolinsky, and A. Zohar. Optimal selfish mining strategies in bitcoin. In J. Grossklags and B. Preneel, editors, *Financial Cryptography and Data Security*, pages 515–532, Berlin, Heidelberg, 2017. Springer Berlin Heidelberg.
- [52] L. Shapley. A Value for n-Person Games. *Contributions to the Theory of Games (AM-28)*, II:307–318, 1953. Princeton: Princeton University Press.
- [53] L. Shapley and B. Grofman. Optimizing group judgmental accuracy in the presence of interdependencies. *Public Choice*, 43(3):329–343, 1984.
- [54] L. S. Shapley and M. Shubik. A Method for Evaluating the Distribution of Power in a Committee System. *The American Political Science Review*, 48(3):787–792, 1954.

- [55] C. Stathakopoulou. On scalability and performance of permissioned blockchain systems. Proceedings of the 12th EUROSYS Doctoral Workshop, 2018.
- [56] N. Stifter, A. Judmayer, P. Schindler, A. Zamyatin, and E. R. Weippl. Agreement with Satoshi – On the Formalization of Nakamoto Consensus. *IACR Cryptology ePrint Archive*, 2018:400, 2018.
- [57] G. Vizier and V. Gramoli. ComChain: Bridging the Gap Between Public and Consortium Blockchains. Proceedings of the IEEE International Conference on Blockchain (Blockchain'18), Jul 2018.
- [58] P. Young. Optimal Voting Rules. *Journal of Economic Perspectives*, 9(1):51–64, March 1995.
- [59] J. Yu, D. Kozhaya, J. Decouchant, and P. Verissimo. RepuCoin: Your Reputation is Your Power. Cryptology ePrint Archive, Report 2018/239, 2018. <https://eprint.iacr.org/2018/239>.
- [60] M. Zuckerman, P. Faliszewski, Y. Bachrach, and E. Elkind. Manipulating the quota in weighted voting games. *Artificial Intelligence*, 180–181:1–19, 2012.