Proof of Stake and Activity: Rewarding On-Chain Activity Through Consensus

Karen Terjanian
Fastex
www.fastex.com
Email: karen@fastex.com

Aram Jivanyan
Yerevan State University
Fastex
Email: aram.jivanyan@ysu.am

Dmitry Sidorov
Fastex
www.fastex.com
Email: dmitry@fastex.com

***Abstract*—We are introducing a novel consensus protocol for blockchain, called Proof of Stake and Activity (PoSA) which can augment the traditional Proof of Stake methods by integrating a unique Proof of Activity system. PoSA offers a compelling economic model that promotes decentralization by rewarding validators based on their staked capital and also the business value they contribute to the chain. This protocol has been implemented already into a fully-fledged blockchain platform called Bahamut (www.bahamut.io) which is designed specifically for iGaming and other markets, is actively operating in Emirati and boasts hundreds of thousands of active users.**

***Index Terms*—Distributed consensus protocols, blockchain, distributed ledger, blockchain architecture, PBFT, Proof of Stake, Proof of Stake and Activity**

I. Introduction

In this article, we discuss a novel consensus method and its concrete implementation which is designed to motivate the active network participants and builders to become chain validators and get directly rewarded for their contribution to the chain. In our model, validators still must stake a predefined amount of assets, which acts as collateral to impose penalties for any improper behavior. However, our approach also allows quantifying the network activity of validators if any, and factoring this into the block reward distribution process. This dual consideration allows our method to be seamlessly integrated into any existing Proof of Stake consensus protocol. The problem of quantifying nodes' importance on the blockchain and enabling the most important network participants to get rewarded proportional to their contribution rather than only to their wealth has been an interesting and active exploration topic in the blockchain space [2], [3]. Many successful blockchain companies such is NEM [4] have tried to perfect this approach and make it practical. We believe our method is the first practical approach for easily and unambiguously quantifying the validator's business contribution to the network in a publicly verifiable way and rewarding them fairly through the consensus and block rewarding processes.

A. Motivation

Research of new consensus algorithms has been a very active research space after the emergence of the seminal paper [1] and especially with the explosion of blockchain systems. The Proof of Stake (PoS) consensus algorithms such are [5], [6] have been invented after Bitcoin's Proof of Work [7] and have been gaining unmatched popularity in recent years because of its energy efficiency. In the PoS system, the validators are chosen at random to validate transactions and add blocks to the blockchain based on the amount of cryptocurrency they hold and are willing to "stake" as collateral. The rewards for validating transactions are proportional to the amount of cryptocurrency staked. This fact results in the biggest disadvantage of PoS where the rich are getting richer. The staking mechanism in PoS increases the chances of selection for participants with higher stakes in the process. PoS motivates and incentivizes the cryptocurrency holding but not usage. Our consensus approach has been designed to shift the economic incentives from saving to business activity which results in multiple benefits for the chain security.

Fostering Decentralization: Within networks predominantly driven by either Proof Of Work or Proof Of Stake, there's a tendency for existing wealth to accumulate, which in turn shapes the primary economic motivation of network participants. This in turn impacts the progression towards greater chain decentralization. A consensus protocol, engineered to reward the network participants' business creativity and resulting activities might encourage a multitude of new participants to join in bolstering chain security and stability.

Enabling Economic Incentive Mechanisms through Consensus: The blockchain landscape is rapidly evolving, and we anticipate a shift in how we perceive network validators. In the future, the focus won't solely be on the quantity of their contributions but also on the quality and impact. Take, for instance, how governments currently incentivize environmentally-friendly businesses with tax benefits or reduced loan interest rates. Our system provides a deterministic and publicly verifiable method to score the activities of each network participant. The weight of this score can then be adjusted in a flexible manner to determine the importance of the participant. This flexibility will allow us to prioritize certain actions of validators over others at the application layer in a publicly verifiable manner, which will allow us to enable economic incentives for certain decentralized applications based on their real-time chain economic significance.

B. Related Work

Apart from the Proof of Work [7] and Proof of Stake [5], [6] numerous other consensus designs [6], [11], [13]–[17], [19] have emerged during the last years. Significant progress has been made in designing more scalable and secure fundamental consensus algorithms providing lower latency and fast finality including [11] [14] [13] [19] among many others. Different variations of the existing fundamental consensus algorithms have been adopted by concrete blockchain companies with special use case focuses [4]. The two most relevant consensus protocols are detailed below.

Proof of Activity: The name of this protocol [8] is very similar to ours, but it is based on a totally different design rationale. Proof of Activity (PoA) combines PoW and PoS protocols in a way that can allow participants to both mine and stake their tokens to validate blocks. Under the PoA setups, miners compete to mine new blocks in exchange for token rewards. However, the blocks themselves do not include transactions; rather, they are empty templates embedded with the transaction title and block reward address. The information in the transaction title is used to randomly select a validator node to sign the block and confirm it to the blockchain ledger, and only token holders are eligible to act as validators. The PoA blockchain consensus mechanism helps lower the chance of a 51% attack because its structure makes it practically impossible to predict which validators will sign a block in each future iteration, and competition among both miners and transaction signers helps strike an effective balance between different network participants. However, this system is subject to many of the criticisms often aimed at classic PoW and PoS systems, since a significant amount of energy is still required to mine blocks during this protocol's PoW phase, and major token holders still have a disproportionately high chance of signing new transactions and accumulating rewards. This Proof of Activity consensus is used by the Decred and Espers blockchain projects [4].

Proof of Importance: PoI builds upon the ideas of PoS but looks beyond just the capital a node holds to evaluate its contributions to the network. Instead of only assessing the amount of capital a node has invested as in traditional PoS systems, Proof of Importance (PoI) takes into account multiple factors to determine the influence each node has on the blockchain. While the exact scoring criteria used in PoI varies, many of these consensus mechanisms borrow features from the consensus algorithms used in network clustering and page ranking. Common factors include the number of transfers a node has participated in over a set period of time and the degree to which different nodes are interlinked via clusters of activity.

PoI helps mitigate the risk of excess concentration of agency and wealth on a blockchain network since the network's top token holders do not concentrate absolute power over the network. Since each node's importance score is dynamic and based on network activity, this consensus mechanism also discourages wasteful blockchain forks, since users would need to expend resources to remain active on both forked networks in order to maintain their score. This runs contrary to traditional PoS mechanisms, in which the marginal cost of creating a block is zero and users can continue effortlessly validating blocks in the event of a fork. The PoI consensus mechanism was first introduced by the New Economy Movement (NEM) project and the design relies on the generic network theory techniques [21] [20]. This approach suffers from a major drawback which is calculating trust scores for each node where each trust score, in turn, is derived from the trust scores reported by the neighboring nodes. This algorithm to compute trust suffers from the fact that the faithfulness of the feedback reported from other nodes is unknown.
While the PoI and PoSA share a common motif of quantifying the node's contribution to the network, they utilize totally different approaches for that.

C. Our Contribution

While other consensus algorithm concepts have been explored during the last few years to quantify the network participants' importance scores and consider that in the decision-making processes such as block harvesting or validation, we believe our work's contribution in the process of exploring the feasibility of bringing each node importance into the decision making equations further is threefold.

- **Deterministic Scoring Method:** In the Proof of Importance approach the malicious nodes could collude and report low trust values for honest nodes and high trust values for dishonest nodes. An improvement could be estimating the credibility of the feedback of other nodes and weighting the reported trust values by the credibility score which in turn will significantly complicate the design without fully solving the problem. Our suggested method allows for deriving each node's importance in a very straightforward and unambiguous way relying only on publicly available and finalized transaction data. This score can further be flexibly programmed based on various consensus rules.
- **Seamless integration with any PoS design:** Our proposed approach of quantifying node activities and their significance can seamlessly be integrated with any Proof of Stake consensus algorithm. It stands as a robust augmentation rather than merely serving as an alternative or replacement. This underscores its high adaptability and potential impact.
- **Enterprise-Ready Implementation:** The proposed design has been seamlessly integrated with the Ethereum's Proof of Stake consensus algorithm [5] and the final Proof of Stake and Activity Consensus has been implemented into a fully-fledged blockchain platform [23] which now is very actively operating in the United Arab Emirates market and worldwide. The platform is based on the Ethereum Virtual Machine and replicates its main design components including the block proposer and sync committee selection design, validator penalties, and slashing

approaches. But instead of selecting the validators based only on their staked amount, our implementation is leveraging the Validator Power, a new parameter that is derived from both the validator staked amount and its activity score. The implementation required approximately 7000 lines of code to change in Ethereum's codebase for production-ready deployment.

II. Activity Score Calculations

The goal of our Proof of Stake and Activity (PoSA) consensus protocol is to create a decentralized network that values real, on-chain activity, not just the ability to stake tokens. This method aims to level the playing field, encouraging innovation and business acumen instead of just making the wealthy wealthier. Unlike the Proof of Importance model, which focuses on a node's communication and interaction frequency, we measure a node's importance based on its actual economic activities performed on the blockchain.

In existing blockchain systems, which serve as general platforms for decentralized apps (DApps), the business logic is carried out through smart contracts. Users interact with these smart contracts for various tasks, whether it's exchanging tokens, borrowing assets, or purchasing NFTs. Our PoSA logic encourages DApp developers who have meaningful transaction activity to become validators. They do this by staking tokens and then associating a specific smart contract with their validator account. This smart contract activity is then factored in as an "extra importance score" during the block validation process. This activity is transparent and can be fairly assessed in multiple ways.

Below, we delve into how we calculate this activity score and use it to determine the final "validator power," which will serve as an alternative to the simply staked value in the consensus process.

A. Validator Activity Score Calculation

In the following discussion, we introduce a quantification approach that assigns each validator a deterministic and publicly-verifiable activity score for a given epoch. It's important to note that validators are still required to lock up a certain amount as collateral (stake). Therefore, the activity score shares the same unit of measurement as the stake, which is the native currency of the ledger. Validators can still participate solely with their stake, resulting in an activity score of zero. For validators who are generating on-chain activity through associated smart contracts, their calculated activity is added to the staked amount to define their final decision-making power.
In the realm of blockchain, a new block is appended to the chain at fixed intervals, termed as "block time." In our specific implementation, an epoch consists of 32 blocks, and a window encompasses 1575 epochs. Validator activity scores, and thus their power, are recalculated over these epochs. The activity score stems from the cumulative transaction fees tied to the linked smart contract. According to Ethereum's fee calculation method, each transaction fee is defined by the formula

$$\text{tx}_{\text{fee}} = \text{gas\_fee} \cdot (21000 + \text{tx}_{\text{gas}})$$

where gas_fee is the current gas price defined by the market dynamics, the 21000 is a fixed constant gas amount, and tx_{gas} is assigned in a deterministic way based on the smart contract code instructions. Peer-to-peer asset transfers not invoking smart contracts will have a fee of $\text{tx}_{\text{fee}} = \text{gas\_price} \cdot 21000$ Let's define the activity score assigned to a concrete transaction as a_{tx} and the activity assigned to the smart contract for the given epoch e as $\text{A}^{\text{e}}_{\text{contract}}$. For computing the smart contract transaction activity score, we disregard the constant 21000 gas and assign

$$\text{a}_{\text{tx}} = \ \text{tx}_{\text{gas}} \cdot \text{gas\_fee}$$

The smart contract activity for the given epoch is the aggregation of its transaction activity scores which has happened during that epoch.

$$\text{A}^{\text{e}}_{\text{contract}} = \sum_{\text{tx}^{\text{contract}} \in \text{e}} \text{a}_{tx} = \sum_{\text{tx}^{\text{contract}} \in \text{epoch e}} \text{tx}^{\text{contract}}_{\text{gas}} \cdot \text{gas\_fee}$$

. The final smart contract activity score $\text{A}^{\text{e}}_{\text{contract\_score}}$ is updated at each epoch through Algorithm 1 where at each epoch it is gradually adjusted to the score change dynamics spanned over the window period. algopseudocode

Algorithm 1 Smart Contract Activity Score Calculation

```
window_size = 1575(**window** = window_size × epoch)
epoch = 32 × block
**for** each epoch $e_i$ **do**
    $\text{A}^{e_i} = 0$
    Take the previous epoch's contract score as $\text{A}^{\text{i-1}}_{\text{contract\_score}}$
    Let $\text{TX}^{\text{contract}}_{e_i} = \{\text{tx}_1, \text{tx}_2, \ldots, \text{tx}_{k_i}\}$ be the given
      contract's transactions during that epoch $e_i$
    **for** $\text{tx}_j \in \text{TX}^{\text{contract}}_{e_i}$ **do**
        $A^{e_i} = A^{e_i} + a_{\text{tx}_j}$
    **end for**

    **if** $i >$ window_size **then**
        $\text{A}^{e_i}_{\text{contract\_score}} = \text{A}^{\text{i-1}}_{\text{contract\_score}} - \frac{\text{A}^{\text{i-1}}_{\text{contract\_score}}}{\text{window\_size}} + A^{e_i}$
    **else**
        $\text{A}^{e_i}_{\text{contract\_score}} = \text{A}^{e_{i-1}}_{\text{contract\_score}} + A^{e_i}$
    **end if**

    Output $\text{A}^{e_i}_{\text{contract\_score}}$
**end for**
=0
```

Note that the method for calculating the activity score, based on transaction fees, has a solid economic rationale. While the amount of gas needed for a concrete transaction is fixed by its business logic, the gas fee itself can fluctuate, reflecting the overall economic activity on the blockchain. For example, when blocks fill up, gas fees tend to increase. This forces users to pay more to ensure their transactions are included in the upcoming block. Such a rise in fees serves as an indicator of how much users value their transactions and, by extension, reveals the economic significance of the smart contract activity. Therefore, while aggregating all transaction fees might seem straightforward at the application level, this metric effectively captures multiple dimensions of a smart contract's importance and user engagement.

B. Validator Power

The validator power is the final property that defines V's participation in the block harwesting and validation lotteries. Let's assume there are N active validators $V_1, V_2, \ldots, V_N$ for the given epoch. The effective balance of the i-th validator for the given epoch EB_i is the validator's staked capital after all penalties and slashing. Next, the overall activity happening on-chain over the given epoch can be summarized as the sum of three independent components defined as follows:

- **Validators Activity A**: Let A_e represent the sum of all activity scores from smart contracts that are associated with validators for the given epoch. Mathematically, $A_e = \sum_{i \in 1,\ldots,N} \mathrm{A}^e_{\text{contract\_score}_i}$ is the activity score of the i-th validator. $\mathrm{A}^e_{\text{contract\_score}_i} = 0$ also in cases the validator did not link any smart contract. In our framework, we permit each validator to associate only a single smart contract with their staked assets. However, entities that control multiple smart contracts can initiate multiple stakes, each linked to a different smart contract. It's worth noting that A could be zero, which would occur if no smart contract owner chooses to become a validator on the blockchain.
- **Other Smart Contract Activity B**: Let B_e represent the sum of all activity scores from smart contracts deployed on the chain but not associated with any validator for the given epoch. While we anticipate that most active smart contract owners will be motivated to become validators, it's acknowledged that some on-chain activities may not be linked to any validator. The value of B_e can be zero in two scenarios: either every smart contract deployed on the chain is linked to a distinct validator, or there is no on-chain activity to speak of.
- **Asset Transfer Transactions Scores T**: Let T_e represent the sum of the invariant components of all transaction fees during a given epoch, calculated as $\mathrm{T} = \sum_{\text{tx} \in \text{epoch}} 21000 \cdot$ gas_fee. The value of T will be zero exclusively when there are no transactions occurring on-chain during that specific epoch.

For the given epoch e we define the V_i's power as

$$\mathrm{P}^{\max}_{e_i} = \frac{\mathrm{T_e}}{N} + \mathrm{A}^e_{\text{contract\_score}_i} \quad (1)$$

and we define the effective power as

$$\mathrm{P}_{e_i} = \mathrm{P}^{\max}_i \cdot \frac{\mathrm{EB}_i}{S} \quad (2)$$

where $\mathrm{A}^e_{\text{contract\_score}_i}$ is the activity score assigned to the validator V_i for the epoch e, S is the fixed amount of capital to be staked and EB_i is the effective balance of the V_i-th validator. The effective balance is the validator's initial staked amount S minus all penalties that it has got before the subject epoch. It is important to note, for comparison, that in the Ethereum Proof of Stake consensus algorithm, the power of a validator is essentially determined by their effective balance EB_i.

III. Consensus Implementation

A novel fully fledged blockchain platform has been deployed with a concrete implementation of the proposed Proof of Stake and Activity consensus [23]. This implementation forks the core Ethereum Virtual Machine code to support the same smart contract logic and programming language. It also replicates Ethereum's Proof of Stake concepts [22] with an interesting modification where the network validators are selected based on their **Effective Power** P_{e_i} as computed above instead of only their effective balances EB_i . We leave a detailed description of the full consensus protocol implementation for the full paper and briefly highlight the main design rationale here.

Like in Ethereum, the blockchain implementation will consist of two different layers. The first layer is the execution layer similar to Ethereum's Geth, where the user wallet and smart contracts transactions are executed. The second layer is the consensus layer like [22] where the block validation process is managed.

- The total supply of native tokens is 1 billion. Among the total supply, a pool of 120 million tokens is allocated for future minting according to the yearly issuance formula. The native token of the blockchain is called FTN and 1 FTN = 10^{18} Gwei.
- The block time is 12 seconds like in Ethereum. The fixed staking amount is fixed to be S = 8192 FTN.
- For each block, a block proposer, sync committee members, and attesting validators are selected to finalize the next block choice. All committee members, validators, and the block proposer will be rewarded for their honest and on-time activity and will get penalized for misconducting their actions. The rewarding and penalty mechanism logic is borrowed from the Ethereum design (Section 2.8.5 in [24]).
- The gas fee is dynamically controlled by the overall chain activity like in Ethereum. Each block capacity is 30M gas usage. If the block is half filled, the base gas fee is increased up to 12 percent compared to the price in the previous block. If the capacity is low, so the blocks are mostly empty, the base gas fee is decreased up to 12 percent.

A. Validator Selection Process

In order to become a validator, the network participant must stake a fixed S amount of native tokens. This staked amount is locked up as a security deposit on the network and is used as collateral which compels the validator nodes to behave properly and keep the network secure. The stake size is fixed to be $S = 8192$ which is referred to as the validator's actual balance. The I-th validator's effective balance EB_i is the actual balance deducted by the penalties that happened due to slashing. For each block, certain validators are assigned specific roles to contribute to the block selection process.

Sync Committee Selection A sync committee of 512 members is selected once per every 256 epochs which proposes the proceeding block on the blockchain before the next block proposal. The selection process of sync committee members is executed through a special *shuffle and select* algorithm like in Ethereum, where the list of all validators is first shuffled randomly, and then a fresh randomness $rand$ is chosen between the range 0 and $\text{MaxRand} = 256$ to check if the equation

$$\frac{\text{EB}_i}{S} \geq \frac{rand}{\text{MaxRand}}$$

holds for the next validator in the shuffled list. If the condition holds for the current validator, the latter is selected, otherwise, the selection algorithm moves on to check the next validator in the shuffled list.

Block Proposer Selection. Assuming the number of all validators staked on the chain is N, one block proposer is selected per each block time to propose the next block according to the following selection process.

- Randomly shuffle the list of all N validators and consider the list of effective powers $P_1, P_2, \cdots, P_N$ ordered according to the shuffle. Note that each validator power is computed according to the formula (2) and is updated once per epoch.
- Compute the aggregated effective power as

  $$P = \sum_{i=1}^{N} P_i$$

  .
- Consider the vector

  $$\frac{P_0}{P}, \frac{P_1}{P}, \frac{P_2}{P}, \dots, \frac{P_N}{P}$$

  where $P_0 = 0$ is a void validator added to the list for the sake of integrity
- Chose a random value $x \leftarrow_R [0, 1]$ from the range
- If $x = 0$ then select the first validator with the power P_1, otherwise select the k-th validator from the shuffled sorted list such that

  $$\sum_{i=0}^{k-1} \frac{P_{k-1}}{P} < x \leq \sum_{i=0}^{k} \frac{P_k}{P}$$

This selection algorithm will guarantee each validator's selection probability to be proportional to its effective power. As shown in Fig 1, in Ethereum the top 100 smart contracts among the millions deployed on-chain generate almost 50 percent of the overall chain activity. Enabling these few sharks to be constantly selected as proposers would put the decentralization of decision-making processes at a huge risk. So in future versions of our protocol, we may also consider the validator selection process to be executed through alternative methods. One possible way can be applying a mathematical Sigmoid function to each power which will give extra leverage to small powers. Another approach can be clustering the validators based on certain power thresholds and giving certain quotas to each cluster. This will increase the chances of small powers being selected frequently and will mild the big powers' decision power.

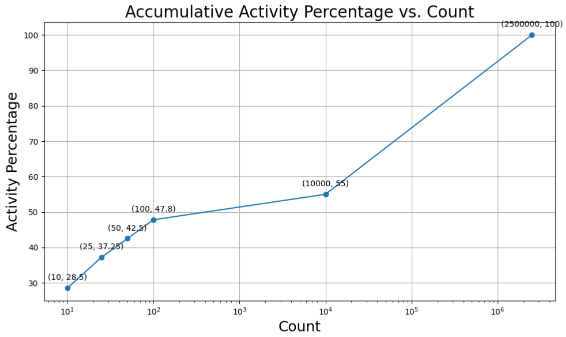

Figure 1. Accumulative Activity percentage of a certain number of validators over 1 month period. There are approximately 2.5m active smart contracts deployed on the Ethereum blockchain now.

Attestators Selection Process The process for selecting attestators is the same as in Ethereum and is quite simple. All validators are grouped into 32 smaller sets. Throughout the epoch, each set is picked in turn to serve as attestators for the next block. This way, every validator gets a chance to act as an attestator for one block in each epoch.

B. Mint and Burn Mechanisms

We burn tokens on the application layer and mint new tokens on the consensus layer to reward the block validators. Rewards are distributed according to specific laws designed in Ethereum and tailored to our PoSA logic to incorporate the validator activity parameters. The overall minted amount per epoch is computed as

$$M = M_1 + M_2$$

where

$$M_1 = \frac{F \cdot \sqrt{\sum_{i=1}^{N} EB_i}}{\text{number\_of\_epochs\_per\_year}}$$

and

$$M_2 = \frac{(A_e + T_e) \cdot f_e}{(\text{window\_size} \cdot \text{epoch\_size})^2} \cdot \text{epoch\_size}$$

The $F \cdot \sqrt{\sum_{i=1}^{N} EB_i}$ is the yearly minted amount according to the specified issuance formula which is proportional to the square root of the validator count and minted tokens. The special constant k is chosen by the development team to hit certain issuance rates under concrete conditions. At this moment $k = 156$ in order to ensure at least 7 % validator

Number of validators	ROI
4096	7%
8192	4.95%
12888	3.9%
16384	3.496%
20480	3.127

Table I
STAKING REWARDS VS NUMBER OF VALIDATORS

return in the setup with 4096 validators. This constant is derived through the following reasoning. New tokens will be minted per each epoch and there are $N_e = 82181.25$ epochs per year. Assuming the effective balance of each validator is $EB_i = 8192$ tokens, the initial number of validators is $N_V = 4096$ and the target ROI is 7%, we can assume the expected number of newly issued tokens in gwei is to be

$$M_e = 4096 \cdot 8192 \cdot 10^9 \cdot 0.07$$

Hence we should have

$$\frac{c \cdot N_v \cdot 8192 \cdot 10^9}{\sqrt{N_v \cdot 8192 \cdot 10^9}} \cdot N_e = M_e$$

This constant can be adjusted to control the minimum return of interest for each validator. The current value of F will ensure the following minimum ROI-s for the fixed number of validators as provided in the table below.

We also implement a deflationary mechanism and burn a certain amount of tokens per block where the number of burned tokens is defined by the formula

$$K_e = (A_e + T_e + B_e)$$

Here

- A_e aggregates the transaction fees comprising the activity scores of all smart contracts linked to validators.
- B_e aggregates the transaction fees comprising the activity scores of all smart contracts not linked to any validator.
- T_e sums up the invariant part of transaction fees corresponding to the fixed 21000 gas spending in each transaction.

K_e is the sum of all transaction fees spent on the blockchain during the epoch. When this K_e amounts of tokens are burned, we will mint

$$M_2 = (A_e + T_e) \cdot f_e$$

fresh new tokens on the execution layer at the end of each epoch where

- A_e approximates the aggregation of the activity scores of all smart contracts linked to validators.
- T_e sums up the invariant part of transaction fees corresponding to the fixed 21000 gas spending in each transaction.
- f_e is the aggregated and normalized gas fee per epoch.

The exact algorithm of computing the values A_e, T_e f_e is depicted in Algorithm 2.

Algorithm 2 Mint amount calculation

```
window_size = 1575(window = window_size × epoch)
epoch_size = 32(epoch = epoch_size × block)
$A_{\text{e}} = 0$
$f_{\text{e}} = 0$
$T_{\text{e}} = 0$
$A_{\text{previous}}$ is the previous epoch aggregated activity score
$f_{\text{previous}}$ is the previous epoch's normalized gas fee score
$T_{\text{previous}}$ is the previous epoch's aggregated invariant activity score.
for each epoch do
    Take the current epoch activity score as $\text{A}_{\text{epoch}}$
    Take the current epoch invariant activity score as $\text{T}_{\text{epoch}}$
    $\text{f}_{\text{epoch}} = 0$
    for i = 1, 2, ..., 32 do
        Take the current block fee as $\text{f}_{\text{block}_i}$
        $\text{f}_{\text{e}} = \text{f}_{\text{e}} + \text{f}_{\text{block}_i}$
    end for
    $f_{\text{e}} = f_{\text{previous}} - \frac{f_{\text{previous}}}{\text{window\_size}} + \text{f}_{\text{epoch}}$
    $A_{\text{e}} = A_{\text{previous}} - \frac{A_{\text{previous}}}{\text{window\_size}} + \text{A}_{\text{epoch}}$
    $T_{\text{e}} = T_{\text{previous}} - \frac{T_{\text{previous}}}{\text{window\_size}} + \text{T}_{\text{epoch}}$
    for i = 1, 2, ..., 32 do
        $\text{block\_reward}_i = \frac{(A_{\text{e}} + T_{\text{e}}) \cdot f_{\text{e}}}{(\text{window\_size} \cdot \text{epoch\_size})^2}$
        Output $M_2 = \text{block\_reward}_i$
    end for
end for
=0
```

C. Rewards and Penalties

Sync committee members, attesting validators and the block proposer are all getting block rewards according to the following rules

- **Sync Committee Member Rewards:** Each sync committee member gets

$$R^i_{\text{scm}} = \frac{\frac{2}{56} M_1}{512}$$

The reward is distributed per block only to attesting validators who participate in the process timely.

- **Attesting Validator Reward** Each attesting validator can get maximum

$$R^i_{av} = (c_1 + c_2 + c_3) BR_i$$

where

$$BR_i = \frac{EB_i \cdot k}{\sqrt{\sum_{j=1}^{N} EB_j}}$$

Here c_1 and c_2 can be either $\frac{14}{56}$ or less up to 0 and C_3 can be $\frac{26}{56}$ or less up to 0 depending on the fair and timely participation of the validator on the attestation process. The details can be found in the Ethereum attestation process design [5]. Easy to notice that in case of fair participation of all sync committee members and attesting validators the overall reward distributed will be equal

$$M_1 = \sum_{i=1}^{512} R^i_{\text{scm}} + \sum_i R^i_{av}$$

- **Block Proposer Reward**. The block proposer is supposed to get all M_2 tokens minted on the blockchain along with all transaction tips left by the users. But in our implementation, the block proposer will get penalties in case any of the sync committee members or attesting validators misconduct his duties during the block validation process. Hence the block proposer reward is normalized through a unique formula that incorporates the fair behavior and penalties of all other validators participating in the process through the following formula

$$R_{\text{bp}} = \frac{M_2^e}{32}(\frac{2}{56}\frac{\sum_{i=1}^{512} R^i_{\text{scm}}}{MSCR} + \frac{54}{56}\frac{\sum_j R^i_{\text{av}}}{(\sum_i BR_i \frac{54}{56}}))$$

In our consensus algorithm, we replicate the slashing and penalty mechanisms designed and used by the Ethereum blockchain with minor modifications [24]. Note that slashing occurs when validators break very specific protocol rules of three different types which could be part of an attack on the chain. We preserve the logic of correlated penalties where a light punishment is happening for isolated incidents, but a severe punishment occurs when many validators are slashed in a short time period. Like in Ethereum, the block proposers will receive rewards for reporting evidence of slashable offenses. The details of slashing mechanism in Ethereum are detailed in [24] but for the sake of paper integrity we will discuss the most relevant aspects here.

The Initial Penalty: Slashing is triggered by the evidence of the offense being included in a block. Once the evidence is confirmed by the network, the offending validator (or validators) is slashed. The offender immediately has $\frac{1}{32}$ of its actual balance deducted from its effective balance. Along with the initial penalty, the validator is queued for exit and has its withdrawability epoch set to around 36 days in the future.

The Correlated Penalty: 18 days post-slashing, at the halfway mark of the withdrawal period, a validator faces a potential second penalty. This is influenced by other slashings over those 18 days. If few slashings occur, the penalty might be zero. This system aims to apply light penalties for minor infractions but severe ones for major threats, like conflicting block finalizations. The execution layer chain keeps a record of slashed validators' balances over the last 8192 epochs (36 days) for this calculation. The correlated penalty calculation is detailed in [24]

The effective balance which is the validator staked balance minus all penalties up to that certain period will play a vital role in calculating the validator's effective power which in turn will define the success probability of the certain validator being selected in the consensus protocol as a block proposer or a sync committee member.

In our blockchain certain amounts of tokens are constantly burned and minted in parallel to ensure concrete economic dynamics. New tokens are minted through two separate and logically independent processes. The first process creates new tokens out of fresh air no matter the existing on-chain activity volume. The second process is linked to the validators' activity and will mint new tokens depending only on the on-chain activity and the transaction fees spent by the validators.

IV. Benchmarks and Performance Analysis

The proposed consensus algorithm has been implemented into a fully-fledged blockchain platform and is running on the Bahamut blockchain's testnet already. Thorough benchmarks and economic tests have been run to check that no validator can speculate with its smart contracts by generating excessive activity in order to get leverage in the selection process and earn more rewards. The most important factor mitigating such risks is the fact that the returned rewards never exceed the cost essentially paid through transaction fees for generating the activity. Other tests have been run to show the overall economic behavior and dynamics under different patterns. Tests have been run with 4000 validators among which 1000 validators also have active smart contracts whose activity is factored in defining their powers. The distribution of these 1000 contract activities mimics the same distribution on the Ethereum blockchain where the majority of the linked smart contract activity is generated with few smart contracts while the other contracts have relatively small and similar weights. The graphics shown in IV show two interesting scenarios. In the first scenario, the combined activity of 1000 smart contracts constitutes just 25% of total activity, while the invariant gas fee portion of all transactions represents 50%. This shows that while highly active smart contracts have diminished power, those with lesser activity have proportionally more influence. This effectively curtails the dominance of major players, granting smaller entities a higher stake in decisions. Such a situation, where many smart contracts are tied to validator accounts, enhances the security of the chain. However, in the second scenario, the influence of validator smart contract activities dips to a mere 16% of the total chain activity. In this scenario, the effective validator powers are more correlated with their actual activity scores. We delve deeper into the economic implications of the provided consensus algorithm and will provide comprehensive security proofs in the full paper.

V. Conclusion

This paper has discussed a novel practical consensus protocol focused on rewarding the important nodes that are contributing to the chain's usability and stability. The proposed protocol has been implemented into a fully-fledged EVM-based blockchain system with over four thousand validators and hundreds of thousands of users. The motivation of the proposed design is to enable rewarding network participants' activity and creativity through the core consensus rules and

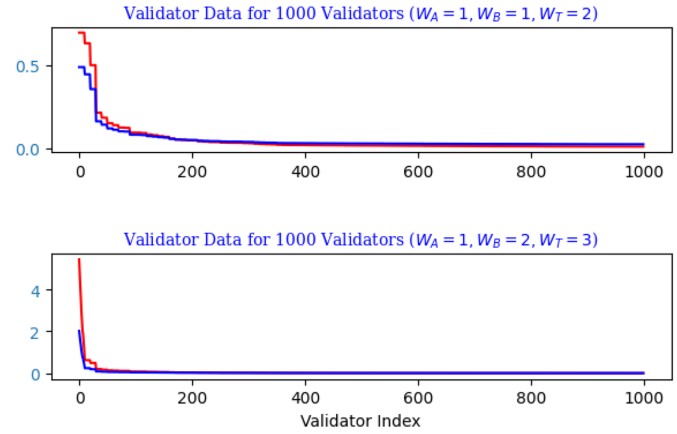

Figure 2. Distribution of validator activities percentages and power percentages over 1 month period. The W_A, W_B and W_T show respectively the weights of A, B and T in the overall gas usage for the given period.

also foster decentralization of the blockchain and allow programming decentralized and publicly verifiable economic incentives for various types of players and causes in the future. The paper leaves the formal security analysis of the proposed method to the full paper. Both the protocol design and the blockchain ecosystem built on top of it are evolving and it remains an interesting question to research other variations of on-chain activity rewarding systems.

References

[1] M. Castro, B. Liskov, Practical byzantine fault tolerance. In OsDI, volume 99, pages 173–186, 1999.

[2] Yuuki Komi and Takayuki Tatekawa: Consensus Algorithm Using Transaction History for Cryptocurrency. https://eprint.iacr.org/2023/373.pdf

[3] Proof of Importance. https://docs.nem.io/pages/Whitepapers/NEM-techRef.pdf

[4] A Guide to Understand Blockchain Consensus Algorithms. https://appinventiv.com/blog/blockchain-consensus-algorithms-guide/

[5] Ethereum Consensus Layer Documentation. https://eth2book.info/bellatrix/part2/incentives/penalties/

[6] F. Yang, W. Zhou, Q. Wu, R. Long, N. N. Xiong and M. Zhou, ”Delegated Proof of Stake With Downgrade: A Secure and Efficient Blockchain Consensus Algorithm With Downgrade Mechanism,” in IEEE Access, vol. 7, pp. 118541-118555, 2019, doi: 10.1109/ACCESS.2019.2935149.

[7] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system” https://bitcoin.org/bitcoin.pdf

[8] Proof of Activity: Extending Bitcoin’s Proof of Work via Proof of Stake https://eprint.iacr.org/2014/452.pdf

[9] C. Cachin. Architecture of the Hyperledger blockchain Fabric. In Workshop on Distributed Cryptocurrencies and Consensus Ledgers, 2016.

[10] Proof of Authority. https://github.com/paritytech/parity/wiki/Proof-of-Authority-Chains.

[11] Yin, M., Malkhi, D., Reiter, M. K., Gueta, G. G., Abraham, I. (2018). HotStuff: BFT consensus in the lens of blockchain. arXiv preprint arXiv:1803.05069.

[12] Rafael Pass and Elaine Shi. Hybrid Consensus: Efficient Consensus in the Permissionless Model. https://eprint.iacr.org/2016/917.pdf

[13] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, Nickolai Zeldovich. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. https://people.csail.mit.edu/nickolai/papers/gilad-algorand-eprint.pdf

[14] Team Rocket. Snowflake to Avalanche: A Novel Metastable Consensus Protocol Family for Cryptocurrencies.

[15] Vitalik Buterin and Virgil Griffith. Casper the friendly finality gadget. CoRR, abs/1710.09437, 2017. URL: http://arxiv.org/abs/1710.09437

[16] Vlad Zamfir. Casper the friendly ghost. a correct-by-construction blockchain consensus protocol. URL: https://github.com/ethereum/research/blob/master/papers/cbc-consensus/AbstractCBC.pdf.

[17] M. Yin, D. Malkhi, M. K. Reiter, G. Golan-Gueta, and I. Abraham. HotStuff: BFT consensus with linearity and responsiveness. In Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, pages 347–356, 2019.

[18] David Schwartz, Noah Youngs, and Arthur Britto. The Ripple Protocol Consensus Algorithm, September 2014.

[19] Erica Blum, Jonathan Katz, Julian Loss, Kartik Nayak, Simon Ochsenreither. Abraxas: Throughput-Efficient Hybrid Asynchronous Consensus, https://eprint.iacr.org/2023/689.pdf

[20] Xinxin Fan, Ling Liu, Mingchu Li and Zhiyuan Su. EigenTrust++: Attack Resilient Trust Management https://docs.nem.io/pages/Whitepapers/EigenTrust.pdf

[21] A.N. Langville and C.D. Meyer. Google’s PageRank and Beyond: The Science of Search Engine Rankings. Princeton University Press, Princeton, USA, 2006.

[22] Vitalik Buterin, Diego Hernandez, Thor Kamphefner, Khiem Pham, Zhi Qiao, Danny Ryan, Juhyeok Sin, Ying Wang, and Yan X Zhang. Combining GHOST and Casper. 2020.

[23] Bahamut. First Emirati Blockchain. URL: www.bahamut.io

[24] Ethereum Technical Overview. The Incentive Layer. Slashing: https://eth2book.info/bellatrix/part2/incentives/slashing/

[25] Circulating Supply Equilibrium for Ethereum and Minimum Viable Issuance during the Proof-of-Stake Era. https://ethresear.ch/t/circulating-supply-equilibrium-for-ethereum-and-minimum-viable-issuance-during-the-proof-of-stake-era/10954