

# Fairness Notions in DAG-based DLTs

Mayank Raikwar,\* Nikita Polyanskii,† Sebastian Müller‡

\*University of Oslo, Norway

Email: mayankr@ifi.uio.no

†IOTA Foundation, Berlin, Germany

Email: nikita.polyanskii@iota.org

‡Aix Marseille Université, CNRS, Centrale Marseille, France

Email: sebastian.muller@univ-amu.fr

**Abstract**—This paper investigates the issue of fairness in Distributed Ledger Technology (DLT), specifically focusing on the shortcomings observed in current blockchain systems due to Miner Extractable Value (MEV) phenomena and systemic centralization. We explore the potential of Directed Acyclic Graphs (DAGs) as a solution to address or mitigate these fairness concerns. Our objective is to gain a comprehensive understanding of fairness in DAG-based DLTs by examining its different aspects and measurement metrics. We aim to establish a shared knowledge base that facilitates accurate fairness assessment and allows for an evaluation of whether DAG-based DLTs offer a more equitable design. We describe the various dimensions of fairness and conduct a comparative analysis to examine how they relate to different components of DLTs. This analysis serves as a catalyst for further research, encouraging the development of cryptographic systems that promote fairness.

## I. INTRODUCTION

DLTs have become quite popular in the last decade due to their broad range of applications. The most well-known DLT architecture is the Blockchain which was introduced in Bitcoin cryptocurrency [1]. After the advent of Bitcoin, there have been a plethora of cryptocurrencies [2] with similar underlying blockchain structures but differing in consensus, security guarantees, performance, and additional functionalities. Nevertheless, the main technical hurdle to the adoption of blockchain in real-world applications is its scalability. Therefore, many methods such as Layer-2 protocols [3], and sharding techniques [4] have been proposed to solve the scalability issue of the blockchain. Another alternative solution to the scalability problem is to have a different structure of DLT than the linear chain structure of blockchain. One such structure is Directed-Acyclic Graph (DAG)-based DLT [5], [6]. The assurance of DAG-based DLT is high scalability and fast confirmation of transactions without significantly compromising security.

The idea of DAG-based DLT is to form a directed acyclic graph structure from the blocks/transactions instead of forming a chain by referencing a single block [7]. In this way, many transactions are verified in parallel and hence provide a more scalable DLT architecture. The goal of these DAG-based DLTs is to achieve higher throughput while keeping or reducing the communication or/and computation complexity, latency, and storage overhead. Most of the existing works in DAG-based DLTs focus on performance improvement, however, fairness notions in DAG-based DLTs have not yet been explored.

Fairness plays a prominent role in DLTs encompassing technical, economic, and social considerations. In a DLT, an unfair

scenario can lead to discontent among participants, thereby impeding the widespread adoption of the DLT. Consequently, it is imperative for DAG-based systems to prioritize the integration of fairness within their frameworks. The prevailing notion of fairness, often acknowledged, entails the eventual acceptance of blocks produced by slow yet honest nodes in the DAG-based DLT. A node can be termed as *slow* if 1) the node has low computing power in Proof-of-Work (PoW)-based DLT; 2) the node has a low stake in Proof-of-Stake (PoS)-based DLT; 3) the node has large communication delay with the rest of the network, for example, due to the network topology or being geographically distant. The above-defined notion is a very general definition of fairness but in DAG-based DLTs, fairness can be expressed in different notions.

While fairness has been examined and addressed in blockchain protocols, the presence of a single miner or validator making decisions regarding the ledger’s progress remains a prominent cause of fairness challenges. However, in DAG-based protocols, the ledger’s state progression is controlled by multiple nodes. In essence, this represents the potential for enhanced fairness in DAG-based protocols. Furthermore, some DAG-based DLTs allow weak references to rescue the old blocks and improve the fairness of the DLT [8], [9].

### A. Related Work

Since the inception of Bitcoin, there has been a substantial amount of work and studies conducted toward improving the scalability and security of the protocol [10]. Nevertheless, in recent years, a few works concentrate on fairness in Bitcoin like linear blockchain models. These works examine fairness in two main aspects: 1) Fairness in reward mechanisms of blockchain systems [11], [12]; 2) Fairness in the ordering of transactions in blockchain systems [13], [14].

Amoussou-Guenou et al. [11] propose the formal notions of fairness in committee-based PoW blockchain protocols. The authors define two fairness notions: 1) Selection fairness which says how fairly the subset of nodes are selected to participate to append a block in a consensus round; 2) Reward fairness which describes how the reward is distributed among the committee nodes in a fair manner. Huang et al. [12] propose two types of fairness, i.e., exceptional fairness and robust fairness in PoS blockchain protocols. These fairness notions characterize the relation between a miner’s stake and his/her reward in a consensus round.

Fairness in transaction ordering refers to the ordering of transactions in a block of a blockchain. A fair ordering expresses a fair block proposal where the block proposer follows the random selection of transactions and does not include/exclude/prioritize some transactions. Different fairness notions in transaction ordering have been proposed in blockchain protocols [13], [14]. Even so, manipulation in transaction ordering leads to MEV extraction and MEV attacks [15]. There are different types of MEV attacks. Out of these attacks Sandwich and Long-trail MEV attacks occur more often. In 2022, MEV searchers generated \$128 million revenue alone from Sandwich attacks [16].

Though a few efforts have been made to study and subsequently solve the fairness issues in blockchain protocols, not many works endeavour to look into fairness problems in DAG-based DLTs. The primary goal of these DLTs has been to solve the inherent low throughput problem of blockchain.

The field of DAG-based protocols has experienced continuous growth in research efforts [7]. While many of these works aim to enhance the performance of DAG-based Distributed Ledger Technologies (DLTs), they also explore different network models (synchronous, partially synchronous, asynchronous) and adversarial models (static, adaptive). However, only a limited number of works specifically address fairness in DAG-based DLTs. Birmpas et al. [17] present a parametric model that evaluates the fairness and efficiency of DAG-based DLTs by adjusting various ledger parameters. In their model, fairness is defined as the proportional reward allocation to participating validator nodes based on their hash power. Their findings demonstrate that fairness is significantly influenced by the connectivity levels of validator nodes within the underlying peer-to-peer network. It is important to note that their model focuses on capturing fairness exclusively in DAG-based DLTs that employ Proof-of-Work (PoW) puzzles as their consensus mechanism.

DAG-based DLTs such as DAG-Rider [18], Tusk [19], Bullshark [8] give attention to fairness in their protocol. We will give more details about this in the following sections.

In a PoW-based DLT protocol with high difficulty, slow nodes rarely get a chance to append a block or issue the transactions. Vigneri et al. [20] propose an adaptive rate control mechanism in a PoW-based DLT providing dynamic difficulty for the fair treatment of nodes irrespective of their computational power. Another work [21] in a similar direction shows fairness between low and high power nodes through analytical and simulation results. Müller et al. [22] construct a mathematical model for fairness in a weighted voting consensus protocol in DAG-based DLT. Using their model, a node having low weight also participates in consensus finding. These types of fairness notions are out of the scope of this paper.

## B. Our Contribution

In this paper, we are primarily interested to explore the different notions of fairness in the context of DAG-based DLTs. To the best of our knowledge, no work has been done to systematize fairness notions in DAG-based DLTs. We present informal definitions of these fairness notions and describe the

importance of these notions. We also discuss the relation of fairness with other DLT components.

## C. Structure of the Paper

The rest of the paper is described as follows: Section II presents the description of a DAG-based DLT. Section III demonstrates the different fairness notions in DAG-based DLT protocols. Further, Section IV provides a brief description of the relationship between defined notions and other important components/functionalities in DAG-based DLTs. Finally, in Section V, we conclude the paper with a few interesting research directions.

## II. DAG-BASED DLT

DAG-based DLT systems structure the transactions/blocks in the form of a DAG topology. A directed graph  $G$  is a tuple of vertex set  $V$  and directed edge set  $E$ , such that  $G := (V, E)$ . In this graph, a vertex represents a unit (transaction/block) and a directed edge represents a relationship between two units. That means, an edge  $e \in E$  in the graph  $G$  represents a partial order relationship between a pair of units  $(u, v)$ , where  $u, v \in V$ . An edge  $u \leftarrow v$  can have the meaning that  $v$  verifies/confirms/witnesses  $u$ . An edge can also represent a hash reference from one unit to another. A directed graph  $G$  is a directed acyclic graph if  $G$  has no directed cycles, whereas a *directed cycle* is a sequence of vertices where the first and last vertices in the sequence are the same, and all edges are oriented in the same direction.

The introduction of multiple references per block results in the formation of a Directed Acyclic Graph (DAG). In a traditional linear blockchain, the longest chain rule serves a dual purpose: agreeing on the included blocks and establishing their order. However, when transitioning from linear chains to DAGs, these two functions need to be replaced. To address this, two primary classes of protocols have emerged: those that perform these tasks directly on the DAG itself, and Byzantine Fault Tolerant (BFT) protocols that utilize supplementary broadcast primitives.

The concept of using DAG in a ledger protocol was first introduced by Sompolinsky et al. in GHOST protocol [5] which addresses the concurrency problem of transactions in Bitcoin by building a tree instead of a linear chain. Later, an improved version of the protocol [23] was utilized in Ethereum [24]. In GHOST protocol, a newly created (mined) block does not refer to the leaf block of the longest chain. Instead, starting from the genesis block and at each fork, the branch with the heaviest sub-tree underneath is chosen until a leaf block is found. Once the leaf block is found, the newly mined block connects to it by including the hash pointer of the leaf block. Therefore, in the GHOST protocol, blocks created by the forks also contribute to the final consensus while maintaining robustness and high throughput.

Later, IOTA [6], Byteball [25], and many other DAG-based DLT protocols adopted the paradigm of using individual transactions as vertices in a DAG. With this modification, the transactions are disseminated and confirmed faster. In the following years, many works and studies on DAG-based

DLTs are conducted. These works update and modify the DAG structure, and improve the performance by requiring less computation, communication, and storage.

The structure of the DAG becomes very relevant while achieving the consensus and proving the security guarantees. Many Proof-of-Work (PoW) employed unstructured DAG-based protocols such as SPECTRE [26], PHANTOM [27], and DAG KNIGHT [28] have been proposed. In protocols like SPECTRE, PHANTOM and DAG KNIGHT, a newly created block has to cross-refer (point) to all the visible tips (leaf) in the ledger. The ordering of all these blocks in the DAG becomes the heart of the consensus protocol. Every node locally runs the ordering procedure and returns a linear ordering over the blocks in its local DAG.

Structured DAG-based protocols such as Aleph [29], DAG-Rider [18], Tusk [19], Bullshark [8], Cordial Miners [30] work in rounds in a permissioned setting, i.e., there are fixed  $n$  nodes that can add a block to the DAG, of which  $f$  nodes are possibly faulty or Byzantine. Every round has at most  $n$  blocks (one for each node<sup>1</sup>) where every newly created block must refer to  $n - f$  blocks from the previous round. In these protocols, each node maintains its local version of the DAG and runs the consensus protocol through logical interpretation of the DAG, i.e., blocks represent transaction proposals and references serve as votes. Therefore, consensus does not require any extra communication complexity. These protocols are leader-based where a leader is elected after a constant number of rounds (wave) using the encoded randomness in each wave in the asynchronous model or the round-robin method in the partially synchronous model. Once the elected leader block is committed and every node performs ordering on the leader block's casual history.

Out of the above protocols, only a few stress fairness while catering to high throughput. Under the assumption of their network model, in these DAG-based protocols, transactions and blocks propagated by the nodes/users in the P2P network might take a long time before a validator node might receive them. Due to that, validator nodes only see a portion of all the users' transactions and blocks ever produced in the network. Therefore, providing fairness to all the issued transactions and blocks is a challenging but important task.

### III. DIFFERENT NOTIONS OF FAIRNESS IN DLTs

#### A. Fairness to the Participants

1) *For Validators*: Fairness for the validators refers to having an equal chance for each validator node<sup>2</sup> to append its block to a distributed ledger irrespective of communication latency and the validators' computing power. A slow yet honest validator node cannot be distinguished from an adversarial node, as it becomes challenging to identify the cause of delay in the received block or transaction.

<sup>1</sup>Cordial Miners allows multiple blocks from one faulty node per round. To mitigate equivocating blocks, this protocol suggests the following: once an honest node detects an equivocator, the further equivocator's blocks are not directly referenced by the honest node.

<sup>2</sup>Throughout the paper, we use validator node and node interchangeably unless explicitly specified.

Fairness in this prospect has been taken into account in DLT protocols such as Prime [31], HoneyBadger [32], and Fairledger [33]. These are BFT-based blockchain protocols. To ensure fairness in these protocols, transactions from multiple nodes are consolidated into batches, which are then committed atomically within each epoch. Furthermore, these protocols incorporate mechanisms to identify and penalize nodes that deviate from the prescribed rules, thus bolstering fairness in favor of the honest nodes.

A recent line of work in DAG-based DLTs starting from Aleph [29], DAG-Rider [18], Tusk [19], and Bullshark [8] emphasise on fairness to the validators. During the construction of the DAG in certain DAG-based DLTs, a degree of fairness is attained by mandating the inclusion of a minimum of  $n - f$  blocks from different validator nodes in the previous round. To provide fairness for all the validators, some of these protocols [8], [18] use weak links to append slow validators' blocks. The idea of a weak link is to reference a tip (a block that is not yet referenced) in the distant past together with other "recent" tips while creating a new block in the DAG.

Fairness for validators ensures censorship resistance. It also shows a strong notion of participation equity which refers to the Chain Quality property of DLT, meaning that the ledger includes a certain portion of honest validators' blocks. Therefore, to enforce fairness for the validator nodes, the new block should include references to the slow validator's block either by a weak link in structured DAG-based DLTs or by referencing all the current tips.

2) *For Clients*: Transactions in DLTs can be delayed inappropriately by a malicious validator node. A validator node can prevent sending the transactions received from a client to the rest of the DLT network. This particular phenomenon can be termed as a compromise of fair access due to the prevention of a client's transaction from fairly accessing the network relative to other clients' transactions.

Fairness for a client can be defined as fair access to the client's transaction to the rest of the DLT network. It means when a validator node receives a client transaction, the node should send the transaction to the rest of the network without delay. To provide fairness to multiple clients connected to a validator, the validator can either process the clients' transactions in "First In First Out"(FIFO) order [34] or by monitoring the latency of the transaction requests [35].

In protocols like Narwhal [19], a client transmits its transaction to all the validator nodes to ensure fairness although it increases the bandwidth usage. If the client's transaction is not sequenced in the ledger in time, the client re-submits its transaction. However, re-submitting the transaction to multiple validator nodes not only wastes excessive resources such as bandwidth and CPU but also comes under the radar of request duplication attacks. This attack might harm honest clients if there is a penalty involved in every re-transmission of honest clients' transactions, for example, in Hedera Hashgraph [36]. To prevent this duplication attack, a technique used in Mir-BFT [37] based on hashing can be employed in DAG-based DLTs to provide fairness. On the

positive side, the dissemination of transactions is encoded in the very structure of the Hashgraph. Each transaction carries with it the gossip history which is used to achieve consensus on the validity and order of transactions.

A malicious validator node could violate fair access. It can delete or ignore client transactions. It can also delay a client's transaction and favor other clients' transactions in order to punish the client or front-run the client's transaction. With the lack of delay in terms of fair access compromise, the client's transaction is transmitted and received later in the network and hence the transaction is added later in the ledger than it should have been.

Clients can achieve fair access to their transactions in different ways as follows:

- Client can run a node that relieves the requirement of middleman proxy for the client in the DLT network, e.g., Permissionless Hedera [36].
- Client can submit a transaction to multiple or all validators in the DLT network [19].
- Client can submit a transaction to a node it trusts or the node having a better rating if the nodes maintain a reputation system within the DLT network [38].
- Client can submit a transaction anonymously along with providing proof of correctness to the transaction.

### B. Fairness in Consensus Ordering

Fairness in consensus ordering aka *Order-Fairness* ensures that the order of transactions relative to each other is fair. There are different ways to ensure fairness in transaction ordering. Order-Fairness is described in many DLT protocols such as Helix [39], Hashgraph [36]. Kelkar et al. [14] presents a good overview of order fairness in Byzantine consensus protocols. In a similar direction, Kursawe [13] unveils a group of low overhead protocols, Wendy, which can implement different notions of order-fairness in the blockchain. Another interesting work [40] manifests a notion of content-agnostic order-fairness.

Fairness in ordering can be abused by recording the transaction inappropriately, meaning the transaction that arrived first is recorded as received later. In other words, it is referred to as order manipulation. Particularly in a leader-based consensus of a DLT, the consensus algorithm elects a leader who is responsible for ordering and can delay the transaction of a specific client while ordering. This phenomenon of manipulation of transaction ordering is called the Miner Extractable Value (MEV) where a validator (leader/miner) maximizes its profit by including, excluding, or changing the order of client's transactions within blocks. MEV was first introduced as a measure in Flash Boys 2.0 [41], later it was named as Maximal Extractable Value.

Fair ordering of transactions can solve the problem of front-running attacks and censorship resistance in DLTs. The need for the fair ordering of transactions becomes clear when we look at financial systems such as Ethereum [24]. A recent report [42] delineates a comprehensive study of MEV extraction on Ethereum and an overview of policymakers on MEV. Another latest paper [15] presents a Systematization

of Knowledge (SoK) on MEV countermeasures where a brief description of common types of MEV and transaction ordering methods are shown in a precise way.

In the pursuit of achieving order-fairness and mitigating Miner Extractable Value (MEV) in DLT systems, two primary defensive measures are commonly employed: 1) Time-based Order-Fairness and 2) Blind Order-Fairness.

1) *Time-based Order-Fairness*: This protects from order manipulation by defining certain properties about transaction ordering based on transaction arrival/sent time, and by satisfying these properties. There are multiple definitions for Time-based Order-Fairness in the literature [13], [14]. From these literature, following we exhibit a few informal definitions of this order-fairness. Let  $N$  represent the total number of validator nodes, and  $\alpha$  is a fraction parameter denoting enough number of validator nodes.

#### • Receive-Order-Fairness

*Definition 1: (Receive-Order-Fairness)* It states that if a sufficiently large number  $\alpha N$  (with  $\alpha > 0.5$ ) of validator nodes receive a transaction  $tx_1$  before a transaction  $tx_2$ , then  $tx_1$  should be ordered before  $tx_2$  in the final ordering of transactions agreed upon by the validator nodes.

For the above definition to work, we need to make a few assumptions: 1) The network must be synchronous; 2) The adversary must be non-colluding and non-corrupting. Otherwise, it is almost impossible to achieve receive-order-fairness. In some sense, this describes a distributed FIFO notion.

#### • Send-Order-Fairness

*Definition 2: (Send-Order-Fairness)* It states that if a transaction  $tx_1$  is sent before a transaction  $tx_2$ , then  $tx_1$  should be ordered before  $tx_2$  in the agreed-upon transaction log by the validator nodes.

For the above definition to work, we need a trusted way to timestamp the transaction at the client's end. One potential way to generate a trusted timestamp is by using Trusted Execution Environment (TEE) such as Intel SGX [43] or ARM Trustzone [44] at the client side. Nevertheless, even after having a timestamp mechanism, network synchrony still plays a role to ensure that the transaction is not arbitrarily delayed.

#### • Approximate-Order-Fairness

*Definition 3: (Approximate-Order-Fairness)* For any two transactions  $tx_1$  and  $tx_2$ , let  $n$  validator nodes receive both transactions in a given time-frame and let  $\epsilon$  be the number of rounds  $tx_1$  and  $tx_2$  are apart. Approximate-Order-Fairness states that if  $\alpha n$  nodes receive  $tx_1$  more than  $\epsilon$  rounds before  $tx_2$ , then for every honest node  $j$ ,  $j$  does not deliver  $tx_2$ , unless it has previously delivered  $tx_1$ .

#### • Block-Order-Fairness

*Definition 4: (Block-Order-Fairness)* For any two transactions  $tx_1$  and  $tx_2$ , let  $n$  validator nodes receive both transactions in a given time-frame and let  $\alpha > \frac{1}{2}$ . Block-Order-Fairness states that if at least  $\alpha n$  nodes receive  $tx_1$  before  $tx_2$ , then for every honest node  $j$ ,  $j$  does not deliver  $tx_1$  in a later block than it delivers  $tx_2$ .

The last two definitions are weaker notions of order-fairness. These two definitions have inherent limitations related to the network model and relative measures. To overcome these limitations, Cachin et al. [45] presented a new notion of order-fairness, which they call *Differential Order-Fairness*.

Although most of the above definitions are used in the context of Byzantine consensus, these definitions can also be used to provide order-fairness in DAG-based DLT protocols.

2) *Blind Order-Fairness*: This provides MEV protection by committing to a transaction ordering without seeing the content of the transactions. That means this type of ordering is determined independently of transaction content. It is also referred to as content-agnostic ordering. The natural way of performing such ordering is by realizing through commit-reveal protocols.

The idea of a commit-reveal protocol for ordering is to first commit the transactions and reveal the committed transactions once the ordering has already been determined. Nevertheless, there is an inherent leakage of metadata in this ordering which makes the ordering weaker compared to time-based ordering. Helix protocol [39] employs commit-reveal to provide fair order. Although Helix provides censorship resistance it suffers from order manipulation due to metadata leakage.

The commit-reveal is performed by clients using some cryptographic primitive. Clients send their encrypted transactions as commitments along with metadata to the validator nodes. Then the consensus protocol commits to an ordering of transactions. Later, the commitments are opened and the transactions are executed. The commit-reveal step can be instantiated using different cryptographic primitives. Heimbach and Wattenhofer [46] present a Systematization of Knowledge of commit-reveal approaches using different cryptographic primitives in on-chain and off-chain. These approaches can also be helpful in DLT to perform the commit-reveal step. We present a brief overview of some of these primitives and their usage in DLTs to achieve blind order-fairness as follows.

- **Threshold Encryption** Threshold encryption allows a threshold number of honest validator nodes to retrieve committed transactions. In a threshold encryption scheme, during setup, the validator nodes generate a public key, and the corresponding secret key is shared among validator nodes. Using this scheme, a client encrypts a transaction using a public key in the commit phase, later in the reveal phase the message is decrypted using the decryption shares from a threshold number of validator nodes.

Malkhi and Szalachowski [40] construct a DAG-based protocol Fino which employs a threshold encryption scheme to achieve blind order-fairness. They also present other approaches, i.e., verifiable secret sharing scheme [47], AVID-M [48] to achieve blind order-fairness in DAG-based DLTs.

- **Trusted Execution Environment (TEE)** A TEE is a hardware-protected secure area that provides an isolated execution environment for code and data. Hence, it provides confidentiality and integrity of code and data. The state-of-the-art popular implementation of TEE includes Intel SGX [43] and ARM Trustzone [44].

To provide blind order-fairness, for a round, a client first generates a key pair inside TEE and encrypts its transaction using the public key. TEE can be programmed in a way such that the decryption key for the round is released only after the ordering for the round has already been done. TEE-employed order-fairness can be used in the DAG-based DLT such as Teegraph [49] which is already designed using TEE.

- **Time-lock Puzzle** Time-lock puzzle [50] can be used to encrypt the transaction as a commitment and the decryption only takes place after enough time has elapsed. For the time-lock puzzle to work in the ordering protocol, the global time-lock parameters must be set so that only after the transaction ordering, the encrypted transactions can be decrypted. Moreover, an adversary should not be able to solve a time-lock puzzle faster than an honest client.

Another cryptographic primitive related to the time-lock puzzle is delay encryption [51] where all clients encrypt their transactions under the same key for a given round. Delay encryption has been used to provide fairness in FairPoS consensus protocol [52]. These primitives have the potential to be used in any DLTs.

### C. Fairness in terms of DLT Components

- 1) *Block-level Fairness*: Achieving block-level fairness can be approached through various methods. One approach involves establishing weak links that connect the blocks of slow yet honest validators during the creation of new blocks in the DLT. Attaining block-level fairness is challenging. However, some degree of fairness can be achieved through the tip selection mechanism. If the mechanism selects a tip from a distant past, there is a higher likelihood of including the block from a slow honest validator in the ledger. However, achieving this level of fairness necessitates validator nodes to possess substantial storage capacity to store the complete DLT history, which can be practically challenging to implement without any relaxation. Additionally, discarding old blocks poses difficulties, as it may inadvertently remove blocks from slow validators that were never added to the ledger. DAG-Rider and Bullshark are protocols that employ weak links to achieve block-level fairness.

- 2) *Transaction-level Fairness*: This principle asserts that transactions received by an honest validator node will eventually be included in the ledger. Transaction-level fairness is particularly relevant for ensuring censorship resistance and is considered a more realistic measure than block-level fairness. To achieve transaction-level fairness, it is crucial for honest validator nodes to promptly relay client transactions to the DLT network upon receiving them. However, due to network connectivity issues, such as those encountered by slow validator nodes, transactions or blocks may not reach all nodes in the network. In such cases, transaction-level fairness can be upheld by re-injecting the transactions from a non-committed block belonging to a slow honest validator into a newly created block. This process ensures that the transactions ultimately find their way into the ledger.

## IV. DISCUSSION

In the ensuing discussion, we delve into various aspects and implications of the findings presented in the preceding sections, shedding light on the broader implications of the research and exploring potential avenues for further investigation. In a DLT, fairness notions are directly or indirectly affected by different components or functionalities plugged into it. In this section, we elaborate on the relation of some of these components with fairness notions.

### A. Writing Access

Writing access permits nodes to propose and write blocks, hence progressing the state of the ledger. A DLT protocol can pertain to a high degree of parallelism by having concurrent writing access by all the participants.

The fairness experienced by participants in a system is influenced by the writing access policies in place. In transaction-based DAG ledgers, where nodes add transactions to the ledger, providing writing access to all nodes ensures fairness among them. On the other hand, block-based DAG ledgers involve different node roles (validator, miner, ordinary, follower), and only a subset of nodes (such as miners) possess the permission to append blocks. This restricted access can introduce fairness limitations to the system. Thus, a more open writing access approach leads to a fairer system overall. Currently, the prevailing writing access solutions in the field encompass PoW and PoS-based lotteries, alongside the permissioned setting.

Let us mention a few works that allow more general writing access. Zhao et al. [53] compose an improved access control mechanism for DAG-based DLTs to improve the security and robustness of the network. The work focuses on fairness in the aspect of network parameters such as fairness in dissemination rate and latency. Cullen et al. [54] achieve fairness in an adversarial environment by proposing an access control mechanism for DAG-based DLTs. They ensure fairness during congestion using a notion of weighted max-min fairness that shows each node gets a fair share of throughput weighted by the reputation score of each node.

### B. Garbage Collection

In DAG-based DLT protocols, nodes require unbounded memory to guarantee validity and fairness. This arises the practical challenges for these protocols to be deployed. Due to the requirement of unbounded memory, nodes are not allowed to garbage collect the old rounds. Otherwise, an honest slow node block might be garbage collected before being added to the DAG. As per this scenario, garbage collection directly conflicts with fairness, especially in an asynchronous network model where a block or message can be delayed for an indefinite time. Garbage collection is a trade-off between fairness and performance.

Narwhal [19] employs a garbage collection mechanism to clean the DAG up to a certain round in the past from the genesis. Nevertheless, fairness at the block-level is sacrificed because slow nodes might be garbage collected before they get a chance to be ordered. Bullshark [8] on the other hand, achieves fairness for all the nodes in an asynchronous

network while having a garbage collection mechanism in place. Bullshark gives a practical implementation where nodes have bounded memory and fairness is achieved after Global Stabilization Time (GST) during the period of synchrony.

The garbage collection mechanism is rather new in the DAG-based DLT space. Mechanisms similar to the one in Bullshark can be adopted in other DAG-based DLTs. This kind of mechanism is also required for any high-throughput DLT since eventually, the underlying data structure has to be pruned.

### C. Reward Mechanism

In distributed ledgers, the reward (incentive) mechanism plays a crucial role to ensure fairness among the participants. The reward mechanism should reward the nodes according to their effort put forward to make the protocol progress.

In the majority of DLTs, expected rewards are typically proportional to the delivered Proof-of-Work (PoW) or Proof-of-Stake (PoS), as well as the performance of the nodes, resulting in a fair distribution. However, as a second-order effect, nodes with higher PoW or PoS tend to exhibit less variance in their profits and incur lower costs in running their nodes. This phenomenon can lead to centralization, where a small number of nodes with significant resources dominate the network.

To address this potential centralization, many PoS systems impose a cap on the maximum amount of staked funds. This limitation aims to create a more decentralized and fairer distribution of rewards.

Several DAG-based DLTs do not have a reward mechanism as being “feeless” is an advantage for those DAG-based DLTs, e.g., [6]. Moreover, a few DAG-based DLTs e.g., Inclusive [23], Sphinx [55] employ reward mechanisms to increase network participation, and Graphchain [56] introduces its reward mechanism to maintain the DAG.

Some DLT protocols do not uniformly distribute the rewards and hence lack fairness. For example, scalable BFT protocols [57] employing threshold signature distribute the rewards to every node no matter whether the node participated or not. Nonetheless, Kokoris-Kogias [58] builds a fair reward mechanism in the novel design of robust and scalable BFT protocol which can be deployed in BFT-based DLT.

In SUI protocol [59], which is based on Bullshark [8], fairness is achieved by rewarding validators based on their behavior and performance, as perceived by other validators. This is done through subjective reporting which is then used as input into the stake reward distribution rule.

### D. Tip Selection

In the Nakamoto consensus mechanism [1], each block refers to a single previous block, resulting in an ever-extending chain. Conversely, DAG-based systems allow blocks to reference multiple preceding blocks, leading to a more intricate structure as opposed to a straightforward, linear chain. As a result of this partial order of vertices in a constructed DAG, the tip selection algorithm becomes an integral part of the associated consensus mechanism. Fairness in the context of tip

selection algorithms generally refers to an equal opportunity for all transactions to get selected and validated regardless of their origin, ensuring that no transaction gets left out. We list a few tip selection algorithms:

- **Uniform Random Tip Selection** To issue a new block, a node chooses tips to approve uniformly at random from all *good* tips in the DAG until a fixed number  $k$  of references is created, e.g., IOTA [9], [60]. This algorithm is specifically designed for synchronous network models and permissionless DAG-based DLTs since the size of the references should be negligible compared to the block size. A good tip refers to a tip that is valid and consistent with the finalized ledger and issued rather recently. Such tip selection gives a fair chance to all good tips in the tip pool to be referenced and eventually approved. Moreover, it leads to a Nash Equilibrium, i.e., knowing that other nodes select tips with equal probabilities, a given node has nothing to gain from deviating from this behavior, see [61].
- **Round-based Tip Selection** To issue a new block at round  $r$ , a node chooses at least  $n - f$  tips in the DAG from round  $r - 1$ , e.g., Aleph [29], Bullshark [8], Cordial Miners [30], DAG-Rider [18], Tusk [19]. Such round-based tip selection algorithms are used in permissioned DAG-based DLT protocols which work under partially synchronous or asynchronous network models. The number  $n - f$  is the maximum guaranteed number of tips one can expect if  $f$  nodes are faulty or Byzantine. Moreover, certain protocols [8], [18], offer some sort of fairness to slower nodes. This is achieved by allowing nodes to reference blocks from previous rounds using weak links. These weak links facilitate the inclusion of transactions from slower nodes into the ledger, while not being utilized for voting purposes.
- **Heaviest Cluster Tip Selection** To issue a new block, a node selects all tips from the heaviest cluster, the largest subset of blocks in the DAG, in which each block is not comparable by the partial order with at most a constant number of other blocks, e.g., Phantom [27], DAG-KNIGHT [28]. This approach achieves certain fairness as a new block references all existing tips in the heaviest subDAG.
- **Sub-sampled Voting Tip Selection** To issue a new block, a node first queries small random samples of other nodes and based on their responses finds the tips that are likely to be approved by the network, e.g., [62] and [9]. Due to its nature, the algorithm works with permissionless DAG-based DLT protocols. This tip selection algorithm is considered fair as with a high probability it reflects the preference of the majority of the network.

### E. Smart Contract Architecture

The increased level of parallelism in DAG-based DLTs shows promise in addressing contention issues within UTXO, Cardano, and object-based smart contracts like SUI [63]. These smart contracts offer distinct advantages compared to account-based smart contracts, particularly in terms of determinism. The deterministic nature of these smart contracts significantly reduces the occurrence of Miner Extractable Value (MEV) possibilities.

The inherent deterministic execution model and parallelism in object-based smart contract systems provide notable advantages in terms of fairness, scalability, and security. The deterministic nature of object-based smart contracts ensures predictable outcomes, facilitating auditing and verification processes. Overall, these advantages position object-based smart contracts as an attractive option for achieving fairness and performance in decentralized applications within the context of DAG-based DLTs.

## V. CONCLUSION

In this study, we have conducted a thorough analysis of fairness in DAG-based DLTs, defining different notions and examining their entanglement with DLT components. Promising research directions include exploring alternative methods for fairness without weak references and investigating cryptographic schemes for blind order-fairness. By addressing these areas, we can enhance the understanding and implementation of fairness in DAG-based DLTs, fostering more equitable and robust systems.

## REFERENCES

- [1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized business review*, p. 21260, 2008.
- [2] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2084–2123, 2016.
- [3] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, "Sok: Layer-two blockchain protocols," in *Financial Cryptography and Data Security: 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10–14, 2020 Revised Selected Papers 24*. Springer, 2020, pp. 201–226.
- [4] G. Wang, Z. J. Shi, M. Nixon, and S. Han, "Sok: Sharding on blockchain," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 2019, pp. 41–61.
- [5] Y. Sompolinsky and A. Zohar, "Accelerating bitcoin's transaction processing. fast money grows on trees, not chains," *Cryptology ePrint Archive*, 2013.
- [6] S. Popov, "The tangle," *White paper*, vol. 1, no. 3, p. 30, 2018.
- [7] Q. Wang, J. Yu, S. Chen, and Y. Xiang, "Sok: Dag-based blockchain systems," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023.
- [8] A. Spiegelman, N. Giridharan, A. Sonnino, and L. Kokoris-Kogias, "Bullshark: Dag bft protocols made practical," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2705–2718.
- [9] S. Popov, H. Moog, D. Camargo, A. Caposelle, V. Dimitrov, A. Gal, A. Greve, B. Kusmierz, S. Mueller, A. Penzkofer *et al.*, "The coordicide," *Accessed Jan*, pp. 1–30, 2020.
- [10] M. Conti, E. S. Kumar, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3416–3452, 2018.
- [11] Y. Amoussou-Guenou, A. Del Pozzo, M. Potop-Butucaru, and S. Tucci-Piergiovanni, "On fairness in committee-based blockchains," *arXiv preprint arXiv:1910.09786*, 2019.
- [12] Y. Huang, J. Tang, Q. Cong, A. Lim, and J. Xu, "Do the rich get richer? fairness analysis for blockchain incentives," in *Proceedings of the 2021 International Conference on Management of Data*, 2021, pp. 790–803.
- [13] K. Kursawe, "Wendy, the good little fairness widget: Achieving order fairness for blockchains," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 25–36.
- [14] M. Kelkar, F. Zhang, S. Goldfeder, and A. Juels, "Order-fairness for byzantine consensus," in *Advances in Cryptology—CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part III 40*. Springer, 2020, pp. 451–480.
- [15] S. Yang, F. Zhang, K. Huang, X. Chen, Y. Yang, and F. Zhu, "Sok: Mev countermeasures: Theory and practice," *arXiv preprint arXiv:2212.05111*, 2022.

- [16] EIGENPHI, “MEV Outlook 2023: Walking Through the Dark Forest,” Feb 2023, accessed: 15-Jun-2023. [Online]. Available: <https://eigenphi.substack.com/p/mev-outlook-2023>
- [17] G. Birmpas, E. Koutsoupias, P. Lazos, and F. J. Marmolejo-Cossío, “Fairness and efficiency in dag-based cryptocurrencies,” in *Financial Cryptography and Data Security: 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10–14, 2020 Revised Selected Papers*. Springer, 2020, pp. 79–96.
- [18] I. Keidar, E. Kokoris-Kogias, O. Naor, and A. Spiegelman, “All you need is dag,” in *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, 2021, pp. 165–175.
- [19] G. Danezis, L. Kokoris-Kogias, A. Sonnino, and A. Spiegelman, “Narwhal and tusk: a dag-based mempool and efficient bft consensus,” in *Proceedings of the Seventeenth European Conference on Computer Systems*, 2022, pp. 34–50.
- [20] L. Vigneri, W. Welz, A. Gal, and V. Dimitrov, “Achieving fairness in the tangle through an adaptive rate control algorithm,” in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2019, pp. 146–148.
- [21] L. Vigneri and W. Welz, “On the fairness of distributed ledger technologies for the internet of things,” in *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2020, pp. 1–3.
- [22] S. Müller, A. Penzkofer, D. Camargo, and O. Saa, “On fairness in voting consensus protocols,” in *Intelligent Computing: Proceedings of the 2021 Computing Conference, Volume 2*. Springer, 2021, pp. 927–939.
- [23] Y. Lewenberg, Y. Sompolinsky, and A. Zohar, “Inclusive block chain protocols,” in *Financial Cryptography and Data Security: 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26–30, 2015, Revised Selected Papers 19*. Springer, 2015, pp. 528–547.
- [24] G. Wood *et al.*, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.
- [25] A. Churyumov, “Byteball: A decentralized system for storage and transfer of value,” URL <https://byteball.org/Byteball.pdf>, p. 11, 2016.
- [26] Y. Sompolinsky, Y. Lewenberg, and A. Zohar, “Spectre: A fast and scalable cryptocurrency protocol,” *Cryptology ePrint Archive*, 2016.
- [27] Y. Sompolinsky and A. Zohar, “Phantom,” *IACR Cryptology ePrint Archive, Report 2018/104*, 2018.
- [28] Y. Sompolinsky and M. Sutton, “The dag knight protocol: A parameterless generalization of nakamoto consensus,” *Cryptology ePrint Archive*, 2022.
- [29] A. Gagol, D. Leśniak, D. Straszak, and M. Świątek, “Aleph: Efficient atomic broadcast in asynchronous networks with byzantine nodes,” in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 2019, pp. 214–228.
- [30] I. Keidar, O. Naor, and E. Shapiro, “Cordial miners: A family of simple, efficient and self-contained consensus protocols for every eventuality,” *arXiv preprint arXiv:2205.09174*, 2022.
- [31] Y. Amir, B. Coan, J. Kirsch, and J. Lane, “Prime: Byzantine replication under attack,” *IEEE transactions on dependable and secure computing*, vol. 8, no. 4, pp. 564–577, 2010.
- [32] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, “The honey badger of bft protocols,” in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 31–42.
- [33] K. Lev-Ari, A. Spiegelman, I. Keidar, and D. Malkhi, “Fairledger: A fair blockchain protocol for financial institutions,” *arXiv preprint arXiv:1906.03819*, 2019.
- [34] S. Duan, K. Levitt, H. Meling, S. Peisert, and H. Zhang, “Byzid: Byzantine fault tolerance from intrusion detection,” in *2014 IEEE 33rd International Symposium on Reliable Distributed Systems*. IEEE, 2014, pp. 253–264.
- [35] P.-L. Aublin, S. B. Mokhtar, and V. Quéma, “Rbft: Redundant byzantine fault tolerance,” in *2013 IEEE 33rd International Conference on Distributed Computing Systems*. IEEE, 2013, pp. 297–306.
- [36] L. Baird, M. Harmon, and P. Madsen, “Hedera: A public hashgraph network & governing council,” *White Paper*, vol. 1, pp. 9–10, 2019.
- [37] C. Stathakopoulou, T. David, M. Pavlovic, and M. Vukolić, “Mir-bft: High-throughput robust bft for decentralized networks,” *arXiv preprint arXiv:1906.05552*, 2019.
- [38] C. Huang, Z. Wang, H. Chen, Q. Hu, Q. Zhang, W. Wang, and X. Guan, “Repchain: A reputation-based secure, fast, and high incentive blockchain system via sharding,” *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4291–4304, 2020.
- [39] A. Asayag, G. Cohen, I. Grayevsky, M. Leshkowitz, O. Rottenstreich, R. Tamari, and D. Yakira, “A fair consensus protocol for transaction ordering,” in *2018 IEEE 26th International Conference on Network Protocols (ICNP)*. IEEE, 2018, pp. 55–65.
- [40] D. Malkhi and P. Szalachowski, “Maximal extractable value (mev) protection on a dag,” *arXiv preprint arXiv:2208.00940*, 2022.
- [41] P. Daiian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels, “Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability,” in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 910–927.
- [42] M. Barczentewicz, “Mev on ethereum: A policy analysis,” *ICLE White Paper*, pp. 01–23, 2023.
- [43] V. Costan and S. Devadas, “Intel sgx explained,” *Cryptology ePrint Archive*, 2016.
- [44] S. Pinto and N. Santos, “Demystifying arm trustzone: A comprehensive survey,” *ACM computing surveys (CSUR)*, vol. 51, no. 6, pp. 1–36, 2019.
- [45] C. Cachin, J. Micić, N. Steinhauer, and L. Zanolini, “Quick order fairness,” in *Financial Cryptography and Data Security: 26th International Conference, FC 2022, Grenada, May 2–6, 2022, Revised Selected Papers*. Springer, 2022, pp. 316–333.
- [46] L. Heimbach and R. Wattenhofer, “Sok: Preventing transaction reordering manipulations in decentralized finance,” *arXiv preprint arXiv:2203.11520*, 2022.
- [47] A. Shamir, “How to share a secret,” *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
- [48] Y. Zhang, S. Setty, Q. Chen, L. Zhou, and L. Alvisi, “Byzantine ordered consensus without byzantine oligarchy,” *Cryptology ePrint Archive*, 2020.
- [49] X. Fu, H. Wang, P. Shi, and X. Zhang, “Teegraph: A blockchain consensus algorithm based on tee and dag for data sharing in iot,” *Journal of Systems Architecture*, vol. 122, p. 102344, 2022.
- [50] R. L. Rivest, A. Shamir, and D. A. Wagner, “Time-lock puzzles and timed-release crypto,” *Massachusetts Institute of Technology. Laboratory for Computer Science*, 1996.
- [51] J. Burdges and L. De Feo, “Delay encryption,” in *Advances in Cryptology—EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I*. Springer, 2021, pp. 302–326.
- [52] J. H.-y. Chiang, B. David, I. Eyal, and T. Gong, “Fairpos: Input fairness in proof-of-stake with adaptive security,” *Cryptology ePrint Archive*, 2022.
- [53] L. Zhao, L. Vigneri, A. Cullen, W. Sanders, P. Ferraro, and R. Shorten, “Secure access control for dag-based distributed ledgers,” *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 10 792–10 806, 2021.
- [54] A. Cullen, P. Ferraro, W. Sanders, L. Vigneri, and R. Shorten, “Access control for distributed ledgers in the internet of things: A networking approach,” *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2277–2292, 2021.
- [55] Q. Wang and R. Li, “A weak consensus algorithm and its application to high-performance blockchain,” in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.
- [56] X. Boyen, C. Carr, and T. Haines, “Graphchain: A blockchain-free scalable decentralised ledger,” in *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts*, 2018, pp. 21–33.
- [57] G. Gueta, I. Abraham, S. Grossman, D. Malkhi, B. Pinkas, M. REITER, D. SEREDINSCHI, O. Tamir, and A. Tomescu, “Sbft: a scalable decentralized trust infrastructure for blockchains, 2018,” 1804.
- [58] E. Kokoris-Kogias, “Robust and scalable consensus for sharded distributed ledgers,” *Cryptology ePrint Archive*, 2019.
- [59] The Mysten Labs Team, “The sui smart contracts platform: Economics and incentives,” 2022, accessed: 17.06.2023. [Online]. Available: <https://github.com/MystenLabs/sui/blob/main/doc/paper/tokenomics.pdf>
- [60] S. Müller, A. Penzkofer, N. Polyanski, J. Theis, W. Sanders, and H. Moog, “Tangle 2.0 leaderless nakamoto consensus on the heaviest dag,” *IEEE Access*, vol. 10, pp. 105 807–105 842, 2022.
- [61] S. Popov, O. Saa, and P. Finardi, “Equilibria in the tangle,” *Computers & Industrial Engineering*, vol. 136, pp. 160–172, 2019.
- [62] T. Rocket, M. Yin, K. Sekniqi, R. van Renesse, and E. G. Sirer, “Scalable and probabilistic leaderless bft consensus through metastability,” *arXiv preprint arXiv:1906.08936*, 2019.
- [63] The Mysten Labs Team, “The sui smart contracts platform,” 2022, accessed: 17.06.2023. [Online]. Available: <https://github.com/MystenLabs/sui/blob/main/doc/paper/sui.pdf>