Peer-to-Peer Networking and Applications (2021) 14:3660–3673
https://doi.org/10.1007/s12083-021-01165-8

Enforcing Fairness in Blockchain Transaction Ordering

Ariel Orda[1] · Ori Rottenstreich[1]

Received: 2 December 2020 / Accepted: 11 April 2021 / Published online: 7 July 2021

Abstract
In Blockchain networks involving multiple applications, the quality of service of an application is affected by the transaction ordering. For instance, upon issuing payment transactions, users of an application would like to be notified quickly on the transactions approval. The application can be a financial institution (such as a bank), sharing the blockchain with other such applications and is represented by a node. A node might attempt to prioritize its own transactions by including them early in blocks added to the blockchain. A fair block proposal of a node follows a random selection of the transactions among the set of pending transactions the node is aware of. On the contrary, a dishonest node includes more of its transactions at the expense of transactions of other applications. In this work, we propose a toolbox of techniques to enforce such a fair block selection. First, we design an accurate statistical test for the honesty of a proposal and explain it. We conduct experiments demonstrating the accuracy of the new validation scheme. We also describe a technique that enforces fair block selection through concise commitments on the set of pending transactions known to a node. We clarify the advantages of the new mechanisms over state-of-the-art methods.

Keywords Blockchain · Consensus mechanisms · Block Selection · Fairness

1 Introduction

Blockchain is a growing list of records (often called *transactions*), managed in a distributed manner among multiple participants. Transactions can either be simple money transfers or some more general pieces of code such as Ethereum smart contracts [26]. The blockchain is organized in *blocks*, each composed of multiple transactions. The blockchain can be shared among multiple independent financial institutions such as banks or more general applications. Blockchain applications are diverse and beyond money transfers include supply chains, electronic voting and medical informatics. Recently, applications have been described in areas such as data sharing for Industrial Internet of things (IIoT), 5G-based crowdsourcing, integration in Large-Scale Heterogeneous Networks (LS-HetNet) and even fighting COVID-19 [28–31].

Typically, participants reach an agreement on the blockchain content through an agreed-upon addition of a new block. The new block is determined as a selection of transactions among the pool of pending transactions, namely transactions that have been issued by one of the participants but do not yet appear in the blockchain. The block selection process implies an order on the transactions. In addition to the agreement on the block order, a consensus is also required for implying updates to the state of the blockchain, namely either account balances or memory accessed by smart contracts.

While in some networks the block is jointly determined by multiple participants (e.g., HoneyBadger [23]), typically a block is proposed by a selected node. A node might have complete freedom in the selection of the transactions blocks (e.g., as in Bitcoin [24] and Ethereum [26], where a miner can select those of the highest fees, thus maximizing its profit). Another approach restricts the freedom in block selection by implying a review process for the block selection by other participants [15]. These nodes, often organized as a *committee*, can validate the selection according to some required criteria, such that each node indicates whether or not to accept the proposal. Incentives might be used to encourage nodes to avoid manipulations in the block selection. When there is a particular node allowed to present a block proposal, we refer to that node as the *primary*.

As the block rate is bounded, an important aspect in the block selection is *fairness* [3, 4, 27]. By nature, nodes that share the same blockchain can have contradicting considerations and thus might imply a competition to fast include in

✉ Ori Rottenstreich
or@technion.ac.il

Ariel Orda
ariel@ee.technion.ac.il

1 Technion, Haifa, Israel

blocks transactions prioritized by each of the nodes. A recently suggested protocol, named *Helix* [2], suggested a concrete method for the block selection that the primary is expected to follow for implementing a random block selection. The primary should sort its local pool of pending transactions according to a sorting function that changes every round of block proposal. Then, given a maximal block size of b transactions, the primary should simply include in the proposed block the b transactions with the lowest ranking based on the computed order.

n this paper, we study *how such a desired fairness in the block selection can and should be enforced.* An inherent challenge in validating a block proposal follows directly from the nature of distributed blockchain networks. While different nodes (typically) agree on the content of the blockchain, they are not fully aware of all pending transactions, such that nodes are often exposed to non-identical sets of pending transactions. This makes it hard, or indeed impossible, for a validator to simply reject a proposal when it does not include a transaction that, by the view of the validator, was expected to be included. The validator cannot clearly indicate that the primary was aware of that transaction and ignored it on purpose to serve other transactions it prioritizes.

Helix [2] describes a statistical test to examine whether a proposed block followed these instructions. It relies on a (simplifying) model where a node has a fixed probability to be aware of a pending transaction. For a given transaction, this event is independent among the various nodes. In Helix, a committee member examines some level of similarity between the proposed block and the locally computed one (while making use of information from the actual block proposal). This helps the committee member to make a *binary* decision whether to accept the proposal or not. A minimal number of accepting nodes among the committee members is required for the block proposal to be approved and added to the blockchain. We note that the validation performed by Helix does not take advantage of all available information and we detail ways it can be improved in order to strengthen the fairness.

Contributions. In this paper we make two main contributions, as follows.

Our *first contribution* is an improvement of Helix's block validation scheme. Specifically, we explain that, following the statistical model for transaction dissemination, the validation process in Helix does not fully utilize the information of the committee members. We show that a more accurate decision should be a *joint decision* of the various committee members rather than simply being based on the number of independent approvals. We propose how to determine the validity of the proposal based on the aggregated information from the members. Specifically, we describe a simple formula for the probability that a proposal is honest following the complete information (unlike the difficulty to do so given the independent decisions of Helix with partial information). We conduct experiments to evaluate the accuracy of the new scheme in comparison with that of Helix.

The *second contribution* takes a different approach and enforces honesty upfront. Specifically, we establish a technique that effectively eliminates nodes from the option to ignore any transactions for increasing the number of prioritized transactions in a proposed blocks. We explain that a periodic *concise* report of the set of known transactions to a node can be highly useful towards such a goal. Our scheme relies on the observation that, since the transaction sorting cannot be predicted, a node is not aware in advance of the specific transactions it would like to ignore in a particular round. The techniques we describe make use of various data structures, such as Bloom filters [9] or Merkle trees [8, 22] and their variants. Hashing is a useful technique in such data structures, for mapping elements to areas of the report as well as for providing a signature for a reported element.

The rest of the paper is organized as follows. Section 2 summarizes related work. Section 3 overviews the considered settings and the problem statement. Section 4 overviews the block proposal and validation procedure in Helix. Then, Section 5 details the suggested alternative validation process of a block proposal. In Section 6 we describe experiments that evaluate the accuracy of the proposed block validation process. Then, Section 7 presents the scheme for enhancing fairness in block proposals through node declarations regarding their pending transactions. Finally, concluding remarks are presented in Section 8.

2 Related Work

Fairness in Blockchain Systems and Beyond: As mentioned, the approach of Helix [2] provides fairness through allowing pending transactions to be selected with the same probability to a block through random block selection [23]. Sokolik et al. suggested to reduce the tail-latency of the time it takes a transaction to be included in a block by giving priority in the block selection to transactions observing high latency [17]. The notion of fairness among transactions is defined differently in [3], as each node gets a fair share of the ledger. Namely, each block contains the same number of transactions from each node assuming that they have infinite streams of transactions. Another related definition is due to *Receive-order-fairness* [4] which enforces transaction selection such that if many nodes learned about a transaction before some other transactions, such an order should be reflected in the ledger. Weaker potential definitions refer to the unfairness of the order of two transactions only if they were received sufficiently apart in time. Another option is to ignore the internal order of transactions within blocks and only refer to the order of transactions in different blocks. Wendy [27] handles *relative order fairness* and claims for fairness requirements only for subsets of transactions, e.g., those belonging to each of several existing markets. This approach is different than that of Helix, where the fairness is kept between the different applications and it is assumed that a transaction has no priority regarding the internal order of its transactions. Table 1 overviews these different fairness aspects.

Beyond blockchain, aspects of fairness have been studied in other networking and computer system settings where a restricted resource is shared among multiple entities. Examples include a queue with a bounded service rate or a link with limited capacity [5]. A well-known notion is that of *max-min fairness*, which suggests how to determine a resource partition based on the demands of multiple users, summing up to more than the resource availability. Intuitively, such a fair allocation tries to maximize the share of users of small demands. Generalizations of the definition have also been suggested [6].

Fee-based Block Selection: In commonly used blockchains based on Proof of Work (PoW) consensus, e.g. Bitcoin [24] and Ethereum [26], the selection process is not part of the protocol and can be decided by the proposer regardless of any fairness considerations. Since transactions might be associated with different fees, the block proposer typically selects those of maximal fee it is aware of in order to maximize its profit [12]. Our work assumes no fee per transaction so that fees have no influence on considerations whether to include a transaction in a block.

Primary Node Election: Various approaches exist for the election of the node with the right to propose a block. This can be based on computational power in PoW [11] through solving a mathematical puzzle that requires multiple hash computations, or according to a distribution implied by the balance of the nodes in Proof of Stake (PoS) [16, 18]). Additional alternatives in similar notions are Proof-of-Space [7] and Proof-of-Elapsed-Time [1], referring to the amount of memory being held or computation time spent by the participants, respectively. In the present work we refer to random node selection, as well as to selection based on other criteria, such as a reputation scheme.

3 Settings and Problem statement

Our blockchain network is a fully mesh network of multiple nodes. In the network, *a round* occurs every given time. In each round, some node is chosen as the block proposer, either randomly or by some criteria. The primary should construct a new block from its pending transactions pool *randomly in a fair manner, without prioritizing neither its own transactions nor those of any of the other nodes*. The primary then propagates the block through the network. A small fraction of nodes serve as committee members, and their task is to validate the proposed block.

The problem we study is to design a framework that allows an effective validation of the fairness in the block selection. The mechanism should be simple and refrain from consuming a large amount of communication. Moreover, it should be accurate, namely fair block selections should be approved while selections that are not performed according to the random selection guidelines should not pass validation.

Table 2 summarizes the main notations employed in this work.

4 Helix Block Selection and Validation - Background and Motivation

We begin this section by explaining the importance we see in Helix that motivated us to study its potential improvements. With the increasing popularity of the blockchain technology, more applications would be based on it in the near future. To increase governance and reduce centrality, as well as due to the technical implementation challenges of the technology, multiple applications would share the same infrastructure [14, 20]. To allow this sharing and avoid any abuse by some application, fairness of the transaction ordering among the applications must be guaranteed. Moreover, to allow massive use, an application is expected not to charge fees for transactions from its users, hence expenses for the use of the technology are covered by the application. Helix addresses this common scenario and aims to provide ordering fairness among transactions of all applications sharing the same blockchain infrastructure. The importance of transaction ordering was

Table 1 Overview of existing fairness aspects in blockchain systems

	Fairness aspect
Helix [2]	Similar probability for a transaction to be selected for a block
Age-aware fairness [17]	Prioritizing transactions with a large observed latency in block selection
Fair share [3]	Similar block parts among nodes
Receive-order-fairness [4]	Transaction order is based on time nodes learn on each transaction
Relative order fairness (Wendy) [27]	Internal fairness within subsets of transactions

Table 2 Summary of main notations

Symbol	Meaning
b	block size (number of transactions)
α	similarity parameter, probability for a pending transaction to appear in a pool
EB	Proposed block of transactions
EP	Pool of pending transactions
b'	Number of transactions ($\geq b$) in pool among which the block is selected

recently demonstrated in studies that appeared following the proposal of Helix, including [3, 4, 17, 27].

We proceed to provide an overview of the block selection and its validation process in Helix [2]. A block is selected by a primary p from the pool of pending transactions it is aware of EP_p. In such a pool, transactions are maintained in an encrypted form. Due to network latency, the pools EP_i and EP_j of two different nodes i, j might differ. However, we may assume a measure of similarity between two pools of pending transactions. To model this similarity, we use a probabilistic model satisfying the following property. For any two correct nodes i, j, each etx in EP_i is in EP_j with probability at least α. We refer to α as the *similarity parameter* of the network.

The Helix block selection scheme uses a hash function in order to serialize candidate *etx*s for the next block. The hash function is tweaked with a random seed RS to eliminate its predictability, yielding a common, random and unpredictable serialization of the *etx*s. The random seed is a function of the content the block from the previous round. Till a block is selected, transactions appear in an encrypted form so that it is difficult to predict the random seed from the transactions following their decryption. Formally, when considering the block in term r, the nodes order the pending *etx*s according to the values $H(RS^{r-1}, etx)$, and refer to these values as the *hash values* of the *etx*s. We use the notation $H(etx)$ for brevity.

Denote by b the maximal allowed block size (number of transactions in a block). Let EB_p be a block proposed by the primary p and T_p be the maximal hash of an etx in EB_p, i.e., $T_p := \max\{H(etx)|etx \in EB_p\}$. Likewise, denote by $EB_i^{'} := \{etx \in EP_i|H(etx) \leq T_p\}$ the set of *etx*s in EP_i with hash values lower than T_p, and $b_i := \max\{|EB_i^{'}|, b\}$. We further denote by $b^{'}$ the size of $EB_p^{'} = \{etx \in EP_p|H(etx) \leq T_p\}$ and say that EB_p was constructed under a $b^{'}$ -construction. This illustrates the fact that EB_p was selected as a subset of size b among the $b^{'}$ lowest hashed *etx*s in EP_p such that the b'^{th} etx was included. The setting is illustrated in Fig. 1.

Under these notations, the validation checks (in the context of selection fairness) performed by a committee member i upon receiving a proposed block, EB_p (from primary p), are:

1. $|EB_p| = b$
2. $|EB_p \cap EB_i^{'}| \geq \beta_\alpha(b_i)$ for $\beta_\alpha(b_i) := \alpha b_i - \sqrt{10 b_i}$

The second condition encourages primaries to construct blocks with low b'. The minimal value of b' is b; in the event that the value of b' is in fact b, the selection scheme is perfectly fair. Intuitively, a larger b' allows the primary more freedom in the selection of EB_p (rather than selecting it as the b minimal *etx*s). However, since we can expect $|EB_p^{'}| \approx |EB_i^{'}|$, large b' yields large b_i and accordingly large $\beta_\alpha(b_i)$, reducing the chances of EB_p to pass validation. $\beta_\alpha(b_i)$ is the maximal value for which blocks constructed with $b' = \mathrm{b}$ pass validation w.o.p., as implied by Hoeffding's bound.

The pseudocode of the Helix block validation process is presented in Algorithms 1-2. Algorithm 1 refers to the process conducted by a committee while the validation described in Algorithm 2 refers to each node in the committee.

The extra validation process dictated by the Helix block selection scheme bears a risk to the liveness of the protocol. Blocks that would have passed validation might get rejected once the statistical validation is enforced. It was shown that, w.o.p., a block compliant with the b-construction passes validation of a committee member that follows the protocol.

Property 1 Let EB_p be a block constructed according to the b-construction, and let i be a committee member following the protocol. Then, EB_p passes i's validation w.o.p. (under the assumption that α bounds from below the similarity parameter of the network).

5 Alternative Joint Block Validation

In this section we present an improvement of the Helix scheme. As summarized in Section 4, the block validation of Helix relies on an independent evaluation of the proposed block by each of the committee members. A committee member votes in favor of the proposal when the conditional probability for the block proposal to be fair is above a required lower bound. The probability is computed based on the content of the transaction pool of the committee member. A minimal number of votes in favor of the proposal are required for the block to be approved but finding the optimal value of this

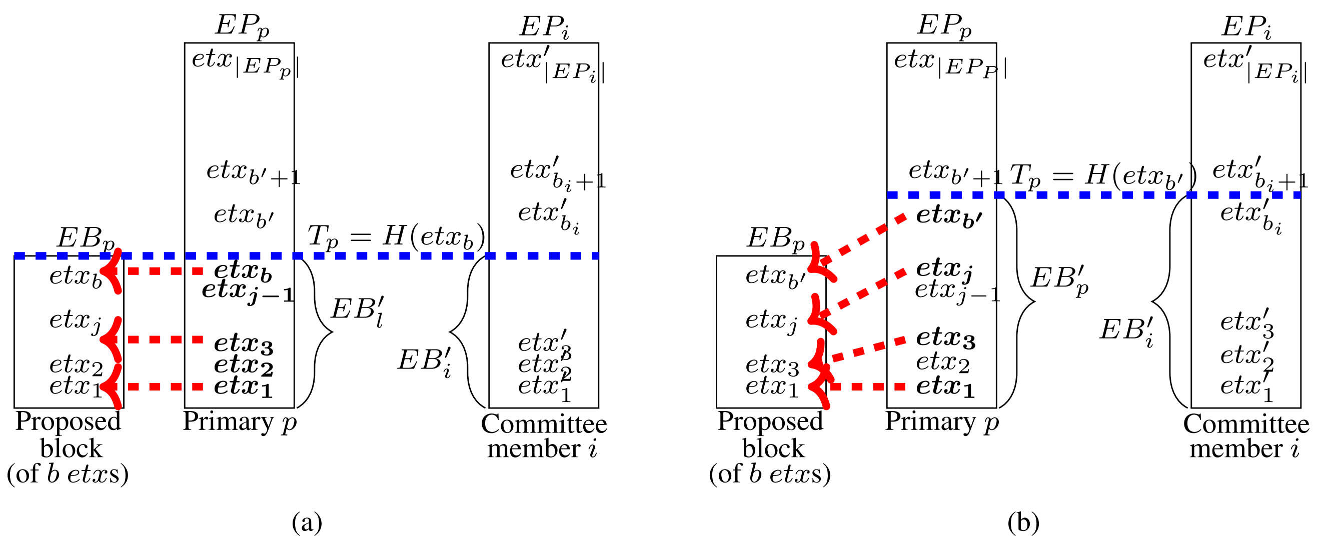


Fig. 1 Illustration of a random block selection scheme. In each Epool, *etx*s are sorted based on their hash values. A honest primary fairly constructs a block from the b transactions with the minimal hash value. An unfair block selection includes skipping transactions, selecting the b block transactions among some $b' > $ b transactions with minimal hash values. A committee member i examines the block proposal by computing the overlap between the proposed block EB_p and the set of *etx*s in a block computed locally based on the local pool of pending transactions EP_i

(minimal) number seems difficult and is not addressed by Helix. On the positive side, this scheme does not require communication between the committee members prior to their voting.

In this section, we explain that such a voting criterion does not utilize all information available by the committee members. We follow the assumption of Helix, namely that an issued transaction has the fixed probability α to appear in a pool of pending transactions by each node, such that for a given transaction this probability is independent among the various nodes. Accordingly, a block proposal of a honest primary should include, with probability α, a transaction issued by other nodes satisfying a bound on its hash value. On the other hand, a proposal of a dishonest primary would be selective and its number of such included transactions is expected to be lower than that implied by such a distribution. We proceed to describe an alternative and more accurate validation process than that proposed for Helix. The process requires communication among the committee members earlier to the *joint* indication regarding the honesty of the block proposal.

Fig. 2 illustrates the intuition of the new scheme and the way it differs from previous approaches. As shown in (a), the traditional scheme of Helix block validation is individual for each committee member. Conversely, in the proposed alternative shown in (b), block validation is performed jointly, and communication among the committee members is required prior to providing a joint validation decision.

Intuitively, consider a transaction that the primary could be aware of, yet it did not include it in its block proposal. Intuitively, in Helix, each committee member that identifies that the transaction should have been included yet it is missing would have a lower chance to indicate the block as valid.

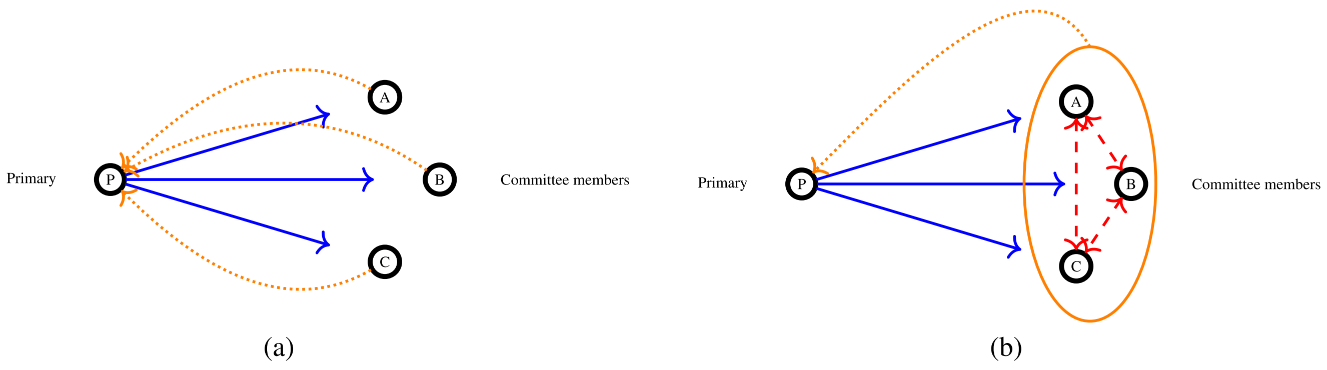


Fig. 2 Illustration of the traditional block validation procedure (a) vs. the suggested alternative joint Block validation (b) that includes communication among committee members that produce a joint decision. The joint validation can take into account additional information that allows a more accurate validation

Consider for simplicity the case where a block proposal of the primary p is evaluated by two committee members i, j. Following the definition of the probability α and its assumed model, we claim that the evaluation should make a distinction between the following cases:

- *Case I* - Node i holds a single transaction etx_1 that was not included in the block although being expected to, and another node j also holds a transaction $etx_2 \neq etx_1$ that was also not included in the block although being expected to.
- *Case II* - Both nodes i, j hold a single identical transaction etx_3 not included in the block although being expected to.

The reason a distinction should be made between the cases follows the different probabilities for the scenarios to occur with a honest primary given the existence of the transactions by the committee members. While in Case I, this happens with probability of $(1-\alpha)^2$, in Case II the corresponding probability is larger, namely it is $1-\alpha$. This means that when Case II is detected the primary has higher chances to be honest than in Case I.

The block validation of Helix does not distinguish between the two cases. The reason is that Helix conducts an independent validation at each of the committee members. Such validation does not make any distinction based on the identity of the missing transactions and accordingly cannot distinguish between such cases, as in both each committee member observes the missing of a single transaction. We propose an alternative approach, according to which a right evaluation should be based on examining the ratio of included transactions among those expected given the content of the local pools among the committee members. We show that the related probability of the primary being honest is affected by the identity of missing transactions in the block according to each committee member and not just by the numbers observed by each of the members. Namely, the probability is a function of the total *number of distinct missing transactions* rather than their sum among the committee members.

More specifically, the approval decision should determine whether the selection is fair. Thus, each transaction that has to be included should contribute equally to that decision, either positively if it is indeed included, or negatively if it is not included. This can be computed by the number of expected transactions to be included and the number of those among them that are indeed included. A precise determination of these numbers should ignore multiplicities of the same transaction among nodes and thus requires communication among them. Accordingly, the joint decision for the nodes is made based on these numbers. Unlike the scheme of Helix, in the proposed scheme there is no notion of a block approval by a single committee member and thus computing the minimal required number of such nodes is not necessary.

We proceed to analyze the probability for a particular scenario based on the above mentioned transaction numbers. Again, let EB_p be a block (of size $|EB_p| = b$) proposed by the primary p such that $T_p = \max\{H(etx)|etx \in EB_p\}$ is the maximal hash of an etx in EB_p. For a committee member i we denote by $EB_i^{'} := \{etx \in EP_i | H(etx) \leq T_p\}$ the set of etxs in EP_i with hash values lower than T_p and also denote $b_i :=$ max $\{|EB_i^{'}|, b\}$. Let $EB' = \bigcup_{\mathrm{i}} \mathrm{EB}_{\mathrm{i}}^{'}$ (where the union is computed over the committee members) be the set of all transactions that the committee members are aware of and should be included in the block as implied by T_p. Denote $b' = |\mathrm{EB}'|$. Let $EB' \cap \mathrm{EB_p}$ be the set of transactions among those expected that are indeed included in the block and let K be the random variable for their number. For a honest primary, K should follow a binomial distribution (α, b') such that $Pr(K=k) = \binom{b'}{k}\alpha^k (1-\alpha)^{b'-\mathrm{k}}$ and a probability for a honest node that an intersection of size k or less appears is given by $\Phi_{\alpha}^{k,b'} = \sum_{m=0}^{k} \times \binom{b'}{m}\alpha^m(1-\alpha)^{b'-\mathrm{m}}$. The joint decision of the committee members should then be to accept the block proposal whenever the computed probability satisfies some lower bound. The bound is selected as a tradeoff between the required liveness of the protocols and the probability to reject a proposal of a dishonest primary. A lower bound Φ_{min} guarantees that a proposal of a honest leader is accepted with at least such probability.

Algorithm 3 presents the pseudocode of the proposed joint block validation process.

We now provide some intuition regarding when we expect the joint validation to imply a meaningful improvement in the accuracy. These are precisely the instances when the original accuracy of Helix block validation is relatively low. Two major parameters that have an impact on the accuracy are the pool similarity parameter α and the number of committee members k. In Helix, a low pool similarity parameter α makes the accuracy of the indication of each node to be less accurate. Moreover, adding more committee members allows more votes, making the overall decision more accurate. Accordingly, we expect the joint block validation to allow a particularly significant improvement in the accuracy when either the pool similarity parameter is low or when committees are small.

6 Experimental Comparison of the Block Validation Schemes

We proceed to evaluate the accuracy of the suggested joint block validation from Section 5 in comparison with that of the traditional block validation in Helix, where each committee member provides its own decision regarding the honesty of a proposal. We recall that the two validation schemes consider as their input different information provided by the committee members. The schemes were illustrated in Fig. 2, where Fig. 2(a) corresponds to Helix's and Fig. 2(b) corresponds to the new (joint) scheme. The strictness of a validation scheme implies a tradeoff between two possible kinds of validation errors, namely: *False negative* - identifying a fair primary as unfair; and *False positive* - identifying an unfair primary as fair. In particular, we would like to examine the strength of the approach for: *(i)* different pool similarity levels; and *(ii)* variable block sizes. Table 3 summarizes the range of values of the main parameters examined in the experiments.

We examine five ways to select block transactions. The first is a fair selection (selecting the b block transactions as those of minimal hash values), while the other four refer to unfair selections with various levels of bias. They describe the selection of the b block transactions among a larger number $b' >$ b for $b'/\text{b} \in \{1.03, 1.06, 0.09, 1.12\}$, such that larger b $'/$b values refer to higher levels of unfairness with larger flexibility to prioritize transactions owned by the primary.

A transaction issued by a node that is still pending appears in the pool of each of the other nodes with probability α. We assume a block size of $b = 1000\,etx$s. We refer to a committee size of three nodes and, in the Helix block validation process, for a block to be approved, its (individual) approval by each of the committee members is required. The results are based on $10000 = 10$ K runs of the experiment.

In Fig. 3 we assume a pool similarity level of $\alpha = 0.9$ and compare the accuracy of the validation schemes. Fig. 3a depicts the accuracy of schemes based on Helix, while Fig. 3b depicts the accuracy of the suggested joint validation scheme. Each scheme can use various levels of strictness to balance between the possible errors. For each scheme and level of strictness, we first count the number of times that the fair block selection was wrongly identified as being unfair. This false negative error stands for the x-axis of the two figures. In applying the same test (with the same strictness) on the four ways of unfair block selection, we also measure the number of times that unfair block selections were wrongly identified as fair selections (false positives, shown in the y-axis). Note that, for both figures, the y-axis appears in logarithmic scale.

For both validation schemes, allowing more errors of one type reduces the number of occurrences of the other error. Moreover, the accuracy of the schemes is higher when the block selection unfairness is more extreme, as expressed by larger $b'/$b values. We can see that it is quite difficult to identify unfair selection with $b'/\text{b} = 1.03$, such that the number of false positives among these cases is in the range [6307, 9284] (among the 10K examined instances) for the Helix validation scheme, assuming a bounded number of at most 100 false negatives. It decreases to be in the range [4753, 9285] for the joint validation scheme. Larger $b'/$b values, such as 1.06, 1.09, express higher levels of unfairness of the primary and allow better distinction between fair and unfair selections. For instance, with $b'/\text{b} = 1.06$, the number of false positives is in [385, 1599] for Helix validation and only in [223, 1510] in joint validation. When $b'/\text{b} = 1.12$, it is easy for both schemes to correctly identify such a selection as unfair, and at most two cases of false positives were observed (for a particular false positive value), even when the number of false positives was close to 0.

Fig. 3c emphasizes the differences between the results in Fig. 3a and Fig. 3b, and it presents the partial reduction in the number of false positives. A clear reduction of up to 21.5% appears even for the smallest examined level of unfairness with $b'/\text{b} = 1.03$. Interestingly, the partial improvement is even more significant for $b'/\text{b} = 1.06, 1.09$, with mean partial improvement values of 38.6% and 29.3%, respectively. Due to the high level of unfairness with $b'/\text{b} = 1.12$, it can be well observed by both schemes hence they typically achieve similar performance.

Next, we examine the impact of network quality and demonstrate that the superiority of the suggested joint block validation scheme becomes significant when the pool similarity α is relatively small, as can be implied by larger transaction propagation delays. Fig. 4 compares the accuracy of the schemes when the probability of a transaction to appear in a pool of a node that did not issue the transaction is $\alpha = 0.75$. The lower similarity between pools reduces the quality of the block validation schemes. Consider for instance the Helix validation scheme (Fig. 4a) and block selection unfairness $b'/$b $= 1.06$. For 20, 40 and 80 false negatives, the number of false positives increases from 729, 582 and 458 with $\alpha = 0.9$, to 4852, 4044 and 3182 with $\alpha = 0.75$. Similarly, for the joint

Table 3 Values of main parameters

Symbol	Meaning	Range of values
b	block size (number of transactions)	125 - 8K
α	similarity parameter, probability for a pending transaction to appear in a pool	0.75 - 0.9
$b'/$b	Number of transactions in pool among which block is selected vs. original block size	1.03 - 1.12
Instances	Experiment length	10K

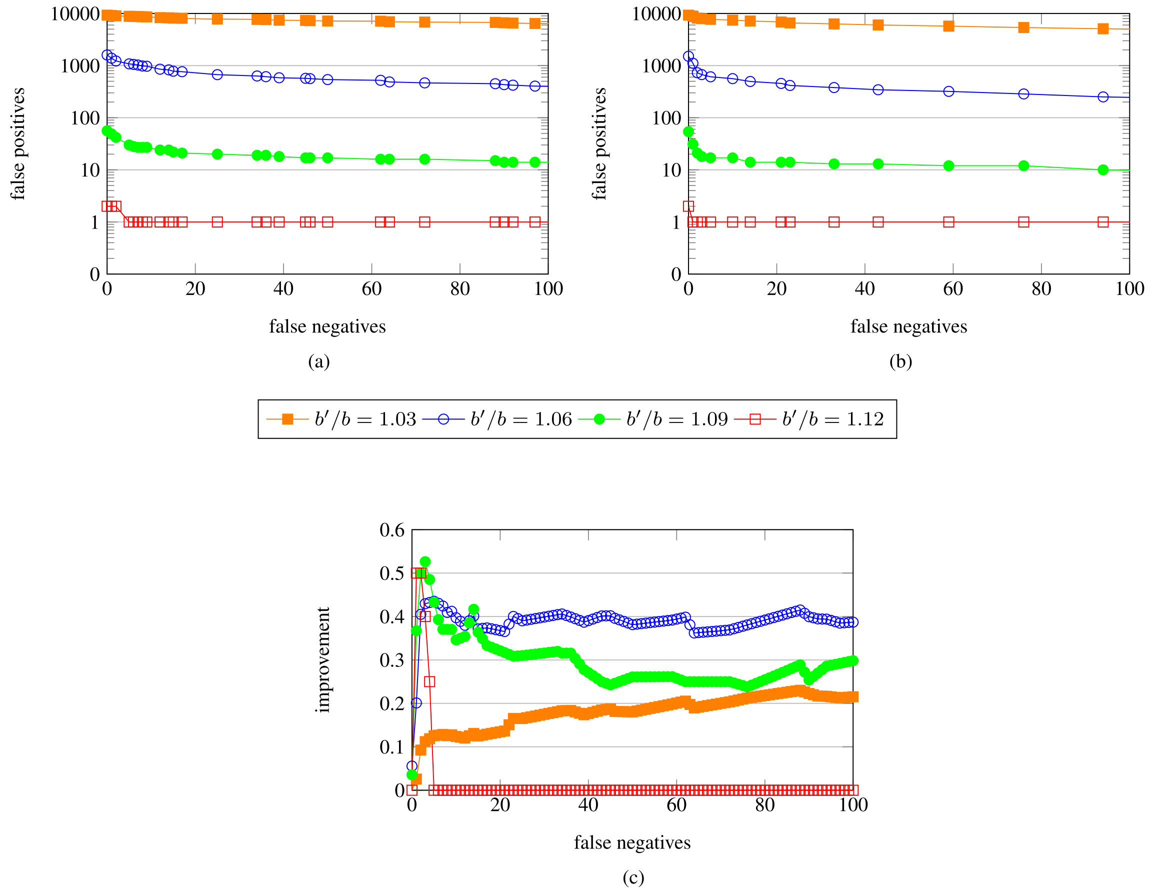


Fig. 3 Comparison of the accuracy of the block validation of Helix and the suggested joint validation. Pool similarity $\alpha = 0.9$, block size $b = 1000$. Evaluation included 10000 instances of block selection validation

validation scheme (Fig. 4b), the values are lower but still increase with the reduction of α, specifically: From 461, 355 and 278 with $\alpha = 0.9$, to 2889, 2256 and 1711 with $\alpha = 0.75$. Here, even for a high unfairness level $b'/\mathrm{b} = 1.12$, both schemes are not fully accurate, and the number of observed false positives is not negligible. The number of false positives is in the range [3, 74] for the Helix scheme and in the range [0, 24] for the joint scheme. Interestingly, as shown in Fig. 4c, the advantage of the suggested joint validation scheme becomes more significant for such a lower pool similarity value α, and the improvement is typically stronger for larger values of b'/b. The mean partial improvement values are 8.9% and 43.6% for $b'/\mathrm{b} = 1.03, 1.06$, and they grow to 70.9% and 81.4% for $b'/\mathrm{b} = 1.09, 1.12$. Note that, in a small number of instances for $b'/\mathrm{b} = 1.12$, no false positives were observed for the joint validation when the number of false negatives was relatively large.

We also examine the impact of block size on the accuracy of the validation schemes. In Fig. 5 we measure their accuracy while considering various block size of $b \in [125, 250, 500, 1$ K, 2K, 4K, 8K$]$ transactions. Intuitively, for a given level of block selection unfairness, a larger block size involves the ignorance of a larger number of transactions that could be included in the block by a fair block selection. Thus, larger blocks can simplify the detection of unfair selections and improve the accuracy of both schemes. The four sub-figures refer to the different levels of selection unfairness with $b'/\mathrm{b} \in \{1.03, 1.06, 1.09, 1.12\}$. Consider for instance an unfairness level of $b'/\mathrm{b} = 1.03$, as shown in Fig. 5a. We recall that, as was shown in Fig. 3, the accuracy of the schemes was not very high for a block size of $b = 1000 = 1$ K. For 100 false negatives for instance, we have 6394 false positives for the Helix scheme and 4864 for the joint validation scheme. As shown here, the situation is even worse for a smaller block size. For $b = 500$ for instance, allowing 100 false negatives implies 8962 false positives for Helix and 8380 false positives for the joint scheme. On the other hand, increasing block size to $b = 2$ K reduces false positives to 2102 and 1139 in the Helix and joint schemes, respectively. A block size of $b = 4$ K implies very small numbers, of only 53 and 17 false positives, respectively. Indeed, no false positives were observed for both schemes for a block size of $b = 8$ K, although the level of unfairness is as low as $b'/\mathrm{b} = 1.03$.

Higher levels of selection unfairness (Fig. 5b-(d)) make it easier to identify the unfairness. For instance, Fig. 5c refers to $b'/\mathrm{b} = 1.09$. Here, for block size $b = 500$, with 100 false

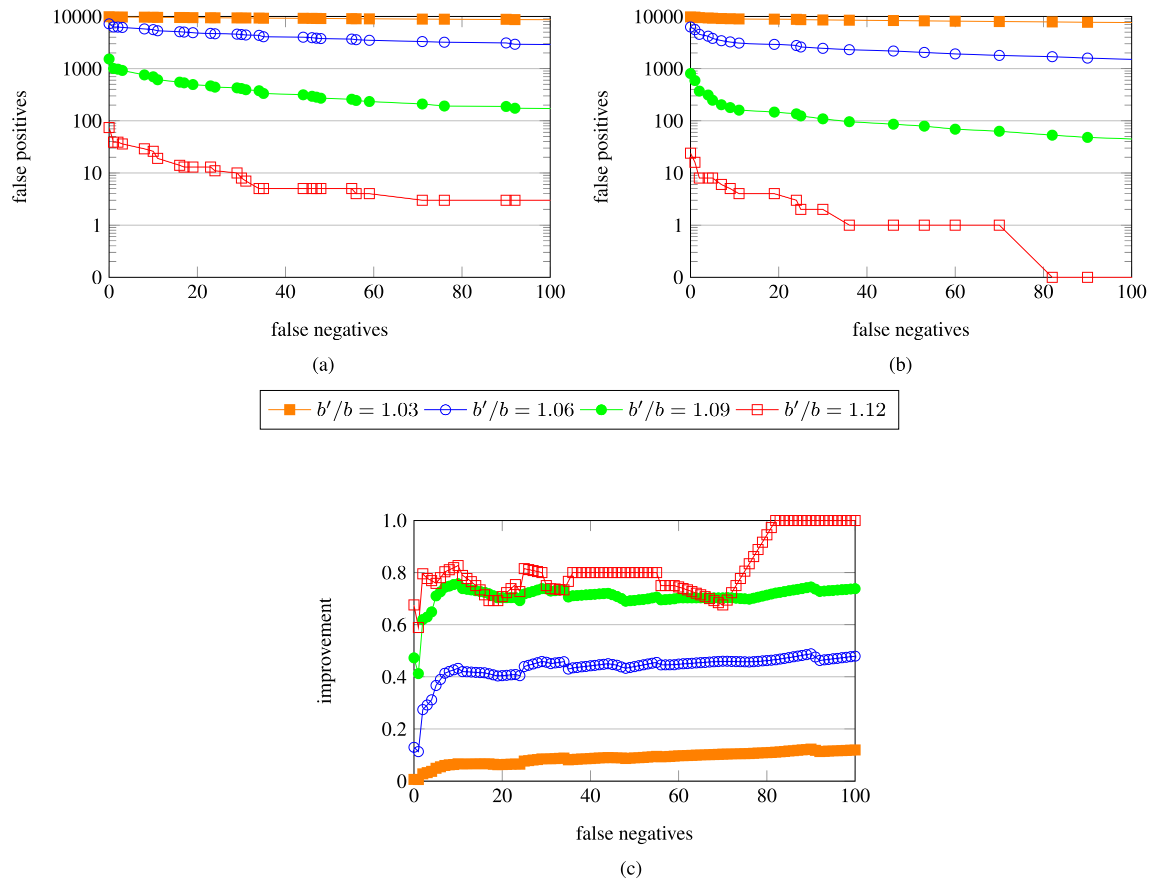


Fig. 4 Impact of higher transaction propagation delay with implying lower pool similarity $\alpha = 0.75$ (same block size $b = 1000$ and number of 10000 instances)

negatives the number of false positives is 664 for Helix and 495 for the joint scheme. For block size $b = 1$ K they reduce to just 13 and 7, respectively. No false positives were observed for a block size of 2K or more.

Fig. 6 shows the relative improvement (reduction in the number of false positives) of the joint validation schemes vs. Helix. We compare the reduction in false positives when the number of false negatives is either 5 or 100 (among the 10K instances). Again, for all examined block sizes, the number of false positives was smaller or equal for the joint scheme. As shown in Fig. 6a, for $b'/\mathrm{b} = 1.03$ the more significant improvements were obtained for block sizes $b = 1\mathrm{K}, 2\mathrm{K}, 4\mathrm{K}$. The maximal relative improvement was obtained for $b = 4$ K and was equal to 61.3% and 67.9% for 5 and 100 false negatives, respectively. As mentioned, for a smaller block size, the accuracy of both schemes was not very high, while for a large block size, namely $b = 8$ K, both achieved high accuracy. For a larger unfairness level of $b'/\mathrm{b} = 1.09$ (Fig. 6b), where schemes perform better, the partial improvement is more modest and is more enhanced for smaller block sizes such as $b = 500, 1$ K. For instance the improvement equals 3 0.1% for 5 false negatives and block size $b = 500$, and it reaches 46.2% for 100 false negatives and block size $b = 1$ K.

7 Declarations on Pending Transactions

7.1 Intuition for Declarations

We would like to further enhance the fairness of the block selection. Indeed, even with the alternative examination of the block proposal presented in Section 5, it might be possible for a primary to ignore a few particular transactions without being detected. In particular, when the block is small, it can be easier to manipulate some part of it. Also, when the network conditions imply low values of α, the validation process cannot be strict and manipulation in the block selection is easier to perform.

Accordinfly, we proceed to suggest an approach that is based on periodically asking nodes to *declare the set of transactions they are aware of*. Then, a block proposal would be tested based on a recent declaration of the primary. When provided, a declaration would be examined to include enough transactions of other nodes. A crucial point in this scheme is that, at the time of the declaration, a node is not aware of the random ordering of the transactions in a specific future round, thus it cannot predict those particular transactions of others it has to include and would like to ignore upon being selected as

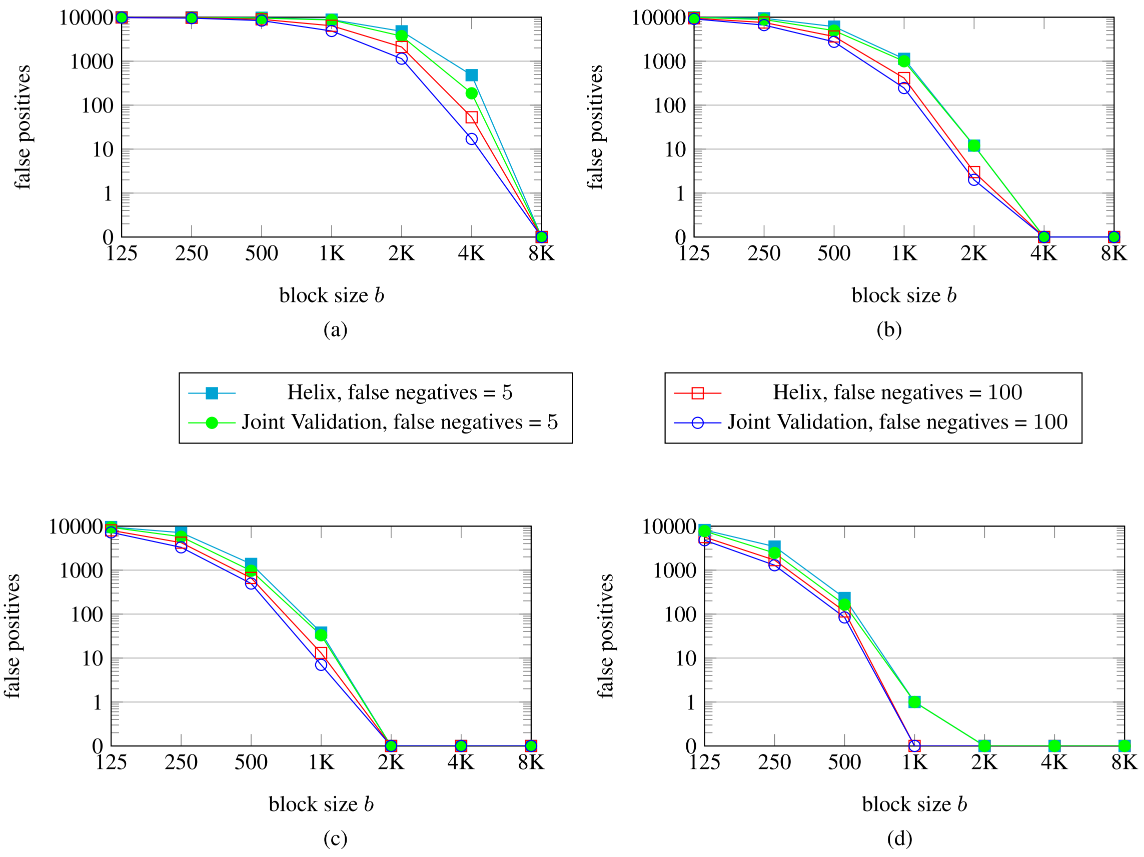


Fig. 5 Impact of block size on the accuracy for various unfairness levels (pool similarity $\alpha = 0.9$ based on a total of 10000 instances)

a primary. To allow itself to make some meaningful manipulation, it would have to make in advance a major self-adjustment to its declaration, hence significantly increasing the chance of being detected.

A clear correlation exists between the effectiveness of a declaration and its size, as follows. On the one hand, a detailed declaration can be more helpful for better validation, yet it may require a large communication overhead. On the other hand, a concise declaration (e.g., including partial information, or compressed information with loss) reduces the opportunity to detect missing transactions in the proposal. We believe that a restriction on the allowed amount of communication overhead typically exists, thus we focus on communication-efficient declarations.

A declaration on the set of pending transactions known to a node can be seen as a description of a set. Designing such representations, either exactly or concisely, while losing some information, is a well-studied research area with many

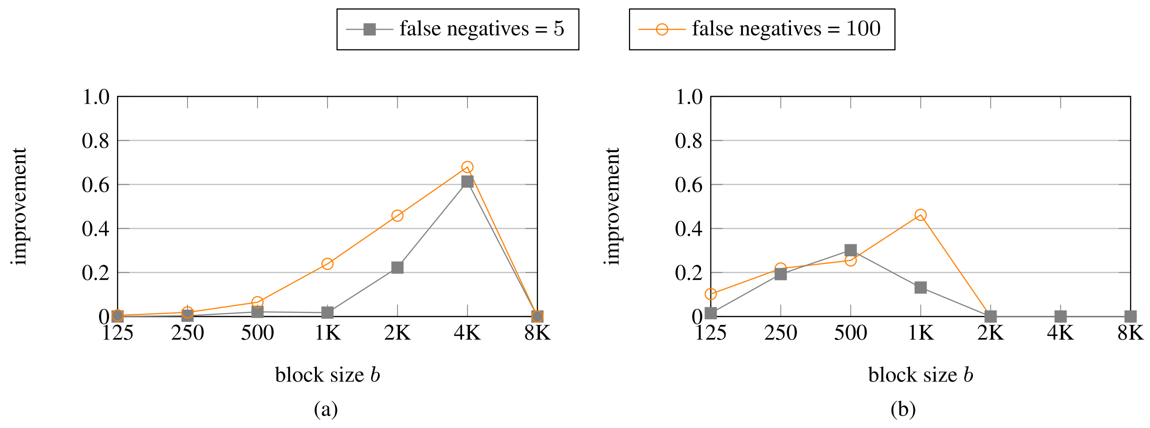


Fig. 6 Relative improvement of the joint validation scheme vs. Helix for various block size and unfairness levels (pool similarity $\alpha = 0.9$, 10000 instances)

applications [9, 21, 22]. The choice of the representation scheme is based on the application requirements, such as the support of answering membership queries, the allowed types of errors, the ability to prove the inclusion or exclusion of an element, and whether the order of elements has significance. For a representation of a set S, there are two kinds of errors in membership queries: a false positive (when an element $x \notin S$ is reported as a member of S) and a false negative (when an element $x \in S$ is reported as a non-member).

In the following, we present an overview of potential declaration schemes. We examine multiple criteria, such as the declaration size and the ability to easily examine a declaration or a block proposal. We also refer to the ability of a node to prove its honesty (in the block selection) by being able to show that a missing transaction was not included in a declaration. A high-level summary of the results is presented in Table 4.

7.2 Baseline - Reporting Complete List

As a baseline declaration, one might consider a declaration including the complete list of transactions known to a node. Such a detailed declaration would be long, implying a large communication overhead. On the positive side, such a declaration can be easily tested when proposed and later be useful in a simple validation of the honesty of a block proposal. To validate the declaration, a node examines that the complete list includes a large portion of its transactions. A block proposal is examined by a node by making sure that each of its transactions, missing although being expected to appear in a block following the block hash threshold, does not appear in the declaration, namely is not known to the primary.

7.3 Bloom Filter based Declarations

The Bloom filter [9, 19] is a popular data structure for set representation, supporting element insertion and answering membership queries. It is used for multiple blockchain purposes, such as summarizing the set of transactions in an Ethereum block [26] or representing the addresses a Bitcoin SPV (light) client is interested in [13]. Beyond blockchain, it is also common in many networking schemes [10].

Table 4 Fundamental properties of various declaration schemes

Scheme	Declaration size	Declaration testing	Block testing	Proving honesty (non-membership)
Complete list	Large	Local	Local	Complete
Bloom filter	Small	Local	Partial	Partial
Merkle tree	Small	Commun.	Commun.	Complete

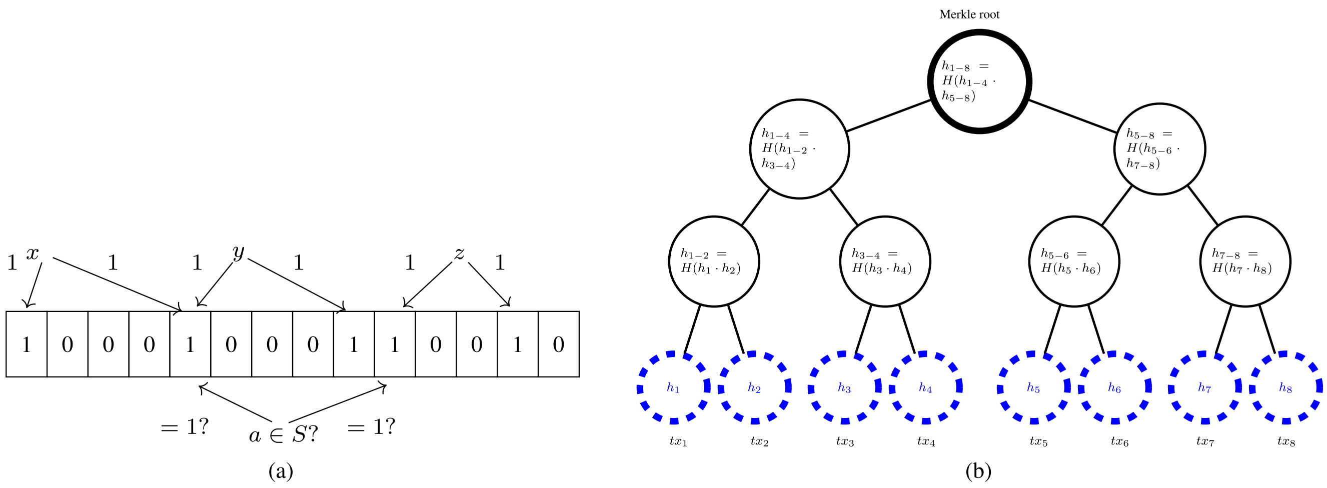
The Bloom filter encounters false positives and has no false negatives. The probability of an error (ratio of non-member elements reported as members) decreases when more memory is allocated for the data structure and increases when a larger set S is represented. The Bloom filter, illustrated in Fig. 7a, stores an array of bits, where a set of hash functions is used to map elements to locations in the bit array. With initial values of zero bits, the elements of S are first inserted to the filter, setting to a value of 1 all bits pointed by the hash functions. Upon a membership query, the bits mapped by the queried element are examined and a positive answer is returned only when all these bits have a value of 1.

We proceed to describe how the Bloom filter can be utilized for the enforcement of fairness. Every few rounds of block selections, each node summarizes in a filter the pending transactions from its local pool that were issued by other nodes (rather than by the node itself). The node then distributes the filter to other nodes, while reporting the Bloom parameters (namely the number of elements, filter length in bits and hash function number). Other nodes (e.g., those that belong to a committee, which is selected for this purpose in a hard-to-predict way) examine the filter validity. They would like to see that the filter includes, in the set represented by it, some portion of their transactions. The filter is approved upon achieving some minimal required support from the committee. A node whose filter was not approved is not selected as the primary for the next several rounds. For a node i, we denote by S_i the represented set of known transactions and by F_i the set of transactions with a positive indication in the Bloom filter such that $S_i \subseteq F_i$.

In the following rounds, one node is selected as the primary among those with an approved filter. The primary p proposes a block EB_p selected from the pending transactions that were available to the primary by the time of computing its most recent (and recently reported) Bloom filter. Assume that the maximal hash value of the block is $T_p = \max\{H(etx)|etx \in EB_p\}$. A committee member i with local pool EP_i computes $EB_i^{'} = \{etx \in EP_i|H(etx) \leq T_p\}$, the set of etxs in EP_i with hash values lower than T_p. In contrast to the Helix scheme, in its examination, node i does not simply compare $|EB_p \cap EB_i^{'}|$ to some lower bound but rather it carefully examines $\Delta_i = EB_i^{'} \setminus EB_p$, the set of transactions not included in the proposal although being expected to. If the primary is honest it must not be familiar with any such transactions.

Specifically, node i checks whether Δ_i is aligned with the Bloom filter reported by the primary p. Namely, for each $etx \in \Delta_i$ one out of the following two conditions should hold:

- $etx \notin F_p$, and the transaction was not reported by the primary as a pending transaction in its pool,
- $etx \in F_p$ and the transaction was reported as known to the primary, yet there was a false positive, namely $etx \notin S_p$.



Merkle root
$h_{1-8} = H(h_{1-4} \cdot h_{5-8})$
$h_{1-4} = H(h_{1-2} \cdot h_{3-4})$
$h_{5-8} = H(h_{5-6} \cdot h_{7-8})$
$h_{1-2} = H(h_1 \cdot h_2)$
$h_{3-4} = H(h_3 \cdot h_4)$
$h_{5-6} = H(h_5 \cdot h_6)$
$h_{7-8} = H(h_7 \cdot h_8)$
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8
tx_1 tx_2 tx_3 tx_4 tx_5 tx_6 tx_7 tx_8
x y z
1 0 0 0 1 0 0 0 1 1 0 0 1 0
$= 1?$ $a \in S?$ $= 1?$
(a)
(b)

Fig. 7 Illustration of the Bloom filter and the Merkle tree, two popular data structures for set representation

The validation of the declaration restricts the number of transactions of the first type, namely those satisfying $etx \in \Delta_i$, $etx \notin F_p$. The committee member i examines the transactions in $\Delta_i \subseteq S_i$. It skips those not in F_p, namely those for which the filter returns a negative membership indication. For those in F_p, since they are not included in the proposed block of the primary, they must not be known to the primary and the fact they belong to F_p is due to false positives. The committee member challenges the primary with the list of such transactions in $\Delta_i \cap F_p$. The primary has to demonstrate that they are indeed false positives by indicating on other transactions that the hash functions map to the same bits of the filter. Failing to do so indicates on the primary dishonesty.

Moreover, the properties of the Bloom filter do not enable the primary to (completely) prove its honesty, showing that all missing transactions were not among the set for which the declaration was computed for. The reason is that the Bloom filter does not enable proving non-membership of an element in the represented set for such elements causing the false positives, namely transactions in Δ_i that also appear in $F_p \setminus S_p$.

7.4 Merkle Tree based Declarations

Another common data structure in Blockchain networks is the Merkle tree [8, 22], also used for concise set representation while supporting different functionality than the Bloom filter. As illustrated in Fig. 7, the Merkle tree is a binary tree where a leaf is associated with a set element and its hash value. An internal node hash value is computed based on those of its direct children. The hash value of the root is the *Merkle root.*

Upon declaring the Merkle root for a represented set S, it is later possible to prove the inclusion of an element in S. The Merkle inclusion proof consists of the values of all the siblings of the nodes in a path to the root from the leaf corresponding to the element. Moreover, elements can be maintained in a sorted manner in the tree leaves. This enables to also demonstrate exclusion of an element through an exclusion proof, showing the inclusion as adjacent leaves of a predecessor and successor, with lower and higher hash values, respectively.

A declaration of a node i includes publishing the Merkle root for its pool of pending transactions. Testing the declaration by another node j cannot be done locally by node j and requires sending challenges to node i. Based on sampling, node j repeatedly selects a transaction it issued and asks node i to demonstrate it is included within the tree through a membership proof. This should also include statistically verifying the tree values are sorted through examining the locations of the queried transactions. Node j approves the declaration if a large portion of its challenges are answered by node i. Given a block proposal, a committee member identifies the missing transactions and demands from the primary an exclusion proof demonstrating that such transactions were not included in the declaration. Providing such proofs for all missing transactions establishes the honesty of a primary node.

8 Conclusions and Future Work

In this work, we proposed a set of new techniques to enhance the fairness of block selection. First, we described an accurate evaluation of a block proposal through a joint decision of committee members. The joint validation allows more accurate testing whether a block proposal was done honestly, based on random selection, or through prioritizing particular transactions. Experimental results showed a clear advantage of the suggested block validation over the existing approach. The cost for joint validation is additional communication overhead among committee members. Then, we showed how declarations of nodes on their pools of pending transactions can dramatically limit their ability to manipulate the block selection. Declarations on transactions restrict unfair block selection through ignoring transactions. The various declaration schemes imply different communication overhead based on the size of declarations, their frequency and the ability to test a block proposal locally or through queries. For future work, we would like to find optimal tradeoffs among the characteristics of transaction declaration schemes. In particular, we would like to determine the existence of a scheme implying short declarations that enables local testing of a declaration and a block proposal, while maintaining the ability to show the non-membership of any transaction not part of the declaration.

Acknowledgements This research was partially supported by the Technion Hiroshi Fujiwara Cyber Security Research Center, the Israel Cyber Bureau and the Israel National Cyber Directorate. Likewise, it was supported by the Taub family fellowship and by the Gordon Fund for System Engineering.

References

1. Proof of Elapsed Time, http://sawtooth.hyperledger.org, 2016
2. Avi Asayag, Gad Cohen, Ido Grayevsky, Maya Leshkowitz, Ori Rottenstreich, Ronen Tamari and David Yakira. A Fair Consensus Protocol for Transaction Ordering. IEEE International Conference on Network Protocols (ICNP), 2018
3. Lev-Ari Kfir, Spiegelman Alexander, Keidar Idit, Malkhi Dahlia (2019) FairLedger: A Fair Blockchain Protocol for Financial Institutions. International Conference on Principles of Distributed Systems (OPODIS)
4. Kelkar Mahimna, Zhang Fan (2020) Steven Goldfeder and Ari Juels. Order-Fairness for Byzantine Consensus, IACR Cryptology ePrint Archive
5. Jain Raj (1984) Dah-Ming Chiu and William R Hawe. Digital Equipment Corporation, Hudson, MA, A quantitative measure of fairness and discrimination. Eastern Research Laboratory
6. Emilie Danna, Avinatan Hassidim, Haim Kaplan, Alok Kumar, Yishay Mansour, Danny Raz and Michal Segalov. Upward Max-Min Fairness. Journal of the ACM (JACM), vol. 64, no. 1, pp. 2:1–2:24, 2017
7. Giuseppe Ateniese, Ilario Bonacina, Antonio Faonio and Nicola Galesi. Proofs of Space: When Space Is of the Essence. International Conference on Security and Cryptography for Networks, 2014
8. Becker Georg (2008) Merkle signature schemes, Merkle trees and their cryptanalysis. Ruhr-University Bochum, Tech. Rep
9. Bloom Burton H (1970) Space/time trade-offs in hash coding with allowable errors. Commun. ACM 13(7):422–426
10. Broder Andrei Z, Mitzenmacher Michael (2003) Network Applications of Bloom Filters: A Survey. Internet Mathematics 1(4):485–509
11. Cynthia Dwork and Moni Naor. Pricing via Processing or Combatting Junk Mail. Springer CRYPTO, 1992
12. Wenbo Wang, Dinh Thai Hoang, Peizhao Hu, Zehui Xiong, Dusit Niyato, Ping Wang, Yonggang Wen and Dong In Kim. A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks. IEEE Access, vol. 7, pp. 22328–22370, 2019
13. Arthur Gervais, Srdjan Capkun, Ghassan O. Karame and Damian Gruber. On the privacy provisions of Bloom filters in lightweight bitcoin clients. ACM Annual Computer Security Applications Conference (ACSAC), 2014
14. Oleksii Konashevych. The concept of the blockchain-based governing: Current issues and general vision. European Conference on Digital Government (ECDG), 2017
15. Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos and Nickolai Zeldovich. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. ACM Symposium on Operating Systems Principles (SOSP), 2017
16. Aggelos Kiayias, Alexander Russell, Bernardo David and Roman Oliynykov. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. Annual International Cryptology Conference, 2017
17. Yaakov Sokolik and Ori Rottenstreich. Age-aware Fairness in Blockchain Transaction Ordering. IEEE/ACM International Symposium on Quality of Service (IWQoS), 2020
18. Sunny King and Scott Nadal. PPCoin: Peer-to-peer crypto-currency with proof-of-stake. 2012
19. Lailong Luo, Deke Guo, Richard T. B. Ma, Ori Rottenstreich and Xueshan Luo. Optimizing Bloom Filter: Challenges, Solutions, and Comparisons. IEEE Communications Surveys and Tutorials, vol. 21, no. 2, pp. 1912–1949, 2019
20. IBM Blockchain Blog. Executing on the vision for a true blockchain network of networks. ibm.com/blogs/blockchain/2019/09/executing-on-the-vision- for-a-true-blockchain-network-of-networks/, 2019
21. MacDavid Robert, Birkner Rüdiger, Rottenstreich Ori, Gupta Arpit (2017) Nick Feamster and Jennifer Rexford. Concise Encoding of Flow Attributes in SDN Switches, ACM Symposium on SDN Research (SOSR)
22. R. C. Merkle. Secrecy, Authentication, and Public Key Systems. PhD thesis, Stanford, 1979
23. Andrew Miller Yu, Xia Kyle Croman, Shi Elaine, Song Dawn (2016) The Honey Badger of BFT Protocols. ACM Special Interest Group on Security, Audit and Control (SIGSAC)
24. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Bitcoin white paper, http://bitcoin.org/bitcoin.pdf, 2008
25. Orda Ariel, Rottenstreich Ori (2019) Enforcing Fairness in Blockchain Transaction Ordering. IEEE International Conference on Blockchain and Cryptocurrency (ICBC)
26. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper, https://gavwood.com/paper.pdf, 2014
27. Klaus Kursawe. Wendy, the Good Little Fairness Widget: Achieving Order Fairness for Blockchains. ACM Conference on Advances in Financial Technologies, 2020
28. Keping Yu, Tan Liang, Aloqaily Moayad (2021) Hekun Yang and Yaser Jararweh. Blockchain-Enhanced Data Sharing with Traceable and Direct Revocation in IIoT, IEEE Transactions on Industrial Informatics
29. Shi Na, Tan Liang, Li Wenjuan (2020) Xin Qi and Keping Yu. A blockchain-empowered AAA scheme in the large-scale HetNet, Digital Communications and Networks
30. Tan Liang, Xiao Huan, Keping Yu, Aloqaily Moayad, Jararweh Yaser (2021) A blockchain-empowered crowdsourcing system for 5G-enabled smart cities. Computer Standards & Interfaces 76:
31. Keping Yu, Tan Liang, Shang Xinglin, Huang Junjie, Srivastava Gautam, Chatterjee Pushpita (2021) Efficient and Privacy-Preserving Medical Research Support Platform Against COVID-19: A Blockchain-Based Approach. IEEE Consumer Electronics Magazine 10(2):111–120

Ariel Orda received the B.Sc. (summa cum laude), M.Sc., and D.Sc. degrees from the Technion – Israel Institute of Technology, Haifa, Israel, in 1983, 1985, and 1991, respectively, all in electrical engineering. Since 1994, he has been with the Viterbi Faculty of Electrical Engineering, Technion – Israel Institute of Technology, where he is currently the Herman and Gertrude Gross Professor of Communications. During 2014–2017, he has been the Dean of the Viterbi Faculty of Electrical Engineering at the Technion – Israel Institute of Technology. His research interests include network routing, the application of game theory to computer networking, survivability, QoS provisioning, wireless networks, and network pricing. He received several awards for research, teaching and service.

Ori Rottenstreich is an assistant professor at the department of Computer Science and the department of Electrical Engineering of the Technion, Haifa, Israel. Previously, he was a Postdoctoral Research Fellow at Princeton university. Ori received his B.Sc. degree in Computer Engineering and Ph.D. degree in Electrical Engineering from Technion.