A Game Theoretical Analysis of DAG-based ledgers Backbone

Yackolley Amoussou-Guenou[1], Simone Galimberti[2], and Maria Potop-Butucaru[2]

[1] Université Paris-Panthéon-Assas, CRED, Paris, France
[2] Sorbonne Université, CNRS, LIP6, Paris, France

Abstract. We study the rational behaviors of agents in the backbone of *DAG*-Based Distributed Ledgers. We analyze generic algorithms that encapsulate the main actions of agents in a *DAG*-based distributed ledger: voting for a block, and checking its validity. Knowing that those actions have costs, and validating a block rewards agents who participated in the validation procedure, we study, using game theory how strategic agents behave while trying to maximize their gains. We overview possible settings with gain-maximizing agents and analyze if there exist equilibria satisfying the properties of the protocol. We found that there exist equilibria where protocol properties may be violated. Our work paves the way for designing correct-by-construction *DAG*-based protocols.

Keywords: DAG-based ledger · DLT · Game-theory

1 Introduction

Blockchains and generally distributed ledger technologies [8], have emerged as one of the most revolutionary developments in recent years, with the goal of eliminating centralised intermediaries and installing distributed trusted services. Traditionally, blockchain systems maintain a continuously growing list of ordered blocks that include one or more transactions that have been verified by the members of the system, called miners or validators. Blocks are linked using cryptography and the order of blocks in the blockchain is the result of a form of agreement among the system agents. After the release of the most popular blockchains (e.g., Bitcoin or Ethereum) with a specific focus on economic transactions, their huge potential for various other applications became evident. However, quickly after their release, blockchains reached their limits in terms of throughput, block size and unforeseen behaviors. Therefore, academic and non-academic research further concentrate on proposing alternatives by improving some performance aspects but with non-zero costs either on security or throughput. One research direction extends the blockchain to a DAG (directed acyclic graph) overlay and provides an ordering over all blocks and transactions, and outputs a consistent set of accepted transactions. The research along this line (e.g. Spectre [26], GhostDAG [27], Phantom [28], IOTA [23], Avalanche [25], Inclusive Block [16], Colordag [2], Glacier [5], DAG-Rider [15], OHIE [30], Sycomore [7], Prism [9], Fino [18] etc.) structures blocks of transactions in a DAG where each block can include one or several references to its predecessors. Interstingly, while not being a DAG-based blockchain, Fruitchains [22] takes advantage of the DAG structure of transactions in a Bitcoin-like blockchain to increase fairness. The so-called fruits in Fruitchains are like a DAG-based blockchain.

Interestingly, most of these blockchains share the same structure. It is therefore important to study and understand the key components of that structure, and analyze its correctness properties when taking into account various behaviors (including rational and malicious) of the blockchain participants. It should be noted that the analysis of strategic behaviors in blockchains (see for example, [6, 12, 13, 19, 29]) mostly focuses on chain-based ledgers. To the best of our knowledge, no work has been dedicated to analyzing the rational behaviors in *DAG*-based ledgers. The only work proposed in the context of the IOTA ledger is [24] which studies the attachment preferences. Our work, however, analyzes the backbone of a protocol which is shared by multiple DAG-based ledgers.

Our contribution is twofold. First, we formalise the generic backbone protocol of DAG-based distributed ledgers and specify its correctness properties. Second, we analyze the correctness of this backbone protocol with various combinations of behaviors (including rational and faults).

Type of agents considered	**Protocol without communication** (Algorithm 2)	**Protocol with communication** (Algorithm 3)
Obedient	Lemma 1	Lemma 1
Obedient + Crash	Lemma 2	Lemma 2
Obedient + Crash + Malicious	Theorem 1	Theorem 2
Strategic	Theorem 3 and Corollary 2	Theorem 4 and Theorem 8
Strategic + Obedient	Not Applicable	Theorem 9
Strategic + Crash	Not Applicable	Theorem 11
Strategic + Cost-indifferent malicious	Not Applicable	Theorem 12
Strategic + Cost-sensitive malicious	Not Applicable	Theorem 13

Table 1. Summary of the positive results (satisfying the DAG-based ledger correctness properties). Notice that in the presence of strategic agents, the positive results show the existence of a Nash equilibrium. When strategic agents are considered, it is not interesting to discuss the case with no communication, because each behaviour is only local and there is no coordination required, marked in the table as "Not Applicable".

On the positive side, our findings summarized in Table 1 show that in all considered settings under some thresholds on the number of agents and on the costs, the DAG-based backbone correctness properties hold. On the negative side Theorem 7 and Theorem 10 show that there are equilibria that do not satisfy the correctness properties even in the simplest case with a single issuer.

In this paper, we consider the core aspect of DAG-based distributed ledgers which we model using game-theoretic tools. We believe this formalization is a step in understanding more clearly the interactions in these systems. We show that it is possible to design systems respecting their correctness properties, even in the presence of rational miners or validators, however, care should be taken since there exist many equilibria violating these properties.

The rest of this paper is organised as follows. In Section 2, we present the model, formalize the correctness properties of DAG-based ledgers and extract the backbone protocol of these ledgers. In Section 3, we prove the correctness of the backbone when agents are obedient and crash faulty. In Section 4, we analyze the cases where strategic (rational) agents are in play, and we conclude in Section 5. All the missing proofs appear in the addix.

2 Model and Problem Definition

We consider a system composed of a finite set Π of $n > 2$ synchronous and sequential agents denoted by their index, namely $\Pi = \{1, \ldots, n\}$.

Agents communicate in a message passing fashion by sending and receiving messages through a *synchronous network*. There is a finite and known upper bound $\Delta > 0$ on message transmission delay. We assume that each agent proceeds in rounds. A *round* consists of three sequential phases, in order: the send, the delivery, and the compute phases. The delivery phase has a fixed duration that allows collecting all the messages sent by all agents. The system being synchronous, agents can set the duration of their delivery phase to collect all previously sent messages. We assume that messages are signed, and signatures are unforgeable, i.e., when an agent receives a message, the creator of the message is known. Finally, we assume the existence of a *reliable broadcast*.

In classical distributed systems, agents are either correct (they follow exactly the given protocol) or faulty. We that agant can fail by crashing. We consider an additional type of agent, which is

called rational. Intuitively, rational agents are those who can deviate from the given protocol if and only if such a deviation would increase their gain. Notice that those agents have a notion of gain, and will try to maximise their gains and deviate if needed. Rational agents were first discussed in distributed systems in [3, 4, 20] and later considered in blockchain systems [6, 10, 12, 13, 17]. In this work, we consider two sub-classes of rational agents, strategic agents and malicious agents who have different objective functions. Both could be considered as sub-classes of Byzantine agents [1].

Here, we consider that each agent has exactly one of the following types:

- **Obedient** (correct, honest, or acquiescent): those that blindly follow the given protocol until the end. Many terms exist in the literature to describe them, they are also called correct agents;
- **Crash faulty** (faulty by crash or simply crash): those that follow the given protocol and from a point in time, stop working. The moment when a crash faulty agent stops working is not defined nor decided in advance;
- **Rational**: those that want to maximise their expected gain. We divide them into 2 types with different objectives:
  - **Strategic**: Rational agents who do not desire to break the correctness guarantees of the protocol, only to benefit from it; and
  - **Malicious**: Rational agents who have a gain function different than strategic agents, in which they aim to break the correctness properties.

We assume malicious and strategic agents do not crash. While agents that crashed were obedient till their crash, those considered obedient never crash and always follow the protocol. In the following, when we talk about faulty agents, we mean only crash agents.

2.1 Problem definition

Before giving the problems we analyze in this work, let us first describe what are DAG-based ledgers.

A DAG-based ledger, for directed acyclic graph based ledger, is a generalisation of blockchains where instead of maintaining ideally a chain-like structure, each agent maintains a DAG. In the DAG, each vertex is a "block" containing one (e.g., IOTA [23], Ghost [27], Avalanche and Glacier [5,25]) or more (e.g., Spectre [26]) transactions. Locally, each agent updates its local DAG, and the goal is to guarantee some eventual consistency. A new vertex is added to the DAG with respect to the ledger protocol. For example, it can be after a consensus algorithm, or simply when one sees a submitted transaction not already in the DAG.

Vertices of a DAG, which we will simply call blocks hereafter, contain (i) the *identifier* of the agent which built it, this agent is called the *issuer*, (ii) one or more transactions depending on the ledger, and (iii) some hashes representing the parents blocks in the DAG structure. When a block is not the parent of any other block, we say that it is a *leaf* block or simply call it a *tip*.

To consider a DAG-based protocol as correct, it should guarantee that the local DAG of each non-malicious agent consists only of valid blocks, i.e., blocks respecting the application used for the ledger (for example no double spending in Bitcoin); the local DAG of each non-faulty and non-malicious agent is growing, in the sense that when new valid blocks are submitted, some are eventually added to the local DAG of the agents, and finally, the intersection of the ledger of any two agents must be non empty and eventually growing. Let i be an agent, and t a time. We denote by DAG_i the current state of i's local DAG, and by DAG_i^t the state of i's DAG at time t. We say that a DAG D_1 is a prefix of DAG D_2, if all blocks and their links appearing in D_1 are present in D_2, and we denote it $D_1 \subseteq D_2$. We use this definition to encompass the basis of DAGs, however, we must note that some protocols ensure stronger safety properties.

Formally, a DAG-based ledger must have the following properties:

Algorithm 1: Issuer i Block $\mathcal{B}$

1 $Tips := select_tips(DAG_i)$
2 $\mathcal{B} := generate\ (Tips, tr)$
3 $DAG_i := update(DAG_i, \mathcal{B})$
4 **Broadcast**$(< \mathcal{B} >)$
5 **Wait** 2Δ

Definition 1 (DAG-based ledger correctness properties). *A protocol satisfies the DAG-based ledger correctness properties if and only if the following properties hold:*

- DAG-Prefix. *For every non-malicious agent i given two different instants of time $t_2 > t_1$ it follows that $DAG_i^{t_1} \subseteq DAG_i^{t_2}$, which means $DAG_i^{t_1}$ is a prefix of $DAG_i^{t_2}$;*
- DAG-Uniformity. *For every two non-malicious agents i and j, in the absence of new blocks being issued, eventually $DAG_i \subseteq DAG_j$ (or $DAG_j \subseteq DAG_i$). If i and j are obedient, the local DAGs are eventually the same: $DAG_i = DAG_j$.*
- DAG-Liveness. *If a correct or strategic agent i adds a valid block $\mathcal{B}$ to DAG_i, then all non-faulty agents eventually add $\mathcal{B}$ to their own local DAG;*
- DAG-Validity. *For any non-malicious agent i, each block in DAG_i is valid; each block in the DAG satisfies a predefined validity predicate for the ledger.*

In this paper, we study whether or not a DAG-based ledger backbone protocol (properly described in subsection 2.2) satisfies the DAG-based ledger correctness properties (DAG-Prefix, DAG-Uniformity, DAG-Liveness, and DAG-Validity) in the presence of the different types of agents detailed above.

The definition of DAG-based ledger correctness properties raises a question: how do these properties apply to strategic agents, who may act differently from obedient agents? Given that strategic agents aim to maximize their gain, their optimal actions (describe more precisely in subsection 2.2) might still align with these correctness properties. This study seeks to explore if and when such situations occur, then showing that the protocol support the presence of strategic agents.

Moreover, we focus on the strategic behaviour of a single step of the algorithm; therefore, we consider that before executing that step of the protocol, all agents have the same view of the DAG, and we analyze the impact of a new block issued.

While our assumption may seem strong, it is a first step in the analysis of the dynamic setting. Indeed, to study the dynamic game, one needs to study the single stages of the game. Our analysis fills this gap, and we believe the results to be worthwhile on their own, and we let the dynamic analysis as future work.

2.2 Generic protocol studied

A *selection tip algorithm* is run by the function $select_tips(DAG_i)$ (*line 1* of Algorithm 1) to select the tip(s)[3] to approve in the local DAG_i (of the agent i who is the issuer). For example, the protocol IOTA performs a *random walk algorithm* into the local DAG of each node to select the best two tips to which the newly issued block will be added. This procedure is computationally expensive compared to a randomized choice but the former has many advantages: *(i)* Discourages lazy behaviour and encourages approving fresh tips, *(ii)* continuously merges small branches into a single large branch, thus increasing confirmation rate and *(iii)* in case of conflicts, kills off all but one of the conflicting

[3] We recall that a tip is a leaf block in the DAG, i.e., a block that does not have a child block.

Algorithm 2: Receiver i Block $\mathcal{B}$ with no communication

1 **Initialisation:**
2 $action^{check} :=$ `false`
// $action^{check} \in \{$`true`, `false`$\}$
3 $validValue :=$ `true` // $validValue \in \{$`true`, `false`$\}$
4 **Upon receiving** $\mathcal{B}$ **:**
5 $action^{check} \leftarrow \sigma_i^{check}()$ // $\sigma_i^{check}() \in$ {true,false} sets the action of checking or not the validity of the block
6 **if** $action^{check}$ **then**
7 $validValue \leftarrow isValid(\mathcal{B})$ // The execution of $isValid(\mathcal{B})$ has a cost c_{check}
8 **Compute phase:**
9 **if** $validValue$ or not $action^{check}$ **then**
10 $DAG_i := update(DAG_i, \mathcal{B})$

branches. Block $\mathcal{B}$ issued by i contains i's id, the transaction tr and the hashes referring to the parent's blocks, i.e., the previously selected tips (*line 2* of Algorithm 1).

Finally, the issuer updates the local DAG (*line 3* of Algorithm 1), broadcasts (*line 4* of Algorithm 1) the block to other agents, and, finally, waits to ensure that everyone receives the block.

As we can notice, in Algorithm 1, there is no choice given to the issuer. We assume that the issuer is not directly an agent in our analysis, therefore, we assume that it does not fail by crashing or does not exhibit rational behaviour (neither strategic nor malicious). We, however, consider two cases in our study. First, when the block issued is always valid, and second, when there is a probability that the block issued may be invalid. In order to make a sound analysis through game theory, we assume in the following that the probability is known by all the rational agents. Our game-theoretic analysis does not consider the manipulation of rational agents within Algorithm 1 and is left as future work. We aim at highlighting the core step of DAG-based ledger.

We now present the case of Algorithm 2. In Algorithm 2, there is no communication between agents before adding a block to their DAG. It means that no consensus is required a priori. This is the case of the initial version of IOTA [23]. However, such a protocol was not consistent [14]. Therefore, updates or other protocols require a (lightweight) consensus mechanism before inserting a new block. Such a mechanism is what we call *communication among agents* in the following.

Algorithm 3 represents the receiver's algorithm with communication among agents. In this case, each agent broadcasts or not a vote for the issued block. To add a block to a local DAG, the block issued should have a majority of votes.

In this version of the protocol, agents can communicate, and for this purpose, two more variables are needed. *timerVote* (*line 2*) is a timer which allows an agent to receive every vote expressed by the others, this is strictly required since every agent has his own time clock and maybe some delay is present between votes' arrival. From *line 14* to *line 17* the receiver must wait till the end of the round and then stores the votes in the vector *votes* which were initialised as empty. In the *Compute* phase, each agent receives the opinions of the others and inserts the block $\mathcal{B}$ according to the majority of the received messages.

In both Algorithms 2 and 3, *obedient* agents set their $action^{check}$ to `true`, and set $action^{send}$ to `true` no matter the value given by $\sigma^{send}(bool_var)$. It means that obedient agents send negative votes for invalid blocks. Contrarily, the *strategic* agents have the choice to play with such parameters.

Lastly, we note that in our analysis, whenever an agent i inserts a block in DAG_i, the block cannot be removed anymore.

Algorithm 3: Receiver i Block $\mathcal{B}$ with communication

Initialisation:
  $timerVote := 0$ // Timer for receiving votes
  $votes[] := \{\varnothing,...,\varnothing\}$ // Vector storing the votes about the validity of a block
  $action^{check} :=$ `false` // $action^{check} \in \{$`true`, `false`$\}$
  $action^{send} :=$ `false` // $action^{send} \in \{$`true`, `false`$\}$
  $validValue :=$ `true` // $validValue \in \{$`true`, `false`$\}$
Upon receiving $\mathcal{B}$ **:**
  $action^{check} \leftarrow \sigma_i^{check}()$ // $\sigma_i^{check}() \in$ {true,false} sets the action of checking or not the validity of the block
  if $action^{check}$ **then**
    $validValue \leftarrow isValid(\mathcal{B})$ // The execution of $isValid(\mathcal{B})$ has a cost c_{check}
    $action^{send} \leftarrow \sigma_i^{send}(validValue)$ // $\sigma_i^{send}() \in$ {true,false} sets the action of sending or not the validity of the block
    if $action^{send}$ **then**
      broadcast $(< validValue >)$ // The execution of the broadcast has a cost c_{send}
set $timerVote$ to Δ
while $TimerVote$ not expires **do**
  upon receiving $< validValue_j >$ **:**
    $votes[j] \leftarrow validValue_j$ // The agents collect all the opinions on the validity of the block
Compute phase:
  if nbr *votes received for* `true` $>$ nbr(*votes received*)$/2$ **then**
    DAG_i:= $update(DAG_i, \mathcal{B})$

3 DAG-Based Ledger Backbone Protocol with Obedient and Crash Faulty Agents

We first propose the backbone of a DAG-Based Ledger with only obedient and crash faulty agents and one correct issuer.

We note that in this setting, the agent receives a block, without checking the validity since it is guaranteed, and then she updates her own local DAG_i. The following Lemma 1 proves that Algorithms 2 and 3 verify the properties of a DAG-based ledger when all agents are obedient, even if the issued block can be invalid. The following Lemma 2 shows that even in the presence of crash faulty agents, the algorithms verify the properties of DAG-based ledgers.

Lemma 1. *If there is a single issuer, and all agents are obedient, then Algorithms 2 and 3 verify the DAG-based ledger correctness properties.*

Intuitively, Lemma 1 holds because before adding a block to the local DAG, any obedient agent checks that block's validity in Algorithm 2, and votes `true` for the block in Algorithm 3. If the block is invalid, the obedient agent does not add it. The protocol is also tolerant to crash faulty agents.

Lemma 2. *If there is a single issuer, and agents are crash faulty or obedient, then Algorithms 2 and 3 verify the DAG-based ledger correctness properties, no matter the number of crash faulty agents.*

It is interesting to analyze the resilience of the protocol with only obedient, crash faulty and malicious agents. To do so, we must restrict or define all possible actions for the malicious agents.

This will be done in the next section. However, no matter the actions taken by malicious agents (i.e., $action^{check}$ and $action^{send}$), the algorithms ensure DAG-based ledger correctness properties.

The next two theorems generalize our findings on correct and crash faulty agents when there is no rationality in Algorithms 2 and 3.

Lemma 3. *With a single issuer, and agents can be obedient, or crash faulty, then Algorithm 2 verifies the DAG-based ledger correctness properties.*

Since the reasoning with malicious agents in this setting is similar, for concision, we include them in the statements.

Theorem 1. *With a single issuer, and agents can be obedient, crash faulty, or malicious, then Algorithm 2 verifies the DAG-based ledger correctness properties.*

Theorem 1 holds trivially for Algorithm 2 since, before including a block, an obedient (or crash faulty) agent checks the validity of the block, and does not rely on what the other agents are saying. Algorithm 3, needs more assumptions on the number of crash faulty and malicious agents to guarantee correctness.

Theorem 2. *If there is a single issuer, and agents can be obedient, crash faulty or malicious, with at least* $\lfloor n/2 \rfloor + 1$ *obedient agents, then Algorithm 3 verifies the DAG-based ledger correctness properties.*

Theorem 2, shows that the correctness of Algorithm 3, holds thanks to the fact that all obedient agents always send their vote, even if the block is invalid. Therefore, if there are less than $n/2$ malicious and crash faulty agents, obedient agents' votes will always be the majority and will dictate the decision.

4 DAG-Based Distributed Ledger Backbone with Strategic agents

We now study the backbone of DAG-based ledgers considering strategic agents. In this framework, agents try to maximize their gain, so they may decide not to follow the protocol. To be able to analyze the behaviour of strategic agents, first, we need to properly define their possible actions.

Strategic agents' choice is represented in Algorithm 2 and Algorithm 3 by a dedicated variable, namely, $action^{check}$. The action, $action^{check}$, initialized at a default value `false`, can take value from the set {`false`, `true`}. For agent i, the value of $action^{check}$ is set by calling respectively the functions $\sigma_i^{check}()$, returning its strategy. The strategy σ_i^{check} determines if i, the receiving agent, chooses to check the validity of the proposal or not, which is a costly action, represented by c_{check}. The strategy σ_i^{send} determines if i, the receiving agent, chooses to send a vote or not according to the validity information for the potential previous check. This is also a costly action, represented by c_{send}.

The strategy of agent i is a mapping from her information set into her actions. At the point at which the agent can decide to check block validity, her strategy is given by σ_i^{check}. In the case of Algorithm 3, after making the choice to check, the agent must decide whether to send a message or not, and that decision is given by σ_i^{send}.We denote by $\sigma = (\sigma_1, ..., \sigma_n)$, a *strategy profile* where $\forall i \in \{1, ..., n\}$, agent i uses strategy σ_i, where σ_i is the pair (σ_i^{check}, σ_i^{send}).

We assume that when an invalid block is inserted, all agents are punished by incurring in a cost κ. A reward R, is given to the agents when a block is produced. In addition to the cost of checking c_{check}, the agent can send a message at a cost c_{send}. We assume that $\kappa > R > c_{check} \geq c_{send} \geq 0$. Moreover, we need to have the reward bigger than the execution costs to ensure some interest in working, that is: $R > c_{check} + c_{send}$. However, in the case where blocks can be invalid with a known probability p,

we need the rewards to be greater than the cost of execution, which is: $(1-p)R > c_{check}+(1-p)c_{send}$, where p is the probability that the issued block may be invalid.

These are the only costs considered. Indeed, we believe they represent all the existing costs in these protocols. The expected gain of strategic agent i is defined as follows:

$$U_i = E\Big[R \times \mathbb{1}_{(\sigma_i^{send}=\texttt{true})} \times \mathbb{1}_{(block\ produced)} - \kappa \times \mathbb{1}_{(invalid\ block\ produced)} - c_{check} \times \mathbb{1}_{(\sigma_i^{check}=\texttt{true})} - c_{send} \times \mathbb{1}_{(\sigma_i^{send}(block)=\texttt{true})}\Big]$$

where $\mathbb{1}(.)$ denotes the indicator function, taking the value 1 if its argument is `true`, and 0 if it is `false`. In the case of Algorithm 2, we assume that when a strategic agent adds the block, there is automatically the reward R.

Note that when a block is produced, in protocols which allow communication only the agents which sent a message are rewarded (and receive R). This assumption is in line with what is observed in real life and in other distributed ledger systems [6,11,12,21]. Indeed, it does seem such a mechanism will be an incentive for agents to send votes for valid messages, while rewarding everyone at the same time could lead to scenarios of free-riding, i.e., some agents will not participate, hoping the others will do the job, but still be rewarded.

We use the concept of *Nash equilibrium* for our analysis. Intuitively, a Nash equilibrium can be seen as a strategy profile where no agent can increase its gain by deviating alone from the strategy profile. Let $\sigma = (\sigma_1, \ldots, \sigma_n)$ be a strategy profile, and let $\sigma_i' \in \mathcal{S}_i$ be a strategy of agent i. We denote the strategy profile $(\sigma_1, ..., \sigma_{i-1}, \sigma_i', \sigma_{i+1}, ..., \sigma_n)$ by (σ_{-i}, σ_i'). (σ_{-i}, σ_i') is the strategy profile where only agent i deviates by doing σ_i' instead of σ_i. Formally, a strategy profile σ is a Nash equilibrium if and only if $\forall i \in N, \forall \sigma_i' \in \mathcal{S}_i, U_i(\sigma) \geq U_i((\sigma_{-i}, \sigma_i'))$.

In the rest of this paper, we will analyze whether or not the backbone protocol is a Nash equilibrium, and compute the equilibria that exist in the game.

4.1 Analysis with only Strategic Agents

In this section, we consider the setting where all agents are strategic.

Issued block is always valid. First, we consider the situation where the block issued is valid and strategic agents do not communicate with each other.

Strategic agent's choice is represented in Algorithms 2 and 3 by the variable $action^{check}$. *When the block issued is known to be valid, no strategic agent will choose to check the validity, paying a cost c_{check}. It is not in their interest to pay such a cost when they know for sure that the block is valid.*

Theorem 3. *If there is a single issuer, when the issued block is always valid, and all agents are strategic, the strategy profile where strategic agents do not check the block validity but add all blocks in Algorithm 2 is the only Nash equilibrium.*

Theorem 4. *When the issued block is always valid, and all agents are strategic, the strategy profile where strategic agents do not check the block validity but send a vote for the issued block in Algorithm 3 is the only Nash equilibrium.*

A corollary of each Theorem 3 or Theorem 4 is that when the block is always valid, the prescribed strategy followed by obedient agents is not a Nash equilibrium. Indeed, knowing that the block issued is always valid, using the costly action of checking validity is useless. It only yields a cost, where the answer is already known.

Corollary 1. *If there is a single issuer, when the issued block is always valid, the strategy profile where a strategic agent checks the block validity in Algorithms 2 and 3 is not a Nash equilibrium. However, they guarantee at equilibrium the DAG-based ledger correctness properties.*

In the rest of the section, we analyze the situation where the issued block may be invalid with probability p, and valid with probability $1 - p$.

Issued block may be invalid and no communication. We now study the case with no communication between the agents, i.e., agents follow Algorithm 2.

Theorem 5. *In Algorithm 2, if there is a single issuer, when issued blocks may be invalid with a given probability p, and there are only strategic agents, there is an equilibrium in which no agent will check the block validity but they will add the block to their DAG.*

If agents expect that the others will not check the block validity, it is better for them not to check the validity too. This situation breaks the *DAG*-Validity property. However, if the cost of adding an invalid block is high enough, there exists an equilibrium in which strategic agents check block validity.

Theorem 6. *If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq (1-p)R/p + c_{check}/p$, the strategy profile where strategic agents check the block validity and add only a valid block in Algorithm 2 is a Nash equilibrium.*

Corollary 2. *If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq (1-p)R/p + c_{check}/p$, the equilibrium where strategic agents check the block validity and add only a valid block in Algorithm 2, verifies the DAG-based ledger correctness properties.*

Issued block may be invalid and communication between agents. We now consider that all agents follow Algorithm 3.

Theorem 7. *If there is a single issuer, when the issued block may be invalid with any given probability p, there are only strategic agents, in Algorithm 3, there is a Nash equilibrium where all agents do not check the validity of the proposal and vote in favour of the issued block. The DAG-Validity property is not guaranteed.*

While there exists an equilibrium in which the DAG-Validity property does not hold, if the cost of adding an invalid block is high enough, there exists a "good" equilibrium in all DAG-correctness properties hold.

Theorem 8. *If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq R + c_{check}/p$, in Algorithm 3, there exists a Nash equilibrium which satisfies the DAG-based ledger correctness properties.*

The strategy profile where the first $\lfloor n/2 \rfloor + 1$ strategic agents check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, and the others do not check the validity but send a vote in favour of the block, is such a Nash equilibrium.

4.2 Analysis with Multiple Types of Agents in setting With Communication

For the rest of the paper, we consider setting of Algorithm 3 with different types of agents, strategic, obedient, crash faulty and malicious agents.

Equilibria with Strategic and Obedient Agents First, let us discuss the case where there are only obedient and strategic agents.

Theorem 9. *If there is a single issuer, with $\lfloor n/2 \rfloor$ obedient agents, the rest being strategic, in Algorithm 3, if there is a probability p that issued block is invalid, if the following condition holds, $\kappa \geq (pR + c_{check})/(p \Pr(\text{"all obedient agents have index lower or equal to } \lfloor n/2 \rfloor + 1\text{"}))$ there is a Nash equilibrium in which the DAG-ledger correctness properties hold.*[4]

The following strategy profile is such a Nash equilibrium. Strategic agents with index less or equal than $\lfloor n/2 \rfloor + 1$ will check the block validity and vote in favour of the issued block iff it is valid, and against if it is invalid, while the strategic agents with index strictly greater than $\lfloor n/2 \rfloor + 1$ will send a vote in favour of the block without checking its validity.

While Theorem 9 is stated with $n/2$ obedient participants, the results can be extended to cover the cases with less obedient agents.

Equilibria with Strategic and Crash Faulty Agents We analyze the situation with some crash faulty agents. The two next results are similar to Theorem 7.

Theorem 10. *If there is a single issuer, when the issued block may be invalid with probability p, and there are only strategic and crash faulty agents, in Algorithm 3, then there exists a Nash equilibrium in which strategic agents do not check the block validity but vote in favour for the issued block. In that case, the DAG-Validity properties may be violated.*

Theorem 10 holds because only messages received count in the majority rule. If an agent fails (by crashing) and does not send a message, the majority computed will not take into account that message.

Theorem 11. *If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic and f crash faulty agents, and if we have $\kappa \geq (pR + c_{check})/p \Pr(\text{"all crash faulty agents have index lower or equal to } f + \lfloor n/2 \rfloor + 1\text{"})$, in Algorithm 3, there exists a Nash equilibrium in which the DAG-based ledger correctness properties are satisfied.*

The strategy profile where the first $\min(f + \lfloor n/2 \rfloor + 1, n)$ strategic agents check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, and the others do not check the validity but send a vote in favour of the block is such a Nash equilibrium.

In Theorem 11, since agents can fail by crashing, we need to ensure that enough strategic agents are sending a correct vote.

4.3 Equilibria with Strategic and Malicious Agents

Lastly, in this section, we consider the setting where we have a mix of strategic and malicious agents. In more detail, we study the equilibrium behaviour in two cases. First, when the malicious agents do not care about their costs (cost-indifferent malicious agents), and lastly when they try to reduce their costs (cost-sensitive malicious agents). Recall that we assume that there are at least $n/2$ strategic agents, i.e., strategic agents are the majority.

[4] Notice here that we do not impose any repartition on the obedient agents, only the probability is computed.

Cost-indifferent malicious agents Let us assume that the number of cost-indifferent malicious agents is $\mathbf{m}$. These agents do not care about their costs, no matter their actions Their sole objective is damaging the system by validating invalid blocks. For this reason, since they do not have any cost in their gain function, they will always check the block validity and vote for invalid blocks, and against valid ones. Strategic agents are aware of the presence of these other agents and they know their precise number but not their index.

Payoff of cost-indifferent malicious agents. Let ω be an outcome of the game. If ω does not satisfy *DAG-Validity*, then the malicious agents have a payoff of $\tilde{\kappa}_{Validity} > 0$, we also assume that malicious do not care about the cost of their actions, therefore, such costs do not appear in their "gain" function.

If ω does not satisfy *DAG-Validity*, we have $\tilde{\kappa}(\omega) = \tilde{\kappa}_{Validity}$, otherwise, if ω satisfies *DAG-Validity*, $\tilde{\kappa}(\omega) = 0$. If ω is clear from the context, we simply write $\tilde{\kappa}$. The expected gain of a cost-indifferent malicious agent i is:

$$\tilde{U}_i = E\Big[\tilde{\kappa}_{Validity} \times \mathbb{1}_{(\textit{invalid block is produced})}\Big].$$

The cost-indifferent malicious agents could have a preferred strategy which will maximize their chances of breaking the system, which is always checking and voting only in favour of invalid blocks. We show that in the following lemma.

Lemma 4. *For any cost-indifferent malicious agent, the strategy of always checking and vote only in favour of invalid blocks, and not vote for valid blocks is dominant strategy, i.e., there is no other strategy giving strictly greater gain, whatever the action of the other agents.*

Proof. Since a cost-indifferent malicious agent does not incur any cost from their actions, they can always check the block validity to get an information about the block.

Then, if the block is valid, it will be counter-productive for them to vote for it, since they could make the block added to the DAG, defying their purpose. On the contrary, if they vote for invalid blocks, they can help making such a block accepted (especially when 1 vote remains), with no additional cost. Hence, they will prefer it than not voting for them, which could reduce their chance of accepting an invalid block.

Knowing the preferred strategy of the cost-indifferent malicious agents, we now exhibit a Nash equilibrium satisfying the DAG-ledger correctness properties.

Theorem 12. *Assume that there is a single issuer. Let us be in the setting of Algorithm 3 where the issued block may be invalid with probability p and there are only strategic and cost-indifferent malicious agents. We indicate with $\mathbf{m}$ the number of cost-indifferent malicious agents and with $\mathbf{r} \geq \lfloor n/2 \rfloor + 1$ the number of strategic ones. If the cost κ is high enough in the sense that $\kappa \geq (pR + c_{check})/p \Pr(\text{"all malicious agents have index strictly greater than } \lfloor n/2 \rfloor + \mathbf{m}\text{"})$, there exists a Nash equilibrium where the DAG-ledger correctness properties hold.*

The strategy profile where malicious agents check the block validity and send a vote in favour of invalid blocks and no vote for valid blocks; strategic agents with index lower or equal to $\lfloor n/2 \rfloor - \mathbf{m}$ send a vote in favour without checking the issued block validity, and the remaining strategic agents check the block's validity, for in favour of it if valid, and against it if invalid is such a Nash equilibrium.

Proof. Thanks to Lemma 4, we know that in the proposed strategy profile, the cost-indifferent malicious agents cannot deviate and gain more. In particular, in this strategy profile, the gain of the malicious agents is 0. We can turn to the strategic agents. Therefore, let us prove that the strategy profile described in the theorem is a Nash equilibrium by focusing on strategic agents.

In that strategy profile, the gain of the non-checkers (strategic agents with index $\lfloor n/2 \rfloor - \mathbf{m}$) is $(1-p)R - c_{send}$, and the expected gain of the checkers (remaining strategic agents) is $(1-p)R - c_{send} - c_{check}$.

- First, notice that since the malicious agents do not care about the cost of their actions, it is better for them to send vote in favour of invalid blocks. There is no point sending votes in favour of valid blocks since that will help making such block accepted. Lastly, voting against valid block have no impact on the decision too. So, they are better off by sending votes only in favour of invalid blocks.
- Now, we can turn to the non-checkers. If a non-checker deviates by checking the block validity and send a vote accordingly, the expected gain is $(1-p)R - c_{send} - c_{check}$ which is lower than $(1-p)R - c_{send}$ since $c_{check} > 0$. If a non-checker deviates by checking the block validity and send only a vote in favour of the block if it is valid, and no vote if the block is invalid, the expected gain is $(1-p)(R - c_{send}) - c_{check}$ which is lower than $(1-p)R - c_{send}$ since $c_{check} \geq c_{send}$. Other deviations are trivially dominated.
- Finally, let us turn to the checkers. Let us denote by $\mathcal{M}_{\mathbf{m}}$ the random event *"all malicious agents have index strictly greater than* $\lfloor n/2 \rfloor - \mathbf{m}$*"*.
  If a checker deviates by sending without checking, the expected gain is $\Pr(\mathcal{M}_{\mathbf{m}})(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{M}_{\mathbf{m}}))(R - c_{send})$. Indeed, if a checker votes without checking, there are two cases. Either all malicious are part of the indices that should check (which happens with probability $\Pr(\mathcal{M}_{\mathbf{m}})$), in which case by deviating, invalid blocks get exactly $\lfloor n/2 \rfloor + 1$ votes and are therefore accepted, and valid blocks get $n - \mathbf{m}$ votes. If malicious agents are more spread out (which happens with probability $1 - \Pr(\mathcal{M}_{\mathbf{m}})$), then the checker deviating is better off, since the votes does not make invalid blocs added, while valid blocks will still be added. This gain at deviation is lower than the gain at equilibrium iff

$$\begin{aligned}
&(1-p)R - c_{send} - c_{check} - \Pr(\mathcal{M}_{\mathbf{m}})(R - c_{send} - p\kappa) \\
&\qquad - (1 - \Pr(\mathcal{M}_{\mathbf{m}}))(R - c_{send}) \geq 0 \\
\iff\ & -pR - c_{check} + \Pr(\mathcal{M}_{\mathbf{m}})p\kappa \geq 0 \\
\iff\ & \Pr(\mathcal{M}_{\mathbf{m}})p\kappa \geq pR - c_{check} \\
\iff\ & \kappa \geq \frac{pR + c_{check}}{p \Pr(\mathcal{M}_{\mathbf{m}})}
\end{aligned}$$

  which is the condition in the theorem.
  If a checker deviates by checking and sending a vote in favour iff the block is valid, and no vote if invalid, the gain at deviation is $(1-p)(R - c_{send} - c_{check})$ which is lower than $\Pr(\mathcal{M}_{\mathbf{m}})(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{M}_{\mathbf{m}}))(R - c_{send})$ iff

$$\begin{aligned}
&(1-p)(R - c_{send}) - c_{check} - \Pr(\mathcal{M}_{\mathbf{m}})(R - c_{send} - p\kappa) \\
&\qquad + (1 - \Pr(\mathcal{M}_{\mathbf{m}}))(R - c_{send}) \geq 0 \\
\iff\ & \Pr(\mathcal{M}_{\mathbf{m}})p\kappa - p(R - c_{send}) - c_{check} \geq 0 \\
\iff\ & \Pr(\mathcal{M}_{\mathbf{m}})p\kappa \geq p(R - c_{send}) + c_{check} \\
\iff\ & \kappa \geq \frac{pR + c_{check}}{p \Pr(\mathcal{M}_{\mathbf{m}})} - \frac{c_{send}}{p \Pr(\mathcal{M}_{\mathbf{m}})}
\end{aligned}$$

  This is guarantee under the condition in the theorem. So no checkers has the incentive to deviate. All other deviations are trivially dominated. Therefore, the proposed strategy is a Nash equilibrium.

Now, we show that the above equilibrium satisfies the DAG-based ledger correctness properties. In the above strategy profile, only valid blocks get the strict majority of votes and gets added, and no invalid block is added by any strategic agent. All strategic agents, checker or not receive the same votes and are adding the same block if valid, and no block if invalid, therefore, we can show that in that equilibrium, all DAG-based ledger correctness properties to hold.

Cost-sensitive malicious agents We now consider the malicious agents that are cost-sensitive. We denote by $\mathbf{b}$ their numbers. The objective of these malicious agents is to harm the system but they are sensitive to their costs, contrarily to cost-indifferent malicious agents.

Payoff of cost-sensitive malicious agents. Let ω be an outcome of the game. If ω does not satisfy *DAG-Validity*, then the malicious agents have a payoff of $\tilde{\kappa}_{Validity} > 0$. If ω does not satisfy *DAG-Validity*, $\tilde{\kappa}(\omega) = \tilde{\kappa}_{Validity}$, otherwise, if ω satisfies *DAG-Validity*, $\tilde{\kappa}(\omega) = 0$. If ω is clear from the context, we simply write $\tilde{\kappa}$. The expected gain of a cost-sensitive malicious agent i is:

$$\tilde{U}_i = E\Big[\tilde{\kappa}_{Validity} \times \mathbb{1}_{(\textit{invalid block is produced})} - c_{check} \times \mathbb{1}_{(\sigma_i^{check} = \texttt{true})} - c_{send} \times \mathbb{1}_{(\sigma_i^{send}(block) = \texttt{true})}\Big].$$

The strategy profile which was an equilibrium strategy for Theorem 12 is not a Nash equilibrium when malicious agents are cost-sensitive. Since cost-sensitive malicious agents try to reduce their costs, the deviation of not checking nor sending is profitable for them. There, however, exists an equilibrium with cost-sensitive malicious agents guaranteeing the DAG-ledger correctness properties.

Theorem 13. *Assume that there is a single issuer. Let us be in Algorithm 3 where the issued block may be invalid with probability p and there are only strategic and cost-sensitive malicious agents. We indicate with $\mathbf{b}$ the number of malicious agents and with $\mathbf{r}$ the number of strategic ones. If:*

$$\kappa > R - c_{send}/p + \frac{c_{send} - R + c_{check} + \mathbb{P}(\mathbf{r} > \mathbf{b})[R(1-p) - c_{send}]}{p(1 - \mathbb{P}(\mathbf{r} > \mathbf{b}))}$$

The following strategy profile is Nash equilibrium satisfying the DAG-ledger correctness. Strategic agents with index at most $\min(\mathbf{b} + \lfloor n/2 \rfloor + 1, n)$ check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, the others strategic agents do not check the validity but send a vote in favour of the block, and cost-sensitive malicious agents do nothing.

Proof. Let us show that the strategy profile is a Nash equilibrium.

This resembles the strategy profile used to demonstrate Theorem 11. And except the case of the malicious agents, the proof follows the same steps.

Let us call the strategic agents with index smaller or equal to $\mathbf{b} + \lfloor n/2 \rfloor + 1$, the checkers.

The expected gain of each checker is $(1-p)R - c_{send} - c_{check}$, and the expected gain of non-checkers is $(1-p)R - c_{send}$, and the gain of malicious agents is 0.

- First, let us show that cost-sensitive malicious agents are better off not doing anything. Indeed, if a cost-sensitive agent deviated and send a vote in favour of the issued block, if the block is invalid, it will get at most $\lfloor n/2 \rfloor$ votes in favour, which is not enough. Therefore, the gain will be at best $-c_{send}$. So, the cost-sensitive malicious agent prefers doing nothing. All other deviations are trivially dominated.
- We now show that non-checkers have no interest to deviate. If a non-checker deviates by checking the block validity and send a vote accordingly, the expected gain is $(1-p)R - c_{send} - c_{check}$ which is lower than $(1-p)R - c_{send}$ since $c_{check} > 0$. All other deviations are trivially dominated.

– Finally, turns to the checkers. We show that they also have no interest in deviating. Let us denote by $\mathcal{B}_{\mathbf{b}}$ the random event *"all malicious agents have index lower or equal to* $\mathbf{b}+\lfloor n/2 \rfloor+1$*"*. Indeed, if a checker deviates by sending without checking, the expected gain is $\Pr(\mathcal{B}_{\mathbf{b}})(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{B}_{\mathbf{b}}))(R - c_{send})$. This is lower than the gain at equilibrium iff

$$\begin{aligned}
&(1-p)R - c_{send} - c_{check} - \Pr(\mathcal{B}_{\mathbf{b}})(R - c_{send} - p\kappa) \\
&\qquad - (1 - \Pr(\mathcal{B}_{\mathbf{b}}))(R - c_{send}) \geq 0 \\
\iff\ & -pR - c_{check} + \Pr(\mathcal{B}_{\mathbf{b}})p\kappa \geq 0 \\
\iff\ & \Pr(\mathcal{B}_{\mathbf{b}})p\kappa \geq pR + c_{check} \\
\iff\ & \kappa \geq \frac{pR + c_{check}}{p \Pr(\mathcal{B}_{\mathbf{b}})}
\end{aligned}$$

which is the condition in the theorem.
If a checker deviates by checking and sending a vote in favour iff the block is valid, and no vote if invalid, the gain at deviation is $(1-p)(R - c_{send} - c_{check})$ which is lower than $\Pr(\mathcal{B}_{\mathbf{b}})(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{B}_{\mathbf{b}}))(R - c_{send})$ iff

$$\begin{aligned}
&(1-p)(R - c_{send}) - c_{check} - \Pr(\mathcal{B}_{\mathbf{b}})(R - c_{send} - p\kappa) \\
&\qquad + (1 - \Pr(\mathcal{B}_{\mathbf{b}}))(R - c_{send}) \geq 0 \\
\iff\ & \Pr(\mathcal{B}_{\mathbf{b}})p\kappa - p(R - c_{send}) - c_{check} \geq 0 \\
\iff\ & \Pr(\mathcal{B}_{\mathbf{b}})p\kappa \geq p(R - c_{send}) + c_{check} \\
\iff\ & \kappa \geq \frac{pR + c_{check}}{p \Pr(\mathcal{B}_{\mathbf{b}})} - \frac{c_{send}}{p \Pr(\mathcal{B}_{\mathbf{b}})}
\end{aligned}$$

This is guarantee under the condition in the theorem. So no checkers has the incentive to deviate. All other deviations are trivially dominated.

Now, we show that the above equilibrium satisfies the DAG-based ledger correctness properties. In the above strategy profile, only valid blocks get the strict majority of votes and gets added, and no invalid block is added by any strategic agent. All strategic agents, checker or not receive the same votes and are adding the same block if valid, and no block if invalid, therefore, we can show that in that equilibrium, all DAG-based ledger correctness properties to hold.

5 Conclusion

We formalize properties of *DAG* based distributed ledgers and analyze their resilience to various behaviors (including rational and malicious) by studying a generic backbone protocol shared by multiple DAG-based ledgers. We prove positive and negative results. Interestingly, intuitive strategies are usually not equilibria, and there are some equilibria in which the correctness properties are broken. We found that when invalid blocks are proposed (even in the simplest scenario with a single issuer), there exist equilibria where the backbone properties may be violated. Our findings are summarized in Table 1.

This work helps to understand the limitations of existing DAG-based blockchains face to rational behaviors. Moreover, our findings can be used by DAG-based protocols designers to choose the appropriate mechanism (in terms of mechanism design theory) in order to avoid equilibria where their protocols do not satisfy the specification. A future line of work is to consider the behavior of rational and faulty agents with respect to the tip's selection and the block proposal.

References

1. Abraham, I., Alvisi, L., Halpern, J.Y.: Distributed computing meets game theory: Combining insights from two fields. SIGACT News (2011)
2. Abraham, I., Dolev, D., Eyal, I., Halpern, J.Y.: Colordag: An incentive-compatible blockchain. In: 37th International Symposium on Distributed Computing, DISC 2023 (2023)
3. Abraham, I., Dolev, D., Gonen, R., Halpern, J.: Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing. PODC '06 (2006)
4. Aiyer, A.S., Alvisi, L., Clement, A., Dahlin, M., Martin, J.P., Porth, C.: Bar fault tolerance for cooperative services. In: SOSP '05 (2005)
5. Amores-Sesar, I., Cachin, C., Tedeschi, E.: When is spring coming? A security analysis of avalanche consensus. In: 26th International Conference on Principles of Distributed Systems, OPODIS 2022 (2022)
6. Amoussou-Guenou, Y., Biais, B., Potop-Butucaru, M., Tucci-Piergiovanni, S.: Committee-based blockchains as games between opportunistic players and adversaries. The Review of Financial Studies (2023)
7. Anceaume, E., Guellier, A., Ludinard, R., Sericola, B.: Sycomore: A permissionless distributed ledger that self-adapts to transactions demand. In: 17th IEEE International Symposium on Network Computing and Applications, NCA 2018 (2018)
8. Anta, A.F., Georgiou, C., Herlihy, M., Potop-Butucaru, M.: Principles of Blockchain Systems. Morgan & Claypool Publishers (2021)
9. Bagaria, V.K., Kannan, S., Tse, D., Fanti, G., Viswanath, P.: Prism: Deconstructing the blockchain to approach physical limits. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019 (2019)
10. Dhamal, S., Ben-Ameur, W., Chahed, T., Altman, E., Sunny, A., Poojary, S.: Strategic investments in distributed computing: A stochastic game perspective. Journal of Parallel and Distributed Computing (2022)
11. Ethereum: Proof-of-stake rewards and penalties. `https://web.archive.org/web/20240315224239/https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/rewards-and-penalties/` (2024)
12. Eyal, I., Sirer, E.G.: Majority is not enough: Bitcoin mining is vulnerable. Commun. ACM (2018)
13. Fooladgar, M., Manshaei, M.H., Jadliwala, M., Rahman, M.A.: On incentive compatible role-based reward distribution in algorand. In: 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) (2020)
14. Foundation, I.: The coordicide (2020), `https://files.iota.org/papers/20200120_Coordicide_WP.pdf`
15. Keidar, I., Kokoris-Kogias, E., Naor, O., Spiegelman, A.: All you need is DAG. In: PODC '21: ACM Symposium on Principles of Distributed Computing (2021)
16. Lewenberg, Y., Sompolinsky, Y., Zohar, A.: Inclusive block chain protocols. In: 19th Financial Cryptography and Data Security International Conference, FC (2015)
17. Liao, M., Golab, W., Zahedi, S.M.: Irs: An incentive-compatible reward scheme for algorand. In: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems. AAMAS '23 (2023)
18. Malkhi, D., Szalachowski, P.: Maximal extractable value (MEV) protection on a DAG. In: 4th International Conference on Blockchain Economics, Security and Protocols, Tokenomics 2022 (2022)
19. McMenamin, C., Daza, V., Pontecorvi, M.: Achieving state machine replication without honest players. In: Proceedings of the 3rd ACM Conference on Advances in Financial Technologies. AFT '21 (2021)
20. Moscibroda, T., Schmid, S., Wattenhofer, R.: When selfish meets evil: byzantine players in a virus inoculation game. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing. PODC '06 (2006)
21. Neuder, M., Moroz, D.J., Rao, R., Parkes, D.C.: Selfish behavior in the tezos proof-of-stake protocol. CoRR **abs/1912.02954** (2019)
22. Pass, R., Shi, E.: Fruitchains: A fair blockchain. In: Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017 (2017)
23. Popov, S.: The tangle (2015)
24. Popov, S., Saa, O., Finardi, P.: Equilibria in the tangle. Comput. Ind. Eng. (2019)
25. Rocket, T., Yin, M., Sekniqi, K., van Renesse, R., Sirer, E.G.: Scalable and probabilistic leaderless BFT consensus through metastability. CoRR (2019)
26. Sompolinsky, Y., Lewenberg, Y., Zohar, A.: Spectre: A fast and scalable cryptocurrency protocol. IACR Cryptol. ePrint Arch. **2016** (2016)
27. Sompolinsky, Y., Wyborski, S., Zohar, A.: PHANTOM GHOSTDAG: a scalable generalization of nakamoto consensus. In: AFT '21: 3rd ACM Conference on Advances in Financial Technologies (2021)
28. Sompolinsky, Y., Zohar, A.: PHANTOM: A scalable blockdag protocol. IACR Cryptol. ePrint Arch. (2018)
29. Tsabary, I., Eyal, I.: The gap game. In: Proceedings of the 2018 ACM SIGSAC conference on Computer and Communications Security (2018)

30. Yu, H., Nikolic, I., Hou, R., Saxena, P.: OHIE: blockchain scaling made simple. In: 2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco (2020)

A Appendix: Proofs

Lemma 1 If there is a single issuer, and all agents are obedient, then Algorithms 2 and 3 verify the DAG-based ledger correctness properties.

Proof (Proof of Lemma 1). We decompose the proof.

- First, assume that the issued block is always valid.
  - We show that Algorithm 2 satisfies the DAG-based ledger correctness properties. We do so by showing that each property holds.
    - *DAG-Prefix*: *DAG-Prefix* property is straightforward. Since the block issued is valid by assumption, the obedient agent receiving will add it the local DAG (lines 10 and 11 of Algorithm 2). Therefore, the local DAG will grow by receiving the new block.
    - *DAG-Uniformity*: Since by assumption, all agents receive the same message, it means that they all collect the same block. Moreover, since blocks are valid by assumption, it means that obedient agents will add them to their local DAG, and therefore their local DAGs are the same.
    - *DAG-Liveness*: Since the block issued is valid by assumption, the obedient agent receiving it will add it the local DAG (lines 10 and 11 of Algorithm 2). Therefore, the local DAG will grow by receiving the new block.
    - *DAG-Validity*: Trivial, since by assumption the issued block is valid.
  - We now show that Algorithm 3 satisfies the DAG-based ledger correctness properties.
    - *DAG-Prefix*: We recall that the block issued is valid. By following the algorithm, each obedient agent will send a vote in favor of the block. Since the communication is synchronous and all agents are obedient, all agents will receive n votes and all of them are at `true`. The condition at line 19 of Algorithm 3 is satisfied, therefore the obedient agent will update the local DAG by adding the block (line 20 of Algorithm 3). The local DAG is therefore growing.
    - *DAG-Uniformity*: All agents receive the same message, and since they are obedient, they all follow the same action, moreover, the block is valid so they vote in favour of it, and they all add the issued block to their local DAG.
    - *DAG-Liveness*: The proof is similar to the *DAG-Prefix* and *DAG-Uniformity* cases. All agents add the same issued block to their local DAG.
    - *DAG-Validity*: Trivial, since by assumption the issued block is valid.
- Since all agents are obedient and therefore they exact same actions. They all check the block's validity and vote in favour of it only if they block is valid, and vote against the block if it is invalid. All obedient agents have the same view and do the same actions and know after checking whether the block is invalid or not.
  If the proposed block is valid, Lemma 1 concludes the proof. If however the block is invalid, no obedient agents will add the block to the local DAG. Therefore all properties all, in particular the *DAG-Validity* property since no block is added.

Lemma 2 If there is a single issuer, and agents are crash faulty or obedient, then Algorithms 2 and 3 verify the DAG-based ledger correctness properties, no matter the number of crash faulty agents.

Proof (Proof of Lemma 2). We decompose this proof.

- First, we show that Algorithm 2 satisfies the DAG-based ledger correctness properties. We do so by showing that each property holds.

Since there is no communication between agents, and since the properties do not concern crash faulty agents, applying Lemma 1 show that *DAG-Prefix*, *DAG-Liveness*, and *DAG-Validity* hold for Algorithm 2 even with the presence on crash faulty agents. Crash faulty agents, until they crash have an obedient behaviour, therefore either the do the same action as the obedient agents (adding the block if it is valid, and not adding if invalid) if they crashed after the update, or they do not add the block at all if they crashed before the update. Therefore in the best case, they have the same DAG as obedient agents, or their DAG is included in the one of obedient agents.

- We now show that Algorithm 3 satisfies the DAG-based ledger correctness properties. The case of Algorithm 3 also follows from Lemma 1. Indeed, the decision of the agents depends on the number of votes they received. If one agent crashes before sending a vote, it will not have an impact on the agents that are non crash faulty, especially since all agents voting will vote the same value (they are crash faulty or obedient). Moreover, as in the first part of this proof, crash faulty agents' DAG are necessarily included in the one of obedient agents.

Theorem 1 With a single issuer, and agents can be obedient, crash faulty, or malicious, then Algorithm 2 verifies the DAG-based ledger correctness properties.

Proof (Proof of Theorem 1). This is a corollary of Lemma 2. The DAG-correctness properties explicitly exclude malicious agents, hence, if the system is composed of them, they properties hold by default. Moreover, the decision of an agent does not depend on the actions of the others.

Theorem 2 If there is a single issuer, and agents can be obedient, crash faulty or malicious, with at least $\lfloor n/2 \rfloor + 1$ obedient agents, then Algorithm 3 verifies the DAG-based ledger correctness properties.

Proof (Proof of Theorem 2). We now show that Algorithm 3 satisfies the DAG-based ledger correctness properties.

We first show that obedient agents will block to their DAG if and only if they are valid. Since there are at least $n/2$ obedient agents in the system, the majority of votes (at least $n/2$) that will be received are from them.

If the block was invalid, at least $n/2$ votes will be against the block (`false`), and it is impossible in that situation to satisfy (line 19 of Algorithm 3). Therefore, no agent will add the block.

If the block is valid, at least there will be $n/2$ votes for the block, no matter what the other will do (or if there are crashes), votes for `true` are the majority and by lines 19 and 20 of Algorithm 3, the block will be added. Since the communication is synchronous and all the obedient agents follow the same actions, they will have the same view.

Therefore, if the total number of malicious and crash faulty agents is strictly lower than the majority, then malicious agents cannot impact the view or decision of obedient agents. The rest of the proof follows by Lemma 2.

Theorem 3 If there is a single issuer, when the issued block is always valid, and all agents are strategic, the strategy profile where strategic agents do not check the block validity but add all blocks in Algorithm 2 is the only Nash equilibrium.

Proof (Proof of Theorem 3). The gain of each strategic agent at equilibrium is R. If a strategic agent checks the validity, that will yield a gain of $R - c_{check}$ which is lower than the gain at equilibrium, so the strategy profile where all agents accept the block with checking its validity is a Nash equilibrium.

Moreover, in all other strategy profiles, at least one agent checks validity. That agent will have a gain of $R - c_{check}$. By deviating and not sending, that agent will have a gain of R, so it is not an equilibrium since $c_{check} \geq 0$. All other strategy profiles are therefore not Nash equilibria.

Theorem 4 When the issued block is always valid, and all agents are strategic, the strategy profile where strategic agents do not check the block validity but send a vote for the issued block in Algorithm 3 is the only Nash equilibrium.

Proof (Proof of Theorem 4). No strategic agent has to check the block validity since the cost will be paid for nothing, and if there is no vote, the agent will not get a reward, so it is better to send the vote. That means that the strategy where the strategic agent sends a vote for the block without checking validity dominates all other strategies. This implies that the strategy profile where all strategic agents send a vote for the block without checking validity dominates all other strategy profiles. Therefore, all other strategy profiles are not a Nash equilibrium.

Now, we show that the strategy profile is indeed a Nash equilibrium. The gain of each strategic agent at equilibrium is $R - c_{send}$. If a strategic agent checks the validity and sends a vote, the gain is $R - c_{check} - c_{send}$, which is lower than the gain at equilibrium. If the agent deviates by not sending, either the gain will be $-c_{check}$ (if there was a check of validity), or 0 if there was not validity check; in both cases, the gain is lower than the gain at equilibrium. Since there is no profitable deviation for any strategic agent, the strategy profile where all strategic agents send a vote for the block without checking validity is a Nash equilibrium.

Theorem 5 In Algorithm 2, if there is a single issuer, when issued blocks may be invalid with a given probability p, and there are only strategic agents, there is an equilibrium in which no agent will check the block validity but they will add the block to their DAG.

Proof (Proof of Theorem 5).

In the strategy profile where all strategic agents do not check the block validity, the gain of each strategic agent is $R - p\kappa$.

If an agent deviates and checks the validity, the gain of that agent is $(1-p)R - c_{check} - p\kappa$ which is lower than $R - p\kappa$ since $R - p\kappa - ((1-p)R - c_{check} - p\kappa) = pR + c_{check} \geq 0$. Therefore, the proposed strategy profile is a Nash equilibrium.

Theorem 6 If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq (1-p)R/p + c_{check}/p$, the strategy profile where strategic agents check the block validity and add only a valid block in Algorithm 2 is a Nash equilibrium.

Proof (Proof of Theorem 6).

In the strategy profile where all strategic agents check the block validity, the gain of each strategic agent is $pR - c_{check}$.

If an agent deviates and does not check the validity, the gain of that agent is $R - p\kappa$ which is lower than $pR - c_{check}$ iff $pR - c_{check} - (R - p\kappa) \geq 0$ which is equivalent to $\kappa \geq (1-p)R/p + c_{check}/p$. Therefore, the proposed strategy profile is a Nash equilibrium under $\kappa \geq (1-p)R/p + c_{check}/p$.

Corollary 2 If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq (1-p)R/p + c_{check}/p$, the equilibrium where strategic agents check the block validity and add only a valid block in Algorithm 2, verifies the DAG-based ledger correctness properties.

Proof (Proof of Corollary 2). In the equilibrium of Theorem 6, the strategic agents behave as if they were obedient, by Lemma 1, we can show that the DAG-ledger correctness properties hold.

Theorem 7 If there is a single issuer, when the issued block may be invalid with any given probability p, there are only strategic agents, in Algorithm 3, there is a Nash equilibrium where all agents do not check the validity of the proposal and vote in favour of the issued block. The DAG-Validity property is not guaranteed.

Proof (Proof of Theorem 7).

- First, let us show that the strategy profile where no agent checks the issued block validity but sends a vote in favour does not satisfies the DAG-Validity property.
  The expected gain of each agent is $R - c_{send} - p\kappa$.
  If the agent does not send a message, the expected gain will be at best $-p\kappa$, since $n > 1$ which is lower than $R - c_{send} - p\kappa$.
  If the agent decided to check the block's validity, and for in favour of valid block, and against invalid one, since $n > 2$ the vote against will not change the decision, therefore, the gain at expectation will be $(1-p)R - c_{check} - c_{send} - p\kappa$, which is lower than $R - c_{send} - p\kappa$.
  Lastly, if the agent decides to deviate by checking validity and only sending vote in favour of valid blocks, the expected gain is $(1-p)(R - c_{send}) - c_{check} - p\kappa$; while this deviation is better than the previous, it is still lower than $R - c_{send} - p\kappa$. Therefore, the strategy profile is a Nash equilibrium.
- Now, let us show that the strategy profile where no one checks but sends a vote in favour or the issued block is a Nash equilibrium.
  In this equilibrium, if the issued block is invalid, it will be added by all strategic agents since they do not check the block's validity. Adding an invalid block violates the DAG-Validity property.

Theorem 8 If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq R + c_{check}/p$, in Algorithm 3, there exists a Nash equilibrium which satisfies the DAG-based ledger correctness properties.

The strategy profile where the first $\lfloor n/2 \rfloor + 1$ strategic agents check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, and the others do not check the validity but send a vote in favour of the block, is such a Nash equilibrium.

Proof (Proof of Theorem 8).

- First, let us show that a strategy profile where the first $\lfloor n/2 \rfloor + 1$ strategic agents check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, and the others do not check the validity but send a vote in favour of the block is a Nash equilibrium. Let us call the strategic agents with index smaller or equal to $\lfloor n/2 \rfloor + 1$, the checkers.
  The expected gain of each checker is $(1-p)R - c_{send} - c_{check}$, and the expected gain of non-checkers is $(1-p)R - c_{send}$.
  We show that non-checkers have no interest to deviate. If a non-checker deviates by checking the block validity and send a vote accordingly, the expected gain is $(1-p)R - c_{send} - c_{check}$ which is lower than $(1-p)R - c_{send}$ since $c_{check} > 0$. All other deviations are trivially dominated.
  Now turns to the checkers. We show that they also have no interest in deviating. Indeed, if a checker deviates by sending without checking, the expected gain is $R - c_{send} - p\kappa$ if n is even, and $-c_{send}$ if n is odd. If n is even, since $c_{send} \geq 0$, then it is lower than $(1-p)R - c_{send} - c_{check}$.
  In the even case, this is lower that $(1-p)(R - c_{send}) - c_{check}$ iff

$$\begin{aligned}(1-p)R - c_{send} - c_{check} - (R - c_{send} - p\kappa) \geq 0 &\iff p\kappa - pR - c_{check} \geq 0 \\ &\iff p\kappa \geq pR + c_{check} \\ &\iff \kappa \geq R + c_{check}/p\end{aligned}$$

which is the condition at equilibrium.
If the agent deviated by checking and sending a vote in favour iff the block is valid, and no message when the block is invalid, the gain at deviation is $(1-p)(R-c_{send}-c_{check})$ (thanks to adding only blocks getting a strict majority for the block) which is lower than $(R-c_{send}-p\kappa)$ iff

$$\begin{aligned}
&(1-p)(R-c_{send})-c_{check}-(R-c_{send}-p\kappa)\geq 0\\
&\iff p\kappa-p(R-c_{send})-c_{check}\geq 0\\
&\iff p\kappa\geq p(R-c_{send})+c_{check}\\
&\iff \kappa\geq R+c_{check}/p-c_{send}
\end{aligned}$$

This is guarantee under $\kappa\geq R+c_{check}/p$. So no checkers has the incentive to deviate. All other deviations are trivially dominated.

– Now, we show that the above equilibrium satisfies the DAG-based ledger correctness properties. In the above strategy profile, only valid blocks get the majority of votes and gets added, and no invalid block is added by any strategic agent. Since all agents have the same view, and are adding the same block if valid, and no block if invalid, we can show that in that equilibrium, all DAG-based ledger correctness properties to hold.

Theorem 9 If there is a single issuer, with $\lfloor n/2\rfloor$ obedient agents, the rest being strategic, in Algorithm 3, if there is a probability p that issued block is invalid, if the following condition holds, $\kappa\geq(pR+c_{check})/(p\Pr(\text{“all obedient agents have index lower or equal to }\lfloor n/2\rfloor+1\text{”}))$ there is a Nash equilibrium in which the DAG-ledger correctness properties hold.[5]

The following strategy profile is such a Nash equilibrium. Strategic agents with index less or equal than $\lfloor n/2\rfloor+1$ will check the block validity and vote in favour of the issued block iff it is valid, and against if it is invalid, while the strategic agents with index strictly greater than $\lfloor n/2\rfloor+1$ will send a vote in favour of the block without checking its validity.

Proof (Proof of Theorem 9). We study the following strategy profile. All strategic agent with index less or equal to $\lfloor n/2\rfloor+1$ will check the block validity and vote in favour of the issued block iff it is valid, and against if it is invalid, and the strategic agents with index strictly greater than $\lfloor n/2\rfloor+1$ will send a vote in favour of the block without checking its validity.

– Let us call the strategic agents with index smaller or equal to $\lfloor n/2\rfloor+1$, the checkers.
The expected gain of each checker is $(1-p)R-c_{send}-c_{check}$, and the expected gain of non-checkers is $(1-p)R-c_{send}$.
We first show that non-checkers have no interest to deviate. If a non-checker deviates by checking the block validity and send a vote accordingly, the expected gain is $(1-p)R-c_{send}-c_{check}$ which is lower than $(1-p)R-c_{send}$ since $c_{check}>0$. All other deviations are trivially dominated.
Now turns to the checkers. We show that they also have no interest in deviating. Let us denote by $\mathcal{O}$ the random event *“all obedient agents have index lower or equal to* $\lfloor n/2\rfloor+1$*”*. Indeed, if a checker deviates by sending without checking, the expected gain is $\Pr(\mathcal{O})(R-c_{send}-p\kappa)+(1-\Pr(\mathcal{O}))(R-c_{send})$. This is lower than the gain at equilibrium iff

$$\begin{aligned}
&(1-p)R-c_{send}-c_{check}-\Pr(\mathcal{O})(R-c_{send}-p\kappa)-(1-\Pr(\mathcal{O}))(R-c_{send})\geq 0\\
&\iff -pR-c_{check}+\Pr(\mathcal{O})p\kappa\geq 0\\
&\iff \Pr(\mathcal{O})p\kappa\geq pR+c_{check}\\
&\iff \kappa\geq\frac{pR+c_{check}}{p\Pr(\mathcal{O})}
\end{aligned}$$

[5] Notice here that we do not impose any repartition on the obedient agents, only the probability is computed.

which is the condition in the Theorem.
If a checker deviates by checking and sending a vote in favour iff the block is valid, and no vote if invalid, the gain at deviation is $(1-p)(R-c_{send}-c_{check})$ which is lower than $\Pr(\mathcal{O})(R-c_{send}-p\kappa)+(1-\Pr(\mathcal{O}))(R-c_{send})$ iff

$$\begin{aligned}
&(1-p)(R-c_{send})-c_{check}-\Pr(\mathcal{O})(R-c_{send}-p\kappa)+(1-\Pr(\mathcal{O}))(R-c_{send})\geq 0\\
&\iff \Pr(\mathcal{O})p\kappa-p(R-c_{send})-c_{check}\geq 0\\
&\iff \Pr(\mathcal{O})p\kappa\geq p(R-c_{send})+c_{check}\\
&\iff \kappa\geq\frac{pR+c_{check}}{p\Pr(\mathcal{O})}-\frac{c_{send}}{p\Pr(\mathcal{O})}
\end{aligned}$$

This is guarantee under the condition in the Theorem. So no checkers has the incentive to deviate. All other deviations are trivially dominated.

– Now, we show that the above equilibrium satisfies the DAG-based ledger correctness properties. In the above strategy profile, only valid blocks get the majority of votes and gets added, and no invalid block is added by any strategic agent. Since all agents have the same view, and are adding the same block if valid, and no block if invalid, we can show that in that equilibrium, all DAG-based ledger correctness properties to hold.

Theorem 10 If there is a single issuer, when the issued block may be invalid with probability p, and there are only strategic and crash faulty agents, in Algorithm 3, then there exists a Nash equilibrium in which strategic agents do not check the block validity but vote in favour for the issued block. In that case, the DAG-Validity properties may be violated.

Proof (Proof of Theorem 10). We decompose this proof.

– First, let us show that the strategy profile where no agent checks the issued block validity but sends a vote in favour does not satisfies the DAG-Validity property.
The expected gain of each agent is $R-c_{send}-p\kappa$.
If the agent does not send a message, the expected gain will be at best $-p\kappa$, since $n>1$ which is lower than $R-c_{send}-p\kappa$.
If the agent decided to check the block's validity, and for in favour of valid block, and against invalid one, since $n>2$ the vote against will not change the decision, therefore, the gain at expectation will be $(1-p)R-c_{check}-c_{send}-p\kappa$, which is lower than $R-c_{send}-p\kappa$.
Lastly, if the agent decides to deviate by checking validity and only sending vote in favour of valid blocks, the expected gain is $(1-p)(R-c_{send})-c_{check}-p\kappa$; while this deviation is better than the previous, it is still lower than $R-c_{send}-p\kappa$. Therefore, the strategy profile is a Nash equilibrium.
– Now, let us show that the strategy profile where no one checks but sends a vote in favour or the issued block is a Nash equilibrium.
In this equilibrium, if the issued block is invalid, it will be added by all strategic agents since they do not check the block's validity. Adding an invalid block violates the DAG-Validity property.

Theorem 11 If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic and f crash faulty agents, and if we have $\kappa\geq(pR+c_{check})/p\Pr($*"all crash faulty agents have index lower or equal to* $f+\lfloor n/2\rfloor+1$*"*$)$, in Algorithm 3, there exists a Nash equilibrium in which the DAG-based ledger correctness properties are satisfied.

The strategy profile where the first $\min(f+\lfloor n/2\rfloor+1,n)$ strategic agents check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, and the others do not check the validity but send a vote in favour of the block is such a Nash equilibrium.

Proof (Proof of Theorem 11). We decompose this proof.

- First, let us show that a strategy profile where the first $\min(f + \lfloor n/2 \rfloor + 1, n)$ strategic agents check the block's validity and send a message for the block if it is valid and a vote against the block if it is invalid, and the others do not check the validity but send a vote in favour of the block is a Nash equilibrium. Let us call the strategic agents with index smaller or equal to $f + \lfloor n/2 \rfloor + 1$, the checkers. Moreover, assume that the strategic agents are averse to the fault in the sense that they believe crash faulty agents will crash before sending their votes. We note that since the position of crash faulty agents is not known, we cannot simply apply Theorem 8 to demonstrate this Theorem. However, it does resemble the of Theorem 9.
  The expected gain of each checker is $(1-p)R - c_{send} - c_{check}$, and the expected gain of non-checkers is $(1-p)R - c_{send}$.
  We first show that non-checkers have no interest to deviate. If a non-checker deviates by checking the block validity and send a vote accordingly, the expected gain is $(1-p)R - c_{send} - c_{check}$ which is lower than $(1-p)R - c_{send}$ since $c_{check} > 0$. All other deviations are trivially dominated.
  Now turns to the checkers. We show that they also have no interest in deviating. Let us denote by $\mathcal{F}_f$ the random event *"all crash faulty agents have index lower or equal to* $f + \lfloor n/2 \rfloor + 1$*"*. Indeed, if a checker deviates by sending without checking, the expected gain is $\Pr(\mathcal{F}_f)(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{F}_f))(R - c_{send})$. This is lower than the gain at equilibrium iff

$$\begin{aligned}
&(1-p)R - c_{send} - c_{check} - \Pr(\mathcal{F}_f)(R - c_{send} - p\kappa) - (1 - \Pr(\mathcal{F}_f))(R - c_{send}) \geq 0 \\
&\iff -pR - c_{check} + \Pr(\mathcal{F}_f)p\kappa \geq 0 \\
&\iff \Pr(\mathcal{F}_f)p\kappa \geq pR + c_{check} \\
&\iff \kappa \geq \frac{pR + c_{check}}{p \Pr(\mathcal{F}_f)}
\end{aligned}$$

  which is the condition in the Theorem.
  If a checker deviates by checking and sending a vote in favour iff the block is valid, and no vote if invalid, the gain at deviation is $(1-p)(R - c_{send} - c_{check})$ which is lower than $\Pr(\mathcal{F}_f)(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{F}_f))(R - c_{send})$ iff

$$\begin{aligned}
&(1-p)(R - c_{send}) - c_{check} - \Pr(\mathcal{F}_f)(R - c_{send} - p\kappa) + (1 - \Pr(\mathcal{F}_f))(R - c_{send}) \geq 0 \\
&\iff \Pr(\mathcal{F}_f)p\kappa - p(R - c_{send}) - c_{check} \geq 0 \\
&\iff \Pr(\mathcal{F}_f)p\kappa \geq p(R - c_{send}) + c_{check} \\
&\iff \kappa \geq \frac{pR + c_{check}}{p \Pr(\mathcal{F}_f)} - \frac{c_{send}}{p \Pr(\mathcal{F}_f)}
\end{aligned}$$

  This is guarantee under the condition in the Theorem. So no checkers has the incentive to deviate. All other deviations are trivially dominated.
- Now, we show that the above equilibrium satisfies the DAG-based ledger correctness properties. In the above strategy profile, only valid blocks get the majority of votes and gets added, and no invalid block is added by any strategic agent. Since all agents have the same view, and are adding the same block if valid, and no block if invalid, we can show that in that equilibrium, all DAG-based ledger correctness properties to hold.