A Game Theoretical Analysis of DAG-based ledgers Backbone

Yackolley Amoussou-Guenou* Simone Galimberti† Maria Potop-Butucaru†

*Université Paris-Panthéon-Assas, CRED, Paris, France
†Sorbonne Université, CNRS, LIP6, Paris, France

Abstract

We study the rational behaviors of agents in *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 gives rewards to agents who participated in the validation procedure, we study using game theory how strategic agents behave while trying to maximize their gains. We consider scenarios with different type of agents and investigate if there exist equilibria where the properties of the protocols are guaranteed. The analysis is focused on the study of equilibria when invalid blocks may be issued. We found that in such a case, there exist equilibria where protocols properties may be violated. However, we also show that in all studied cases, there exist equilibria satifisfying the protocol properties.

Keywords: Blockchains, Game Theory, Directed Acyclic Graph

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. They facilitate trustworthy trades and exchanges over the Internet, power cryptocurrencies, ensure transparency for documents, and much more. 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 releasing of the most popular blockchains (e.g., Bitcoin or Ethereum) with a specific focus on economical transactions, their huge potential for various other applications became evident. However, quickly after their release, blockchains reached their limits in terms of throughput, blocksize and unforeseen behaviors. Therefore, non academic research further concentrate in proposing alternatives by improving some performance aspects but with non zero costs either on security or throughput. One of these solutions extends the blockchain to a DAG (directed acyclic graph) overlay and provide an ordering over all blocks and transactions, and outputs a consistent set of accepted transactions. In the research along this line, transactions are still organized on blocks. All these DAG-based systems structure blocks in a Directed Acyclic Graph (e.g. Spectre [19], GhostDAG [20], Phantom [21], IOTA [16], Avalanche [18], Glacier [4], DAG-Rider [12], Sycomore [7], Fino [14] etc.), where each block can include one or several references to its predecessors.

Interestingly, most of these blockchains share the same structure. It is therefore interesting to study and understand the key component sof that structure, and analyse its correctness properties. Moreover, as in all blockchains, even in DAG-based ledgers, there is an inherent financial aspect to consider. Indeed tokens and crypto-currencies may be in place, more so since nodes building the ledger (miners, validators) receive a reward. As these rewards serve to incentivise nodes to work properly, it should be analyse whether they satisfy the said objectives.

The objective of this paper is then twofold. First, we explicitly formalise the generic backbone protocol of DAG-based distributed ledgers and define their correctness properties, and second, we analyse the correctness of this backbone protocol when considering correct agents and crash faulty, as well as nodes behaving rationnal such as to increase their expected gain given some reward function. For the latter, we use game theory.

In this paper we extract the backbone of a DAG-based blockchain we analyze the behavior of rational agents. We show the different equilibria that exist given different scenarios where other type of agents are present. We analyze if the equilibria do satisfy the DAG-backbone invariants.

It should be noted that the analysis of strategic behaviors in blockchains (see for example, [5, 9, 10, 13, 15, 22]) mostly focuses on blockchains which are chain-based ledgers and not on DAGs. To the best of our knowledge, no work has been dedicated to analyze or discuss the rational behaviors among agents in *DAG*-based blockchains. The only work proposed in the context of the IOTA ledger is [17] which studies the attachment preferences. Contrarily to [17] which is complementary, our work analyses using game theory the backbone of a protocol which is shared by multiple DAG-based ledgers. We did not focus on how to choose the parent(s).

In this work, we show that in all considered settings, no matter the different types of agents, if some security thresholds (for example on the number of faulty or malicious nodes) hold, then there exist situations where the DAG-based distributed ledgers correctness properties hold. We pinpoint in Table 1 the properties demonstrating our positive findings for each scenario considered.

The rest of this paper is organised as follows. In Section 2, we present the model, show the formalisation of the DAG-based distributed ledger algorithms, and defined the problem we study, the DAG-based ledger correctness properties. In Section 3, we show the correctness of the protocols when agents are classical, i.e., obedient and crash faulty. In Section 4, we analyse the cases where strategic agents are in play, and we conclude in Section 5.

		Without communication	With communication
Valid	Obedient	Lemma 2	Lemma 2
	Strategic	Lemma 9	Lemma 10
Invalid	Obedient	Lemma 3	Lemma 3
	Obedient + Crash	Lemma 4	Lemma 4
	Obedient + Crash + Malicious	Proposition 5	Proposition 6
	Strategic	Corollary 14	Proposition 16
	Strategic + Obedient	▨	Proposition 17
	Strategic + Crash	▨	Proposition 20
	Strategic + Cost-indifferent malicious	▨	Proposition 21
	Strategic + Cost-indifferent malicious	▨	Proposition 22

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, there may be some equilibria not satisfying the correctness properties, as in Proposition 15 and Proposition 19. Moreover, when strategic agents are considered, it is not interesting to discuss the case with no communication, since each behaviour is only local and there is no coordination required. That zone is greyed out.

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 $\Pi = \{1, \ldots, n\}$. For the computation of equilibria, we assume in the following that the set of agents is ordered. The order chosen is done at random before the beginning of the execution.

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. At the end of a round, an agent exits from the current round and starts the next one. The system being synchronous, agents can set the duration of their delivery phase to collect all previously sent messages. We assume as usual that messages are signed, and signature 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*. Messages are not lost, altered, nor duplicated.

In classical distributed systems, agents are either correct (they follow exactly the given protocol) or faulty. Agent can be faulty for example by crashing (they stop executing the protocol and do not receive nor send messages anymore). We only only consider faults by crash in this work. We consider an additional type of agents, 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 [2, 3]. Last but not the least, we also consider malicious agents. Malicious agents have the objective of breaking the system, or trying their best to break the safety or liveness properties of the system.

In this work, 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 where a crash faulty agent stops working

is not defined nor decided in advance;

- **Strategic** (rational, or selfish): those that want to maximise their expected gain by deviating if needed, but they do not desired to break the correctness guarantees of the protocol; and

- **Malicious**: those that want to hurt the system by trying to preclude the protocol to achieve the correctness properties. Given our communication assumptions, we do not allow the malicious agents to equivocate, i.e., they are not allowed to send multiple messages corresponding to the same step, or not allowed to send messages that can only be received by some and not others. In some sense, we can think of it as if there is no point to point communication.

  Notice that malicious agents are not per se Byzantine agents, since in the general fashion, Byzantine agents' objective is either unknown or totally arbitrary [1]. However, they are a subclass of Byzantine agents whose behaviour is clearly defined to break the correctness properties.

We assume malicious, and selfish 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 analyse 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 [16], Ghost [20], Avalanche and Glacier [18, 4]) or more (e.g., Spectre [19]) 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 see a submitted transaction not already in the DAG.

Vertices of a DAG, which we will simply call blocks hereafter, contains (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. The leaves in the DAG are called *tips*.

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 ledger 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. Formally, a DAG-based ledger verifies the following properties:

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-Growth. *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: $DAG_i = DAG_j$.*

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** $\Delta$
```

- DAG-Liveness. *If a correct or strategic agent* i *appends a valid block* $\mathcal{B}$ *to* DAG_i*, then all non faulty agents eventually append* $\mathcal{B}$ *to their own local DAG;*

- DAG-Validity. *For any non malicious agent* i*, each block in* DAG_i *is valid; it 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-Growth, DAG-Uniformity, DAG-Liveness, and DAG-Validity) in the presence of the different types of agents detailed above. Moreover, in this work, we focus on the strategic behaviour on 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 analyse the impact of a new block issued.

While our assumption may seem strong, it is a first step in the analyses of the dynamic setting. Indeed, to study the dynamic game, one needs to study the single stages of the game. This analysis poses these basis, and we believe the result to be worthwhile on their own, and we let the dynamic analysis as further 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)[1] 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 will be added the new issued block. This procedure is computational 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 branches. The block $\mathcal{B}$ issued by i contains i's *id*, the transaction *tr* and the hashes referring to the parents 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 a time Δ to be sure that anyone receives the block.

As we can notice, in Algorithm 1, there is no choice given to the issuer. We assume that it is not directly an agent of our analysis, and we will consider two cases. First, when the block issued is always valid, and second, when there is a know probability that the block issued may be invalid.

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 initial the version of IOTA [16]. However, such protocol was not consistent [11]. Therefore, updates or other protocols require a (lightweight) consensus mechanism before inserting a new block. Such mechanism is what we call *communication among agents* in the following.

[1]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    $action^{check} := \varnothing$
                                        // $action^{check} \in \{$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     $DAG_i := update(DAG_i, \mathcal{B})$
```

Algorithm 3 represents the receiver's algorithm when we consider communication among agents. In this case, each agent broadcasts or not a vote for the issued block, and to add a block to one's 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 agents has his own time clock and maybe some delay is present between votes' arrival. From *line 14* to *line 17* the receiver must wait Δ and 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 insert the block $\mathcal{B}$ according to the majority of the received messages.

In both Algorithms 2 and 3, the actions given to *obedient* agents is to set their $action^{check}$ to `true`, and to 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. We let the choice there to emulate the costly action of checking a block, and deciding whether to send a vote knowing the validity of the block.

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

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

We propose the backbone of a DAG-Based Ledger with only obedient agents and one correct issuer. Since there is no crash nor malicious agents we can simplify

We note that in this simplify version of the protocol the agent receives a block, without checking the validity since it is guaranteed, then she updates her own local DAG_i. The following lemma proves that Algorithms 3 and 2 verify the properties of a DAG-based ledger when all agents are obedient.

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

Assuming that with some probability p, the issuer generates an invalid proposal, some blocks can now be invalid. The above lemma does not hold anymore. The following lemma analyses that case.

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
  $action^{check} := \varnothing$
                                                     // $action^{check} \in \{$true, false$\}$
  $action^{send} := \varnothing$
                                                     // $action^{send} \in \{$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 $\Delta$
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
      $DAG_i := update(DAG_i, \mathcal{B})$
```

Lemma 3. *If there is a single issuer, the block issued may be invalid with probability p, and all agents are obedient, then Algorithms 2 and 3 verifies the DAG-based ledger correctness properties.*

Intuitively, both lemmas 2 and 3 hold 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 as shown by the following lemma.

Lemma 4. *If there is a single issuer, the block issued may be invalid with probability p, 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 could be interesting to analyse the resilience of the protocol with only obedient, crash faulty and malicious agents. To do so, we need to restrict or properly define all possible actions for the malicious. 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 do guarantees the DAG-based ledger correctness properties.

Proposition 5. *If there is a single issuer, the block issued may be invalid, and agents can be obedient, crash faulty, or malicious, then Algorithm 2 verifies the DAG-based ledger correctness properties.*

Proposition 5 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, however, needs more assumption on the number of crash faulty and malicious agents to guarantee correctness.

Proposition 6. *If there is a single issuer, the block issued may be invalid, 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.*

In more details, Proposition 6, showing 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.

In this section we show the correctness of the backbone protocol we propose. However, as we will see in future sections, replacing obedient agents by strategic, the correctness is not necessarily guaranteed anymore.

4 DAG-Based Distributed Ledger Backbone with Strategic agents

In this section we study the backbone of DAG-based ledgers considering strategic agents. In this framework agents try to maximize their own gain, so they may decide not to follow the protocol. To be able to analyse 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, initialized at a default value $\varnothing$, can take value from the set $\{\texttt{false}, \texttt{true}\}$. For agent i, the value of $action^{check}$ is set by calling respectively the functions $\sigma_i^{check}()$, returning its strategy. The strategy $\sigma_i^{check}()$ determines if i, the receiving agent, chooses to check the validity of the proposal or not, which is a costly action. The cost of such action is represented by c_{check}.

When communication between agents is required before updating one DAG, the strategic agents have the choice of sending or not a vote for the issued block, assessing its validity. That action is represented in Algorithm 3 by the dedicated variable $action^{check}$. 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. The cost of such action is represented by c_{send}.

To summarise, after receiving the proposal block, each strategic agent decides whether to check the block's validity or not (at cost c_{check}), and second, in Algorithm 3 decides whether to send a message (at cost c_{send}) or not.

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 $\sigma_i^{check}()$. 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 $\sigma_i^{send}(validValue)$, where $validValue$ is the information the agent has about the proposal validity.

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 $(\sigma_i^{check}, \sigma_i^{send})$.

The expected gain of strategic agent i is:

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

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, since in that case, they are not asked to send a vote.

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]. Indeed, it does seem such mechanism will be an incentive for agents to send good messages, while rewarding everyone at the same time could lead to scenario of free-riding, i.e., some agents will not participate, hoping the others will do the job, but still be rewarded.

In our analysis, we assume that when an invalid block is inserted, all agents are punished by incurring in a cost κ. While a reward R, is given to the agents when a block is produced. We already presented before, the cost c_{check} of checking validity, moreover another action will be possible for agents which will be able to communicate by paying c_{send} of sending a message in order to coordinate. Additionally, we assume that the reward obtained when a block is produced is smaller than the cost κ of producing an invalid block and the cost of checking is greater than sending. That is,

$$\kappa > R > c_{check} \geq c_{send} \geq 0.$$

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 case 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.

Remark 7. *These are the only costs considered in this work. Indeed, we think that they represent all the existing costs in these protocols, and to the best of our knowledge, there is no other cost.*

The game being defined, as often using game theory, we will use the concept of *Nash equilibrium* to analyse the behaviour of agents. 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 agent i deviates by doing σ'_i instead of σ_i, and all other agents continue with the same strategies. Therefore, formally, a Nash equilibrium can be defined as follows:

Definition 8. *(Nash Equilibrium). A strategy profile σ is a Nash equilibrium iff:*

$$\forall i \in N, U_i(\sigma) \geq U_i((\sigma_{-i}, \sigma'_i)).$$

In the rest of this paper, we will analyse 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 situation where we only have strategic agents in the system.

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 agents 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.

Proposition 9. *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.*

The same holds also in Algorithm 3. The agents do not have the incentive to check the block validity, since it is known to be valid, however, they have to send a vote in favour of the block, otherwise they will not get a reward.

Proposition 10. *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 Lemma 9 or Lemma 10 is that when the block is always valid, the strategy imposed to obedient agents is not a Nash equilibrium. Therefore, in the situation where all proposed blocks are always valid, there is no point of checking blocks' validity.

Corollary 11. *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 this rest of the section, we analyse 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. First, in the situation where the issued block may be invalid, we study the case where there is no communication between the agents. It means that strategic agents follow Algorithm 2.

As we can notice, in this case, the agents now may have a certain interest in checking the validity of the issued block, otherwise, they will insert invalid block in their DAG, and therefore suffering cost $-\kappa$.

Proposition 12. *In Algorithm 2, if there is a single issuer, when issued blocks may be invalid with 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.*

The above proposition shows that 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, as shown by the next proposition, if the cost of adding an invalid block is high enough, there exists an equilibrium in which strategic agents check block validity.

Proposition 13. *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.*

A direct corollary is therefore that in this situation, the backbone protocol satisfies all DAG-based ledger correctness properties.

Corollary 14. *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. Now we add the broadcast of the opinion of agents about the validity of the block issued. After the checking phase each agent will receive the decisions of the others (by their votes) and according to a simple majority rule the agent will add or not the block to the local DAG. Here, all agents follow Algorithm 3.

Proposition 15. *If there is a single issuer, when the issued block may be invalid with 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 in this setting.*

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

Proposition 16. *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

The previous section helps showing what can happen when strategic behaviours enter into action. Satisfying the properties is not necessarily guaranteed anymore, but there are situations where the DAG-based ledger properties are satisfied. Now, for the rest of the paper, we study the case where we mix strategic agents with some other agents (obedient, crash faulty and malicious). Moreover, we only analyse the situation of Algorithm 3 since adding multiple agents in Algorithm 2 does not seem to have an impact on any individual agents.

4.2.1 Equilibria with Strategic and Obedient Agents

First, let us discuss the case where there are only obedient and strategic agents.

Proposition 17. *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 $\kappa \geq (pR + c_{check})/p \Pr($"all obedient agents have index lower or equal to $\lfloor n/2 \rfloor + 1$"$)$ there is a Nash equilibrium in which the DAG-ledger correctness properties hold.*

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, is such a Nash equilibrium.

Remark 18. *This case can be extended to cover the cases with less obedient agents.*

4.2.2 Equilibria with Strategic and Crash Faulty Agents

We analyse the situation where crash faulty agents are present. A crash faulty agent is an agent that crashes and stops following the protocol. The two next results are similar to Proposition 15.

Proposition 19. *If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic and crash faulty agents, in Algorithm 3, 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.*

Indeed, Proposition 19 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. In other words, the majority we talk about is not a majority over n, but a majority over the number of received votes.

Proposition 20. *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.

In Proposition 20, since agents can fail by crashing, there is a 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. More in details, 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.

Cost-indifferent malicious agents Let us assume that the number of cost-indifferent malicious agent 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.

$$\tilde{\kappa}(\omega) = \begin{cases} \tilde{\kappa}_{Validity}, & \text{if } \omega \text{ does not satisfy } \textit{DAG-Validity} \\ 0, & \text{if } \omega \text{ satisfies } \textit{DAG-Validity} \end{cases} \tag{1}$$

If ω is clear from the context, we simply write $\tilde{\kappa}$. The expected gain of cost-indifferent malicious agent i is:

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

Proposition 21. *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 $\kappa \geq (pR + c_{check})/p\Pr($ "all malicious agents have index stictly greater than $\lfloor n/2 \rfloor + \mathbf{m}$"$)$, 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 of Proposition 21. First, let us prove that the strategy profile described in the proposition is a Nash equilibrium.

In that strategy profile, the gain of the malicious agents is 0, 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 of, 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

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

  which is the condition in the proposition.

  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

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

  This is guarantee under the condition in the proposition. 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 harming the system but they incur, differently from cost-indifferent malicious ones, the costs for checking and sending.

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 $\tilde{\kappa}_{Validity} > 0$.

$$\tilde{\kappa}(\omega) = \begin{cases} \tilde{\kappa}_{Validity}, & \text{if } \omega \text{ does not satisfy } \textit{DAG-Validity} \\ 0, & \text{if } \omega \text{ satisfies } \textit{DAG-Validity} \end{cases} \tag{2}$$

If ω is clear from the context, we simply write $\tilde{\kappa}$. The expected gain of cost-sensitive malicious agent i is:

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

It is important to note that the strategy profile which was an equilibrium strategy for Proposition 21 is not a Nash equilibrium when cost-indifferent malicious agents are replaced by cost-sensitive malicious agent. In fact, since cost-sensitive ones try to reduce their cost, the deviation of not checking nor sending is profitable, since they were checking and sending vote in favour of invalid blocks, while no invalid block could be accepted in that strategy profile. The next proposition, however, shows that there is another strategy profile which is a Nash equilibrium with cost-sensitive malicious agents that guarantees the DAG-ledger correctness properties.

Proposition 22. *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:*

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

There exists a Nash equilibrium where the DAG-ledger correctness properties hold.

The strategy profile where the 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, and the others strategic agents do not check the validity but send a vote in favour of the block is a Nash equilibrium, and cost-sensitive malicious agents do nothing.*

Proof of Proposition 22. Let us show that the strategy profile is a Nash equilibrium.

This resemble the strategy profile used to demonstrate Proposition 20. 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 prefer 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

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

which is the condition in the proposition.

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

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

This is guarantee under the condition in the proposition. 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 analyze the behavior of rational agents in *DAG* based distributed ledger protocols. This paper studies the case where there is one block issued at the time. This helps understanding the backbone of multiple DAG-based distributed ledgers. We found that when invalid blocks may be proposed, there exist equilibria where the backbone properties may be violated. The results of our work can be used in improving the design of DAG-based ledgers. Indeed, this work shows that the design of such protocols, in particular even simple reward systems may not always have the desired properties. These results could serve to build the dynamic version of these protocols which are the case of multiple blocks issued in parallel. We let such a study as future work, as well as letting the option to agents to accept or not a block regardless of its validity or the number of votes received.

References

[1] Ittai Abraham, Lorenzo Alvisi, and Joseph Y. Halpern. Distributed computing meets game theory: Combining insights from two fields. *SIGACT News*, 42(2):69–76, jun 2011. doi:10.1145/1998037.1998055.

[2] Ittai Abraham, Danny Dolev, Rica Gonen, and Joe Halpern. 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, page 53–62, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1146381.1146393.

[3] Amitanand S. Aiyer, Lorenzo Alvisi, Allen Clement, Michael Dahlin, Jean-Philippe Martin, and Carl Porth. Bar fault tolerance for cooperative services. In *SOSP '05*, 2005.

[4] Ignacio Amores-Sesar, Christian Cachin, and Enrico Tedeschi. When is spring coming? A security analysis of avalanche consensus. In Eshcar Hillel, Roberto Palmieri, and Etienne Rivière, editors, *26th International Conference on Principles of Distributed Systems, OPODIS 2022, December 13-15, 2022, Brussels, Belgium*, volume 253 of *LIPIcs*, pages 10:1–10:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[5] Yackolley Amoussou-Guenou, Bruno Biais, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. Rational vs byzantine players in consensus-based blockchains. AAMAS '20, page 43–51. International Foundation for Autonomous Agents and Multiagent Systems, 2020.

[6] Yackolley Amoussou-Guenou, Bruno Biais, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. Committee-based blockchains as games between opportunistic players and adversaries. *The Review of Financial Studies*, page hhad051, 2023.

[7] Emmanuelle Anceaume, Antoine Guellier, Romaric Ludinard, and Bruno Sericola. Sycomore: A permissionless distributed ledger that self-adapts to transactions demand. In *17th IEEE International Symposium on Network Computing and Applications, NCA 2018, Cambridge, MA, USA, November 1-3, 2018*, pages 1–8. IEEE, 2018.

[8] Antonio Fernández Anta, Chryssis Georgiou, Maurice Herlihy, and Maria Potop-Butucaru. *Principles of Blockchain Systems*. Synthesis Lectures on Computer Science. Morgan & Claypool Publishers, 2021. doi:10.2200/S01102ED1V01Y202105CSL014.

[9] Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. *Commun. ACM*, 61(7):95–102, jun 2018. doi:10.1145/3212998.

[10] Mehdi Fooladgar, Mohammad Hossein Manshaei, Murtuza Jadliwala, and Mohammad Ashiqur Rahman. On incentive compatible role-based reward distribution in algorand. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 452–463, 2020. doi:10.1109/DSN48063.2020.00059.

[11] IOTA Foundation. The coordicide. 2020. URL: https://files.iota.org/papers/20200120_Coordicide_WP.pdf.

[12] Idit Keidar, Eleftherios Kokoris-Kogias, Oded Naor, and Alexander Spiegelman. All you need is DAG. In *PODC '21: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, July 26-30, 2021*, pages 165–175. ACM, 2021.

[13] Ziyao Liu, Nguyen Cong Luong, Wenbo Wang, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. A survey on blockchain: A game theoretical perspective. *IEEE Access*, 7:47615–47643, 2019. doi:10.1109/ACCESS.2019.2909924.

[14] Dahlia Malkhi and Pawel Szalachowski. Maximal extractable value (MEV) protection on a DAG. In *4th International Conference on Blockchain Economics, Security and Protocols, Tokenomics 2022, December 12-13, 2022, Paris, France*, volume 110 of *OASIcs*, pages 6:1–6:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[15] Conor McMenamin, Vanesa Daza, and Matteo Pontecorvi. Achieving state machine replication without honest players. In *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, AFT '21, page 1–14. Association for Computing Machinery, 2021. `doi:10.1145/3479722.3480986`.

[16] Serguei Popov. The tangle. 2015.

[17] Serguei Popov, Olivia Saa, and Paulo Finardi. Equilibria in the tangle. *Comput. Ind. Eng.*, 136:160–172, 2019. `doi:10.1016/j.cie.2019.07.025`.

[18] Team Rocket, Maofan Yin, Kevin Sekniqi, Robbert van Renesse, and Emin Gün Sirer. Scalable and probabilistic leaderless BFT consensus through metastability. *CoRR*, abs/1906.08936, 2019. URL: `http://arxiv.org/abs/1906.08936`, `arXiv:1906.08936`.

[19] Yonatan Sompolinsky, Yoad Lewenberg, and Aviv Zohar. Spectre: A fast and scalable cryptocurrency protocol. *IACR Cryptol. ePrint Arch.*, 2016:1159, 2016.

[20] Yonatan Sompolinsky, Shai Wyborski, and Aviv Zohar. PHANTOM GHOSTDAG: a scalable generalization of nakamoto consensus: September 2, 2021. In Foteini Baldimtsi and Tim Roughgarden, editors, *AFT '21: 3rd ACM Conference on Advances in Financial Technologies, Arlington, Virginia, USA, September 26 - 28, 2021*, pages 57–70. ACM, 2021. `doi:10.1145/3479722.3480990`.

[21] Yonatan Sompolinsky and Aviv Zohar. PHANTOM: A scalable blockdag protocol. *IACR Cryptol. ePrint Arch.*, page 104, 2018. URL: `http://eprint.iacr.org/2018/104`.

[22] Itay Tsabary and Ittay Eyal. The gap game. In *Proceedings of the 2018 ACM SIGSAC conference on Computer and Communications Security*, pages 713–728, 2018.

A Appendix: Proofs

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

Proof of Lemma 2. We decompose the proof.

- First, we show that Algorithm 2 satisfies the DAG-based ledger correctness properties. We do so by showing that each property holds.
  - *DAG-Growth*: *DAG-Growth* property is straightforward. Since the block issued is valid by assumption, the obedient agent receiving will add it 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 messages, it means that they all collect the same blocks. Moreover, since all blocks are valid by assumption, it means that obedients 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-Growth*: 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 set of messages, 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-Growth* and *DAG-Uniformity* cases. All agents appends the same issued block to their local DAG.
  - *DAG-Validity*: Trivial, since by assumption the issued block is valid.

□

Lemma 3 If there is a single issuer, the block issued may be invalid with probability p, and all agents are obedient, then Algorithms 2 and 3 verifies the DAG-based ledger correctness properties.

Proof of Lemma 3. 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 agent 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 2 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 4 If there is a single issuer, the block issued may be invalid with probability p, 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 of Lemma 4. 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 communication between agents, and since the properties do not concern crash faulty agents, applying Lemma 3 show that *DAG-Growth*, *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 3. 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.

□

Proposition 5 If there is a single issuer, the block issued may be invalid, and agents can be obedient, crash faulty, or malicious, then Algorithm 2 verifies the DAG-based ledger correctness properties.

Proof of Proposition 5. This is a corollary of Lemma 4. 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. □

Proposition 6 If there is a single issuer, the block issued may be invalid, 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 of Proposition 6. 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 their 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 4. □

Proposition 9 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 of Proposition 9. 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. □

Proposition 10 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 of Proposition 10. 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. □

Proposition 12 In Algorithm 2, if there is a single issuer, when issued blocks may be invalid with 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 of Proposition 12. 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.

□

Proposition 13 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 of Proposition 13. 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 14 If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic agents, and if $\kappa \geq (1-p)R/p + c_{check}/p$, in equilibrium, Algorithm 2 verifies the DAG-based ledger correctness properties.

Proof of Corollary 14. In the equilibrium of Proposition 13, the strategic agents behave as if they were obedient, by Lemma 3, we can show that the DAG-ledger correctness properties hold. □

Proposition 15 If there is a single issuer, when the issued block may be invalid with 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 in this setting.

Proof of Proposition 15.

- 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 equillibrium.

- 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.

□

Proposition 16 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.

Proof of Proposition 16.

- 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 is 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.

$\square$

Proposition 17 *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 $\kappa \geq (pR + c_{check})/p\,\Pr($"all obedient agents have index lower or equal to $\lfloor n/2 \rfloor + 1$"$)$ there is a Nash equilibrium in which the DAG-ledger correctness properties hold.*

Proof of Proposition 17. 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

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

which is the condition in the proposition.

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 proposition. 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.

□

Proposition 19 If there is a single issuer, when the issued block may be invalid with probability p, there are only strategic and crash faulty agents, in Algorithm 3, the only Nash equilibrium is for strategic agents to not check the block validity but vote in favour for the issued block. In that case, the DAG-Validity properties may be violated.

Proof of Proposition 19. 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 equillibrium.

- 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.

$\square$

Proposition 20 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.

Proof of Proposition 20. 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 Proposition 16 to show this proposition. However, it does resemble the proof of Proposition 17.

  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 proposition.

  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 proposition. 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.

$\square$