Resolution of Blockchain Conflicts through Heuristics-based Game Theory and Multilayer Network Modeling

Alessandro Di Stefano
Computer Laboratory
University of Cambridge
Cambridge, UK
distefano.alessandro@gmail.com

Damiano Di Francesco Maesa
Computer Laboratory
University of Cambridge
Cambridge, UK
dd534@cam.ac.uk

Sajal K. Das
Department of Computer Science
Missouri University of Science and Technology
Rolla, USA
sdas@mst.edu

Pietro Liò
Computer Laboratory
University of Cambridge
Cambridge, UK
pl219@cam.ac.uk

April 22, 2020

Abstract

A blockchain is a fully distributed system in which the user behavior, actions and decisions are crucial for its operation. This paper discusses how to handle conflict situations affecting a blockchain system. Specifically, we model two real-world conflict scenarios – the Lazy Miner dilemma and the Impatient Seller dilemma – by proposing a novel multi-layer framework coupled with a heuristics-based game-theoretic modeling. The multi-layer approach provides a way to include cross-modality integration (human quality factors, such as reliability) and human actions on the blockchain. We design a multi-agent game-theoretic methodology combined with some statistical estimators derived from the heuristics. Our model also includes the concept of homophily, a human-related factor connected to the similarity and frequency of interactions on the multi-layer network. Based on the heuristics, a dynamically evolving measure of weights is further defined such that an agent increases or decreases the link weights to its neighbours according to the experienced payoffs. We show how data mining in blockchain data could be incorporated into a heuristic model which provides parameters for the game-theoretic payoff matrix. Thus, this work represents a platform for simulating the evolutionary dynamics of the agents' behaviors, including also heuristics and homophily on a multi-layer blockchain network.

***K*eywords** Game theory, Blockchain, heuristics, multi-layer networks, multi-agent systems

1 Introduction

The blockchain has emerged as a very promising technology attracting attention to both the academic and industry community [1, 2, 3]. Indeed, research efforts on blockchain technology have been on the rise due to its applications to a wide variety of domains, such as finance and insurance [4], biomedical and healthcare [5], information processing and Internet of Things (IoTs) [6], and access control systems [7], to name a few.

An important feature of blockchain lies in its distributed nature, allowing the users to build and manage a shared data repository without relying on any third party or central authority.

Such decentralised nature relies on a chosen distributed consensus algorithm so that the strategies and interactions between the users become crucial to achieve an agreement on the current state of the system among mutually untrusted entities. Through these strategic interactions on the blockchain, a user/node is able to learn and predict the behavior of other users during the consensus step. To this end, game-theoretic models offer an appropriate tool to formulate and explore the dynamics of user behaviors [8].

As many other real-world systems, composed of components interacting with each other, the blockchain also possesses rich multi-level structures and exhibit complex dynamics. The source of complexity is due to the presence of interactions among various components having autonomous decision-making capabilities.

Furthermore, the fully distributed nature of blockchain systems where many interacting elements or agents participate in transactions, lead to the emergence of collective properties that cannot be simply associated with the individual constituting elements. Therefore, the study of such systems requires new tools and modeling approaches in a holistic manner, with a goal to capture the collective emergence of behaviors. Hence, we propose to represent a blockchain system as a complex network similar to social networks, financial markets, and transportation networks. More precisely, we model blockchain as a multi-layer network, characterised by heterogeneous and complex agents interconnected on different layers. In this multi-layer network, the participants strategically interact with each other at different scales (time and volume) and at different frequencies directly or indirectly. Moreover, the various agents and components interact non-linearly in presence of perturbations.

Blockchains are also characterised by some conflict situations arising from the complex interactions between the agents and elements of the blockchain. The complexity exhibits the agent behaviors that often follow non-linear and nontrivial patterns, and additionally the statistical properties change with time and space. Thus, the multi-scale and evolving nature of blockchain justifies our complex networks approach in dealing with such systems. To the best of our knowledge, our work is the first to tackle two real-world conflict scenarios – the Lazy Miner dilemma and the Impatient Miner dilemma – that have strong impacts not only on the blockchain system development, but also on the user payoffs (gains or losses) and their future behavior.

To address the conflict scenarios, we propose a novel game-theoretic framework based on multi-agent heuristics. The framework models the blockchain network as a weighted multi-layer network between agents, and defines some statistical measures (based on heuristics) for each agent in the network. These measures dynamically change the link weights of the multi-layer network. Using a game-theoretic model based on an Iterated Prisoner's Dilemma, we quantify the evolutionary dynamics of the agent behaviors with the help of homophily between nodes, which allows us to measure the tendency to interact more with similar agents and how it impacts on their behaviors.

From a mathematical perspective, the novelty of our proposed methodology lies in combining game-theory with multi-agent heuristics. In general, two systems are typically used for decision making, which reflects the "dual process" of human thinking: one is rapid, intuitive but error-prone; while the other one is slower, reflective and statistical [9, 10]. Fast and frugal heuristics have been used to explain the intuitive part of human reasoning based on simple, efficient rules, learned or hard-coded by the evolutionary process. These heuristics allow human beings to take decisions and solve complex problems, in spite of incomplete information [10, 9]. Our game-theoretic model is based on the strategic interactions and the players' rationality capable of making corrections to the heuristics.

The contributions of our work can be summarised as follows.

- We model two novel real-world conflict scenarios characterising a cryptocurrency enabling blockchain. The scenarios are mapped into an analytical multi-agent framework based on a game-theoretic model, which allows us to quantify the evolutionary dynamics of agents' behaviors. The framework also includes human-related factors derived from the heuristics and homophily between agents, measuring their role in the emergent dynamics.
- We model a blockchain system as a weighted multi-layer network, which provides a way to include cross-modality integration (human quality factors, such as reliability) and human actions on the blockchain.
- We define a dynamical weighted multi-layer network, where link weights change over time based on the heuristics. We further introduce a strategy update based on the Fermi distribution, which

includes an utility function that combines the link weight and the payoffs obtained at each round of the Prisoner's Dilemma game.

- Heuristics allow us to connect multi-layer and game theory in the presented blockchain conflict scenarios. We integrate trust and actions on blockchain with the payoff matrix, allowing us to interpret the decision-making processes. Human heuristics model the human quality factors, showing a fluctuating behavior as illustrated in results.

The rest of the paper is organized as follows. Section 2 summarizes recent literature on blockchain relevant in our context and game-theoretic models for blockchain networks. Section 3 defines the two blockchain conflict scenarios we analyse. Section 4 deals with multi-layer representation of the blockchain network, and focuses on the proposed model and how it handles conflict scenarios. Section 5 presents preliminary simulation results while Section 6 concludes the paper with directions of future research.

2 Background and Related Works

A Blockchain protocol is considered as a possible implementation of the so called Distributed Ledger Technology (DLT), in which trust is a key concept. The DLT allows mutually untrusted entities to collaborate on a common goal, i.e., the management of a shared information repository. The trust in a central entity (or third party) in traditional systems is replaced by trust in the protocol itself. This disruptive aspect of DLT helps create trust at a technology level, so much so that it has been dubbed *trust machine* [11]. As an example, we can think of Bitcoin, the first Blockchain protocol proposed. The goal of Bitcoin is to provide any user the freedom to exchange value with any other untrusted user without the need for any kind of external supervision or guarantee.

Blockchains create trust among reciprocally untrusted Byzantine users by employing a distributed consensus algorithm. Achieving consensus is often the most expensive step (in terms of time and resources) in a Blockchain protocol. Cheaper solutions are possible if the concept of untrusted entities is relaxed and some level of trust is introduced in some chosen users, which leads to the concept of permissioned blockchains. In a permissioned blockchain, only a potentially dynamic set of trusted entities is allowed to update the ledger, while on permission-less blockchains (e.g., Bitcoin) any one can update. The more trust assumptions are placed on the entities, the more efficient would the system be, albeit more centralised.

Given trust is a key concept in any blockchain, with significant impacts on various aspects of a given protocol, many of the expensive security steps necessary in a public blockchain protocol are based on the trustless assumption. Thus the protocol can be made more efficient if some level of trust between users is introduced. We have already shown the example of permissioned blockchains obtained by relaxing the trustless assumption on the distributed consensus step, but such example regards the entire blockchain protocol and the derived community as a whole. Instead, in this paper, we show two examples where any user can individually choose to relax the strict trustless assumption (i.e., every other user is treated as an adversary) to achieve some personal gain. In such scenarios, a user may choose to trust other users outside of the consensus step, i.e., on operations not protected by it. Users choosing to do so risk to incur in a loss in case they get cheated by those users they choose to trust, but gain a direct benefit if the trusted user behaves honestly. This clearly leads to a complex game theory problem, in which the users play rationally to decide if and whom to trust. In the rest of this paper, we present two such scenarios common to any cryptocurrency based blockchain (such as Bitcoin or Ethereum) that we name *the lazy miner dilemma* and *the impatient seller dilemma*.

In traditional blockchain, *mining* means the set of operations that the validator nodes (i.e., *miners*) perform during each distributed consensus step. On the other hand, in traditional proof of work (PoW) protocols (e.g., Bitcoin), mining means performing a brute force attack to attempt a cryptographic hash partial inversion. It is an highly expensive process both in terms of initial investment (to buy purpose built efficient ASIC hardware), and run time cost (to pay for electricity to power the hardware). Since the best strategy to invert a cryptographic hash is brute force and only the first solution found is considered, the users take part in an arms race trying to perform as many hashes per millisecond as possible.

Users are incentivised to take part in the mining race by a variable reward of newly minted coins for each solution they find. More precisely, the miners have to partially invert the hash of the header of a block encapsulating a set of transactions not yet validated. The first miner that manages to solve such task, gets credited a reward equal to the fees paid by all the transactions in their block, plus a fixed reward (say, in Bitcoin halving every 210,000 blocks, i.e., about four years at the fixed expected generation rate of one block each 10 minutes, a little more in practice since blocks are found faster if the hash power grows).

As soon as a new block is *mined* (i.e., its hash puzzle is solved by a miner), it is broadcasted by the miner to all other users so that they can add it on top of their own chains. All other compliant miners would then check the newly received block by checking if it is well formed (i.e., contains only valid transactions and all hash pointers are correct), then stop mining their own, and start mining a new block on top of the newly received one. Whether or not a miner should behave honestly and stop mining its own block is an interesting game theoretic problem by itself; many attack models have been proposed in the literature as discussed in the following.

2.1 Game-Theoretic Models for Blockchain

There exist many works on the application of game theory analysis to blockchain technology. In fact, the antagonistic nature of the mining process, coupled with its highly valuable rewards is the perfect terrain for participants to be modelled as rational entities. The high stakes of the mining race have been first studied in [12] and subsequent works [13, 14]. (For a comprehensive survey on the most relevant results, see [8].) Almost all of these studies are focused around the study of optimal mining strategies and the adequacy of offered mining rewards; testing whether the protocol compliant mining strategy is the rational one to follow or not. Thus the topic is strictly intertwined with the study of fraudulent mining techniques, such as the notorious selfish mining strategy [15] or more general block withholding attacks [16]. Those kind of attacks can in turn be seen as application of a more general strategy of Information Withholding Attacks, either trying to gain an unfair advantage over other miners or attempting to be rewarded more than one's fair share from the mining pools [16]. The adequacy of miner rewards are even considered in the face of *Goldfinger Attack*, when an entity has a gain from taking part in the mining process not derived from the value of the mining rewards alone [12].

Our proposed scenario is concerned with the mining process as well, but does not consider directly the choice of a mining strategy or evaluation of the incentives mechanism. Instead, we study a possible non-compliant strategy that can speed up mining independently from the adopted mining strategy (compliant or fraudulent). To the best of our knowledge, we are the first to study from a game-theoretic point of view, the miners approach to blocks checking after new blocks notification.

In contrast, our second scenario does not concern mining at all, and is about the behavior of regular users. Despite the problem being well known (see [17]), to the best of our knowledge, no study has attempted to model the user strategies with game theory in such a scenario. Although a recent work aimed to solve this problem [18], the proposed solution is not generally applicable in our context because it requires a smart contract enabled blockchain (e.g., excluding the Bitcoin) and it works only for digital assets (of which an hash can be computed) whose exchange can be verified. As such, our broader application scenario is very different in that, for example, it includes regular vending machines accepting Bitcoins.

3 Blockchain Conflict Scenarios

3.1 The Lazy Miner Dilemma

It is worth remarking how miners pay a high initial investment to buy dedicated hardware and repay such investment by hoping to mine as many blocks as possible. The mining power of each miner increments only discretely and, during any fixed period, the only strategy to increase the chances to mine a new block is to mine for longer time. Any time spent not mining at all, or mining on a stale (old) block while a new block has already been received but not yet checked, should be minimized as much as possible.

In particular, when a new block is found by another miner, while the current miner has not yet verified it or been notified of its existence, the current miner keeps mining on a stale block with little chances of being accepted and hence provides a reward (since a new block has already been found and only one block can be accepted at a time). In general, this is the time when natural forks may occur (i.e., when two different miners find a new block at almost the same time); even if no fork occurs, all mining work is actually wasted during the time a newly found blocks is to be known and accepted by all other miners. As such, miners have an economic interest in trying to minimise such wasted mining time, and most already try to do so by establishing direct connections to other miners in an attempt to lower the latency that a newly found block is notified to all other miners. Still miners can not stop mining their potentially stale, local block as soon as they are notified about a new block. In fact, the new block might not be valid for two reasons: either the block structure (e.g., its header) is malformed, or the block content (i.e., its transactions) is not valid. In general, checking the validity of a block structure means computing a set of hashes, e.g., verifying that the hash puzzle was properly solved or the root of the hash tree of the content of the block is correct. On the

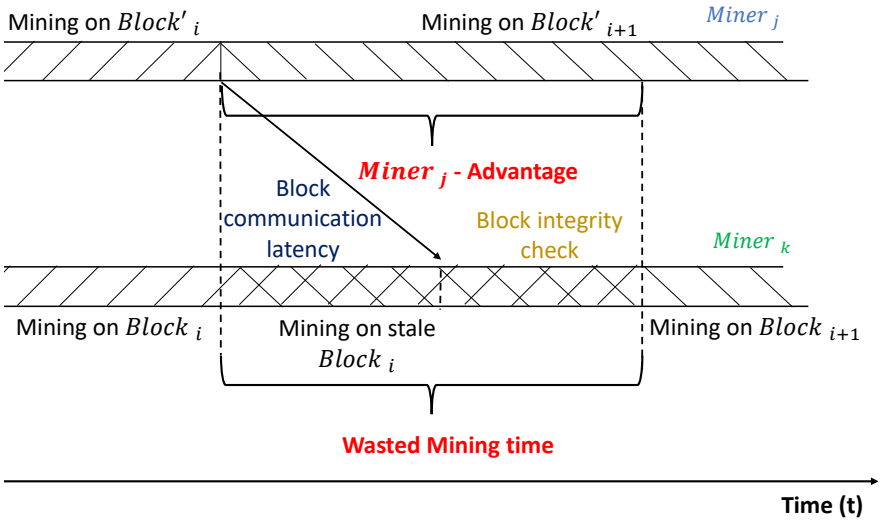

Figure 1: **Lazy Miner scenario** describing the wasted mining time by a miner (say $Miner_k$) when a new block is found by a different miner (say $Miner_j$).

other hand, checking the validity of a block content might be more expensive since it requires checking the validity of each transaction it contains.

Checking a transaction validity means verifying that the value spent by the transaction existed and belonged to the user attempting to spend it (often done by checking digital signatures) and executing the transaction content. In a simple blockchain, such as Bitcoin, this implies simply running a few script instructions contained in the transaction and checking if they terminate correctly. Whereas, in smart contract enabling blockchains, such as Ethereum, this can lead to more complex and time consuming computations. In general, all computations are bounded in order not to stall a miner in endless computations. But often such computations can be more complex, hence time consuming, as long as they pay a proportionate fee. The problem is that only the miner finding a new block gets such fees rewarded, while all other miners have to redo the same computations to check if the first miner did them correctly but got no reward whatsoever for the time spent in doing so. Moreover, all the time spent for checking is not the time spent in mining a new block and so a lowered chance of being the first one to mine the next block. Fig. 1 schematically describes this blockchain scenario.

Now we are ready to present the so called *lazy miner* strategy, in which a miner accepts new blocks from users that they deem trustworthy without checking for their content validity first. A lazy miner can then immediately start mining on the next block after simply checking its header integrity. The advantage of such strategy is that a lazy miner is cutting on checking downtime and thus mining more often for a possibly valid solution (i.e., no mining on possibly already stale blocks), maximising its expected gain. The associated risk is that a newly received block could be invalid and hence no further block mined on top of it would be valid and rejected by other compliant miners, as well as the first one. So the lazy miner is risking to waste all their mining time until the next block is found by some other miner.

Note that the checking time is usually much shorter than the average time between two subsequent blocks found. However, we argue that if the received block was not valid, all other honest but not lazy miners would not accept it. In other words, they would keep on mining their old blocks until they find a valid solution. Thus, the actual time a lazy miner would have to wait to find out if they were cheated would be, on average, much shorter than the normal time to wait for finding two consecutive valid blocks (e.g. 10 minutes in Bitcoin and between 10 and 19 seconds in Ethereum, as indicated respectively by the two protocols).

We also note that invalid blocks are generally not further forwarded to the neighbours when received by honest miners. This means that invalid blocks have a lower chance to naturally diffuse farther in the network than valid ones. Moreover, by checking the structural validity of a block, a Lazy Miner still refuses blocks with a wrong mining puzzle solution. For an adversary, mining invalid blocks is usually expensive (and may not be worth) since the attacker has to forfeit the rewards although performing the same amount work to produce valid blocks.

3.2 The Impatient Seller Dilemma

The distributed consensus algorithms employed to maintain the global state consistent in a distributed ledger technology (DLT) often provide only guarantees of eventual consistency. This means that, at any given time, there might be conflicting histories (called *fork* in a blockchain scenario) that will be re-conciliated eventually. In a Blockchain protocol, there is no guarantee that the state defined by the last block added to the chain would become the official state. The users wait until an unknown number of blocks is added on top of a given block yet to be considered definitive. Due to the lack of a strong consistency proof, the number of blocks to wait (called the *number of confirmations*) cannot be known at run time. In theory, a block at any depth could be invalidated by an even deeper fork.

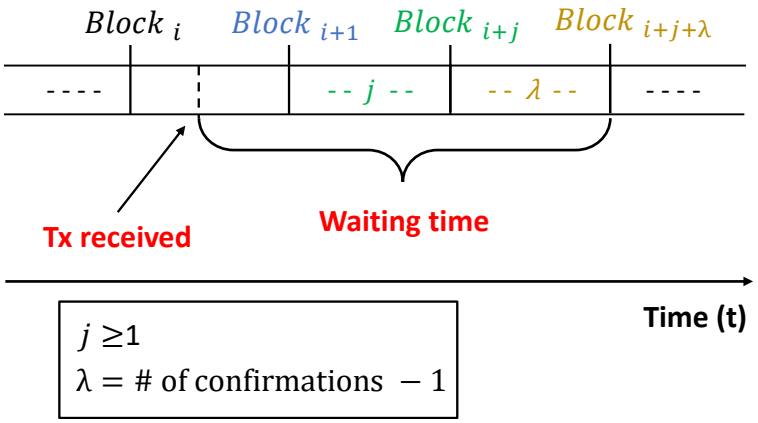

Figure 2: **Impatient Seller scenario.** It describes the conflict of an impatient seller, where j denotes the number of blocks to wait for the transaction to be included in a block.

In practice, natural forks are rare and resolved fairly quickly, while intentional forks (e.g., derived from history revision attempts) are not common at all in the most established blockchain protocols. An arbitrary number of confirmations (i.e., those blocks added on top of the chosen one) is often chosen as a safe practical compromise (e.g., six confirmations rule in Bitcoin).

A user waiting for a transaction to be validated has first to broadcast such transaction, waiting for the latency needed for the transaction to reach and be validated by enough nodes. The user then needs to wait for a miner to choose the transaction to be added to a block, as well as the mining time needed for such block. When a new block containing the transaction gets mined and added as new head of the chain, the user is still advised to wait for the recommended number of confirmations to avoid temporary inconsistencies. In general, the users are advised to use a number of confirmations proportional to the value exchanged, so the actual number of confirmations to wait can be even longer.

The waiting time may be acceptable in certain application scenarios such as e-commerce, where the shipment is the dominant time, but not in some other scenarios involving face to face exchange, such as buying coffee from a vending machine. For small amounts of value transacted, a user could choose to wait for a number of confirmations smaller than the advised safe one, but it would still have to wait for at least one (i.e., mining of the first block containing their transaction). For Bitcoin, given the average block generation time of 10 minutes and six confirmations rule, a user should expect to wait for at least one hour for full confirmation and more than 10 minutes for a *fast* one. This issue is further exacerbated in cases of transactions congestion times, when the blockchain receives more transactions than it can fit and hence the transactions have to wait for a long time to be validated inside the blocks. In practice, this makes some applications simply impossible like the aforementioned vending machine problem.

Such potentially long waits occur due to the zero trust design of public blockchains. If such assumption is relaxed and some level of trust is credited to given users, then it is possible to adopt *fast payments*. A fast payment is the one contained in a transaction that has not yet been verified in any block. Instead of waiting for the transaction to be mined, the recipient simply checks that it is well formed (e.g., spending existing funds belonging to the payer) and accepts the payment as if it was validated by the miners. Such *impatient seller* could then act as if the payment was final, for example, delivering the desired item or service. The risk of an impatient seller is that they may become victim of a *double spending attack*, i.e., the payer may attempt to spend the same funds with at least another transaction at the same time. Among all such

conflicting transactions attempting to spend the same funds, only one of them will be validated; if it is not the one directed to the impatient seller, they would not be credited any value because of their transaction becoming void.

It is possible to attempt to secure a fast payment by employing the so called *listening period*, i.e., waiting for a fixed amount of time to listen on the network for any other conflicting transaction attempting a double spending. But such techniques can only be effective by trusting a third party to perform the listening on behalf of the user or deploy a complex listener's network to protect the user against network partitioning or Sybil attacks [17]. In general, however, it is hard to make fast payments work since the goal is to avoid mining, despite mining being the process trough which transactions are validated in a blockchain distributed environment. Fig. 2 schematically describes this blockchain scenario.

In our evaluation scenario we consider an impatient seller as an user that finalizes a transaction (i.e. acts as if it had been validated in a block), as soon as it receives and checks it, without actually waiting for any mining. As such the user risks being victim of a double spending attempt (i.e. delivering the goods or services without receiving a valid payment for them), but avoids waiting for the long and potentially costly mining times. Of course such scenario is referred mainly to users involved in time sensitive and low value economic transactions.

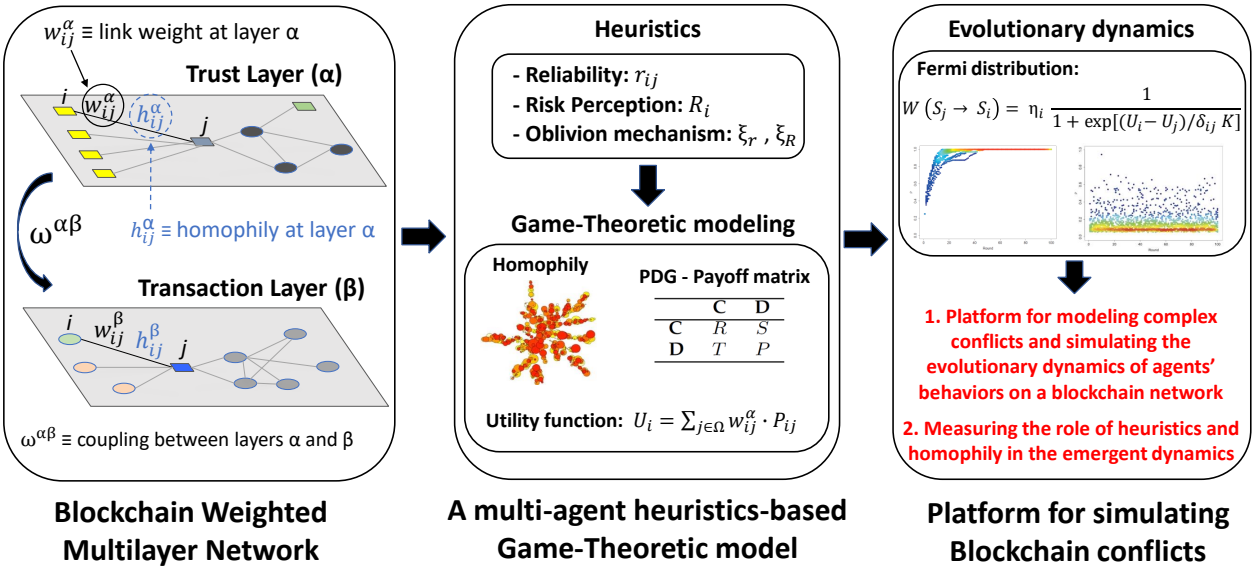

Figure 3: The modeling framework - from the Blockchain Weighted multi-layer Network to the multi-agent heuristics-based game-theoretic model which help explore the evolutionary dynamics of behaviors in the two dilemmas (conflict scenarios).

4 Blockchain as a Multi-layer network and Proposed Models

Given relationships between individual users and agents are typically multi-dimensional in nature, social behaviors and their dynamics can be represented by taking into account more than a single network of interactions. Recently many research works have focused on multi-layer networks [19, 20] as the most suitable way to describe a variety of complex networks, from social networks to biological networks to transportation networks, to name a few. Indeed, a complex system characterises an interdisciplinary field that deals with systems composed of many interacting units, often called "agents". In a multi-layer network, multiple interactions between agents occur in different contexts (scenarios) and an agent's behavior can be different in each layer, although it is determined from the simultaneous interactions on all the layers of the multiplex structure [19, 21, 20, 22, 23].

In the context of blockchain, our modeling approach considers a two-layer structure where we have an upper (logical) layer named as *Trust layer*, and a lower (physical) layer called as *Transaction layer* (see Fig. 3). We define a weighted multi-layer network for the blockchain, since each edge at both layers is weighted. Specifically, the edges (or links) on the Trust layer are defined according to the proposed heuristics, while the

weight of edges on the Transaction Layer depend on the number of transactions that two agents share (see model details in the following subsections).

A crucial aspect of our multi-layer modeling approach is that other than intra-layer relationships in each single network (or layer), we also need to encompass the interplay and the coupling between the two defined layers (i.e., inter-layer relationships). The multi-layer structure allows us to model and quantify the impact of human-related factors, such as the agents' heuristics and homophily in the emergent behaviors on the two layers. Furthermore, it allows us to understand the macro-, meso- and microscopic properties of a blockchain network [20, 24, 25].

Our proposed modeling approach is illustrated in Fig. 3, the components of which are described in the following.

4.1 Multi-agent Heuristics-based Game-theoretic Model

Consider an example of N agents (denoted as $i = 1, .., N$) and $M = 2$ layers, such as the Trust layer and Transaction layer denoted by α and β respectively. At both layers, an agent interacts with other K randomly chosen agents corresponding to its neighbors.

In our model, being neighbors implies not having a physical or social interaction with other agents. Instead, the interaction is indirect, namely the agents participate in the same transaction on the Transaction layer, or they have a certain reliability value against an agent on the Trust layer. Specifically, the edges between agents on the Transaction Layer indicate transactions on the blockchain involving both agents. Two agents may simultaneously participate in multiple transactions with a given neighbor, such that the more the number of transactions (N_{tr}) where both agents are involved, the higher is the link weight on the Transaction layer. Thus, at each time step t, the weight on the Transaction layer (β), denoted by $w_{ij}^{\beta}(t)$, is defined as follows:

$$w_{ij}^{\beta}(t) = N_{tr}(t) + 1 \tag{1}$$

where $N_{tr}(t)$ is the number of transactions in which both agents participate. The Trust layer represents a sort of 'logical' layer and the link weight between two agents at time step t is defined by a statistical estimator, called *reliability*, defined as follows.

Reliability between two agents i and j at time step t, denoted as r_{ij}(t), is a measure of trust between the agents (nodes), and it represents a dynamical memory that decides at each time step whether an agent is reliable or not. We assume that the reliability values are in the range 0 and 1, i.e., $0 \leq r_{ij} \leq 1$. The greater the value of r_{ij}, the more is the number of reliable neighbors.

Even though the reliability may be asymmetric in reality, here we assume that the reliability is a symmetric measure, i.e., $r_{ij} = r_{ji}$. With the help of reliability, let us now define the weight $w_{ij}^{\alpha}(t)$ of a link between two agents i and j in the Trust layer α at each time step t. The weight depends on the reliability $r_{ij}(t)$, defined as:

$$w_{ij}^{\alpha}(t) = r_{ij}(t) + 1 \tag{2}$$

However, the trust on an agent is not an absolute value, and has to be compared with the *risk perception* (i.e., being in a risky situation) in the blockchain. Let $R_i(t)$ denote the risk perception of a node/agent i at a certain time step t, which represents the perceived probability of a risky situation of the agent i. In our case, the risk perception of an agent represents either the risk of accepting invalid blocks in the case of Lazy Miner dilemma, or being a victim of a double spending attack in the case of Impatient Seller dilemma.

In order to combine risk perception with uncertainty, we assume that each individual i decides about a node/agent j according with its previous knowledge of $r_{ij}(t)$. Thus, $r_{ij}(t) < R_i(t)$ implies that from the perspective of the agent i, the reliability of the agent j is relatively low (and vice versa), and hence it will check against the database (to get to know the truth). Thus, the higher the value of $R_i(t)$, the more the agent i will be suspicious and check against the database. The opposite is true for small values of R_i(t); in other words, $r_{ij}(t) \geq R_i(t)$ implies the agent i will trust its neighbor j.

Only after checking against the database, the agent i will know the truth about its neighbor j. This information is used to increase or decrease both the reliability $r_{ij}(t)$ of its neighbor and the risk perception measure $R_i(t)$. In particular, if the check is positive, $r_{ij}(t)$ will be increased by a given amount v_r; otherwise it will be decreased by the same quantity. Similarly, the risk perception $R_i(t)$ is increased or decreased by q_R based on the result of the check.

In the Lazy Miner dilemma, an agent receives a block from its connecting agents, verifies it and then forwards the block to its neighbors. Let us assume for simplicity that this occurs in a synchronous way and at discrete time steps t. Nevertheless, as explained in Section 3.1, the block received might be not valid. Thus, any farther block mined on top of it would not be valid and rejected by other compliant miners as well as the first one. Thus, if the miner decides to start mining on the next block without checking its validity, it would be mining on top of invalid blocks. Alternatively, an agent may decide to verify the block against a central database, which means checking the validity of blocks before mining the next block. However, as explained in section 3.1, this operation is costly, at least in terms of time.

Similarly, the Impatient Seller dilemma poses an analogous challenging issue for agents of the blockchain network. The agent, a seller in this case, may decide to trust its neighboring agent, thus behaving as an impatient seller and finalizing the transaction without waiting for any mining. This allows the agent to avoid waiting for the mining times, but the risk is that of becoming victim of a double spending attack without getting any credit from that transaction. On the other hand, it may decide to wait for mining times, i.e., checking against a database. As explained in section 3.2, this case also implies paying a cost, at least in terms of time.

To deal with these two conflict situations or dilemmas, similar to [9], we propose a multi-agent heuristic-based game theoretic model aimed at balancing between the costs (time) and the risk of accepting invalid blocks (Lazy Miner dilemma) or being victim of a double spending attack (Impatient Seller dilemma).

In our model, other than considering weights of the indirect interactions between agents/nodes at each layer, we define a measure of *homophily* h_{ij}. Specifically, homophily is defined as an interaction-based measure, depending on the frequency of interactions between agents on the Transaction layer; while on the Trust layer, it is defined as a similarity-based measure, which entails the different dimensions of homophily, to derive an Euclidean distance δ_{ij} between the agents. Overall, we define homophily as follows:

Homophily, denoted as h_{ij}, is a measure of similarity between two nodes/agents i and j such that:

$$h_{ij} = \frac{1}{1 + \delta_{ij}} \tag{3}$$

where δ_{ij} is the homophily difference (distance) between i and j.

4.2 Evolutionary Dynamics

To explore and quantify the evolutionary dynamics of behaviors in the multi-layer network, we adopt the Prisoner's Dilemma (PD), a pairwise social dilemma where the nodes can select one of two strategies - cooperation or defection. The symmetric game is defined in function of its payoff matrix, described in Table 1. Players will receive both a *reward* R for mutual cooperation or a *punishment* P for mutual defection. A

	C	**D**
C	R	S
D	T	P

Table 1: **Prisoner's dilemma - payoff matrix**, where R, S, T and P correspond to the Reward, Sucker, Temptation and Punishment payoffs respectively.

defector will get the *temptation* payoff T when playing against a cooperator, while the cooperator obtains the so-called *sucker* payoff S. In the case of Prisoner's dilemma, the payoffs are ordered as $T > R > P > S$, meaning that the defection is the best strategy regardless of the opponent's decision [19, 26, 25]. Overall, social dilemmas, such as Prisoner's dilemma, describe all those conflict situations where the strategy with the highest individual fitness does not represent the most convenient strategy in terms of social community [19, 26, 25]. Indeed, cooperation results in a benefit to the opposing player, but incurs a cost to the cooperator. On the other hand, defection provides a benefit to the individual but incurs a cost to the community. In both cases, it is best to defect for rational and selfish individuals in a single round of the game, regardless of the opponent strategy. However, mutual cooperation leads to a higher payoff for the community than mutual defection, but cooperation is irrational if we consider only the individual player's point of view.

In the two considered dilemmas, cooperation means behaving altruistically and doing a database (dB) check, so that an agent decides to pay a cost in terms of time for the community represented by the other agents in the blockchain. Specifically, in the Lazy Miner dilemma, checking against the dB means checking the

validity of blocks and avoiding the risk of accepting invalid blocks. Whereas, in the Impatient Seller dilemma, checking against the dB means waiting for the recommended number of confirmations in order to avoid being a victim of a double spending attack.

On the other hand, defection corresponds to the selfish behavior. In the case of Lazy Miner dilemma, it corresponds to start mining on the next block without checking the validity of blocks. In the case of Impatient Seller dilemma, defections mean to finalize the transaction without waiting for any mining.

In both dilemmas, acting selfishly and behaving as a lazy miner or an impatient seller is more convenient for single agents. Nevertheless, reasoning in terms of community, the players (agents) would benefit more from mutual cooperation, yielding a total benefit (individual and community benefits) higher than that of mutual defection. The social dilemma is therefore established. Although cooperation should not naturally emerge under these conditions, in nature and real-world networks we observe how the cooperation does exist.

One of the main targets of this work is to explore the emergent dynamics and better understand what underlying mechanisms in terms of human-related factors, such as homophily, along with the role of weights and heuristics may lead to a resolution of blockchain conflicts. To explore and quantify the evolutionary dynamics of behaviors on the blockchain weighted multi-layer network, we take into account the iterated forms of the Prisoner's Dilemma, where at each round of the game, agents can change their strategies or behaviors, based on the imitation dynamics of the fittest strategies [19, 26, 25]. We quantify the evolutionary process in accordance with the standard Monte Carlo simulation procedure, composed of elementary steps [19, 26, 25], so that at each round a player i changes its strategy S_i and adopts the strategy S_j from player j with a probability determined by the Fermi function, defined as follows [19]:

$$W(S_j \to S_i) = (\omega^{\alpha\beta}) \cdot \frac{1}{1 + exp[(U_i - U_j)/(\delta_{ij} \cdot K)]} \tag{4}$$

Therefore, a player i on the blockchain weighted multi-layer network adopts strategy S_j of another player j as a function of the difference $U_i - U_j$ between the utility functions of the two nodes (as defined below), according to δ_{ij} and the coupling factor $\omega^{\alpha\beta}$ characterising the multi-layer network. Specifically, δ_{ij} is the homophily difference between two agents i and j. If the value of homophily is high, player i is more likely to imitate the strategy of j at each round. Let K be the selection intensity quantifying the uncertainty in the strategy adoption process. Defined as in [19], the homophily can vary in the range $[0, 1]$. The selected value of K is a traditional and frequently employed choice that does not qualitatively affect the evolutionary outcomes, as shown in the preceding works and reviews [27]. $\omega^{\alpha\beta}$ defines a measure of interdependence (i.e., interplay) between the two layers of the multi-layer network [19] and it helps investigate the evolution of behaviors on the multi-layer structure.

The strategy update of each player at each round of the game on the multi-layer network strongly depends on the difference between the utility functions of the two agents. The utility function of an agent i at time step t is defined as:

$$U_i(t) = \sum_{j \in \Theta} w_{ij}(t) P_{ij}(t) \tag{5}$$

where Θ is the set of neighbors j of the agent i. Also, $w_{ij}(t)$ is the link weight, and $P_{ij}(t)$ is the payoff of agent i after having played with neighbor j at time step t. Therefore, by definition, the accumulated utility function for each agent combines the link weight and the aforementioned payoff. Let us observe how we have introduced an utility function, since both payoffs and link weights dynamically change over time in accordance with the interactions on the multi-layer network. Thus, at the beginning ($t = 0$), we assume that the the reliability r_{ij} follows a power-law distribution $P(r_{ij}) * r_{ij}^{-\gamma}$, with the exponent γ assuming values in the range $2 \leq \gamma \leq 3$. This corresponds to the realistic assumption that in the blockchain network, the agents are more inclined to interact with the few hubs in the network, i.e., those having higher reputation.

The reliability values are updated at each time step t of the evolutionary dynamics after checking against the database, if the agent i decides to cooperate (C). In this case, the reliability is updated following the strategy adopted in the previous round by agent j:

$$r_{ij}(t+1) = \begin{cases} r_{ij}(t) + v_r, & if S_j = C \\ r_{ij}(t) - v_r, & if S_j = D \end{cases} \tag{6}$$

Similarly, the risk perception values are updated as follows:

$$R_i(t+1) = \begin{cases} R_i(t) + q_r, & if S_j = C \\ R_i(t) - q_r, & if S_j = D \end{cases} \tag{7}$$

Otherwise, if the agent i decides to defect (D) without any check against dB, and it blindly trusts in its neighbor j, the reliability values will be increased or decreased by a quantity $v'_r < v_r$, as a function of the strategy adopted in the previous round by the agent j. Thus, we have:

$$r_{ij}(t+1) = \begin{cases} r_{ij}(t) + v'_r, & if S_j = C \\ r_{ij}(t) - v'_r, & if S_j = D \end{cases} \tag{8}$$

Similarly, the risk perception values will be increased or decreased by a quantity $q'_r < q_r$ according to the strategy adopted in the previous round by the agent j, as follows:

$$R_i(t+1) = \begin{cases} R_i(t) + q'_r, & if S_j = C \\ R_i(t) - q'_r, & if S_j = D \end{cases} \tag{9}$$

The reason behind the choice of values $v'_r < v_r$ and $q'(r) < q(r)$ is that the agent has not actually checked against dB, thus representing an unconditional trust towards the neighboring agent but without any verification.

As explained before, at each time step t of the evolutionary process, the link weights at the Trust layer change according to the reliability (see Eq. 2), and thus the utility function impacting on the evolutionary dynamics of agents' behaviors (see Eq. 5).

4.3 Oblivion Control Mechanism

The heuristics based quantities, r_{ij} and R_i, change smoothly over the rounds of the evolutionary dynamics. As in [9], our model also includes an *oblivion mechanism*, which means that the agents have finite memory and they "forget" their history. This mechanism acts on the heuristic estimators r_{ij} and R_i, and it is implemented with the parameters ξ_r and ξ_R, respectively. Thus, after a certain number of time steps, the information stored τ time steps before the present time has a weight $(1-\xi)^\tau$ and new information is stored with weight o. Consequently, the reliability and risk perception will be updated after τ, as follows:

$$r_{ij}(\tau) = r_{ij}(1-\xi_r)^\tau \tag{10}$$

$$R_i(\tau) = R_i(1-\xi_R)^\tau \tag{11}$$

The oblivion mechanism represents a perturbation to the system. This helps the system move back to its original state, thus "resetting" the evolutionary process, after which the agents re-start the dynamic process they were previously engaged in. Based on the values of ξ_r, the oblivion mechanism may dramatically change the subsequent evolutionary dynamics, pushing the systems towards a totally different equilibrium state. It would be interesting to see how different values of ξ_r and ξ_R and τ impact on the size of the considered temporal windows between these mechanisms.

5 Putting it Together

This section aims to provide a real-world estimation of the payoff matrix in the two conflict scenarios. Since quantifying the human quality factors (benefits and costs) is challenging, through heuristics we incorporate these factors, along with data mining from blockchain, into the modeling approach. Thus, the modelis able to provide statistical parameters for the game-theoretic payoff matrix and the heuristics-based factors impacting on the strategy update process in the evolutionary dynamics.

5.1 Lazy Miner Dilemma Payoffs

As mentioned, a Lazy Miner's strategy is to save time by avoiding the block content validity check, which has a variable complexity block by block. Testing the validity of the block content means executing all the transactions inside a block, against the state of the previous block, maintained in a node's local database. The complexity of executing a single transaction varies between transactions and different blockchain protocols. For example, in Bitcoin, executing a transaction means running its own encapsulated script.

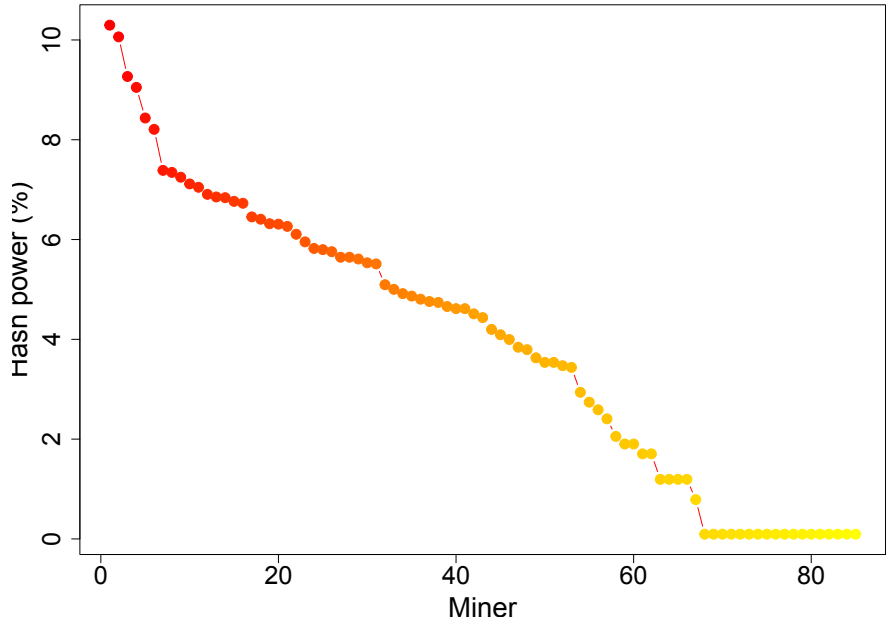

Figure 4: **Hash power (%) of miners**

Most transactions adhere to a few standard script templates that mainly require to compute and verify cryptographic hashes or public key signatures (for details and ratios of script types on the Bitcoin blockchain data, see [28]), which are relatively cheap. Nevertheless, non-standard transactions present in blocks still need to be accepted, as such the complexity can be much higher. Moreover the transaction execution cost

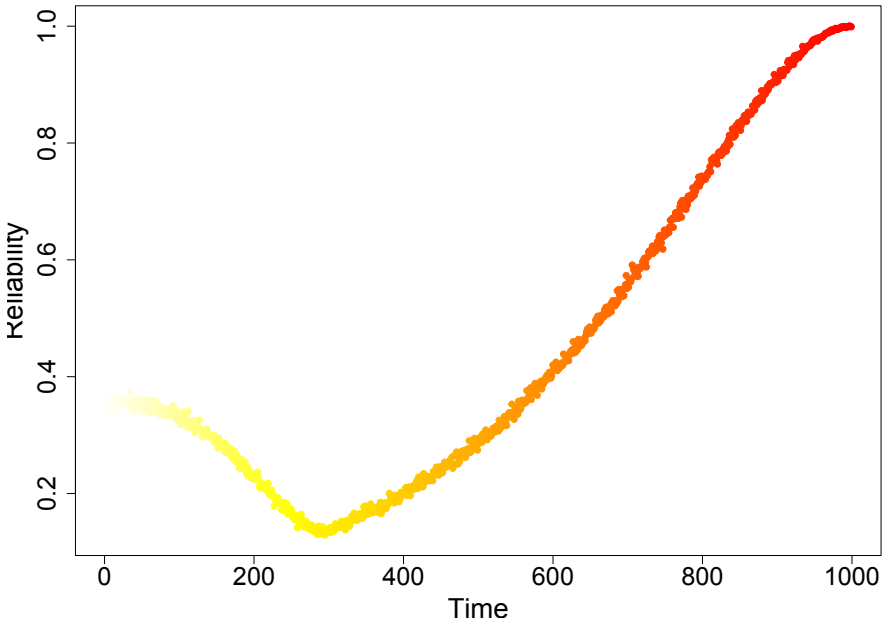

Figure 5: **Reliability over time - low oblivion factor**

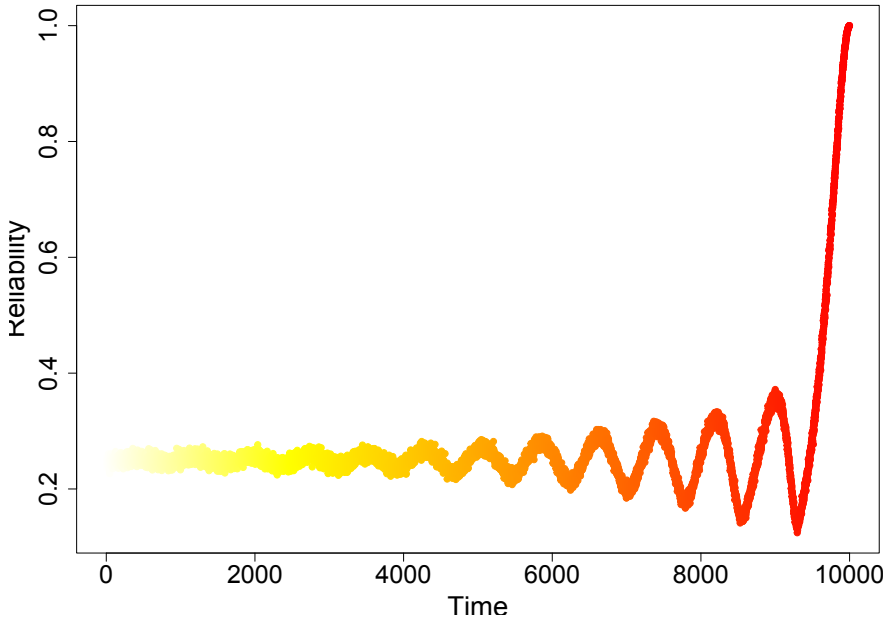

Figure 6: **Reliability over time - medium oblivion factor**

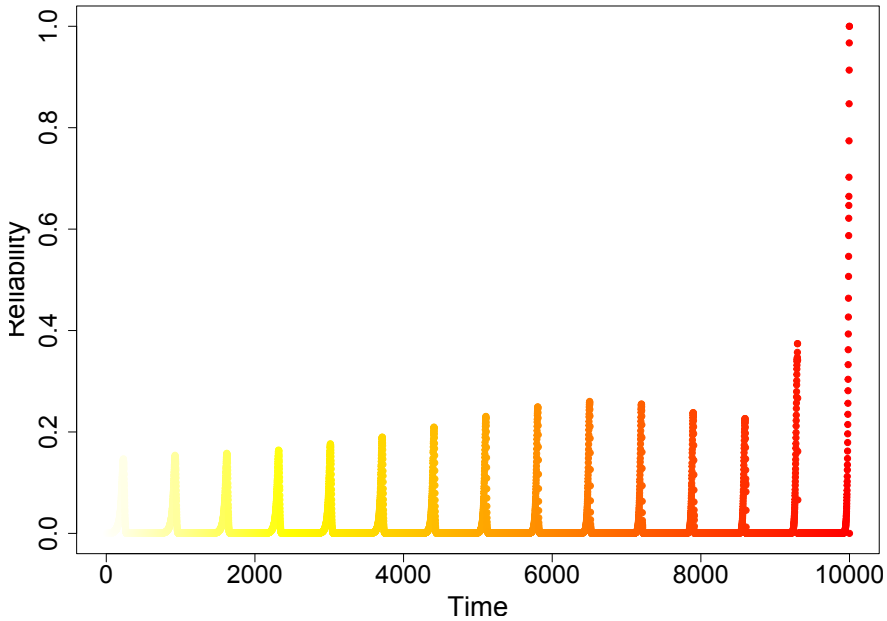

Figure 7: **Reliability over time - high oblivion factor**

can be higher in Smart contract supporting blockchains (e.g. Ethereum), that often support Turing complete languages.

To estimate the cost of block integrity check (see Fig. 1) for the Lazy Miner, we consider the case of the Ethereum protocol. In Ethereum transactions complexity is measured based on the concept of *gas*, the unity of computational cost of the Ethereum Virtual Machine operations [29]. First, we follow the official average benchmark by the Ethereum foundation that takes into account one gas as taking 1 microsec computation time to be completed on average. We can then estimate the cost of each block content integrity check as the average gas usage per block in the recent past (e.g. during October 2019 it has been approximately 6.9 million gas [30]). Combining the two values we obtain that verifying a block would approximately cost 6.9 seconds. This value is unrealistically high, since a lot of the gas costs are referred to storage operations and are meant for miners to compensate for storage space and not time. From [31] we can see how the official estimation is wildly inaccurate for the different instructions executed, and can be overestimated as much as 90% on average on commercial hardware. As such, we can consider a safe rough average time estimation to check a block of 0.7 seconds.

A new block is found, on average, every 14.5 seconds and any miner has a chance to find a new block proportional to the miner relative hash power. A distribution of miners by hash rate during the last blocks can be seen in [32, 30]. A clear power-law distribution (log-scale) is evident (see Fig. 4), with few very big miners and many smaller ones. Reasoning in game-theoretic terms, the Reward payoff corresponds to the case where both agents earn a payoff which is equal to the product of the average time to find a new block and the relative hash power. Moreover, there is a community gain deriving from an overall increase in the value of trust due to the creation of valid blocks. Conversely, the Punishment payoff is different from the Reward payoff because of a lower community gain. In fact, if both miners decide to defect, they will not experience any relative advantage, but they might create an invalid block incurring in a community loss. In the two mixed cases, Sucker and Temptation payoffs, the defecting agents gain a 0.7 seconds advantage over the cooperative ones.

In Fig. 4, during our observation period (October 2019), there were $N = 85$ active miners, corresponding to agents in our model. In order to avoid Distributed Denial of Service attacks (DDoS), miners have to keep the information about connections secret. It is therefore challenging to derive the number of neighboring agents for each agent. However, since the miners are incentivized to be connected to as many neighbors as possible, we consider as a starting value the officially recommended maximum number of connections for a node, i.e., $K = 50$ as estimated average value. Then, we vary this connectivity value in a broad range to see how it impacts on the dynamics of reliability.

We have simulated the Lazy miner dilemma changing the number of neighbors K and all the heuristics parameters introduced in section 4 and we have quantified the distribution of reliability values over the time. As showed in Figures 5, 6 and 7, we can observe how the reliability shows different fluctuating behavior, depending on statistical heuristic parameters given as an input in the model. Specifically, plots highlight the sensitivity of the human heuristics dynamics in function of the various statistical parameters defined in the model: number of nodes, connectivity (i.e. number of neighbours), quantities related to the update of reliability and risk perception values (see section 4.1), and those related to oblivion mechanism (see section 4.3). The latter acts on the reputation of each agent. Indeed, a low oblivion factor (ξ_r) (see Fig. 5) will make the reputation more resilient, so the reliability increases almost linearly over time (see Fig. 5). Vice versa, higher values of the oblivion factor (ξ_r) (see Figures 6 and 7) will make the dynamics very fast and fluctuating, mainly due to the fact that the reliability values change more quicker than their adaptation or updating over the time. Overall, human heuristics allows us to model the human quality factors, and the different observed behaviors demonstrate how we can obtain asymptotic trends or oscillations.

5.2 Impatient Seller Dilemma Payoffs

In the Impatient Seller scenario, we can estimate the number of neighbors considering the real Bitcoin economy as in [33]. Specifically, the distributions of in-degree and out-degrees correspond to the number of received and emitted payments by an user. These distributions follow a power-law behavior in function of time. The number of users, that in our case correspond to nodes/agents in the blockchain multi-layer network, can be estimated by considering the 'active' nodes, defined as in [33], that are about 5.5 millions.

Also in this case, as seen in Fig. 4, we can observe a power-law distribution in terms of connections, with a few highly connected hubs, representing those nodes/users managing the vast majority of transacted value. While the other users are mostly isolated since they take part in very few payments. Similarly as for the Lazy

Miner (see Figures 5, 6 and 7), the values of the payoff matrix can be estimated by exploiting the human heuristics and varying its statistical parameters.

6 Conclusions

In a blockchain, the agents' behaviors play a crucial role in their operational reliability. Thus, it is important to address conflict situations, which are complex and based on multi-agent behaviors. In this work, we have modeled two conflict scenarios and proposed a novel methodology based on game-theoretic modeling and multi-layer networks. Furthermore, we have included human-related factors, such as homophily, impacting on the agents' decisions. We have, also, defined link weights between agents based on human heuristics, modeling how they change according to the experienced payoffs. Some of the main advantages of using game theory and multi-layer networks in a blockchain scenario include the better capability to model complex conflict events under special boundary conditions, and the huge potential to model and manage security against attacks.

The proposed modeling approach belongs to the class of plug and play models [34]. To the best of our knowledge, this is the sole model of that class beyond epidemiological models.

Overall, the multi-layer approach provides a way to obtain a cross-modality integration between human quality factors (such as trust or reliability) and human actions (blockchain). It allows us to integrate the commercial actions on blockchain and these human quality factors. Through the heuristic-based approach, we have incorporated these factors, along with data mining from blockchain, into the modeling approach, which is then able to provide statistical parameters for the game-theoretic modeling, such as payoff matrix values and the quantification of the evolutionary dynamics.

Thus, heuristics allows us to connect multi-layer and game theory in the addressed blockchain conflict scenarios, realising the integration of trust and actions on blockchain within the payoff matrix, which allows us to interpret the decision-making processes. Results in terms of the dynamics of the reliability values shed light on the sensitivity of the human heuristics dynamics.

We envisage that the game-theoretic approach combined with heuristics will be extremely useful in validating the efficacy of the presented modeling approach for the two blockchain conflict scenarios, by considering real data sets. Moreover, it will allow us to better understand from a methodological point of view which is the impact of heuristics and homophily and weights' dynamics on the evolutionary outcomes of the two proposed scenarios.

Acknowledgments: The work of S. K. Das is supported by NSF grants CNS-1818942, CCF-1725755, and DGE-1433659. He is also an International Visiting Professor and SERB-sponsored VAJRA faculty at IIT Kharagpur, India; and Satish Dhawan Visiting Chair Professor at IISc Bangalore.

References

[1] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang. Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375, 2018.

[2] F. Casino, T. K. Dasaklis, and C. Patsakis. A systematic literature review of blockchain-based applications: current status, classification and open issues. *Telematics and Informatics*, 2018.

[3] Tomaso Aste, Paolo Tasca, and Tiziana Di Matteo. Blockchain technologies: The foreseeable impact on society and industry. *Computer*, 50(9):18–28, 2017.

[4] A. Tapscott and D. Tapscott. How blockchain is changing finance. *Harvard Business Review*, 1(9):2–5, 2017.

[5] G. Drosatos and E. Kaldoudi. Blockchain applications in the biomedical domain: a scoping review. *Computational and structural biotechnology journal*, 17:229–240, 2019.

[6] Minhaj A. Khan and K. Salah. Iot security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems*, 82:395–411, 2018.

[7] D. D. F. Maesa, P. Mori, and L. Ricci. A blockchain based approach for the definition of auditable access control systems. *Computers & Security*, 84:93–119, 2019.

[8] Z. Liu, N. C. Luo, W. Wang, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim. A survey on blockchain: a game theoretical perspective. *IEEE Access*, 7:47615–47643, 2019.

[9] F. Bagnoli, A. Guazzini, and P. Liò. Human heuristics for autonomous agents. In *Workshop on Bio-Inspired Design of Networks*, pages 340–351. Springer, 2007.

[10] Gerd Gigerenzer and Peter M Todd. *Simple heuristics that make us smart*. Oxford University Press, USA, 1999.

[11] The Economist, The trust machine. https://www.economist.com/ news/leaders/21677198-technology-behind-bitcoin-could-transform-how-economy-works-trust-machine. Accessed 30 October 2019, 2019.

[12] J. A. Kroll, I. C. Davey, and E. W. Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *Proceedings of WEIS*, volume 2013, pages 1–11. Georgetown University, 2013.

[13] I. Eyal. The miner's dilemma. In *2015 IEEE Symposium on Security and Privacy*, pages 89–103. IEEE, 2015.

[14] O. Schrijvers, J. Bonneau, D. Boneh, and T. Roughgarden. Incentive compatibility of bitcoin mining pool reward functions. In *International Conference on Financial Cryptography and Data Security*, pages 477–498. Springer, 2016.

[15] I. Eyal and E. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *International conference on financial cryptography and data security*, pages 436–454. Springer, 2014.

[16] N. T. Courtois and L. Bahack. On subversive miner strategies and block withholding attack in bitcoin digital currency. *arXiv preprint arXiv:1402.1718*, 2014.

[17] G. Karame, E. Androulaki, and S. Capkun. Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. *IACR Cryptology ePrint Archive*, 2012(248):1–17, 2012.

[18] A. Asgaonkar and B. Krishnamachari. Solving the buyer and seller's dilemma: A dual-deposit escrow smart contract for provably cheat-proof delivery and payment for a digital good without a trusted mediator. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 262–267. IEEE, 2019.

[19] A. Di Stefano, M. Scatà, A. La Corte, P. Liò, E. Catania, E. Guardo, and S. Pagano. Quantifying the role of homophily in human cooperation using multiplex evolutionary game theory. *PloS one*, 10(10), 2015.

[20] S. Boccaletti, G. Bianconi, R. Criado, Del Genio C. I., J. Gómez-Gardeñes, M. Romance, I. Sendiña-Nadal, Z. Wang, and M. Zanin. The structure and dynamics of multilayer networks. *Physics Reports*, 544(1):1–122, 2014.

[21] M. Scatà, A. Di Stefano, A. La Corte, and P. Liò. Quantifying the propagation of distress and mental disorders in social networks. *Scientific Reports*, 8(1):5005, 2018.

[22] F. Battiston, V. Nicosia, and V. Latora. The new challenges of multiplex networks: Measures and models. *The European Physical Journal Special Topics*, 226(3):401–416, 2017.

[23] A. Di Stefano, M. Scatà, A. La Corte, S. K. Das, and P. Liò. Improving qoe in multi-layer social sensing: A cognitive architecture and game theoretic model. In *Proceedings of the Fourth International Workshop on Social Sensing*, pages 18–23. ACM, 2019.

[24] V. Nicosia, S. P. Skardal, A. Arenas, and V. Latora. Collective phenomena emerging from the interactions between dynamical processes in multiplex networks. *Physical Review Letters*, 118(13):138302, 2017.

[25] A. Di Stefano, M. Scatà, S. Vijayakumar, C. Angione, A. La Corte, and P. Liò. Social dynamics modeling of chrono-nutrition. *PLoS computational biology*, 15(1):e1006714, 2019.

[26] M. Scatà, A. Di Stefano, A. La Corte, P. Liò, E. Catania, E. Guardo, and S. Pagano. Combining evolutionary game theory and network theory to analyze human cooperation patterns. *Chaos, Solitons & Fractals*, 91:17–24, 2016.

[27] G. Szabó and G. Fath. Evolutionary games on graphs. *Physics reports*, 446(4-6):97–216, 2007.

[28] D. D. F. Maesa, A. Marino, and L. Ricci. Data-driven analysis of bitcoin properties: exploiting the users graph. *International Journal of Data Science and Analytics*, 6(1):63–80, 2018.

[29] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151, 2014.

[30] Etherchain. https://www.etherchain.org/, Accessed 30 October 2019, 2019.

[31] R. Yang, T. Murray, P. Rimba, and U. Parampalli. Empirically analyzing ethereum's gas mechanism. *arXiv preprint arXiv:1905.00553*, pages 1–10, 2019.

[32] Etherscan. https://etherscan.io/. Accessed 1 October 2019, 2019.

[33] Damiano Di Francesco Maesa, Andrea Marino, and Laura Ricci. Data-driven analysis of bitcoin properties: exploiting the users graph. *International Journal of Data Science and Analytics*, pages 1–18, 2017.

[34] D. He, E. L. Ionides, and A. A. King. Plug-and-play inference for disease dynamics: measles in large and small populations as a case study. *Journal of the Royal Society Interface*, 7(43):271–283, 2009.