The Stability and the Security of the Tangle

Quentin Bramas
ICUBE, University of Strasbourg, CNRS, France
bramas@unistra.fr

Abstract

In this paper we study the stability and the security of the distributed data structure at the base of the IOTA protocol, called the Tangle. The contribution of this paper is twofold. First, we present a simple model to analyze the Tangle and give the first discrete time formal analyzes of the average number of unconfirmed transactions and the average confirmation time of a transaction.

Then, we define the notion of *assiduous honest majority* that captures the fact that the honest nodes have more hashing power than the adversarial nodes *and* that all this hashing power is constantly used to create transactions. This notion is important because we prove that it is a necessary assumption to protect the Tangle against double-spending attacks, and this is true for any tip selection algorithm (which is a fundamental building block of the protocol) that verifies some reasonable assumptions. In particular, the same is true with the Markov Chain Monte Carlo selection tip algorithm currently used in the IOTA protocol.

Our work shows that either all the honest nodes must constantly use all their hashing power to validate the main chain (similarly to the Bitcoin protocol) or some kind of authority must be provided to avoid this kind of attack (like in the current version of the IOTA where a coordinator is used).

The work presented here constitute a theoretical analysis and cannot be used to attack the current IOTA implementation. The goal of this paper is to present a formalization of the protocol and, as a starting point, to prove that some assumptions are necessary in order to defend the system again double-spending attacks. We hope that it will be used to improve the current protocol with a more formal approach.

2012 ACM Subject Classification Networks → Network performance modeling; Networks → Network security

Keywords and phrases Distributed Ledger Technology, Security, Stability

Digital Object Identifier 10.4230/OASIcs.Tokenomics.2019.8

Related Version A previous version of the paper is available at `https://hal.archives-ouvertes.fr/hal-01716111`.

1 Introduction

Since the day Satoshi Nakamoto presented the Bitcoin protocol in 2008 [5], the interest in Blockchain technologies has grown continuously. More generally, this interest concerns *Distributed Ledger Technology*, which refers to a distributed data storage protocol. Usually it involves a number of nodes (or processes, or agents) in a network that are known to each other or not. Those nodes may not trust each-other so the protocol should ensure that they reach a consensus on the order of the operations they perform, in addition to other mechanisms like data replication for instance.

The consensus problem has been studied for a long time [1, 6] providing a number of fundamental results. But the solvability of the problem was usually given in terms of proportion of faulty agents over honest agents. In a trustless network, where anyone can participate, an adversary can simulate an arbitrary number of nodes in the network. To avoid that, proof systems like Proof of Work (PoW) or Proof of Stake (PoS) are used to link the importance of an entity with some external properties (processing power in PoW) or internal properties (the number of owned tokens in PoS) instead of simply the number

© Quentin Bramas;
licensed under Creative Commons License CC-BY
International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2019).
Editors: Vincent Danos, Maurice Herlihy, Maria Potop-Butucaru, Julien Prat, and Sara Tucci-Piergiovanni;
Article No. 8; pp. 8:1–8:15
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of nodes it controls. The consensus problem is now solvable only if the importance of the adversary (given in terms of hashing power or in stake) is smaller than the honest one (the proportion is reduced to 1/3 if the network is asynchronous).

In Bitcoin and in the other blockchain technologies, transactions are stored in a chain of blocks, and the PoW or PoS is used to elect one node that is responsible for writing data in the next block. The "random" selection and the incentive for a node to execute honestly the protocol make the whole system secure, as it was shown by several formal analysis [2, 5]. Usually, there are properties that hold with high probability i.e., with a probability that tends to one quickly as the time increases. For instance, the order between two transactions does not change with probability that tends to 1 exponentially fast over the time in the Bitcoin protocol, if the nodes executing honestly (or rationally) the protocol have more than a third of the hashing total power.

In this paper we study another distributed ledger protocol called *the Tangle*, presented by Serguei Popov [7], that is used in the IOTA cryptocurrency to store transactions. The Tangle is a Directed Acyclic Graph (DAG) where a vertex, representing a transaction, has two parents, representing the transactions it confirms.

According to the protocol a PoW Puzzle must be solved to add a transaction to the Tangle. This PoW prevents an adversary from spamming the network. However, it is not clear in the definition of the Tangle how this PoW impacts its security.

When a new transaction is appended to the Tangle, it references two previous unconfirmed transactions, called tips. The algorithm selecting the two tips is called a *Tip Selection Algorithm* (TSA). It is a fundamental parts of the protocol as it is used by the participants to decide, among two conflicting transactions, which one is valid. It is the most important part in order for the participants to reach a consensus. The TSA currently used in the IOTA implementation uses the PoW contained in each transaction to select the two tips.

Related Work

Very few academic papers exist on this protocol, and there is no previous work that formally analyzes its security. The white paper behind the Tangle [7] presents a quick analysis of the average number of transactions in the continuous time setting. This analysis is done after assuming that the number of tips converge toward a stationary distribution. The same paper presents a TSA using *Monte Carlo Markov Chain* (MCMC) random walks in the DAG from old transactions toward new ones, to select two unconfirmed transactions. The random walk is weighted to favor transactions that are confirmed by more transactions. There is no analysis on how the assigned weight, based on the PoW of each transaction affects the security of the protocol. This MCMC TSA is currently used by the IOTA cryptocurrency.

It is shown in [8] that choosing the default TSA is a Nash equilibrium. Participants are encouraged to use the MCMC TSA, because using another TSA (e.g. a lazy one that confirms only already confirmed transactions) may increase the chances of seeing their transactions unconfirmed.

Finally, the tangle has also been analyzed by simulation [3] using a discrete time model, where transactions are issued every round following a Poisson distribution of parameter λ. Like in the continuous time model, the average number of unconfirmed transactions (called *tips*) seems to grow linearly with the value of λ, but a little bit slower ($\approx 1.26\lambda$ compared to 2λ in the continuous time setting).

Contributions

The contribution of our paper is twofold. First, we analyze formally the number of tips in the discrete time setting, depending on the value of λ by seeing it as a Markov chain where at each round, there is a given probability to obtain a given number of tips. Unlike previous work, we here prove the convergence of the system toward a stationary distribution. This allows us to prove the previous results found by simulations [3] that the average number of tips is stationary and converge towards a fixed value.

Second, we prove that if the TSA depends only on the PoW, then the weight of the honest transactions should exceed the hashing power of the adversary to prevent a double-spending attack. This means that honest nodes should constantly use their hashing power and issue new transactions, otherwise an adversary can attack the protocol even with a small fraction of the total hashing power. Our result is interesting because it is true for any tip selection algorithm i.e., the protocol cannot be more secure by simply using a more complex TSA.

The remaining of the paper is organized as follow. Section 2 presents our model and the Tangle. In Section 3 we analyze the average confirmation time and the average number of unconfirmed transactions. In Section 4 we prove our main theorem by presenting a simple double-spending attack.

2 Model

2.1 The Network

We consider a set $\mathcal{N}$ of processes, called nodes, that are fully connected. Each node can send a message to all the other nodes (the network topology is a complete graph).

We assume nodes are activated synchronously. The time is discrete and at each time instant, called round, a node reads the messages sent by the other nodes in the previous round, executes the protocol and, if needed, broadcast a message to all the other nodes. When a node broadcasts a message, all the other nodes receive it in the next round. Those assumptions are deliberately strong as they make an attack more difficult to perform, which is useful when studying necessary assumptions.

2.2 The DAG

In this paper, we consider a particular kind of distributed ledger called *the Tangle*, which is a Direct Acyclic Graph (DAG). Each node u stores at a given round r a local DAG G_r^u (or simply G_r or G if the node or the round are clear from the context), where each vertex, called *site*, represents a transaction. Each site has two parents (possibly the same) in the DAG. We say a site *directly confirms* its two parents. All sites that are confirmed by the parents of another site are also said to be confirmed (or indirectly confirmed) by it i.e., there is a path from a site to all the sites it confirms in the DAG (see Figure 1). A site that is not yet confirmed is called a *tip*. There is a unique site called *genesis* that does not have parents and is confirmed by all the other sites. For simplicity we identify a DAG simply by a set of sites $G = (s_i)_{i \in I}$

Two sites may be *conflicting*. This definition is application-dependent so we assume that there exists a function $areConflicting(a, b)$ that answer whether two sites are conflicting or not.

If the Tangle is used to store the balance of a given currency (like the IOTA cryptocurrency), then a site represents a transaction moving funds from a sender address to a receiver address and two sites are conflicting if they try to move the same funds to two different receivers i.e., if both executing transactions results in a negative balance for the sender. The

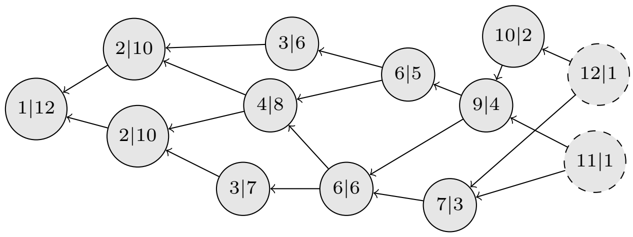

Figure 1 An example of a Tangle where each site has a weight of 1. In each site, the first number is its score and the second is its cumulative weight. The two tips (with dashed border) are not confirmed yet and have cumulative weight of 1.

details of this example are outside the scope of this paper, but we may use this terminology in the remaining of the paper. In this case, signing a transaction means generating a site, and broadcasting a transaction means sending it to the other nodes so that they can include it to their local Tangle.

At each round, each node may sign one or more transactions. For each transaction, the node selects two parents. The signed transaction becomes a site in the DAG. Then, the node broadcasts the site to all the other nodes.

DAG extension

▶ **Definition 1.** *Let G be a DAG and A a set of sites, disjoint from G. If each site of A has its parents in A or in the tips of G, then we say that A is an extension of G and $G \cup A$ denotes the DAG composed by the union of sites from G and A. We also say that A extends G.*

One can observe that if A extends G, then the tips of G form a cut of $G \cup A$.

▶ **Definition 2.** *Let A be a set of sites extending a DAG G. We say A completely extends G (or A is a complete extension of G) if all the tips of $G \cup A$ are in A. In other word, the sites of A confirm all the tips of G.*

▶ **Definition 3.** *A DAG of a node may contain conflicting sites. If so, the DAG is said to be forked (or conflicting). A conflict-free sub-DAG is a sub-DAG that contains no conflicting sites.*

Weight and Hashing Power

When a transaction is signed, a small proof of work (PoW) has to be solved to include it in the DAG. The difficulty of this PoW is called the weight of the site. Initially, this PoW has been added to the protocol to prevent a node from spamming a huge number of transactions. In order to issue a site of weight w a processing power (or *hashing power*) proportional to w needs to be consumed.

With the PoW, spamming requires a large amount of processing power, which increases its cost and reduces its utility. It was shown [7] that sites should have bounded weight and for simplicity, one can assume that the weight of each site is 1.

Then, this notion is also used to compute the *cumulative weight* of a site, which is the amount of work that has been done to deploy this site and all sites that confirm it. Similarly, *the score* of a site is the sum of all weight of sites confirmed by it i.e., the amount of work that has been done to generate the sub-DAG confirmed by it (see Figure 1 for an illustration).

2.3 Tip Selection Algorithm

When signing a transaction s, a node u has to select two parents i.e., two previous sites in its own version of the DAG. According to the protocol, this is done by executing an algorithm called the *tip selection algorithm* (TSA). The protocol says that the choice of the parents must be done among the sites that have not been confirmed yet i.e., among tips. Also, the two selected parents must not confirm, either directly or indirectly, conflicting sites.

We denote by $\mathcal{T}$ the TSA, which can depend on the implementation of the protocol. For simplicity, we assume all the honest nodes use the same algorithm $\mathcal{T}$. As pointed by previous work [7], the TSA is a fundamental factor of the security and the stability of the Tangle.

For our analysis, we assume $\mathcal{T}$ depends only on the topology of the DAG, on the weight of each site in the DAG and on a random source. With those assumptions, we say the TSA is *stateless*. If it also depends on previous output of the TSA it is said to be *stateful*.

The output of $\mathcal{T}$ depends on the current version of the DAG and on a random source (that is assumed distinct for two different nodes). The random source is used to prevent different nodes that has the same view from selecting the same parents when adding a site to the DAG at the same time. However, this is not deterministic and there are not guaranties i.e., it is possible that two distinct nodes issue two sites with the same parents.

Local Main DAG

The local DAG of a node u may contain conflicting sites. For consistency, a node u can keep track of a conflict-free sub-DAG $main_u(G)$ that it considers to be its local *main* DAG. If there are two conflicting sites a and $\bar{a}$ in the DAG G, the local main DAG contains at most one of them.

The main DAG of a node is used as a reference for its own view of the world, for instance to calculate the balance associated with each address. Of course, this view may change over the time. When new transactions are issued, a node can change its main DAG, updating its view accordingly (exactly like in the Bitcoin protocol, when a fork is resolved due to new blocks being mined). When changing its main DAG, a local node may discard a sub-DAG in favor of another sub-DAG. In this case, several sites may be discarded. This is something we want to avoid or at least ensure that the probability for a site to be discarded tends quickly to zero with time.

The tip selection algorithm decides what are the tips to confirm when adding a new site. Implicitly, this means that the TSA decides what sub-DAG is the main DAG. In more detail, the main DAG of a node at round r is the sub-DAG confirmed by the two sites output by the TSA. Thus, a node can run the TSA just to know what its main DAG is, even if no site has to be included to the DAG.

One can observe that, to reach consensus, the TSA should ensure that the main DAG of all the nodes contain a common prefix of increasing size that represents the transactions everyone agree on.

2.4 Adversary Model

Among the nodes, some are honest i.e., they follow the protocol, and some are byzantine and behave arbitrarily. For simplicity, we can assume that only one node is byzantine and we call this node *the adversary*. The adversary is connected to the network and receive all the transactions like any other honest node. He can behave according to the protocol but he can also create (and sign) transactions without broadcasting them, called hidden transaction (or hidden sites).

When an honest node issues a new site, the two sites the $\mathcal{T}$ outputs must be two tips, at least in the local DAG of the node. Thus, one parent p_1 cannot confirm indirectly the other p_2, because in this case the node is aware of p_2 having a child and not being a tip. Also, a node cannot select the same site as parent for two different site, thus the number of honest children cannot exceed the number of nodes in the network. This implies the following property.

▶ **Property 1.** *In a DAG constructed by n honest nodes using a TSA, a site cannot have one parent that confirms the other one. Moreover, the number of children of each site is bounded by the number n of nodes in the network.*

The first property should be preserved by an adversary as it is easy for the honest nodes to check and discard a site that does not verify it. However the adversary can issue multiple sites that directly confirm the same site and the honest nodes have no way to know which sites are honest.

Assiduous Honest Majority Assumption

The cumulative weight and the score can be used by a node to select its main DAG. However, even if it is true that a heavy sub-DAG is harder to generate than a light one, there is no relation yet in the protocol between the weight of sites and the hashing power capacity of honest nodes.

We define the *assiduous honest majority assumption* as the fact that the hashing power of honest nodes is constantly used to generate sites and that it is strictly greater than the hashing power of the adversary. In fact, without this assumption, it is not relevant to look at the hashing power of the honest nodes if they do not constantly use it to generates new sites.

Thus, under this assumption, the cumulative weight of the honest DAG grows according to the hashing power of the honest nodes, and the probability that an adversary generates more sites than the honest nodes in a given period of time tends to 0 as the duration of the period tends to infinity. Conversely, without this assumption, an adversary may be able to generates more sites than the honest nodes, even with less available hashing power.

3 Average Number of Tips and Confirmation Time

In this section we study the average number of tips depending on the rate of arrival of new sites. In this section, like in previous analysis [7], we assume that tip selection algorithm is the simple uniform random tip selection that selects two tips uniformly at random.

We denote by $N(t)$ the number of tips at time t and $\lambda(t)$ the number of sites issued at time t. Like previously, we assume $\lambda(t)$ follows a Poisson distribution of parameter λ. Each new site confirms two tips and we denote by $C(t)$ the number of tips confirmed at time t. We have:

$$N(t) = N(t-1) + \lambda(t) - C(t).$$

We say we are in state $N \leq 1$ if there are N tips at time t. Then, the number of tips at each round is a Markov chain $(N(t))_{t \geq 0}$ with an infinite number of states $[1, \infty)$. To find the probability of transition between two states (given in Lemma 5) we first calculate the probability of transition when the number of new site is known.

▶ **Lemma 4.** *If the number of tips is N and k new sites are issued, then the probability $P_{N \xrightarrow{k} N'}$ of having N' tips in the next round is:*

$$P_{N \xrightarrow{k} N'} = \frac{N!}{N^{2k}(N'-k)!} \left\{ {2k \atop N - N' + k} \right\}$$

where $\left\{ {a \atop b} \right\}$ denotes the Stirling number of the second kind $S(a, b)$.

Proof. If k new sites are issued, then there are up to $2k$ sites that are confirmed. This can be seen as a "balls into bins" problem [4] with $2k$ balls thrown into N bins, and the goal is to see how many bins are not empty i.e. how many unique sites are confirmed.

First, there are N^{2k} possible outcome for this experience so the probability of a particular configuration is $\frac{1}{N^{2k}}$. The number of ways we can obtain exactly $C = N - N' + k$ non empty bins, or confirmed transaction (so that there are exactly N' tips afterward) is the number of ways we can partition a set of $2k$ elements into C parts times the number of ways we can select C bins, in a given order, to receive those C parts (also known as the C-permutations of $2k$).

The first number is called the Stirling number of the second kind and is denoted by $\left\{ {2k \atop N-N'+k} \right\}$. The second number is $\frac{N!}{(N'-k)!}$. ◀

Then, the probability of transition is a direct consequence of the previous lemma

▶ **Lemma 5.** *The probability of transition from N to N' is*

$$P_{N \to N'} = \sum_{k=|N'-N|}^{N'} \mathbb{P}(\Lambda = k) P_{N \xrightarrow{k} N'} = \sum_{k=|N'-N|}^{N'} \frac{N! \lambda^k e^{-\lambda}}{N^{2k}(N'-k)!k!} \left\{ {2k \atop N - N' + k} \right\}.$$

Proof. We just have to observe that the probability of transition from N to N' is null if the number of new sites is smaller than $N - N'$ (because each new site can decrease the number of tips by at most one), smaller than $N' - N$ (because each site can increase the number of tips by at most one), or greater than N' (because each new site is a tip). ◀

▶ **Lemma 6.** *The Markov chain $(N(t))_{t \geq 0}$ has a positive stationary distribution π.*

Proof. First, it is clear that $(N(t))_{t \geq 0}$ is aperiodic and irreducible because for any state $N > 0$, resp. $N > 1$, there is a non-null probability to move to state $N + 1$, resp. to state $N - 1$. Since it is irreducible, we only have to find one state that is positive recurrent (i.e., that the expectation of the hitting time is finite) to prove that there is a unique positive stationary state.

For that we can observe that the probability to transition from state N to $N' > N$ tends to 0 when N tends to infinity. Indeed, for a fixed k, we even have that the probability to decrease the number of tips by k tends to 1:

$$P_{N \xrightarrow{k} N-k} = \frac{N!}{N^{2k}(N-2k)!} = (1 - \frac{1}{N})(1 - \frac{2}{N}) \dots (1 - \frac{2k+1}{N}) \quad (1)$$

$$\lim_{N \to \infty} P_{N \xrightarrow{k} N-k} = 1 \quad (2)$$

so that for any $\varepsilon > 0$ there exists a k_ε such that $\mathbb{P}(\Lambda \geq k_\varepsilon) < \varepsilon/2$ and from (2) an N_ε such that $\forall k < k_\varepsilon$, $P_{N \xrightarrow{k} N-k} < \frac{\varepsilon}{2k_\varepsilon}$. So we obtain:

$$A_{N_\varepsilon} = \sum_{N' > N \geq N_\varepsilon} P_{N \to N'} = \mathbb{P}(N(i+1) > N(i+1) \mid N(i) \geq N) \tag{3}$$

$$< \mathbb{P}(\Lambda \geq k_\varepsilon) + \sum_{k < k_\varepsilon} P_{N \xrightarrow{k} N-k} \tag{4}$$

$$< \varepsilon \tag{5}$$

so that the probability A_N to "move away" from states $[1, N]$ tends to 0 when N tends to infinity. In fact, it is sufficient to observe that there is a number $N_{1/2}$ such that the probability to "move away" from states $[1, N_{1/2}]$ is strictly smaller than 1/2. Indeed, this is a sufficient condition to have a state in $[1, N_{1/2}]$ that is positive recurrent (one can see this by looking at a simple random walk in one dimension with a mirror at 0 and a probability $p < 1/2$ to move away from 0 by one and $(1-p)$ to move closer to 0 by 1). From the irreducibility of $(N(t))_{t \geq 0}$, all the states are positive recurrent and the Markov chain admit a unique stationary distribution π. ◀

The stationary distribution

The stationary distribution π verifies the formula $\pi_N = \sum_{i \geq 1} \pi_i P_{i \to N}$, which we can use to approximate it with arbitrary precision. The stationary distribution, for several values of λ is shown in Figure 2.

When the stationary distribution is known, the average number of tips can be calculated $N_{avg} = \sum_{i>0} i\pi_i$, and with it the average confirmation time *Conf* of a tip is simply given by the fact that, at each round, a proportion λ / N_{avg} of tips are confirmed in average. So $\mathit{Conf} = N_{avg}/\lambda$ rounds are expected before a given tip is confirmed. The value of *Conf* depending on λ is shown in Figure 3.

With this, we show that *Conf* converges toward a constant when λ tends to infinity. In fact, for a large λ, the average confirmation time is approximately 1.26, equivalently, the average number of tips $\mathbb{N}_{avg}$ is 1.26λ. For smaller values of λ, intuitively the time before first confirmation diverges to infinity and N_{avg} converges to 1.

When λ tends to infinity

When λ tends to infinity, we can compute the exact average expected confirmation time. When λ is great enough, one can assume the number of tips is well concentrated around its expectation so that we can do the analysis considering only the expected values. Assume that $\lambda \in \mathbb{N}$ sites are being issued and that there are N tips. Each new site confirms uniformly at random 2 tips. If t_n is the average number of tips confirmed after n sites have been chosen among the N previous tips, one can prove the following recursive formula $t_n = t_{n-1}(1 - 1/N) + 1$. Thus, in average the λ new sites confirms

$$t_{2\lambda} = \sum_{i=0}^{2\lambda - 1} \left(1 - \frac{1}{N}\right)^i = N\left(1 - \left(1 - \frac{1}{N}\right)^{2\lambda}\right).$$

As we said, in average, every round λ tips should be confirmed, hence $t_{2\lambda} = \lambda$. Which gives us asymptotically when λ (and N) tends to infinity

$$\exp\left(-\frac{2\lambda}{N}\right) = 1 - \frac{\lambda}{N}.$$

By using the Lambert function W, the solution to this equation is

$$\frac{N}{\lambda} = \frac{2}{W(-2e^{-2}) + 2} \approx 1.2550009749159754.$$

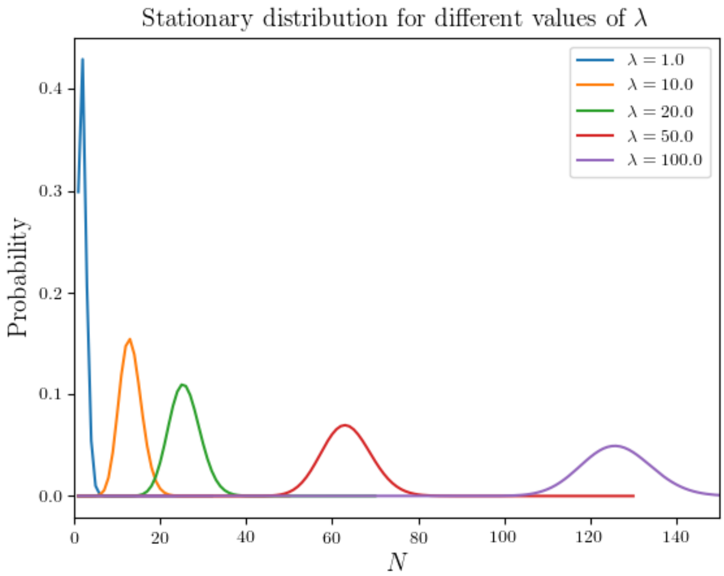

Figure 2 Stationary distribution of the number of tips, for different values of λ. For each value of λ, one can see that the number of tips is really well centered around the average.

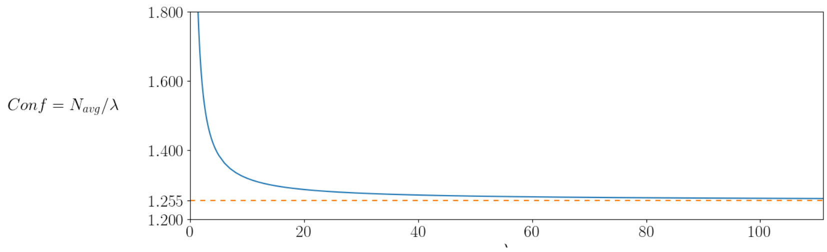

Figure 3 Expected number of round before the first confirmation, depending on the arrival rate of transaction. We see that it tends to 1.26 with λ. Recall that $Conf = N_{avg}/\lambda$ where N_{avg} refers to the average number of tips in the stationary state.

4 A Necessary Condition for the Security of the Tangle

A simple attack in any distributed ledger technology is the double spending attack. The adversary signs and broadcast a transaction to transfer some funds to a seller to buy a good or a service, and when the seller gives the good (he consider that the transaction is finalized), the adversary broadcast a transaction that conflicts the first one and broadcast other new transactions in order to discard the first transaction. When the first transaction is discarded, the seller is not paid anymore and the funds can be reused by the adversary.

The original description of our attack of the Tangle is as follow: after the initial transaction to the seller, the adversary generates the same sites as the honest nodes, forming a sub-DAG with the same topology as the honest sub-DAG (but including the conflicting transaction). Having no way to tell the difference between the honest sub-DAG and the adversarial sub-DAG, the latter will be selected by the honest nodes at some point. This approach may not work with latency in the network, because the sub-DAG of the adversary is always shorter than the honest sub-DAG, which is potentially detected by the honest nodes. To counter this, the adversary can generate a sub-DAG that has not exactly the same topology, but that has the best possible topology for the tip selection algorithm. The adversary can then use all its available hashing power to generate this conflicting sub-DAG that will at some point be selected by the honest nodes.

For this attack we use the fact that a TSA selects two tips that are likely to be selected by the same algorithm thereafter. For simplicity we captured this with a stronger property: the existence of a maximal deterministic TSA.

▶ **Definition 7** (maximal deterministic tip selection algorithm)**.** *A given TSA $\mathcal{T}$ has a maximal deterministic TSA $\mathcal{T}_{det}$ if $\mathcal{T}_{det}$ is a deterministic TSA and for any DAG G, there exists $N_G \in \mathbb{N}$ such that for all $n \in \mathbb{N}$ the following property holds:*

Let A_{det} be the extension of G obtained with $N_G + n$ executions of $\mathcal{T}_{det}$. Let A be an arbitrary extension of G generated with $\mathcal{T}$ of size at most n, conflicting with A_{det}, and let $G' = G \cup A \cup A_{det}$. We have:

$$\mathbb{P}(\mathcal{T}(G') \in A_{det}) \geq 1/2.$$

Intuitively this means that executing the maximal deterministic TSA generates an extension this is more likely to be selected by the honest nodes, provided that it contains more sites than the other extensions. When the assiduous honest majority assumption is not verified, the adversary can use this maximal deterministic TSA at his advantage.

▶ **Theorem 8.** *Without the assiduous honest majority assumption, and if the TSA has a maximal deterministic tip selection, the adversary can discard a transaction that has an arbitrary cumulative weight and achieve double spending.*

Proof. Without the assiduous honest majority assumption, we assume that the adversary can generate strictly more sites than the honest nodes. Let W be an arbitrary weight. One can see W has the necessary cumulative weigh a given site should have in order to be considered final. Let G_0 be the common local main DAG of all node at a given round r_0. At this round our adversary can generate two conflicting sites confirming the same pair of parents. One site a is sent to the honest nodes and the other $\bar{a}$ is kept hidden.

The adversary can use $\mathcal{T}_{det}$ the maximal deterministic TSA of $\mathcal{T}$ to generate continuously (using all its hashing power) sites extending $G \cup \{\bar{a}\}$. While doing so, the honest nodes extend $G \cup \{a\}$ using the standard TSA $\mathcal{T}$. After r_W rounds, it can broadcast all the generated sites to the honest nodes. The adversary can choose r_W so that (i) the probability that it has generated N_G more sites than the honest nodes is sufficiently high, and (ii) transaction a has the target cumulative weight W.

After receiving the adversarial extension, by definition 7, honest nodes will extend the adversarial sub-DAG with probability greater than 1/2. In expectation, half of the honest nodes start to consider the adversarial sub-DAG as their main DAG, thus the honest nodes will naturally converge until they all chose the adversarial sub-DAG as their main DAG, which discard the transaction a.

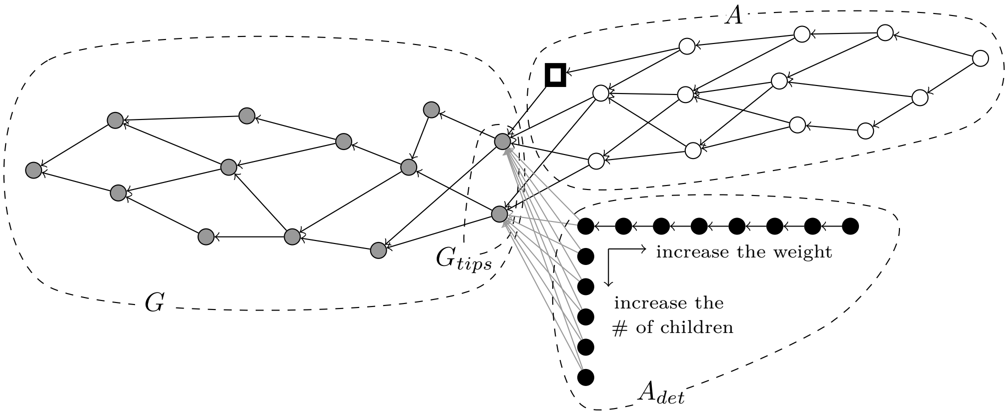

Figure 4 A and A_{det} are two possible extensions of G. The rectangle site conflicts with all site in A_{det} so that when executing the TSA on $G \cup A \cup A_{det}$, tips either from A or from A_{det} are selected. The strategy to construct A_{det} can be either to increase the number of children of G_{tips} or to increase their weight ; both ways are presented here.

If the bandwidth of each channel is limited, then the adversary can start broadcasting the sites of its conflicting sub-DAG at round r_W, at a rate two times greater than the honest nodes. This avoids congestion, and at round $r_W + r_W/2$ all the adversarial sub-DAG is successfully received by the honest nodes. Due to this additional latency, the number of sites in the honest sub-DAG might still be greater than the number of sites in the adversarial sub-DAG, so the adversary continues to generate and to broadcast sites extending its conflicting sub-DAG and at round at most $2r_W$, the adversarial extension of G received by the honest nodes has a higher number of sites than the honest extension.

So the same property is true while avoiding the congestion problem. ◀

Now that we have our main theorem, we show that any TSA defined in previous work (especially in the Tangle white paper [7]) has a corresponding maximal deterministic TSA. To do so we can see that to increase the probability for the adversarial sub-DAG to be selected, the extension of a DAG G obtained by the maximal deterministic TSA should either increase the weight or the number of direct children of the tips of G as shown in Figure 4. We now prove that the three TSA presented in the Tangle white paper [7], (i) the random tip selection, (ii) the MCMC algorithm and (iii) the Logarithmic MCMC algorithm, all have a maximal deterministic TSA, which implies that the assiduous honest majority assumption is necessary when using them (recall that we do not study the sufficiency of this assumption).

4.1 The Uniform Random Tip Selection Algorithm

The uniform random tip selection algorithm is the simplest to implement and the easiest to attack. Since it chooses the two tips uniformly at random, an attacker just have to generate more tips than the honest nodes in order to increase the probability to have one of its tips selected.

▶ **Lemma 9.** *The Random TSA has a maximal deterministic TSA.*

Proof. For a given DAG G the maximal deterministic $\mathcal{T}_{det}$ always chooses as parents one of the l tips of G. So that, after $n + l$ newly added sites A_{det}, the tips of $G \cup A_{det}$ are exactly A_{det} and no other extension of G of size n can produce more than $n + l$ tips so that the probability that the random TSA select a tip from A_{det} is at least 1/2. ◀

▶ **Corollary 10.** *Without the assiduous honest majority assumption, the Tangle with the Random TSA is susceptible to double-spending attack.*

4.2 The MCMC Algorithm

The MCMC algorithm is more complex than the random TSA. It starts by initially putting a fixed number of walkers on the local DAG. Each walker performs a random walk towards the tips of the DAG with a probabilistic transition function that depends on the cumulative weight of the site it is located to and its children. In more details, a walker at a site v has a probability $p_{v,u}$ to move to a child u with

$$p_{v,u} = \frac{\exp(-\alpha(w(v) - w(u)))}{\sum_{c \in C_v} \exp(-\alpha(w(v) - w(c))} \tag{6}$$

where the set C_v is the children of v, and $\alpha > 0$ is a parameter of the algorithm.

The question to answer in order to find the maximal deterministic TSA of MCMC algorithm is: what is the best way to extend a site v to maximize the probability that the MCMC walker chooses our sites instead of another site. The following Lemma shows that the number of children is an important factor. This number depends on the value α. Indeed the following lemma states that if a site v has constant number C_α of children of weight n, then an MCMC walker at v has a probability at least $1/2$ to move to one of those children, even if we add n other sites to the tangle.

▶ **Lemma 11.** *There exists a constant C_α such that, if an MCMC walker is at an arbitrary site v that has C_α children of weight n, then, when extending v with an arbitrary set of sites H of size n, the probability that the walker moves to H is at most* $1/2$.

Proof. When extending v with n sites, one can choose the number h of direct children, and then how the other sites extends those children. There are several ways to extends those children which changes their weights $w_1, w_2, \ldots, w_h$. The probability p_H for a MCMC walker to move to H is calculated in the following way:

$$\begin{aligned}
S_H &= \sum_1^h exp(-\alpha(W - w_i)) \\
\overline{S_H} &= C_\alpha exp(-\alpha(W - n)) \\
S &= S_H + \overline{S_H} \\
p_H &= S_H / S.
\end{aligned}$$

The greater the weight the greater the probability p_H. Adding more children, might reduce their weights (since H contains only n sites). For a given number of children h, there are several way to extends those children, but we can arrange them so that each weight is at least $n - h + 1$ by forming a chain of length $n - h$ and by connecting the children to the chain with a perfect binary tree. The height l_i of a children i gives it more weight. So that we have $w_i = n - h + l_i$. A property of a perfect binary tree is that $\sum_1^h 2_i^l = 1$. We will show there is a constant C_α such that for any h and any $l_1, \ldots, l_h$, with $\sum_1^h 2_i^l = 1$, we have

$$\begin{aligned}\overline{S_H} &\geq S_H \\ C_\alpha \exp(-\alpha(W-n)) &\geq \sum_{i=1}^{h} \exp(-\alpha(W-w_i)) \\ C_\alpha &\geq \sum_{i=1}^{h} \exp(-\alpha(h-l_i)). \end{aligned} \tag{7}$$

Surprisingly, one can observe that our inequality does not depend on n, so that the same is true when we arrange the sites when extending a site v in order to increase the probability for the walker to select our sites.

Let $f_h : (l_1, \ldots, l_h) \mapsto e^{-\alpha h} \sum_1^h \exp(\alpha l_i)$. So the goal is to find an upper bound for the function f_h that depends only on α.

The function f_h is convex (as a sum of convex functions), so the maximum is on the boundary of the domain, which is either

$$(l_1, \ldots, l_h) = (1, 2, \ldots, h)$$

or

$$(l_1, \ldots, l_h) = (\lceil \log(h) \rceil, \ldots, \lceil \log(h) \rceil, \lfloor \log(h) \rfloor, \ldots, \lfloor \log(h) \rfloor).$$

For simplicity, let assume that h is a power of two so that the second case is just $\forall i, l_i = \log(h)$.

In the first case we have

$$f_h(1, \ldots, h) = \exp(-\alpha h) \frac{\exp(\alpha(h+1)) - \exp(-\alpha)}{\exp(-\alpha) - 1} = \frac{\exp(\alpha) - \exp(-\alpha(h+1))}{\exp(-\alpha) - 1}$$

which tends to 0 when h tends infinity, so it admits a maximum C_α^1.

In the second case, we have

$$f_h(1, \ldots, h) = \exp(-\alpha h) h \exp(\alpha \log(h))$$

which again tends to 0 when h tends infinity, so it admits a maximum C_α^2 By choosing $C_\alpha = \max(C_\alpha^1, C_\alpha^2)$ we have the inequality (7) for any value of h. ◀

▶ **Lemma 12.** *The MCMC tip selection has a maximal deterministic TSA.*

Proof. Let G be a conflict-free DAG with tips G_{tips}.

Let T be the number of tips times the number C_α defined in Lemma 11. The T first executions of $\mathcal{T}_{det}$ select a site from G_{tips} (as both parents) until each site from G_{tips} has exactly C_α children.

The next executions of $\mathcal{T}_{det}$ selects two arbitrary tips (different if possible). After T executions, only one tip remains and the newly added sites form a chain.

Let $N_G = 2T$. N_G is a constant that depends only on α and on G. After $N_G + n$ added sites, each site in G_{tips} has a C_α children with weight at least n. Thus, by Lemma 11, a MCMC walker located at a site $v \in G_{tips}$ moves to our extension with probability at least $1/2$. Since this is true for all sites in G_{tips} and G_{tips} is a cut. All MCMC walker will end up in A_{det} with probability at least $1/2$. ◀

One can argue that this is not optimal and we could have improved the construction of the extension to reduce the number of sites, but we are mostly interested here in the existence of such a construction. Indeed, in practice, the probability for a walker to move to our extension would be higher as the honest sub-DAG A is not arbitrary, but generated with the TSA. Our analysis shows that even in the worst configuration, the adversary can still generate an extension with a good probability of being selected.

▶ **Corollary 13.** *Without the assiduous honest majority assumption, the Tangle with the MCMC TSA is susceptible to double-spending attack.*

4.3 The Logarithmic MCMC Algorithm

In the Tangle white paper, it is suggested that the MCMC probabilistic transition function can be defined with the function $h \mapsto h^{-\alpha} = \exp(-\alpha \ln(h))$. In more details, a walker at a site v has a probability $p_{v,u}$ to move to a child u with

$$p_{v,u} = \frac{(w(v) - w(u))^{-\alpha}}{\sum_{c \in C_v} (w(v) - w(c))^{-\alpha}} \tag{8}$$

where the set C_v is the children of v, and $\alpha > 0$ is a parameter of the algorithm. The IOTA implementation currently uses this function with $\alpha = 3$.

With this transition function, the number of children is more important than their weight.

▶ **Lemma 14.** *The logarithmic MCMC tip selection has a maximal deterministic TSA.*

Proof. Let G be a conflict-free DAG with tips G_{tips} and T be the number of tips. We construct $\mathcal{T}_{det}$ in the following way. $\mathcal{T}_{det}$ always selects two sites from G_{tips} in a round-robin manner. After kT executions ($k \in \mathbb{N}$), each site from G_{tips} has $2k$ children.

Let n be the number of sites generated with $\mathcal{T}_{det}$. Let $v \in G_{tips}$. We have that the number of children of v generated by $\mathcal{T}_{det}$ is $C_{det} = 2n/T$ (for simplicity we assume that T divides $2n$). Let A be an arbitrary extension of G of size n and C_v be the number of children of v that are in A.

With $w(v)$ the weight of v and $u = \text{argmax}_{x \in A} w(x)$ the child in A with maximum weight, we have that $w(v) \leq 2n$ and $C_{det} \leq w(v) - w(u)$. Let p be the probability that a walker located at v chooses a site generated by $\mathcal{T}_{det}$. We have

$$p \geq \frac{C_{det}(w(v) - 1)^{-\alpha}}{C_v(w(v) - w(u))^{-\alpha} + C_{det}(w(v) - 1)^{-\alpha}} = \frac{1}{\frac{C_v(w(v) - w(u))^{-\alpha}}{C_{det}(w(v) - 1)^{-\alpha}} + 1}.$$

Then

$$\frac{C_v(w(v) - w(u))^{-\alpha}}{C_{det}(w(v) - 1)^{-\alpha}} \leq \frac{C_v(C_{det})^{-\alpha}}{C_{det}(2n)^{-\alpha}} = \frac{C_v T^{\alpha}}{C_{det}} = \frac{C_v}{2nT^{1-\alpha}}.$$

With T a constant and C_v bounded (by Property 1), we have that the last fraction tends to 0, and thus p tends to 1, as n tends to infinity. This is true for each site of G_{tips}, so after a given number of generated site N_G, the probability that a LMCMC walker located at any site of G_{tips} moves to a site generated by by $\mathcal{T}_{det}$ is greater than 1/2. ◀

▶ **Corollary 15.** *Without the assiduous honest majority assumption, the Tangle with the Logarithmic MCMC TSA is susceptible to double-spending attack.*

5 Conclusion

We presented a model to analyze the Tangle and we used it to study the average confirmation time and the average number of unconfirmed transactions over the time.

Then, we defined the notion of assiduous honest majority that captures the fact that the honest nodes have more hashing power than the adversarial nodes *and* that all this hashing power is constantly used to create transactions. We proved that for any tip selection algorithm that has a maximal deterministic tip selection (which is the case for all currently known TSA), the assiduous honest majority assumption is necessary to prevent a double-spending attack on the Tangle.

Our analysis shows that honest nodes cannot stay at rest, and should be continuously signing transactions (even empty ones) to increase the weight of their local main sub-DAG. If not, their available hashing power cannot be used to measure the security of the protocol, like we see for the Bitcoin protocol. Indeed, having a huge number of honest nodes with a very large amount of hashing power cannot prevent an adversary from attacking the Tangle if the honest nodes are not using this hashing power. This conclusion may seem intuitive, but the fact that it is true for all tip selection algorithms (that have a deterministic maximal TSA) is something new that has not been proved before.

References

1 Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985.

2 Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 281–310. Springer, 2015.

3 B. Kuśmierz. The first glance at the simulation of the Tangle: discrete model, 2016. URL: `http://iota.org/simulation_tangle-preview.pdf`.

4 Michael Mitzenmacher and Eli Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge university press, 2005.

5 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL: `https://bitcoin.org/bitcoin.pdf`.

6 M. Pease, R. Shostak, and L. Lamport. Reaching Agreement in the Presence of Faults. *J. ACM*, 27(2):228–234, April 1980. `doi:10.1145/322186.322188`.

7 S Popov. The tangle. White paper, 2016. URL: `https://iota.org/IOTA_Whitepaper.pdf`.

8 Serguei Popov, Olivia Saa, and Paulo Finardi. Equilibria in the Tangle. *arXiv preprint*, 2017. `arXiv:1712.05385`.