IOTA Maths & Simulations

Philip Staupe

Abstract

This document summaries the mathematical framework behind the Blockchain-free cryptocurrency IOTA, which is based on a DAG (directed acyclic graph). In addition to that, simulation techniques and results are provided.

Last Updated: 26 Nov 2017

Table of Contents

1 Introduction

1.1 Pros and Cons

IOTA, as introduced in [Pop17b], is very different to other cryptocurrencies for the following reasons:

- Unlike Bitcoin [Nak08] or Ethereum [But13], IOTA is not not based on blockchain technology, but uses a DAG (directed acyclic graph) to store transactions. Simplistically, one can think of the DAG as a network of many (much smaller) concurrent blockchains. This allows for much faster confirmation times, because one is one is not bound by a single chain anymore.
- Another consequence of the DAG is that the network scales a lot more efficiently, meaning that more transactions just lead to more concurrent "little blockchains", but they do not increase confirmation times, which is a known issue with blockchains as they get progressively slower the more transactions they have to accommodate.
- Again, unlike most other cryptocurrencies, IOTA offers zero transaction fees. This is crucial in the world of micro-transactions and Machine-to-Machine economy, because any penny (or micro penny) you send via the IOTA network will be received without any fees deducted.
- There are no miners, because every user is its own independent (mini) miner. This also explains why there are no transaction fees. The "transaction fee" you pay is effectively the electricity cost you incur when issuing your own transactions.
- IOTA is quantum-secure. Traditional cryptocurrencies base their security on elliptic-curve cryptography. Classical examples of such algorithms are the RSA Cryptosystem, Diffie-Hellman Key Exchange or Digital Signature Algorithm, all of which are known to be vulnerable to quantum computers. IOTA, on the other hand, uses Winternitz hash-based signatures, which are immune to quantum computers.

But as important as having innovative features is knowing about their potentially downsides (being aware is half the battle):

- Blockchain-based cryptocurrencies have been around for many years and have proven to work. DAG-based cryptocurrencies are still fairly new and have not established themselves yet, which means there is the possibility of unknown flaws/weaknesses. One of the first blockchain-free cryptocurrencies was DAG Coin [Ler15] [RR17], followed by Byteball [Chu15] and more recently the Hasgraph consensus algorithm [Bai16]. So the DAG space is active, but still in its infancy in comparison to Blockchain.
- Using a DAG (versus a single chain of blocks) prevents exact time stamps on each transaction, because transactions cannot be strongly-ordered. This stands in the way of Smart Contracts, which is a major drawback. IOTA has proposed techniques to overcome this issue [Pop17a], but the corresponding development and research is still on-going.
- It can be argued that despite IOTA being a fee-less system, it is not truly fee-less as everyone has to spend computational power (i.e. electricity) on their own transaction's Proof-of-Work.
- The strength of the network is a function of how many people are issuing transactions at any given point in time (not simply how many people are active in the network). So an attacker could attempt to outpower the network during moments of low participation, such off-peak hours or bank holidays.
- Although the network scales very efficiently with increased usage, it requires initial support whilst not being large enough. This is a result of the lack of miners. So the IOTA foundation currently runs what they call the Coordinator, which is a special node that contributes computing power and security measures to the network. It is to be turned off when the network has grown strong enough to sustain itself. Despite good intentions, the fact that the Coordinator is closed-source and there being only one of its kind effectively centralises the entire network, which defeats the purpose of having a de-centralised currency in the first place.

1.2 Basic Framework

When issuing a new transaction, a participant in the IOTA network will choose 2 previously-published (ideally yet unapproved) transactions (also called tips) either at random, using Markov-Chain Monte-Carlo (MCMC) or some other tip-selection algorithm. The user then validates the 2 selected tips, performs a small Proof-of-Work (which serves as spam protection) and ultimately published his or her own transaction to the network. As more users transfer money, this creates a network of linked transactions all referencing each other in the direction of time, leading to a DAG (directed acyclic graph). See Figure 2.

So effectively, there are 3 transaction types:

1. **Approved Transactions**: Those are the ones that have already been published as well as referenced by other transactions, and are thus deemed valid.
2. **Revealed Tips**: Those are transactions that have been revealed to the network, but have not yet been referenced. So they are still unapproved.
3. **Hidden Tips**: Those are the ones that have not yet been revealed to the network, and are still being working on by the user (validation & Proof-of-Work), or are being held up by network latency.

See Figure 1 for a visual aid. In the following we will investigate properties around the long-term stability of the Tangle.

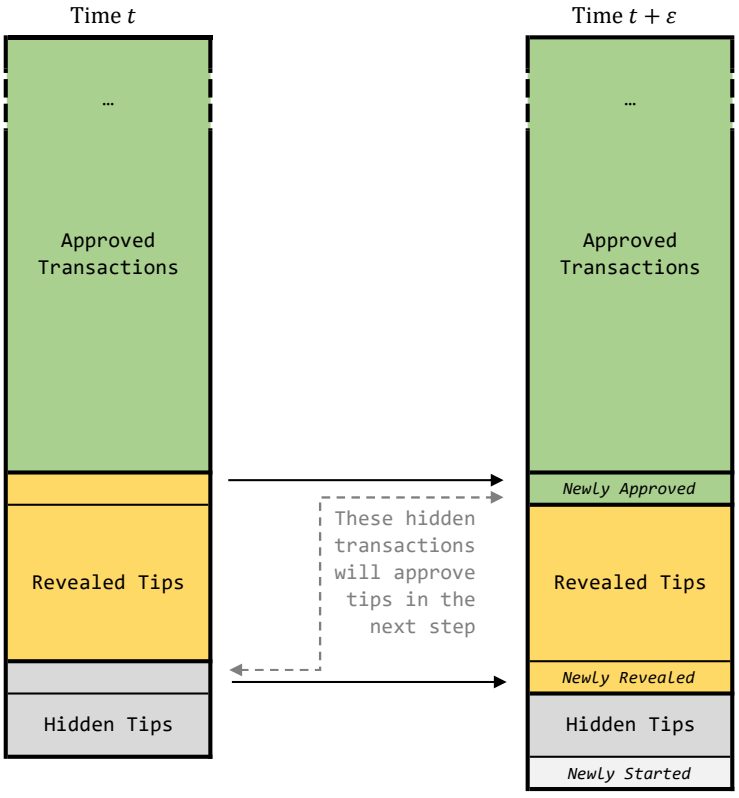

Figure 1: Schema for Tangle dynamics between time t and $t + \varepsilon$

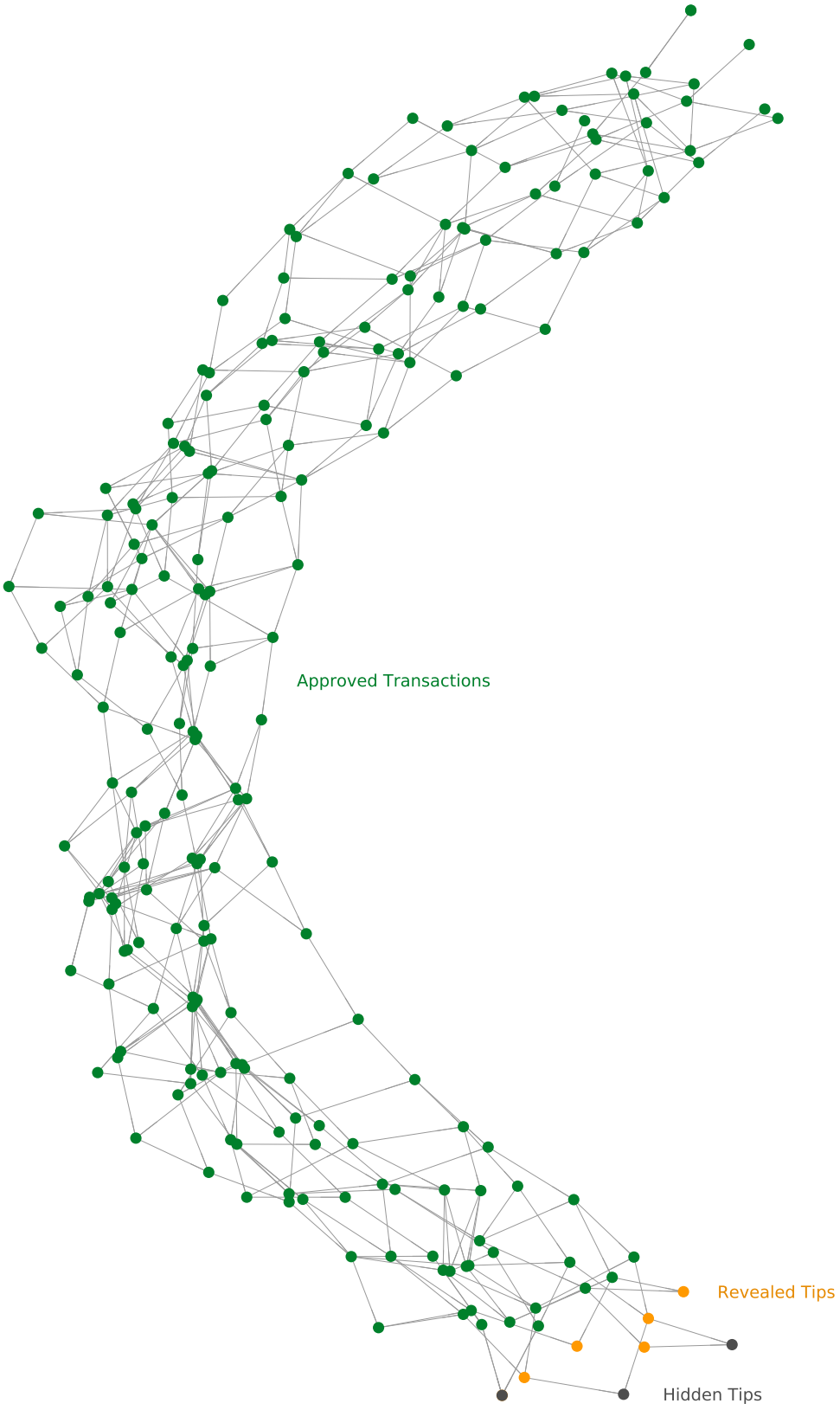

Figure 2: Tangle Structure Example

2 Formal Framework

2.1 Review of Assumptions

In order to analyse stability properties of the Tangle one has to make assumptions as to how the flow of transactions is distributed. Looking at Figures 3 & 4 it is reasonable to assume the occurrence of transactions to be following a Poisson process (in particular because the inter-arrival times closely follow an exponential distribution). So more formally, the total number of transactions $N(\varepsilon)$ issued in any time interval $[t, t+\varepsilon]$ is distributed as follows

$$N(\varepsilon) \sim \text{Poisson}(\lambda \cdot \varepsilon) \quad \text{where} \quad \mathbb{E}[N(\varepsilon)] = \lambda \cdot \varepsilon$$

together with its i.i.d. inter-arrival times τ_i which follow

$$\tau_i \sim \text{Exp}(\lambda) \quad \text{where} \quad \mathbb{E}[\tau_i] = \lambda^{-1}$$

The historic data for October 2017 from Figures 3 & 4 suggests that we currently have

$$\lambda \approx 0.5 \ \frac{\text{transactions}}{\text{second}}$$

i.e. on average a new transaction is issued to the Tangle every 2 seconds. This yields an average of 43 000 transactions per day. From Figure 3 we can see that non-zero transactions make up roughly 10% of all transactions, giving us an average of 4 300 transactions per day that transfer a value greater than zero.

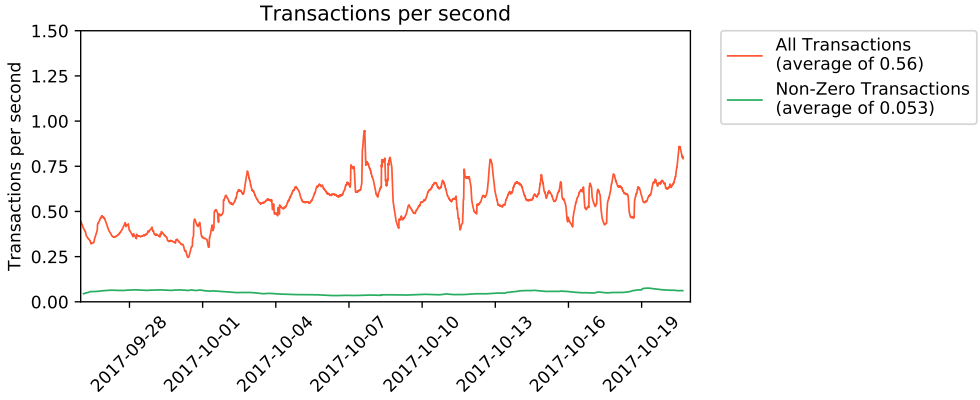

Figure 3: Plot of transactions issued per second for October 2017

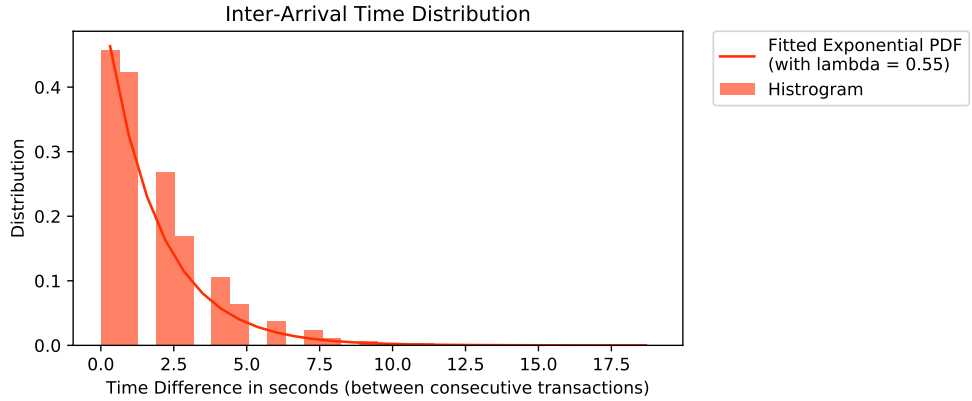

Figure 4: Distribution of inter-arrival times between consecutive transactions for October 2017

2.2 Tip Count Derivation

The IOTA whitepaper [Pop17b] derives a formula for the average number of revealed tips:

$$R = \frac{n}{n-1} \cdot \lambda \cdot h \tag{2.1}$$

where n is the number of transactions referenced by each individual transactions. In IOTA's case we have $n = 2$. In this section we shall derive this formula again. As in the whitepaper, we assume the arrival of new (hidden) transactions follows a Poisson process with magnitude λ, and the time it takes for tips to reveal themselves to the network takes a constant h seconds. Note that different distribution assumptions on h (i.e. anything other than constant) would yield a very different formula to (2.1). In most cases R would still exist, but its analytic solution would be different.

Let H_t and R_t denote the number of hidden and revealed tips at time t respectively. Let us further assume that both stochastic processes are mean-reverting. We say a random process X_t is mean-reverting around some value $X \in \mathbb{R}$ if whenever $X_t > X$ the process will tend to decrease and whenever $X_t < X$ the process will tend to surge. It is further understood that when the process is at its equilibrium $X_t = X$, then there is no drift in either direction. That is, we expect its value to stay at its current level. More formally, this means that

$$\mathbb{E}_t[X_{t+\varepsilon}] = \mathbb{E}[X_{t+\varepsilon} \mid \mathcal{F}_t] = X_t \quad \text{for any} \quad \varepsilon > 0$$

where, loosely speaking, the filtration $\mathcal{F}_t$ of sub-σ-algebras denotes the information available at time t.

We know that the in an interval $[t, t+\varepsilon]$ there will be on average $\mathbb{E}[N(\varepsilon)] = \lambda \cdot \varepsilon$ new (hidden) transactions issued by the users of the network. On top of that, we have

$$\begin{aligned} A_\varepsilon &= \{\text{Some randomly-chosen hidden tip at time } t \text{ is not hidden anymore at time } t+\varepsilon\} \\ \mathbb{P}(A_\varepsilon) &= \begin{cases} \frac{\varepsilon}{h} & , \varepsilon \leq h \\ 1 & , \varepsilon > h \end{cases} \end{aligned}$$

which is due to the constant nature of h. And therefore

$$\mathbb{E}_t[H_{t+\varepsilon}] = H_t = H_t + \lambda \cdot \varepsilon - H_t \cdot \mathbb{P}(A_\varepsilon) \quad \Rightarrow \quad H_t = \frac{\lambda \cdot \varepsilon}{\mathbb{P}(A_\varepsilon)}$$

Simplifying this yields the equilibrium point for hidden tips:

$$H = \lambda \cdot h$$

Now, let $\mathfrak{R}_\mathfrak{t}$ denote the set of revealed tips at some time t when the Tangle is at equilibrium, and let $\mathfrak{R}_{\mathfrak{t}+\varepsilon}$ for $\varepsilon > 0$ denote the very same set of revealed tips but minus the ones that have been approved during $[t, t+\varepsilon]$. So the number of elements in $\mathfrak{R}_\mathfrak{t}$ is decreasing over time, or one might say it is "decaying". Formally, we have

$$|\mathfrak{R}_{\mathfrak{t}+\varepsilon}| = R_t \cdot \mathbb{P}(B_\varepsilon)$$

where

$$\begin{aligned} B_\varepsilon &= \{\text{Some randomly-chosen revealed tip at time } t \text{ has not been approved by time } t+\varepsilon\} \\ \mathbb{P}(B_\varepsilon) &= \begin{cases} \mathbb{P}(B_\varepsilon^0) & , \varepsilon \leq h \\ \mathbb{P}(B_\varepsilon^1) & , \varepsilon > h \end{cases} \end{aligned}$$

$$\begin{aligned} \mathbb{P}(B_\varepsilon^0) &= \frac{R_t - \lambda \cdot \varepsilon}{R_t} \\ \mathbb{P}(B_\varepsilon^1) &= \frac{1}{n} \cdot \exp\left(-\frac{\varepsilon - h}{h} \cdot (n-1)\right) \end{aligned}$$

For B_ε^0 observe that the tip count of $\mathfrak{R}_t$ is linearly decreasing with rate λ, because under the equilibrium assumption we have that $\lambda \cdot \varepsilon$ new (hidden) transactions will be issued, $\lambda \cdot \varepsilon$ will turn from hidden to revealed and, in particular, $\lambda \cdot \varepsilon$ will turn from revealed to approved. This, however, is only true for the first h seconds, because any hidden tip that had picked an element from $\mathfrak{R}_t$ to approve will get revealed in exactly h seconds. So the maximal time when this happens is h later. After that the decay turns exponential.

Now, for the Tangle to be stationary we must have that when a tip gets revealed (and at that moment directly approves its n chosen transactions) only 1 of those n transactions should not have been approved yet, on average. So that the in- and out-flow between hidden, revealed & approved is always expected to be the same. Picking n random items from a bucket of R_t with an individual successes probability of

$$p = \mathbb{P}(B_\varepsilon^0) = \frac{R_t - \lambda \cdot \varepsilon}{R_t}$$

yields a Binomial Distribution $\mathrm{Bin}(n, p)$ with expected value $n \cdot p$. As noted, we must pick on average exactly 1 transaction that has not been approved yet. So

$$n \cdot p = 1 \quad \Rightarrow \quad p = \frac{R_t - \lambda \cdot \varepsilon}{R_t} = \frac{1}{n}$$

Solving this yields the equilibrium point for revealed tips:

$$R = \frac{n}{n-1} \cdot \lambda \cdot h \tag{2.2}$$

We remark that the exact formula of $\mathbb{P}(B_\varepsilon^1)$ was not required to derive the long-term mean of revealed tips. However, it is provided for completeness as it may yield additional insights into the Tangle dynamics and might prove useful for assumptions other than a constant h.

Figure 5 below shows how the cardinality of $|\mathfrak{R}_t|$ behaves for different choices of n over time. The values at $t = 0$ effectively correspond to the long-term mean R from equation (2.2).

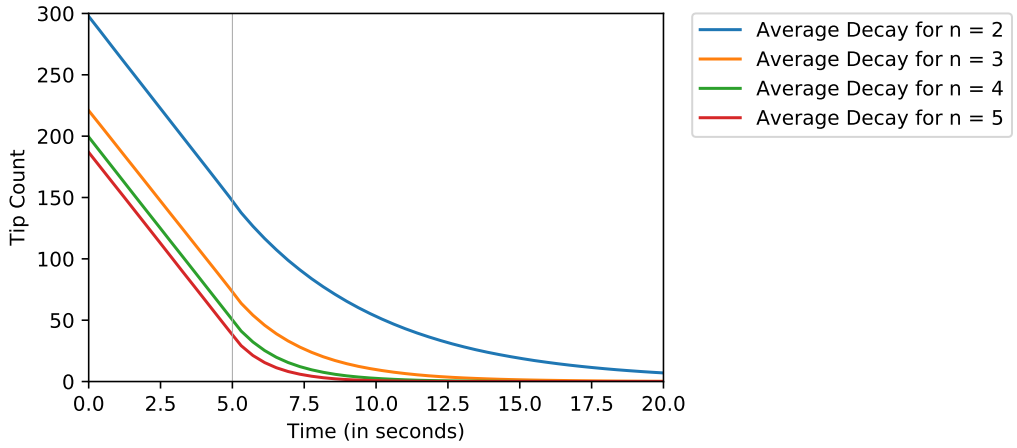

Figure 5: Simulated Decay of Revealed Tip sets using $\lambda = 30$ $\frac{\text{transactions}}{\text{second}}$ and $h \equiv 5$ seconds

3 Simulating the Tangle

3.1 An Efficient Simulation Algorithm

Let us outline a quick way to simulate the Tangle over some closed time-interval $[0, T]$:

1. Generate
   (a) One random number $N(T) \sim \text{Poisson}(\lambda \cdot T)$.
   (b) A total of $N(T)$ inter-arrival times $\tau_1, ..., \tau_{N(T)} \sim \text{Exp}(\lambda)$.
   (c) A total of $N(T)$ issue durations $h_1, ..., h_{N(T)}$ (time it takes for a hidden tx to be revealed). In [Pop17b] these are assumed to be constant, but one may well assume a non-deterministic distribution. For example, one could argue
   $$h_i = c + \tilde{h}_i$$
   where $c \equiv \textit{constant}$ represents deterministic costs such as network latency or validation work, and $\tilde{h}_i \sim \text{Exp}(\beta^{-1})$ which is required for finding the Proof-of-Work Nonce. See Section 3.2 for details as to why an exponential distribution seems adequate.
2. Compute
   (a) Arrival times $a_1, ..., a_{N(T)}$ of each transaction where $a_1 = \tau_1$ and $a_i = a_{i-1} + \tau_i$.
   (b) Reveal times $r_1, ..., r_{N(T)}$ of each transaction where $r_i = a_i + h_i$.
3. Initiate approval times $s_1, ..., s_{N(T)}$ with $s_i = \infty$.
4. Iterate over all transactions in the order of their reveal times r_i (not arrival times). In each iteration:
   (a) Identify set $R_i = \{k \in \{1, ..., N(T)\} \mid r_k < a_i < s_k\}$ of all revealed tips at time a_i.
   (b) Select n random tips $k_{i,1}, ..., k_{i,n}$ from R_i. Typically, we have $n = 2$.
   (c) Set the approval times $s_k := r_i$ for all $k = k_{i,1}, ..., k_{i,n}$.
   (d) Next i.

Note that all of the steps above can be vectorised, thus providing us with a fast and efficient way to simulate the Tangle.

For demonstration purposes one could also identify the set of hidden tips

$$H_i = \{k \in \{1, ..., N(T)\} \mid a_k < a_i < r_k\}$$

at time a_i. The cardinalities $|H_i|$ and $|R_i|$ give us the number of hidden and revealed tips at times a_i respectively. Figure 6 showcases both quantities over time.

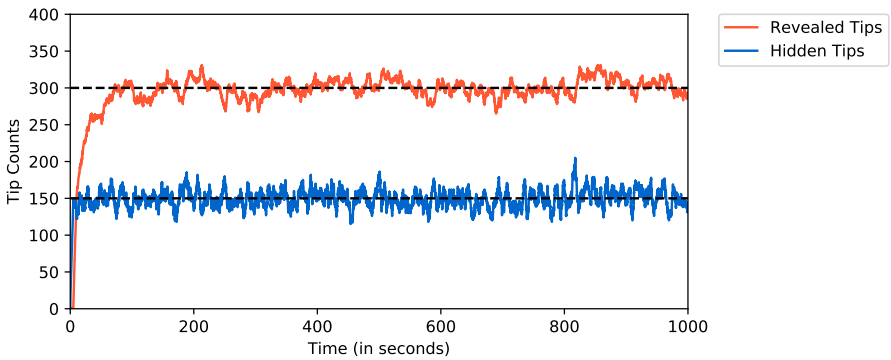

Figure 6: Simulated Tip Count using $\lambda = 30$ $\frac{\text{transactions}}{\text{second}}$, $h \equiv 5$ seconds and $n = 2$

3.2 Nonce Finding Distribution

The number of trials required to find a suitable Proof-of-Work nonce is exponentially distributed. The following Python script demonstrates this.

```
import numpy as np
import hashlib
import matplotlib.pyplot as plt
import scipy.stats

iTrials     = 10000
iDifficulty = 2
sPattern    = '0'*iDifficulty

def RunTrial():
    k = 0
    while True:
        k      = k + 1
        sNonce = str(np.random.randint(100000000))
        sHash  = hashlib.sha256((sNonce).encode()).hexdigest()
        if sHash[0:iDifficulty] == sPattern:
            return k

arr_samples = [RunTrial() for i in range(iTrials)]

plt.hist(arr_samples, 100, normed=1, facecolor='g', alpha=0.7)
lam_fitted = 1.0 / scipy.stats.expon.fit(arr_samples)[1]
x_values   = np.linspace(min(arr_samples),max(arr_samples),100)
y_values   = lam_fitted * np.exp(-lam_fitted * x_values)

plt.plot(x_values, y_values, 'r')
plt.show()
```

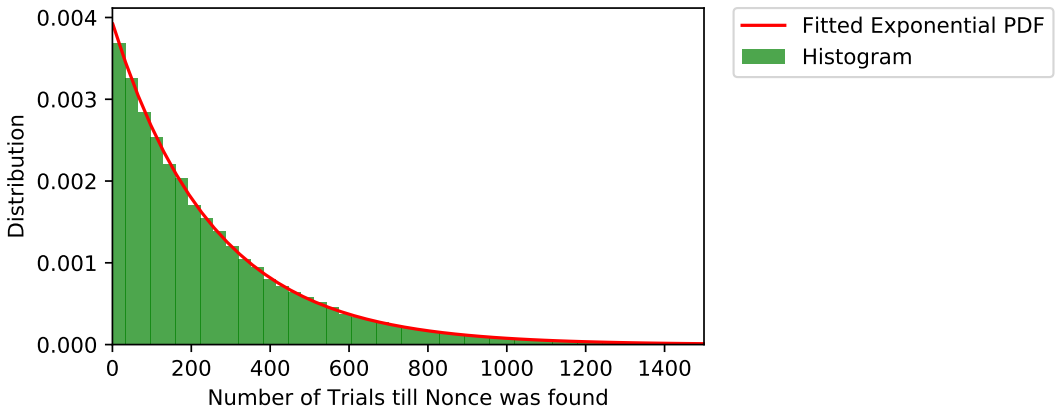

Figure 7: Distribution for finding a Nonce

Bibliography

[Bai16] Leemon Baird. *The Swirlds Hashgraph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance*. `http://www.swirlds.com/downloads/SWIRLDS-TR-2016-01.pdf`, May 2016.

[But13] Vitalik Buterin. *A Next-Generation Smart Contract and Decentralized Application Platform*. `https://github.com/ethereum/wiki/wiki/White-Paper`, late 2013.

[Chu15] Anton Churyumov. *Byteball: A Decentralized System for Storage and Transfer of Value*. `https://byteball.org/Byteball.pdf`, September 2015.

[Ler15] Sergio Lerner. *Dagcoin Draft*. `https://bitslog.files.wordpress.com/2015/09/dagcoin-v41.pdf`, September 2015.

[Nak08] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. `https://bitcoin.org/bitcoin.pdf`, October 2008.

[Pop17a] Serguei Popov. *On the timestamps in the tangle*. `https://iota.org/timestamps.pdf`, August 2017.

[Pop17b] Serguei Popov. *The Tangle*. `https://iota.org/IOTA_Whitepaper.pdf`, October 2017.

[RR17] Yary Ribero and Daniel Raissar. *Dagcoin whitepaper*. `https://dagcoin.org/whitepaper.pdf`, July 2017.