Some Mathematical aspects of DAG-based Distributed Ledger Systems

Olivia Terence Saa

Tese apresentada
ao
Instituto de Matemática e Estatística
da
Universidade de São Paulo
para
obtenção do título
de
Doutor em Ciências

Programa: Matemática Aplicada

Orientador: Prof. Dr. André Salles de Carvalho

Coorientador: Prof. Dr. Serguei Popov

Durante o desenvolvimento deste trabalho a autora recebeu auxílio financeiro da CNPq

São Paulo, abril de 2020

Some Mathematical aspects of DAG-based Distributed Ledger Systems

Esta é a versão original da dissertação/tese elaborada pelo
candidato (Olivia Terence Saa), tal como
submetida à Comissão Julgadora.

Resumo

SAA, O. T. **Aspectos Matemáticos Diversos sobre Sistemas de Ledger Distribuído baseados em DAGs**. 2020. 68 f. Tese (Doutorado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2020.

Na primeira parte do presente trabalho, um sistema de *peering* automático e aleatório é apresentado, modelado e analisado. Este sistema pode ser implementado em qualquer sistema distribuído. Concluímos que ele possui certas propriedades desejáveis (especificamente, um fluxo baixo de mensagens entre os agentes, uma distribuição razoável do número de conexões de cada nó e uma probabilidade desprezável de ser atacado). Na segunda parte do trabalho, apresentamos um artigo publicado no volume 136 do periódico *Computers & Industrial Engineering*, de outubro de 2019 (DOI 10.1016/j.cie.2019.07.025). Neste paper, analisamos os equilíbrios de Nash de um jogo definido de tal maneira a representar as diferentes estratégias que participantes maliciosos podem utilizar para obter certas vantagens em um sistema de Ledger distribuído baseado em DAGs (*Directed Acyclic Graphs*). Provamos a existência de equilíbrios "quase simétricos" para o sistema no qual uma parte dos jogadores usa uma estratégia padronizada e a outra parte tenta otimizar sua estratégia. Também são apresentadas simulações que apontam que os atores "egoístas" não escolherão estratégias excessivamente diferentes das estratégias padrão.

Palavras-chave: Sistemas Distribuídos, DLT, Equilíbrio de Nash, Tangle.

Abstract

SAA, O. T. **Some Mathematical aspects of DAG-based Distributed Ledger Systems**. 2020. 68 f. Tese (Doutorado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2020.

In the first part of this work, we present, model and analyze a randomized automated peering model, that can be implemented to any distributed system. We conclude that the scheme has some desirable properties (specifically, a reasonable message overhead, a reasonable distribution of the numbers of peers of a node, and a negligible probability of an attack by a malicious actor to be successful). In the second part, we present an article published in the volume 136 of the journal *Computers & Industrial Engineering*, in October of 2019 (DOI 10.1016/j.cie.2019.07.025). In the paper, we analyze the Nash Equilibria of a graph attachment game, defined to represent the different strategies that malicious actors can use to take certain advantages in a DAG-based (i.e., based on *Directed Acyclic Graphs*) distributed ledger system. We prove the existence of "almost symmetric" Nash equilibria for the system where a part of players tries to optimize their attachment strategies and another part follows a default one. We also present simulations that show that the "selfish" players will not choose strategies that are considerably different that the "recommended" one.
Keywords: Distributed Systems, DLT, Nash Equilibria, Tangle.

Contents

List of Figures

List of Tables

Organization of the work

This thesis is divided in two parts: "Random Auto Peering" and "Nash Equilibria and Tip Selection Algorithm". Despite being related, the two parts can be read independently.

In the first part, we present, model and analyze an randomized automated peering model, that can be implemented to any distributed system, and not only to DAG-based distributed ledgers.

In the second part, we present an article published in the volume 136 of the journal *Computers & Industrial Engineering*, at October of 2019 (DOI 10.1016/j.cie.2019.07.025). The paper analyzes the Nash Equilibria of a graph attachment game, defined to represent the different strategies that malicious actors can use to take certain advantages in a DAG-based distributed ledger system.

Part I

Part One - Random Auto Peering

Chapter 1

Introduction

1.1 Distributed systems and peer-to-peer networks - from ARPA to DLTs

A *distributed system* can be defined as a set of computers (called *nodes*) working together, independently, towards a common goal. The nodes interact and coordinate their work by messaging each other, without any form of central coordination.

A *peer-to-peer* (P2P) network of a distributed system is formed of a set of equally privileged nodes, which share their resources (as information or processing power) directly to the other participants, without having to share them through servers or another kind of centralized entity [Bar01, Sch01].

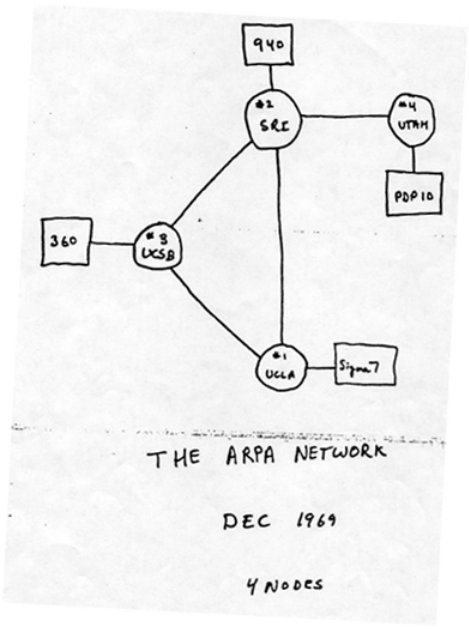

Figure 1.1: *Pencil drawing of the four node Arpanet map, from december 1969. Available at* `https://www.computerhistory.org/collections/catalog/102658020`

P2P networks have been used since the late 60's, with the conception of the ARPANET (an experimental computer network, precursor to the Internet), that originally connected the University of California Los Angeles, the Stanford Research Institute, the University of California Santa Barbara and the University of Utah as equal computing peers. Figure 1.1 is a pencil drawing of the original (1969) ARPANET and Figure 1.2 represents the same network in 1977.

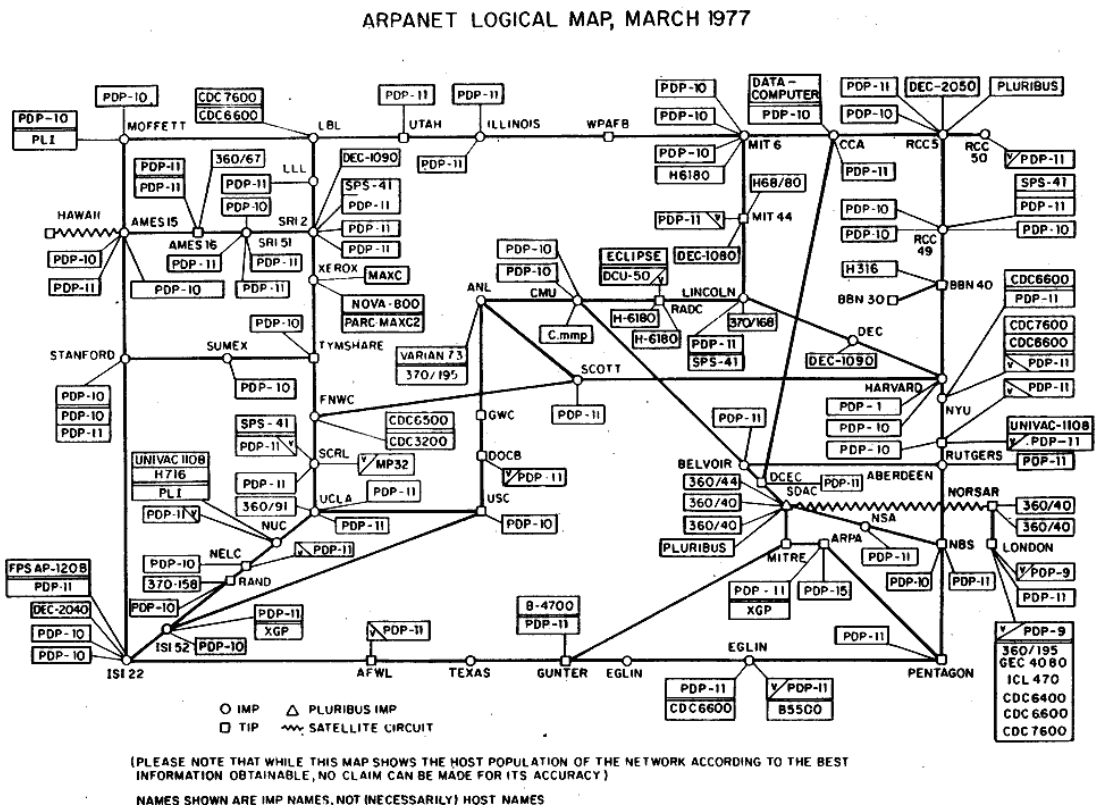

Figure 1.2: *ARPANET logical map, circa 1977. Available (among a collection of other ARPANET logical maps from different years) at* `http://mercury.lcs.mit.edu/~jnc/tech/arpalog.html`

Despite conceptually being more than 50 years old, P2P networks were popularized by the file sharing system Napster -originally released in 1999-, that allowed users to search some of its neighbors' directories for MP3 files and download them directly, without the intervention of a central node. Nevertheless, Napster relied on a centralized indexing server (see Figure 1.3), that was susceptible to being shutdown by regulators. Consequently, Napster ceased its activities in their original form in July 2001.

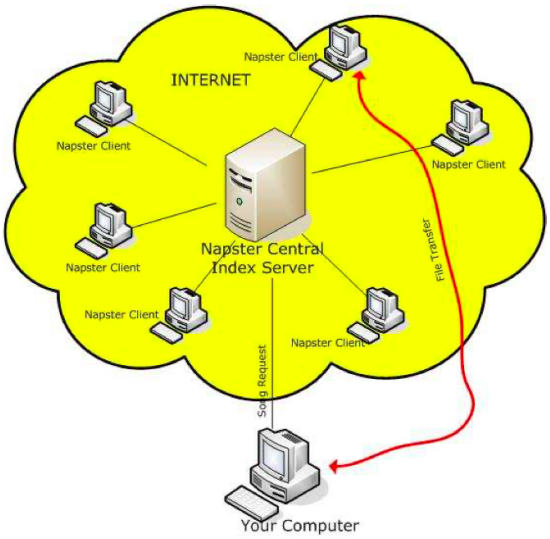

Figure 1.3: *Napster's architecture. Extracted from [Anh08]*

After Napster's shutdown, a new P2P file sharing system called Gnutella became popular; in this system, there is no central database or indexing. Also, there are many different clients available to access this network. These particularities make the Gnutella protocol less vulnerable to being shut down by authorities -or even by attackers- than Napster was.

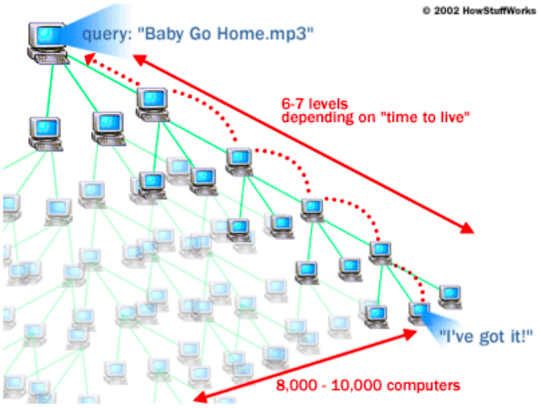

Figure 1.4: *Gnutella's architecture. Extracted from [Anh08]*

Almost one decade later, in 2009, the first functional distributed ledger went live, the Bitcoin Network. Distributed ledgers (or DLTs, from distributed ledger technology) are P2P systems which goal is to store and reach consensus over data, such as financial transactions or any other information that possibly needs to be verified and agreed by the involved parts. The main difference between DLTs and the previous introduced distributed systems is the presence of a *consensus mechanism protocol* besides the decentralized information sharing protocol.

For any of these protocols already introduced, the number of nodes is, in general, relatively large; therefore, in practice, a node cannot share its information directly to all the other nodes. For that reason, in the next section, we introduce the concept of a peering (or connection to peers) algorithm.

1.2 Peering Algorithms

Since we cannot have a complete graph of communication in most of the distributed systems applications, we must introduce the concept of a peering algorithm. When a node first joins the network, it will not have any other node connected to it; in fact, it will not even be aware of the others nodes' existence. In order to establish its first connections, the node will have to rely on a centralized list of nodes, as a bootstrap. Then, it can ask the elements of this list for their known nodes, and so on. This process is what we call a *peer discovering* algorithm.

After the node already discovered a substantial number of possible connections, it will have to choose a small portion of them to be its direct peers. These will be the set of nodes that will directly exchange all kind of information with it. The reason behind the requirement of choosing a small amount of peers from a large list (instead of discovering a small portion of the nodes and connecting to all of them or even sticking with the original bootstrapping list) is, besides the clustering effects of the network around the bootstrap list, the need of an homogeneous flow of information over the graph. Roughly speaking, if a new node connects to others that were introduced to it by a neighbor, the information received by all of them will be very similar. This creates a form of an echo chamber. Moreover, with this kind of network growth, a malicious entity can eclipse a chosen region by just introducing a large amount of his own nodes to the newcomers. We call this second phase of the neighbor choosing process the *peering* algorithm. Furthermore, any peering algorithm where the nodes choose their neighbors without any manual intervention by the node operator is called an auto peering algorithm.

1.3 A Particular Auto Peering Algorithm

Since this work consists in modeling and analyzing only the auto peering algorithm[1], we assume that a suitable peer discovery phase was already performed and finished. Then, each of the nodes $i = 1, \dots, N$ must set:

1. a fixed public *node id* (a 32 bytes string called node_id(i)).
2. a *public salt* (a 20 bytes string called pub_salt(i)).
3. a *private salt* (a 20 bytes string called priv_salt(i)).

The neighbor selection -and, in particular, the decision about which potential neighbors are preferable- is made based on two different distances between nodes, that depend on the node id, private and public salts.

We define the *requesting distance* between node A and B as follows:

$$d_{\text{req}}(A, B) = \text{hash}(\text{node\_id}(A)) \oplus \text{hash}(\text{node\_id}(B) + \text{pub\_salt}(A))$$

where $\oplus$ denotes the *exclusive or* (XOR) operation and any appropriate hash function can be used. To request a new connection, node A must calculate $d_{\text{req}}(A, B)$ for all known nodes and order them by this distance. After that, the node requests connections, from the closest to the farthest, until k of these requests are accepted by the other nodes (and, consequently, the connections are established).

We also define the *accepting distance* between node B and A as

$$d_{\text{acc}}(B, A) = \text{hash}(\text{node\_id}(B)) \oplus \text{hash}(\text{node\_id}(A) + \text{priv\_salt}(B)).$$

A node B will accept a request from a node A whenever:

1. it is currently peered with less than k other nodes or

[1] This algorithm is intended to be used in the IOTA network (see `www.iota.org`)

2. $d_{\text{acc}}(B, A) < d_{\text{acc}}(B, C)$ for some C in the set of B's accepted peers. In this case, node B will drop the connection to its farthest accepted node.

Both private and public salts will be periodically updated; consequently, the distances that depend on them will periodically change. The private salts can be set on each node's volition[2], whereas the public salts must be set using a *hash chain*[3]. When salts are generated with the aid of hash-chains, the moment the first salt is published, the node cannot choose salts at its own volition anymore and has to stick for a long time to the salts generated by the hash chain, since they are publicly verifiable. This way, a node will commit to a certain salt sequence and will not be able to game the requesting process by forcing its requesting distances to always be close to a certain target.

Notice that each node will aim to have $2k$ connections -half of them established by requesting other nodes, and the second half established by accepting other nodes' requests-, but, in general, the nodes will have a random number of connections less or equal than k, implying the non-regularity of the associated graph of connections. Also notice that, since all the public and private salts are independently set, the requesting and accepting distances between a pair of nodes will be uncorrelated. Also, the requesting (or accepting) distances between different pair of nodes are independently and uniformly distributed (since they are obtained from a hash function), implying that every pair of nodes will have the same average probability of being connected over time.

Finally, let us define what does it mean for a malicious node (or set of nodes) to try to attack the system. Since each node aims at $2k$ connections, the probability of a malicious actor successfully peering with a single target will be roughly $2k/N$, where N is the total number of nodes in the network. We say that a node is successfully being attacked if a malicious entity can enforce a probability of peering with it significantly larger than $2k/N$. Observe that the requesting distances depend only on public information. This way, the nodes can verify if a certain request was probably done by following the protocol or not, by evaluating if the distance has an appropriate order of magnitude. Furthermore, if the salts are generated with the aid of hash-chains, the nodes commit to a long sequence of salts, making virtually impossible a successful attack to be sustained for a large amount of time. Now, as the acceptance will depend on a variable only known by the accepting node (the private salt), whether an attacker will or will not establish a connection with a target is completely unpredictable to the attacker itself. So, with this algorithm, network topology does not need to be kept secret to prevent attacks.

Hence, our goal is to analyse if the presented algorithm provides:

1. a negligible probability of an attack by a malicious actor to be successful
2. a reasonable message overhead
3. a reasonable distribution of the numbers of peers of a node

[2]A node can also commit to a sequence of private salts by using a hash chain and revealing the already used private salts. This way, if the node lies about a certain acceptance distance, the other nodes will know that after some time

[3]A Hash Chain is a successive application of a hash function H to a certain secret number S. Then, the sequence will be used **backwards**, meaning that, if $i < j$, the user will always reveal $H^j(S)$ before $H^i(S)$; this way, the next number of the sequence can be only predicted by breaking the hash chain, while the other users can verify if the sequence has been used correctly comparing the hash of the last number to the second to last.

Chapter 2

The Random Auto Peering Model

In this chapter we will mathematically define the random auto peering model used in the current work, and will present two analytical solutions: the first one, a solution to the transient[1] regime of the synchronized expiration problem; the second one, a solution to the equations of the asynchronous problem in its stationary regime (the difference between these two problems will be later well defined).

2.1 Definition of the Mathematical Model

The mathematical model can be informally described as follows: Consider a complete graph K_n with n vertices -which we will denote as nodes- as the network of **possible** connections. Each node's goal is to establish a number k of **realized** connections. The procedure to establish these connections will depend on two different metrics[2] Y(the *requesting metric*) and X(the *acceptance metric*). The algorithm will proceed as follows: Every node i will use an individual random request metric $Y(i,\cdot)$ to determine its order of preferred nodes, and will send requests in this said order. On the other hand, each node j receiving a request from i will accept it either if j has less than k connections or if the acceptance metric $X(j,i)$ is lower than the acceptance metric of at least one of its current accepted nodes. In this case, we drop the connection with the node of largest metric and accept the new request.

Every node will drop all accepted and requested connections periodically and update all their distances, after a fixed amount of time T. We say that a connection has *expired* if it was dropped by this reason. This process can be made asynchronously (meaning that each node will drop its connections at different points in time) or in a synchronized manner (meaning that every node will drop its connections at the same time). If one of your connections was dropped by another node, making your number of connections lower than k, then you will resume the requesting procedure until you obtain k connections again.

In this work, we consider two independent metrics and therefore two independent orderings, one for requests and one for acceptances, both given by i.i.d. uniform random variables in the interval $[0,1]$. Hence, in general, $Y(i,j) \neq Y(j,i)$, $X(i,j) \neq X(j,i)$, and $Y(i,j) \neq X(j,i)$.

Before we formally describe the model, we define the variables $a_i(t)$ and $b_i(t)$), as long as the set $A_i(t)$, as:

1. $a_i(t)$: number of requests that node i has accepted and to which it is still connected. The respective set of such nodes is denoted by $A_i(t)$.

2. $b_i(t)$: number of distinct nodes that requested a connection to i before time t.

[1] Here, we use the word *transient* in the physics sense, i.e., as a synonym of something that has some or all of its properties evolving with time.

[2] The word metric is not used as in the metric spaces theory sense; for instance, we do not claim that the triangle inequality holds.

These variables defined above will also be updated every time each nodes' connections expire.

Definition 2.1.1 (Random Auto Peering Model)**.** *Consider the complete graph K_n (with n vertices), together with the request metric $\{Y(i,j)\}_{(i,j)\in V(K_n)\times V(K_n)}$ and acceptance metric $\{X(i,j)\}_{(i,j)\in V(K_n)\times V(K_n)}$, both given by i.i.d. uniform random variables[3] in the interval $[0,1]$. For every node we have a sequence of non simultaneous incoming requests, forming an Poisson process with rate λ. Since the acceptance metrics are independent of the requesting ones, we only worry here about the accepting process.*

1. *For each node z, on each Poisson event, there is a probability p that one and only one of its connections will vanish[4]. The exact connection that will disappear will be chosen uniformly at random. In that case, the incoming request will be instantly accepted by z, no matter how large the acceptance metric $U[0,1]$ is (since, in this case, node z will have less than k connections).*

2. *Otherwise (i.e., with probability $1-p$), node z will have to compare the acceptance metric of the incoming request to the metric of its current connections. In that case, if node i requested z for a connection at a Poisson event at instant t, then z will accept the connection either if $a_z(t) < k$ or if $a_z(t) = k$ but i satisfies*

$$X(z,i) < \max_{j\in A_z(t)} X(z,j),$$

   In the second case z drops the connection with $\arg\max_{j\in A_z(t)} X(z,j)$ and replace it with i.

3. *z will drop all its connections every T units of time*

In order to calculate all relevant parameters for the practical peering problem, we must evaluate how the probability of a request being accepted evolves with time. In the next two sections we answer to this question, under certain additional simplifications.

2.2 A transient regime solution for the synchronized problem

In this section, we find the distribution of some of the relevant variables of the problem, under the additional hypothesis that all the nodes drop connections at the same time. Without loss of generality, we assume that the last expiration time was at $t = 0$. Since we are interested in the behaviour of the problem in the interval $[0,T)$ -given that after $t = T$ the accepted node will drop connections itself-, we know that none of the accepted connections will expire in that period; in other words, we can assume $p = 0$. For each time t and node z, we define the following parameters:

- $i = b_z(t)$, i.e., the number of requests that node z received in the interval $[0,t)$.
- $(d_j(i))_{j=1,\dots,\min(k,i)}$ is the ordered permutation[5] of the set $\{X(z,j)\}_{j\in A_z(t)}$, meaning that if $j_1 < j_2$, then $d_{j_1}(i) < d_{j_2}(i)$.

This way, for each node, the problem can be summarized as a Poisson process of rate λ of incoming requests, with the following accepting rule: A node z accepts the i^{th} received request, where $i = b_z(t)$, made by m, either if $a_z(t) < k$ or if $a_z(t) = k$ but m satisfies

$$X(z,m) < d_k(i-1);$$

[3] i.e., for each ordered pair of different vertices, we have independent metrics, toting up to $2n^2$ i.i.d variables.

[4] this probability p was added to the model to take into account the possibility of one of the accepted connections being dropped by the requesting node for any reason, as for instance, the expiration of its connections.

[5] the reason behind the dependency on i of these variables is that, the larger is the number of requests that a node has already received, the smaller tend to be the metrics of the accepted connections.

or in other words, if the accepting request of node m as calculated by z is smaller than the largest acceptance metric among z's connections. In the second case z drops the connection with the node associated to $d_k(i-1)$ and replaces it with a connection to m. Notice that since the distributions of the acceptance metric does not actually depend on the requester (neither depends on the requested node), we can identify the acceptance metrics based on their Poisson event instead of based on the associated nodes. For that reason, for each node z, if a request was made in the i^{th} Poisson event by node m, we define $X_i = X(z,m)$. Finally, we assume that z will drop all its connections at time $t = T$.

2.2.1 Probability Distribution of the Acceptance Metrics of the Realised Connections

Theorem 2.2.1. *Let $i = b_z(t)$ and $d_j(i)$ for $j = 1, \ldots, \min(k, i)$ defined as usual. Then, for $i \geq j$:*

$$f_{d_j(i)}(x) = \text{Beta}(j, i-j+1) = \frac{i!}{(i-j)!(j-1)!} x^{j-1} (1-x)^{i-j} \tag{2.1}$$

Also, we have, for $2 \leq j \leq k$ (note that the equation below does not depend on i and therefore is constant over time):

$$f_{d_{j-1}(i)|d_j(i)}(x|y) = \frac{j-1}{y} \left(\frac{x}{y}\right)^{j-2} \tag{2.2}$$

Proof. First notice that the acceptance metric of the realised connections at a certain time do not depend on the order that the requests were made. This means that, for instance, the smallest of the accepted metrics right after the i^{th} Poisson event is the smallest of the metrics of all the requests received until the i^{th} Poisson (this last Poisson event included). In fact, the variable $d_j(i)$ will be the j^{th} smallest of the acceptance metrics among the i first requests. For that reason, in this proof (and only in this proof), we extend the domain of the variable $d_j(i)$ to $j \in \mathbb{N}^+$ and $i \geq j$. Let us now find a useful relation between all the events on this process, noticing the following implications:

1. if $X_i \leq d_{j-1}(i-1)$, then $d_j(i) = d_{j-1}(i-1) \leq d_j(i-1)$
2. if $d_{j-1}(i-1) < X_i \leq d_j(i-1)$, then $d_j(i) = X_i \leq d_j(i-1)$
3. if $d_j(i-1) < X_i$, then $d_j(i) = d_j(i-1)$

One can see that, no matter the value of X_i, if $d_j(i-1) \leq x$, then $d_j(i) \leq x$. Next, if $d_j(i-1) > x$, the only way to have $d_j(i) \leq x$ will be by having $X_i \leq d_{j-1}(i-1) \leq x$ or $d_{j-1}(i) \leq X_i \leq x$; in other words, the only way to have $d_j(i) \leq x$ will be by having $d_{j-1}(i-1) \leq x$ and $X_i \leq x$. Hence, finally, we have the following relation between events:

$$\{d_j(i) \leq x\} = \{d_j(i-1) \leq x\} \cup [\{d_{j-1}(i-1) \leq x < d_j(i-1)\} \cap \{X_i \leq x\}] \tag{2.3}$$

We opt to solve a generalized version of this first part of the Theorem (that will be used later on its generalized form) by considering all random variables $\{X_m\}_{m \in \mathbb{N}^+}$ as uniforms in the interval $[0, d]$ for a fixed $d > 0$ (instead of being uniforms on $[0, 1]$). Consequently, all $d_n(m)$ for $n = 1, \ldots, \min(k, m)$ and $m \in \mathbb{N}^+$ will also be variables in the range $[0, d]$. For the sake of organization, we call these new scaled versions of the variables $\hat{X}_m$ and $\hat{d}_n(m)$. We want to prove that, in this case, the density functions of each $\hat{d}_n(m)$ are given by:

$$f_{\hat{d}_j(i)}(x) = \frac{i!}{(i-j)!(j-1)!} \frac{x^{j-1} (d-x)^{i-j}}{d^i} \tag{2.4}$$

Notice that the relation (2.3) is not affected by this scaling of the variables and it is still valid. Proceeding with the proof, since $\hat{X}_i$ will be given by a uniform random variable in the interval $[0, d]$,

independent of $\hat{d}_{j-1}(i-1)$ and $\hat{d}_j(i-1)$, we have, for $i > j$:

$$P\{\hat{d}_j(i) \le x\} = P\{\hat{d}_j(i-1) \le x\} + \frac{x}{d} P\{\hat{d}_{j-1}(i-1) \le x < \hat{d}_j(i-1)\}$$

Observe that the event $\{\hat{d}_{j-1}(i-1) \le x < \hat{d}_j(i-1)\}$ is equivalent to having exactly $j-1$ out of the $i-1$ i.i.d. uniform random variables $\{\hat{X}_n\}_{n=1,\dots,i-1}$ smaller than x. Then, it follows:

$$\begin{aligned} P\{\hat{d}_j(i) \le x\} &= P\{\hat{d}_j(i-1) \le x\} + \frac{x}{d} P\{\mathrm{Bin}(i-1, x/d) = j-1\} \\ &= P\{\hat{d}_j(i-1) \le x\} + \frac{x}{d} \frac{(i-1)!}{(j-1)!(i-j)!} \frac{x^{j-1}(d-x)^{i-j}}{d^{i-1}} \end{aligned} \tag{2.5}$$

Differentiating both sides and rearranging, we get the following relation:

$$f_{\hat{d}_j(i)}(x) - f_{\hat{d}_j(i-1)}(x) = \frac{(i-1)!}{(j-1)!(i-j)!} \frac{x^{j-1}(d-x)^{i-j-1}(dj-ix)}{d^i} \tag{2.6}$$

Now, let us show by induction that (2.4) is the desired distribution. For $i = j$, it is straightforward[6] to come to the conclusion that $f_{\hat{d}_j(j)}(x) = jx^{j-1}/d^j$, that satisfies (2.4). Now, we use (2.6) for the induction step. Assume (2.4) is valid for some $i \ge j$. Then:

$$\begin{aligned} f_{\hat{d}_j(i+1)}(x) &= f_{\hat{d}_j(i)}(x) + (f_{\hat{d}_j(i+1)}(x) - f_{\hat{d}_j(i)}(x)) \\ &\overset{\text{by}(2.6)}{=} f_{\hat{d}_j(i)}(x) + \frac{i!}{(j-1)!(i+1-j)!} \frac{x^{j-1}(d-x)^{i-j}(dj-(i+1)x)}{d^{i+1}} \\ &\overset{\text{by (2.4)}}{=} \frac{i!}{(i-j)!(j-1)!} \frac{x^{j-1}\,(d-x)^{i-j}}{d^i} \\ &+ \frac{i!}{(j-1)!(i+1-j)!} \frac{x^{j-1}(d-x)^{i-j}(dj-(i+1)x)}{d^{i+1}} \\ &= \frac{(i+1)!}{(j-1)!(i-j+1)!} \frac{x^{j-1}(d-x)^{i-j+1}}{d^{i+1}} \end{aligned}$$

and (2.4) is valid for every $i \ge k$. Applying $d = 1$ to this equation, we have (2.1), that finishes the first part of our proof. To prove (2.2), notice that, for all $x \in [0,1]$, $f_{d_{j-1}(i)|d_j(i)}$ as defined by (2.2) satisfies

$$\begin{aligned} \int_x^1 f_{d_{j-1}(i)|d_j(i)}(x|y) f_{d_j(i)}(y)dy &= \int_x^1 \frac{j-1}{y} \left(\frac{x}{y}\right)^{j-2} \frac{i!}{(i-j)!(j-1)!} y^{j-1}\,(1-y)^{i-j}\,dy \\ &= \frac{i!(j-1)}{(i-j)!(j-1)!} x^{j-2} \int_x^1 (1-y)^{i-j}\,dy \\ &= \frac{i!}{(i-(j-1))!((j-1)-1)!} x^{j-2}(1-x)^{i-j+1} \end{aligned}$$

or, equivalently

$$\begin{aligned} f_{d_{j-1}(i)}(x) &= \int_0^1 f_{d_{j-1}(i),d_j(i)}(x,y)dy \\ &= \int_x^1 f_{d_{j-1}(i)|d_j(i)}(x|y) f_{d_j(i)}(y)dy \end{aligned} \tag{2.7}$$

where $f_{d_{j-1}(i)}$ and $f_{d_j(i)}$ are known. Furthermore, the fact that $f_{d_j(i)}$ is non negative, implies the

[6] Is is sufficient to notice that $P\{\hat{d}_j(j) \le x\}$ is the same as the probability of j uniform random variables in the interval $[0,d]$ being smaller than x/d, i.e. $P\{\hat{d}_j(j) \le x/d\} = (x/d)^j$

uniqueness of $f_{d_{j-1}(i)|d_j(i)}$. In fact, let $f_1(x,y)$ and $f_2(x,y)$ be two solutions[7] for (2.7). Then, subtracting one identity from the other, we have:

$$\begin{aligned} 0 &= \int_x^1 f_1(x,y) f_{d_j(i)}(y)dy - \int_x^1 f_2(x,y) f_{d_j(i)}(y)dy \\ &= \int_x^1 [f_1(x,y) - f_2(x,y)] f_{d_j(i)}(y)dy \end{aligned}$$

Since the equation above must hold for all $x \in [0,1]$, the integrand must be 0 almost everywhere. Moreover, since $f_{d_j(i)}$ is positive a.e., then $f_1 - f_2 = 0$ a.e, that completes the proof. □

Figure 2.1 represents the density functions of $d_k(i)$, for $k = 8$ and some values of i.

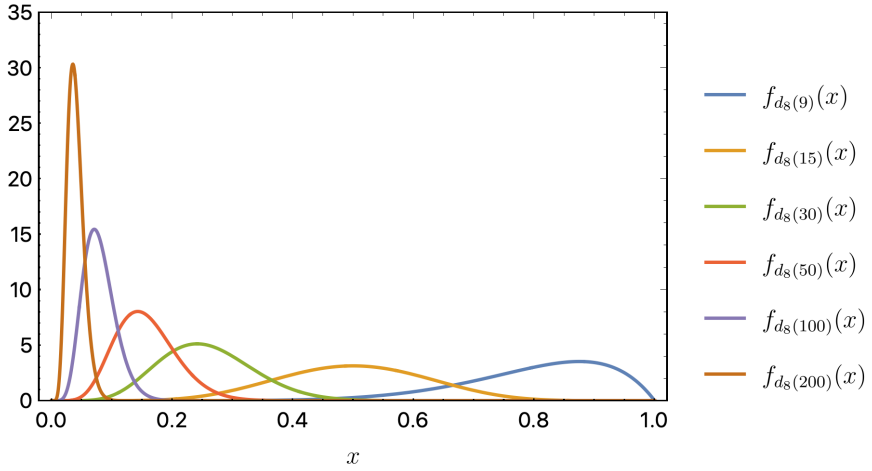

Figure 2.1: *Density functions of $d_k(i)$, for $k = 8$ and some values of i*

Corollary 2.2.2. *The variables $d_j(i)$, $j = 1, \ldots, k-1$ and $i \geq k$ are distributed as $k-1$ ordered uniform random variables in the interval $[0, d_k(i)]$.*

Proof. For a fixed $j < k$ and $d = d_k(i)$, let y_j be the j^{th} smallest of $k-1$ uniform random variables in the interval $[0,d]$. The variable y_j, for a fixed d, has the following distribution[8]:

$$f_{y_j|d}(x|d) = \frac{(k-1)!}{(k-j-1)!(j-1)!} \frac{x^{j-1}\,(d-x)^{k-j-1}}{d^{k-1}}$$

Then, by (2.1) applied to $j = k$, if $d \sim d_k(i)$, we have that:

$$\begin{aligned} f_{y_j,d}(x,d) &= f_{y_j|d}(x|d) f_{d_k(i)}(d) \\ &= \frac{(k-1)!}{(k-j-1)!(j-1)!} \frac{x^{j-1}\,(d-x)^{k-j-1}}{d^{k-1}} \frac{i!}{(i-k)!(k-1)!} d^{k-1}\,(1-d)^{i-k} \\ &= \frac{i!}{(k-j-1)!(j-1)!(i-k)!} x^{j-1}\,(d-x)^{k-j-1}\,(1-d)^{i-k} \end{aligned}$$

And

$$\begin{aligned} f_{y_j}(x) &= \int_x^1 \frac{i!}{(k-j-1)!(j-1)!(i-k)!} x^{j-1}\,(y-x)^{k-j-1}\,(1-y)^{i-k}\,dy \\ &= \frac{i!}{(j-1)!(i-j)!} x^{j-1}(1-x)^{i-j} \end{aligned}$$

[7] i.e., $f_i(x,y)$ is such that $f_{d_{j-1}(i)}(x) = \int_x^1 f_i(x,y) f_{d_j(i)}(y)dy$

[8] It follows from applying $i = k-1$ to (2.4).

that presents the same distribution as $d_j(i)$ (see equation (2.1)). □

2.2.2 Additional Properties of the Process

Theorem 2.2.3. *Given that a node accepts the m^{th} received request (where $m > k$), this connection will survive for exactly the n next requests with a probability given by:*

$$P\{m^{th} \text{ request surviving for exactly } n \text{ other requests}\} = \frac{m}{(n+m+1)(n+m)} \tag{2.8}$$

Proof. First let us calculate the probability $\bar{P}(m,n,k)$ of the m^{th} received request surviving for at least n following Poisson events. Let us denote A_i the event where the m^{th} request becomes the connection with the i^{th} smallest metric. We know that $P(A_i) = \frac{1}{k}$ for $i = 1, \ldots, k$, since, by (2.2) -if the acceptance metric is unknown-, a connection has the same probability of being any of the k current connections (recall that the metrics of the $k-1$ closest neighbors at step i are $k-1$ uniform random variables in the interval $[0, d_k(i)]$, where $d_k(i)$ is the metric of the furthest accepted connection). Then, we have:

$$\begin{aligned}
\bar{P}(m,n,k) &= \sum_{l=1}^{k} P\{m^{\text{th}} \text{ request surviving for at least } n \text{ events} | A_l\} P(A_l) \\
&= \frac{1}{k} \sum_{l=1}^{k} P\{m^{\text{th}} \text{ req. being the } l^{\text{th}} \text{ closest connec. and surviving for at least } n \text{ events}\} \\
&= \frac{1}{k} \sum_{l=1}^{k} P\{\text{at most } k-l \text{ of } X_{m+1}, \ldots, X_{m+n} \text{ being smaller than } d_l(m)\} \\
&= \frac{1}{k} \sum_{l=1}^{k} P\{\text{Bin}(n, d_l(m)) \leq k-l\} \\
&= \frac{1}{k} \sum_{l=1}^{k} \sum_{j=0}^{k-l} P\{\text{Bin}(n, d_l(m)) = j\}
\end{aligned}$$

Then, the probability in which we are interested will be:

$$\begin{aligned}
\bar{P}(m,n,k) &= \frac{1}{k} \int_0^1 \sum_{l=1}^{k} \sum_{j=0}^{k-l} \binom{n}{j} x^j (1-x)^{n-j} f_{d_l(m)}(x) dx \\
&= \frac{1}{k} \int_0^1 \sum_{l=1}^{k} \sum_{j=0}^{k-l} \binom{n}{j} \binom{m}{l} l x^{j+l-1} (1-x)^{n-j+m-l} dx
\end{aligned}$$

Switching the order between integration and sum, we have:

$$\begin{aligned}
\bar{P}(m,n,k) &= \frac{1}{k} \sum_{l=1}^{k} \sum_{j=0}^{k-l} \binom{n}{j} \binom{m}{l} l \int_0^1 x^{j+l-1} (1-x)^{n-j+m-l} dx \\
&= \frac{1}{k} \sum_{l=1}^{k} \sum_{j=0}^{k-l} \binom{n}{j} \binom{m}{l} l \frac{(j+l-1)!(n+m-l-j)!}{(n+m)!}
\end{aligned}$$

Rearranging the sums, with the change of variables $q = l + j$, we find:

$$\begin{aligned}
\bar{P}(m,n,k) &= \frac{1}{k}\sum_{q=1}^{k}\sum_{l=1}^{q}\binom{n}{q-l}\binom{m}{l} l\frac{(q-1)!(n+m-q)!}{(n+m)!} \\
&= \frac{1}{k}\sum_{q=1}^{k}\frac{(q-1)!(n+m-q)!}{(n+m)!}\sum_{l=1}^{q}\frac{n!}{(q-l)!(n-q+l)!}\frac{m!}{(l-1)!(m-l)!} \\
&= \frac{1}{k}\sum_{q=1}^{k}\frac{(q-1)!(n+m-q)!}{(n+m)!}\frac{m}{(q-1)!}\frac{(m+n-1)!}{(m+n-q)!} \\
&= \frac{1}{k}\sum_{q=1}^{k}\frac{m}{(n+m)} = \frac{m}{(m+n)}
\end{aligned}$$

Notice that this probability does not depend on k. Then, the probability $P(m,n)$ of the m^{th} request received surviving for **exactly** n requests will be given by $\bar{P}(m,n) - \bar{P}(m,n+1)$. Using the relation found above, we have:

$$\begin{aligned}
P(m,n) &= \frac{m}{(m+n)} - \frac{m}{(n+m+1)} \\
&= \frac{m}{(n+m+1)(n+m)}
\end{aligned}$$

□

Figure 2.2 shows the density functions $P(m,n)$, for some values of m.

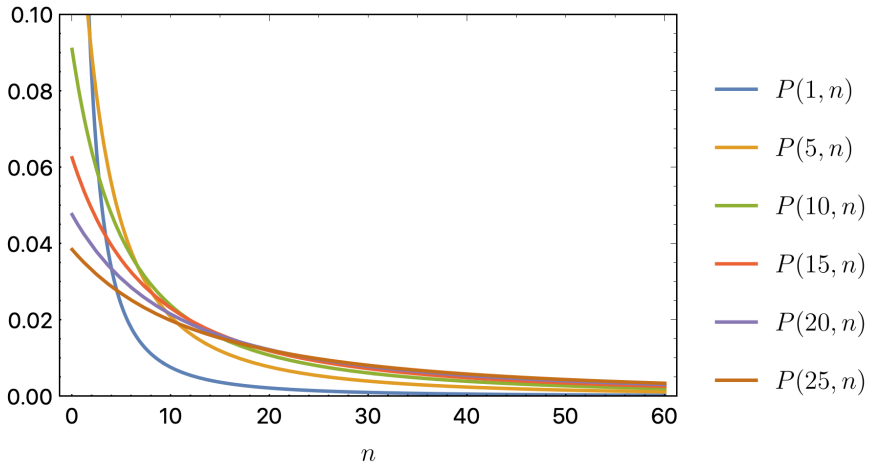

Figure 2.2: *Density functions $P(m,n)$, for some values of m*

Theorem 2.2.4. *Given that a request is the i^{th} one received by a node, for the model with $p = 0$, the probability of acceptance is*

$$p_I(i) = \begin{cases} 1, & \text{if } 1 \leq i \leq k \\ \dfrac{k}{i}, & \text{if } i > k \end{cases} \tag{2.9}$$

Proof. If $i > k$, the probability of acceptance conditioned of the request being the i^{th} received by a node will be the same as the probability of the acceptance metric of the i^{th} request (a uniform

$U[0,1]$) being smaller than $d_k(i-1)$:

$$\begin{aligned} p_I(i) &= P\{U[0,1] < d_k(i-1)\} \\ &= \int_0^1 P\{U[0,1] < d_k(i-1) | d_k(i-1) = y\} f_{d_k(i-1)}(y) dy \\ &= \int_0^1 y \frac{(i-1)!}{(i-1-k)!(k-1)!} y^{k-1} (1-y)^{i-1-k} \, dy \\ &= \frac{k}{i} \end{aligned}$$

If $i \leq k$, then the request will be accepted with probability 1. □

2.3 Another practical application to the synchronized solution

Let us introduce another problem related to the synchronized model studied in the last section. Suppose that the expiration of the connections occur in a asynchronous manner; in other words all the nodes will drop their connections independently and at different points in time. In that case, we cannot assume that $p = 0$ as in the last section, since the expiration effects are not negligible. However, it can be shown that the solution assuming $p = 0$ serves as a bound to the solution for $p > 0$, as in the following proposition:

Proposition 2.3.1. *For the non-simplified problem (i.e., for $p > 0$), the acceptance metrics dominate the analogous acceptance metrics for the case $p = 0$; in other words, if $\hat{d}_j(i)$ is the j^{th} smallest accepting metric among the realised connections at time i for the case $p > 0$, and $d_j(i)$ is the j^{th} smallest accepting metric at time i, for the case $p = 0$, then:*

$$P\{\hat{d}_j(i) \geq x\} \geq P\{d_j(i) \geq x\}$$

whenever these variables are well defined.

Proof. Let $\hat{D}(i) = \{\hat{d}_1(i), \ldots, \hat{d}_k(i)\}$ be the set of accepting metrics of the realised connections of a certain node at a certain point i in time. Let us study the evolution of this set, for the case $p > 0$. If $\hat{X}$ is the acceptance metric of the next incoming request, $Y \sim \text{Ber}(p)$ (related to the event where one of the connections expire) and Z_i is the event where $Y = 1$ and the expired connection is the one with the i^{th} smallest acceptance metric, we have that:

- If $Y = 1$, then $\hat{d}_1(i+1) = \begin{cases} \min(\hat{X}, \hat{d}_2(i)), \text{ on } Z_1 \\ \min(\hat{X}, \hat{d}_1(i)), \text{ on } Z_1^{\complement} \end{cases}$
- If $Y = 1$, then $\hat{d}_j(i+1) = \begin{cases} \max(\min(\hat{X}, \hat{d}_{j+1}(i)), \hat{d}_j(i)), \text{ on } \cup_{l=1}^{j-1} Z_l \\ \min(\max(\hat{X}, \hat{d}_{j-1}(i)), \hat{d}_{j+1}(i)), \text{ on } Z_j \\ \min(\max(\hat{X}, \hat{d}_{j-1}(i)), \hat{d}_j(i)), \text{ on } \cup_{l=j+1}^{k} Z_l \end{cases}$
- If $Y = 1$, then $\hat{d}_k(i+1) = \begin{cases} \max(\hat{X}, \hat{d}_{k-1}(i)), \text{ on } Z_k \\ \max(\hat{X}, \hat{d}_k(i)), \text{ on } Z_k^{\complement} \end{cases}$
- If $Y = 0$ and $1 < j \leq k$, then $\hat{d}_j(i+1) = \min(\hat{d}_j(i), \max(\hat{X}, \hat{d}_{j-1}(i)))$
- If $Y = 0$, then $\hat{d}_1(i+1) = \min(\hat{d}_1(i), \hat{X})$

On the other hand, let $D(i) = \{d_1(i), \ldots, d_k(i)\}$ is the set of accepting metrics of the connections of a certain node at a certain point i in time for the case $p = 0$. If $X = \hat{X}$ is the acceptance metric of the next incoming request, then:

- For any Y and $1 < j \leq k$, we have $d_j(i+1) = \min(d_j(i), \max(X, d_{j-1}(i)))$
- For any Y, we have $d_1(i+1) = \min(d_1(i), X)$

We want to prove by induction that $d_j(i) \leq \hat{d}_j(i)$ for all $1 \leq j \leq k$ and $i \in \mathbb{Z}$. Thus, if we additionally assume that $d_j(i) \leq \hat{d}_j(i)$ for all $1 \leq j \leq k$ and some fixed $i \in \mathbb{Z}$, we have that

- If $Y = 1$, then $d_1(i+1) = \begin{cases} \min(X, d_1(i)) \leq \min(\hat{X}, \hat{d}_2(i)) = \hat{d}_1(i+1), \text{ on } Z_1 \\ \min(X, d_1(i)) \leq \min(\hat{X}, \hat{d}_1(i)) = \hat{d}_1(i+1), \text{ on } Z_1^{\complement} \end{cases}$
- If $Y = 1$ and $1 < j \leq k$, then

$$\begin{aligned} d_j(i+1) &= \min(d_j(i), \max(X, d_{j-1}(i))) \\ &\leq \begin{cases} d_j(i) \leq \max(\min(\hat{X}, \hat{d}_{j+1}(i)), \hat{d}_j(i) = \hat{d}_j(i+1)), \text{ on } \cup_{l=1}^{j-1} Z_l \\ \min(\max(\hat{X}, \hat{d}_{j-1}(i)), \hat{d}_{j+1}(i)) = \hat{d}_j(i+1)), \text{ on } Z_j \\ \min(\max(\hat{X}, \hat{d}_{j-1}(i)), \hat{d}_j(i)) = \hat{d}_j(i+1)), \text{ on } \cup_{l=j+1}^{k} Z_l \end{cases} \end{aligned}$$

- If $Y = 1$, then

$$\begin{aligned} d_k(i+1) &= \min(d_k(i), \max(X, d_{k-1}(i))) \\ &\leq \begin{cases} \max(X, d_{k-1}(i)) \leq \max(\hat{X}, \hat{d}_{k-1}(i)) = \hat{d}_k(i+1), \text{ if } Z = 1 \\ \max(X, d_{k-1}(i)) \leq \max(\hat{X}, \hat{d}_k(i)) = \hat{d}_k(i+1), \text{ if } Z = 0 \end{cases} \end{aligned}$$

- If $Y = 0$, then

$$\begin{aligned} d_j(i+1) &\leq \min(d_j(i), \max(X, d_{j-1}(i))) \leq \min(\hat{d}_j(i), \max(\hat{X}, \hat{d}_{j-1}(i))) = \hat{d}_j(i+1) \\ d_1(i+1) &\leq \min(d_1(i), X) \leq \min(\hat{d}_1(i), \hat{X}) = \hat{d}_j(i+1) \end{aligned}$$

meaning that $d_j(i+1) \leq \hat{d}_j(i+1)$ for all $1 \leq j \leq k$. Now, we want to prove that this is valid for some relatively small $i^* \in \mathbb{N}^+$. Let i^* be the first time step in which all the connection are established for the case $p > 0$ (notice that i^* is a random variable). Then, by construction, $\hat{D}(i^*)$ will be a set of k i.i.d uniform random variables. On the other hand, since for the case $p = 0$ all the connection were already established since $i = k$, we have:

$$d_j(i^*) \leq d_j(k)$$

and since $d_j(k) \sim \hat{d}_j(i^*)$, given that both sets $\hat{D}(i^*)$ and $D(k)$ are constituted by k i.i.d uniform random variables, we finally have

$$P\{\hat{d}_j(i^*) \geq x\} \geq P\{d_j(i^*) \geq x\}$$

that finishes our proof. □

2.3.1 Bounds on the probability of acceptance for $p > 0$

To estimate a bound to the probability of acceptance of a given request, with no further information about the accepting node, we need to calculate the probability distribution of the number of Poisson events of this node's acceptance process since the last expiration of its connections. We first assume that the time since the last expiration is given by a uniform random variable in the interval $(t - T, t)$. Then, we proceed in two different manners:

1. assuming that the requests were received by the accepting node **exactly** every $\frac{1}{\lambda}$ units of time (that is to say, the incoming requests do not configure a Poisson process anymore)

2. assuming that the requests were received as a Poisson process of rate λ

The following two propositions will present bounds to the probability of acceptance of a request, for each of these two cases.

Proposition 2.3.2. *The probability of acceptance $\hat{p}_A$ of a request, if the requests are received by the accepting node exactly every $\frac{1}{\lambda}$ units of time, is bounded by*

$$\hat{p}_A \geq p_A \approx \begin{cases} \frac{k}{\lambda T}\left[1 + \log\left(\frac{\lambda T}{k}\right)\right], & \text{if } \lambda T > k \\ 1, & \text{if } \lambda T \leq k \end{cases} \tag{2.10}$$

Proof. It follows from theorem 2.2.4. We have that, for $\lambda T > k$:

$$\begin{aligned} \hat{p}_A \geq p_A &= \sum_{i=1}^{\infty} p_I(i) P\{\text{asking a node that has been requested } i-1 \text{ times}\} \\ &= \sum_{i=1}^{k} \frac{1}{T\lambda} + \sum_{i=k+1}^{\lambda T} \frac{1}{T\lambda}\frac{k}{i} = \frac{k}{T\lambda}(1 + H_{T\lambda} - H_k) \end{aligned}$$

where H_j stands for the j^{th} harmonic number[9]. In order to estimate the value of this function, we bound the harmonic number as follows ([You91])

$$\frac{1}{2(n+1)} < H_n - \log(n) - \gamma < \frac{1}{2n}$$

where γ is the Euler-Mascheroni constant. Then:

$$\left| H_{T\lambda} - H_k - \log\left(\frac{\lambda T}{k}\right)\right| < \frac{1}{2k} - \frac{1}{2(1+\lambda T)}$$

and finally

$$\hat{p}_A \geq p_A \approx \frac{k}{T\lambda}\left[1 + \log\left(\frac{\lambda T}{k}\right)\right]$$

Analogously, for $\lambda T \leq k$, we will have :

$$\begin{aligned} \hat{p}_A \geq p_A &= \sum_{i=1}^{\infty} p_I(i) P\{\text{asking a node that has been requested } i-1 \text{ times}\} \\ &= \sum_{i=1}^{T\lambda} \frac{1}{T\lambda} = 1 \end{aligned}$$

□

Notice that $\lim_{T\to\infty} p_A = \lim_{T\to\infty} \frac{k}{T\lambda}\left[1 + \log\left(\frac{\lambda T}{k}\right)\right] = \lim_{T\to\infty} \frac{k^2}{\lambda^2 T} = 0$. This behaviour is consistent with the results found in the last section, since if $T \to \infty$, the distribution of the acceptance metric of the farthest connection will tend to get smaller over time (see Figure 2.1).

Analogously to the last proposition, we now present a bound to the probability of acceptance of a request (given the last time since expiration of the queried node's connections), for the case where the requests arrive following a Poisson Process of rate λ.

Proposition 2.3.3. *Given that the auto peering starting point of a given node n was t units of time ago and requests are received following a Poisson process of rate λ, the probability $\hat{p}_t$ of a request*

[9] $H_x = \sum_{i=1}^{x} \frac{1}{i}$

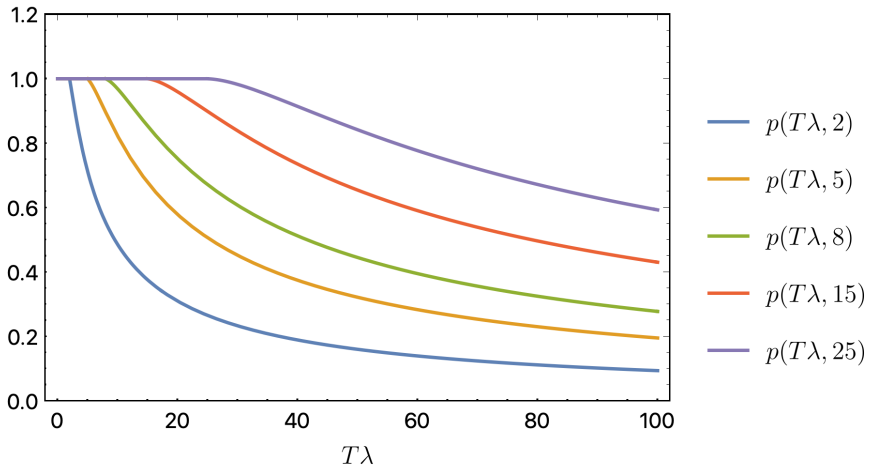

Figure 2.3: *Exact expression for the bound p on the probability of acceptance of a request, for some values of k, under the assumption that the requests are received by the accepting node exactly every $\frac{1}{\lambda}$ units of time.*

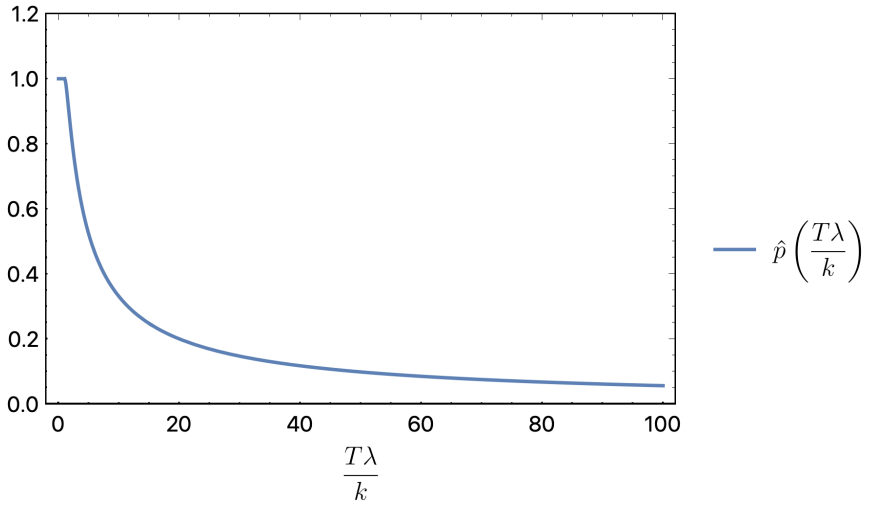

Figure 2.4: *Approximated expression for the bound p on the probability of acceptance of a request, under the assumption that the requests are received by the accepting node exactly every $\frac{1}{\lambda}$ units of time.*

being accepted by this node is bounded by:

$$\hat{p}_t \geq p_t = \frac{\lambda t \Gamma(k, \lambda t) - \Gamma(k+1, \lambda t) + k\Gamma(k)}{\lambda t \Gamma(k)} \tag{2.11}$$

Proof.

$$\begin{aligned}
\hat{p}_t \geq p_t &= \sum_{i=1}^{\infty} P\{\text{a node receiving } i-1 \text{ requests in } t \text{ seconds}\} p_I(i) \\
&= \sum_{i=1}^{k} P\{\text{a node receiving } i-1 \text{ requests in } t \text{ seconds}\} \\
&+ \sum_{i=k+1}^{\infty} P\{\text{a node receiving } i-1 \text{ requests in } t \text{ seconds}\} \frac{k}{i} \\
&= \exp(-\lambda t) \sum_{i=1}^{k} \frac{(\lambda t)^{i-1}}{(i-1)!} + \frac{k \exp(-\lambda t)}{\lambda t} \sum_{i=k+1}^{\infty} \frac{(\lambda t)^i}{i!} \\
&= \frac{\lambda t \Gamma(k, \lambda t) - \Gamma(k+1, \lambda t) + k\Gamma(k)}{\lambda t \Gamma(k)}
\end{aligned}$$

□

2.4 Solutions for the generalized ($p > 0$) model

In this section, we find the equations governing the distribution of the variables $d_j(i)$, for $j = 1, \ldots, \min(k, i)$ and $i \in \mathbb{N}$, as defined previously (in other words, the j^{th} smallest acceptance metric among the realised connections of a node after the i^{th} request has taken place), for the case $p > 0$. A major difference between the two models is that the present one does depend on the order that the requests were made. For instance, take the first accepted request: no matter how small is its acceptance metric, it will have a probability of at least $1 - \left(1 - \frac{p}{k}\right)^n$ of being dropped after n other requests were received. For that reason, we treat the present model differently, finding relations between subsequently sets $(d_j(i))_{j=1,\ldots,\min(k,i)}$ and $(d_j(i+1))_{j=1,\ldots,\min(k,i+1)}$.

Let us call Z_j the event of probability p/k of the j^{th} smallest accepted connection (i.e., the one with the j^{th} smallest acceptance metric $d_j(i)$) being dropped by the requester and $Z = \cup Z_j$ (recall that at most one of the connections might get dropped, so these events are actually disjoint). Let us also call X_i the acceptance metric of the request received at step i of the process. By the law of total probability, we have that, for $1 < n < k$:

$$\begin{aligned}
P\{d_n(i+1) \leq x\} = P\left\{d_n(i+1) \leq x | Z^{\complement}\right\} P\left\{Z^{\complement}\right\} \\
+ P\Big\{d_n(i+1) \leq x | \bigcup_{j=1}^{n-1} Z_j\Big\} P\Big\{\bigcup_{j=1}^{n-1} Z_j\Big\} \\
+ P\{d_n(i+1) \leq x | Z_n\} P\{Z_n\} \\
+ P\Big\{d_n(i+1) \leq x | \bigcup_{j=n+1}^{k} Z_j\Big\} P\Big\{\bigcup_{j=n+1}^{k} Z_j\Big\}
\end{aligned}$$

Since $P\{Z_j\} = \frac{p}{k}$ for each j and $P\{Z^{\complement}\} = 1 - p$, we have:

$$\begin{aligned} P\{d_n(i+1) \leq x\} = & P\left\{d_n(i+1) \leq x | Z^{\complement}\right\}(1-p) \\ & + P\left\{d_n(i+1) \leq x | \bigcup_{j=1}^{n-1} Z_j\right\}\frac{p(n-1)}{k} \\ & + P\{d_n(i+1) \leq x | Z_n\}\frac{p}{k} \\ & + P\left\{d_n(i+1) \leq x | \bigcup_{j=n+1}^{k} Z_j\right\}\frac{p(k-n)}{k} \end{aligned} \tag{2.12}$$

Now, let us reason about the four terms on the right. First, the event of $d_n(i+1)$ being smaller than x, given that none of the connections is dropped will occur if and only if one of the following conditions is satisfied:

- $d_n(i)$ is smaller than x and X_{i+1} is larger than x or
- $d_{n-1}(i)$ is smaller than x and X_{i+1} is smaller than x

Hence, concretely, we have

$$\left\{d_n(i+1) \leq x | Z^{\complement}\right\} = \{d_n(i) \leq x; X_{i+1} > x\} \bigcup \{d_{n-1}(i) \leq x; X_{i+1} \leq x\}$$

Analogously, we can find the following relations for the other three events on equation (2.12):

$$\left\{d_n(i+1) \leq x | \bigcup_{j=1}^{n-1} Z_j\right\} = \{d_{n+1}(i) \leq x; X_{i+1} > x\} \bigcup \{d_n(i) \leq x; X_{i+1} \leq x\}$$

$$\{d_n(i+1) \leq x | Z_n\} = \{d_{n+1}(i) \leq x; X_{i+1} > x\} \bigcup \{d_{n-1}(i) \leq x; X_{i+1} \leq x\}$$

$$\left\{d_n(i+1) \leq x | \bigcup_{j=n+1}^{k} Z_j\right\} = \{d_n(i) \leq x; X_{i+1} > x\} \bigcup \{d_{n-1}(i) \leq x; X_{i+1} \leq x\}$$

Since X_{i+1} is a uniform random variable in the interval $[0, 1]$, independent of $d_n(i)$, $d_{n+1}(i)$ and $d_{n-1}(i)$, we have, for any $1 \leq m \leq k$:

$$\begin{aligned} P\{d_m(i) \leq x; X_{i+1} > x\} &= P\{d_m(i) \leq x\} P\{X_{i+1} > x\} = P\{d_m(i) \leq x\}(1-x) \\ P\{d_m(i) \leq x; X_{i+1} \leq x\} &= P\{d_m(i) \leq x\} P\{X_{i+1} \leq x\} = P\{d_m(i) \leq x\} x \end{aligned}$$

Substituting the above relations into equation (2.12), results:

$$\begin{aligned} P\{d_n(i+1) \leq x\} = & P\{d_n(i) \leq x\}\left[(1-p)(1-x) + \frac{p(n-1)}{k}x + \frac{p(k-n)}{k}(1-x)\right] \\ & + P\{d_{n-1}(i) \leq x\}\left[\frac{p}{k}x + (1-p)x + \frac{p(k-n)}{k}x\right] \\ & + P\{d_{n+1}(i) \leq x\}\left[\frac{p(n-1)}{k}(1-x) + \frac{p}{k}(1-x)\right] \end{aligned}$$

With analogous calculations for $n = 1$ and $n = k$, we find:

$$\begin{aligned} P\{d_1(i+1) \leq x\} &= P\left\{d_1(i+1) \leq x | Z^{\complement}\right\} P\left\{Z^{\complement}\right\} + P\{d_1(i+1) \leq x | Z_1\} P\{Z_1\} \\ &\quad + P\Big\{d_1(i+1) \leq x | \bigcup_{j=2}^{k} Z_j\Big\} P\Big\{\bigcup_{j=2}^{k} Z_j\Big\} \\ &= x + [k - kx - p + px] \frac{1}{k} P\{d_1(i) \leq x\} + \frac{p}{k}(1-x) P\{d_2(i) \leq x\} \end{aligned}$$

$$\begin{aligned} P\{d_k(i+1) \leq x\} &= P\left\{d_k(i+1) \leq x | Z^{\complement}\right\} P\left\{Z^{\complement}\right\} \\ &\quad + P\Big\{d_k(i+1) \leq x | \bigcup_{j=1}^{k-1} Z_j\Big\} P\Big\{\bigcup_{j=1}^{k-1} Z_j\Big\} \\ &\quad + P\{d_k(i+1) \leq x | Z_k\} P\{Z_k\} \\ &= [-pk + p + k] \frac{x}{k} P\{d_{k-1}(i) \leq x\} \\ &\quad + [k - kx - pk + 2pkx - px] \frac{1}{k} P\{d_k(i) \leq x\} \end{aligned}$$

The last three equations describe the evolution of the process and will be used in the next section to find the stationary distribution of the problem.

2.4.1 Stationary distribution

Let us find the stationary distributions $(d_1^S, \ldots, d_k^S)$ of the problem, given by the system of functional equations[10]:

$$\begin{cases} kx + [-kx - p + px] P\{d_1^S \leq x\} + p(1-x) P\{d_2^S \leq x\} = 0 \\ [-pn + p + k] x P\{d_{n-1}^S \leq x\} + [-kx - pn + 2pnx - px] P\{d_n^S \leq x\} \\ \quad + pn(1-x) P\{d_{n+1}^S \leq x\} = 0 \text{ if } 1 < n < k \\ [-pk + p + k] x P\{d_{k-1}^S \leq x\} + [-kx - pk + 2pkx - px] P\{d_k^S \leq x\} = 0 \end{cases} \tag{2.13}$$

Proposition 2.4.1. *The solution to the system (2.13) is, for $1 \leq n \leq k$:*

$$P(d_n^S \leq x) = \frac{\sum_{i=n}^{k} \left(\frac{x}{1-x}\right)^i \prod_{j=1}^{i} \frac{-pj+k+p}{jp}}{1 + \sum_{i=1}^{k} \left(\frac{x}{1-x}\right)^i \prod_{j=1}^{i} \frac{-pj+k+p}{jp}} \tag{2.14}$$

Proof. First notice that the system above can be represented as $M_k x_k = y_k$, where:

$$M_k = \begin{pmatrix} a_1 & b_1 & & & \\ c_1 & a_2 & b_2 & & \\ & c_2 & \ddots & \ddots & \\ & & \ddots & \ddots & b_{k-1} \\ & & & c_{k-1} & a_k \end{pmatrix}, x_k = \begin{pmatrix} P\{d_1^S \leq x\} \\ P\{d_2^S \leq x\} \\ \vdots \\ P\{d_{k-1}^S \leq x\} \\ P\{d_k^S \leq x\} \end{pmatrix} \text{ and } y_k = \begin{pmatrix} -kx \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}.$$

[10] In order to $(d_1^S, \ldots, d_k^S)$ be a stationary distribution, the system (2.13) must hold for all $x \in [0, 1]$

$$\text{and } \begin{cases} a_n = -kx - pn + 2pnx - px \\ b_n = pn(1-x) \\ c_{n-1} = (-pn + p + k)x \end{cases}$$

Then, the determinant of the matrix M_k can be calculated using the following recursive relation:

$$\det(M_n) = a_n \det(M_{n-1}) - b_{n-1}c_{n-1}\det(M_{n-2}),$$
$$\text{with } \begin{cases} \det(M_1) = a_1 \\ \det(M_2) = a_1 a_2 - c_1 b_1 \end{cases}$$

Since a_n, b_n and c_n are always polynomials of degree one on the variable x, $\det(M_n)$ will be a polynomial of degree at most n (also on the variable x); in other words, the determinant of M_k will be a polynomial of degree at most k, that must vanish either only on a set of at most k isolated points or in the whole real line. To check that the polynomial is not identically zero, let $\det(\bar{M}_n)$, $\bar{a}_n$, $\bar{b}_n$ and $\bar{c}_n$ be defined as the particular case of $\det(M_n)$, a_n, b_n and c_n for $x = 1$. Then, we have:

$$\text{and } \begin{cases} \bar{a}_n = -k + pn - p \\ \bar{b}_n = 0 \\ \bar{c}_{n-1} = -pn + p + k \end{cases} \implies \det(\bar{M}_k) = \prod_{i=1}^{k} p(i-1) - k \neq 0$$

Since for $x = 1$, $\det(M_k)$ is not zero, we know that $\det(M_k)$ is not the null polynomial. Then, for almost every $x \in [0, 1]$, the matrix M_k is invertible and then the solution to the system (2.13) must be unique, hence it suffices to verify by substitution that equation (2.14) holds. This substitution is straightforward and is not going to be explicited here. □

Note that, for $p = 1$, the solution will be:

$$\begin{aligned} P(d_n^S \leq x) =& \frac{\sum_{i=n}^{k} \left(\frac{x}{1-x}\right)^i \prod_{j=1}^{i} \frac{-j+k+1}{j}}{1 + \sum_{i=1}^{k} \left(\frac{x}{1-x}\right)^i \prod_{j=1}^{i} \frac{-j+k+1}{j}} \\ =& \frac{\sum_{i=n}^{k} (1-x)^{k-i} x^i \binom{k}{i}}{(1-x)^k + \sum_{i=1}^{k} (1-x)^{k-i} x^i \binom{k}{i}} \\ =& \sum_{i=n}^{k} (1-x)^{k-i} x^i \binom{k}{i} \\ =& P\{\text{Bin}(k, x) \geq n\} \end{aligned}$$

that is the probability of having at least n successes in k attempts of success chance x, i.e., the probability of having n of k uniform random variables in the interval $[0, 1]$ smaller than x, as calculated in the past sections. This was already expected, since for $p = 1$ the set of acceptance metrics of the connections will indeed be distributed as k i.i.d. uniform random variables in the interval $[0, 1]$. On the other hand, for $p \to 0$ and any $x > 0$:

$$\lim_{p \to 0} P(d_n^S \leq x) = \lim_{p \to 0} \frac{\sum_{i=n}^{k} \left(\frac{x}{1-x}\right)^i p^{k-i} \prod_{j=1}^{i} \frac{-pj+k+p}{j}}{p^k + \sum_{i=1}^{k} \left(\frac{x}{1-x}\right)^i p^{k-i} \prod_{j=1}^{i} \frac{-pj+k+p}{j}} = 1$$

and, for $x = 0$, $\lim_{p \to 0} P(d_n^S \leq x) = 0$, meaning that for $p = 0$, the stationary distribution is $(\delta_0, \ldots, \delta_0)$ where δ_0 stands for the Dirac delta around the point $x = 0$. This was also an expected result, since -as already stated in the past sections- the metrics $d_j(i)$ decrease with the increment of i.

2.4.2 Some approximations

Now, let us analyze the process under the point of view of the requester. In the stationary regime, to get its request accepted, the acceptance metric X has to be smaller than the acceptance metric of the farthest realised connection of the requested node. Since the requested nodes are chosen in such a way that the acceptance metrics are not affected (because the requested node's choice is made based only on the **request** metric), we can assume that the acceptance metrics of their farthest connections, in the stationary regime, will follow the distribution $f_{d_k^S}$. Then, on average, the probability of being accepted will be given by

$$\begin{aligned}
P\{U[0,1] < d_k^S\} &= \int_0^1 P\{U[0,1] < x\} f_{d_k^S}(x)dx = E(d_k^S) \\
&= 1 - \int_0^1 P(X_k^S \leq x)dx \\
&= 1 - \int_0^1 \frac{\prod_{j=1}^k \frac{jc+b}{ja}}{1 + \sum_{i=1}^k \prod_{j=1}^i \frac{jc+b}{ja}} dx \\
&= 1 - \int_0^1 \frac{\frac{\prod_{j=1}^k (-jp+k+p)}{k!} x^k}{p^k(1-x)^k + \sum_{i=1}^k p^{k-i}(1-x)^{k-i}x^i \frac{\prod_{j=1}^i (-jp+k+p)}{i!}} dx
\end{aligned}$$

The integral above does not have any simple solution, so we must find some approximations to it. The first hypothesis used is that $p \ll 1$. In this case, $(j-1)p \ll k$ for all $1 \leq j \leq k$ and then $p + k - jp \approx k$, resulting:

$$P\{U[0,1] < d_k^S\} \approx 1 - \int_0^1 \frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^k p^{k-i}(1-x)^{k-i}x^i \frac{k^i}{i!}} dx$$

The fraction on the equation above can be approximated by one with a less complicated integral. First notice that, setting $w = \frac{p}{k}\frac{1-x}{x}$, the integrand will be:

$$\frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^k p^{k-i}(1-x)^{k-i}x^i \frac{k^i}{i!}} = \left[\sum_{i=0}^k w^{k-i}\frac{k!}{i!}\right]^{-1} \tag{2.15}$$

Now, for each of the regions $w > 1$ and $w < 1$, we will find different suitable approximations to this function.

Approximations for $w < 1$

For $w < 1$, we approximate the multiplicative inverse of the function by its two terms of lower order on w, resulting:

$$\begin{aligned}\left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} &= \left[\sum_{i=k-1}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} + \left\{\left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} - \left[\sum_{i=k-1}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1}\right\} \\ &= \left[\sum_{i=k-1}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1}\left[1 - \frac{\sum_{i=0}^{k-2} w^{k-i}\frac{k!}{i!}}{\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}}\right] \\ &= \frac{1}{1+wk}\left[1 - R_2^w(w)\right]\end{aligned}$$

where

$$R_2^w(w) = \frac{\sum_{i=0}^{k-2} w^{k-i}\frac{k!}{i!}}{\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}} = w^2\frac{\frac{w^{k-2}}{0!} + \frac{w^{k-3}}{1!} + \cdots + \frac{w}{(k-3)!} + \frac{1}{(k-2)!}}{\frac{w^k}{0!} + \frac{w^{k-1}}{1!} + \cdots + \frac{w}{(k-1)!} + \frac{1}{k!}} \leq k(k-1)w^2$$

This does not appear, at first sight, to be a really promising approximation, since k might be as large as we want. Nevertheless, notice that our goal is to find an approximation for an integral taken over the variable x, and the part of the domain where the quality of the approximation is not that good (for $w \approx 1$) corresponds to a small interval on the x variable (see figure 2.5, that depicts the transformation $w \longrightarrow x(w) = \frac{p}{p+wk}$ for some values of $\frac{p}{k}$ to check the distortion between the spaces).

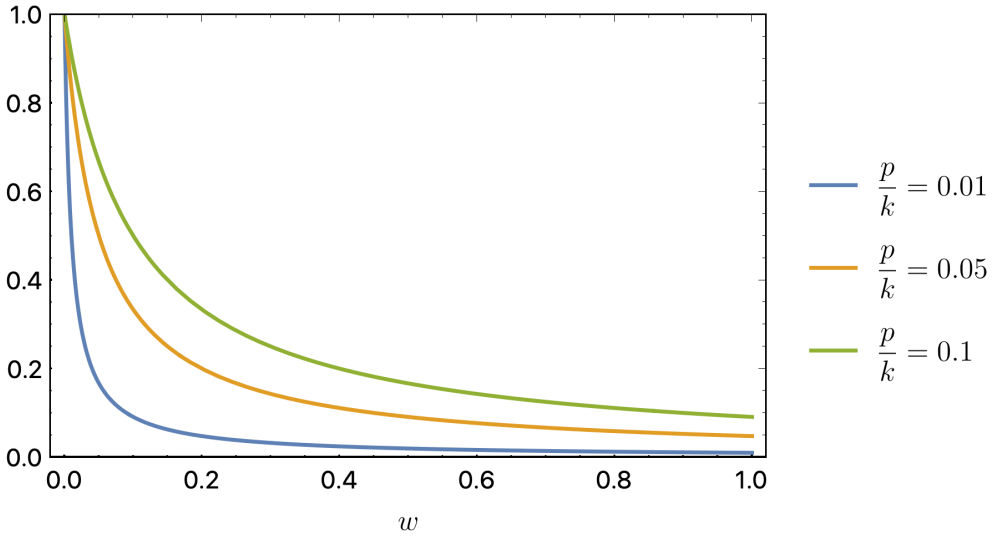

Figure 2.5: *Transformation $w \longrightarrow x(w) = \frac{p}{p+wk}$ for some values of $\frac{p}{k}$*

Then, defining the function $R_2^x(x) = R_2^w\left(\frac{p}{k}\frac{1-x}{x}\right)$ for $\frac{p}{p+k} < x < 1$, we have that

$$R_2^x(x) \leq p^2\frac{k-1}{k}\left(\frac{1-x}{x}\right)^2 < p^2\left(\frac{1-x}{x}\right)^2$$

and

$$\frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^{k} p^{k-i}(1-x)^{k-i}x^i\frac{k^i}{i!}} = \frac{x}{x+p(1-x)}\left[1 - R_2^x(x)\right] \tag{2.16}$$

The approximation given by (2.16) will be later used to estimate the integral of (2.15).

Approximations for $w > 1$

For $w > 1$, we approximate the multiplicative inverse of the function by its two terms of higher order on w, resulting:

$$\begin{aligned}\left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} &= \left[\sum_{i=0}^{1} w^{k-i}\frac{k!}{i!}\right]^{-1} + \left\{\left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} - \left[\sum_{i=0}^{1} w^{k-i}\frac{k!}{i!}\right]^{-1}\right\} \\ &= \left[\sum_{i=0}^{1} w^{k-i}\frac{k!}{i!}\right]^{-1}\left[1 - \frac{\sum_{i=2}^{k} w^{k-i}\frac{k!}{i!}}{\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}}\right] \\ &= \frac{1}{k!}\left(\frac{1}{w}\right)^{k-1}\frac{1}{1+w}\left[1 - R_{-2}^{w}(w)\right]\end{aligned}$$

where

$$R_{-2}^{w}(w) = \frac{\sum_{i=2}^{k} w^{k-i}\frac{k!}{i!}}{\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}} = w^{-2}\frac{\frac{w^{k-2}}{2!} + \frac{w^{k-3}}{3!} + \cdots + \frac{w}{(k-1)!} + \frac{1}{k!}}{\frac{w^{k-2}}{0!} + \frac{w^{k-3}}{1!} + \cdots + \frac{w^{-1}}{(k-1)!} + \frac{w^{-2}}{k!}} \leq \frac{w^{-2}}{2}$$

Then, analogously to the case $w < 1$, defining the function $R_{-2}^{x}(x) = R_{-2}^{w}\left(\frac{p}{k}\frac{1-x}{x}\right)$ for $0 < x < \frac{p}{p+k}$, we have that

$$R_{-2}^{x}(x) \leq \frac{1}{2}\left(\frac{kx}{p(1-x)}\right)^{2}$$

and

$$\frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^{k} p^{k-i}(1-x)^{k-i}x^i\frac{k^i}{i!}} = \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}\left[1 - R_{-2}^{x}(x)\right] \tag{2.17}$$

Hence, finally, we approximate:

$$\frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^{k} p^{k-i}(1-x)^{k-i}x^i\frac{k^i}{i!}} \approx \begin{cases} \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}, & \text{if } 0 < x < \frac{p}{p+k} \\ \frac{x}{x+p(1-x)}, & \text{if } \frac{p}{p+k} < x < 1 \end{cases}$$

The absolute difference between the two solutions is depicted in figure 2.6, for both domains $w \in \mathbb{R}^+$ and $x \in [0, 1]$, for some selected values of k and p. The approximation given by (2.17) will be later used to estimate the integral of (2.15).

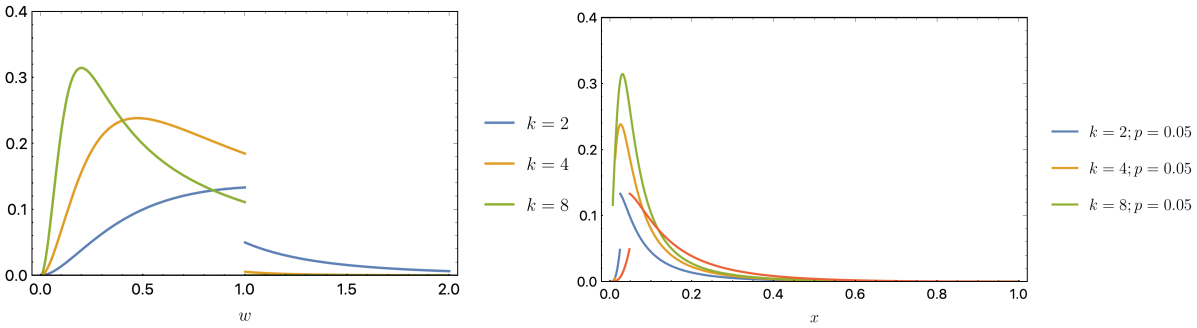

Figure 2.6: *Absolute difference between the exact and approximated solutions for some selected values of k and p*

Calculation of the integral

Using the approximations previously found, we can estimate the integral I^+ over the interval $[\frac{p}{k+p}, 1]$, given by:

$$\begin{aligned} I^+ &= \int_{\frac{p}{p+k}}^{1} \frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^{k} p^{k-i}(1-x)^{k-i}x^i\frac{k^i}{i!}}dx = \int_{\frac{p}{p+k}}^{1} \frac{x}{p(1-x)+x}\left[1 - R_2^x(x)\right]dx \\ &= \frac{1}{1-p}\left[x - \frac{p}{1-p}\log\left(\frac{p}{1-p}+x\right)\right]\Bigg|_{\frac{p}{p+k}}^{1} - \int_{\frac{p}{p+k}}^{1} \frac{x}{p(1-x)+x}R_2^x(x)dx \\ &= \underbrace{\frac{k}{(1-p)(k+p)} + \frac{p}{(1-p)^2}\log\left(\frac{(k+1)p}{k+p}\right)}_{I_1^+} - \underbrace{\int_{\frac{p}{p+k}}^{1} \frac{x}{p(1-x)+x}R_2^x(x)dx}_{I_2^+} \end{aligned}$$

But since

$$0 \leq R_2^x(x) \leq p^2\frac{k-1}{k}\left(\frac{1-x}{x}\right)^2 < p^2\left(\frac{1-x}{x}\right)^2$$

then

$$\begin{aligned} 0 \leq I_2^+ &= \int_{\frac{p}{p+k}}^{1} \frac{x}{p(1-x)+x}R_2^x(x)dx \leq p^2\frac{k-1}{k}\int_{\frac{p}{p+k}}^{1} \frac{x}{p(1-x)+x}\left(\frac{1-x}{x}\right)^2 dx \\ &= p\frac{k-1}{k}\left[\frac{p}{(1-p)}\frac{k}{k+p} + \frac{1}{(1-p)^2}\log\left(\frac{(k+1)p}{k+p}\right) - \log\left(\frac{p}{k+p}\right)\right] \end{aligned}$$

Analogously, in the interval $[0, \frac{p}{k+p}]$ we can integrate the function to calculate I^-, given by:

$$\begin{aligned}
I^- &= \int_0^{\frac{p}{p+k}} \frac{\frac{k^k}{k!}x^k}{\sum_{i=0}^{k} p^{k-i}(1-x)^{k-i}x^i\frac{k^i}{i!}}dx \\
&= \int_0^{\frac{p}{p+k}} \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}\left[1-R^x_{-2}(x)\right]dx \\
&= \underbrace{\int_0^{\frac{p}{p+k}} \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}dx}_{I_1^-} \\
&\quad - \underbrace{\int_0^{\frac{p}{p+k}} \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}R^x_{-2}(x)dx}_{I_2^-}
\end{aligned}$$

Since the function $f(x) = \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}$ is convex[11], then its integral will be bounded by the area of the triangle defined by the points $(0, f(0))$, $(\frac{p}{p+k}, f(\frac{p}{p+k}))$ and $(\frac{p}{p+k}, f(0))$, resulting:

$$0 \le I_1^- = \int_0^{\frac{p}{p+k}} \frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}dx \le \frac{1}{2}\frac{p}{p+k}\left[f\left(\frac{p}{p+k}\right) - f(0)\right] = \frac{1}{4}\frac{p}{p+k}\frac{1}{k!}$$

Analogously, we find that the function $g(x) = \frac{1}{2}\frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}\left(\frac{kx}{p(1-x)}\right)^2$ is convex, resulting:

$$\begin{aligned}
0 \le I_2^- &\le \int_0^{\frac{p}{p+k}} \frac{1}{2}\frac{1}{k!}\left(\frac{kx}{p(1-x)}\right)^{k-1}\frac{kx}{kx+p(1-x)}\left(\frac{kx}{p(1-x)}\right)^2 dx \\
&\le \frac{1}{2}\frac{p}{p+k}\left[g\left(\frac{p}{p+k}\right) - g(0)\right] = \frac{1}{8}\frac{p}{p+k}\frac{1}{k!}
\end{aligned}$$

Finally, combining the bounds for I_1^+, I_2^+, I_1^-, and I_2^-, and defining I as the exact integral of the function (2.15), we have (in the following equation we use the fact that all the terms I_1^+, I_1^-, I_2^+ and I_2^- are positive):

$$\begin{aligned}
I - I_1^+ &= I_1^- - I_2^+ - I_2^- \le I_1^- + I_2^+ + I_2^- \\
&\le p\frac{k-1}{k}\left[\frac{p}{(1-p)}\frac{k}{k+p} + \frac{1}{(1-p)^2}\log\left(\frac{(k+1)p}{k+p}\right) - \log\left(\frac{p}{k+p}\right)\right] + \frac{3}{8}\frac{p}{p+k}\frac{1}{k!} \\
&= \mathcal{O}\left(p\frac{k-1}{k}\left[\frac{p}{(1-p)}\frac{k}{k+p} + \log(k+1)\right] + \frac{3}{8}\frac{p}{p+k}\frac{1}{k!}\right) \\
&= \mathcal{O}\left(p\frac{k-1}{k}\log(k+1) + \frac{3}{8}\frac{p}{p+k}\frac{1}{k!}\right) \\
&= \mathcal{O}\left(p\frac{k-1}{k}\log(k+1)\right)
\end{aligned}$$

[11] Notice that its derivative, given by:

$$f'(x) = \frac{kp\left(\frac{kx}{p(1-x)}\right)^k((k-1)x + p(1-x))}{xk!(kx - px + p)^2}$$

is always positive and monotonically increasing for $k > 1$.

So:

$$I = \frac{k}{(1-p)(k+p)} + \frac{p}{(1-p)^2}\log\left(\frac{(k+1)p}{k+p}\right) + \mathcal{O}\left(p\frac{k-1}{k}\log(k+1)\right)$$

Then, finally, we have:

$$\begin{aligned} E(d_k^S) &= 1 - \frac{1}{1-p}\frac{k}{p+k} - \frac{1}{1-p}\frac{p}{1-p}\log\left(\frac{p(k+1)}{p+k}\right) \\ &\quad + \mathcal{O}\left(p\frac{k-1}{k}\log(k+1)\right) \\ &= \frac{k^2+2k+2}{2(k+1)^2}\sqrt{p} + \mathcal{O}\left(p\frac{k-1}{k}\log(k+1)\right) \end{aligned} \tag{2.18}$$

2.4.3 Heuristic approximations

In this section we present a heuristic approximation, whose error could not be properly calculated but that, in practice, has been shown to achieve better quality than the errors in the last section. Notice that the function being approximated, when represented as a function of the variable $w = \frac{p(1-x)}{kx}$, has the following approximately exponential behaviour:

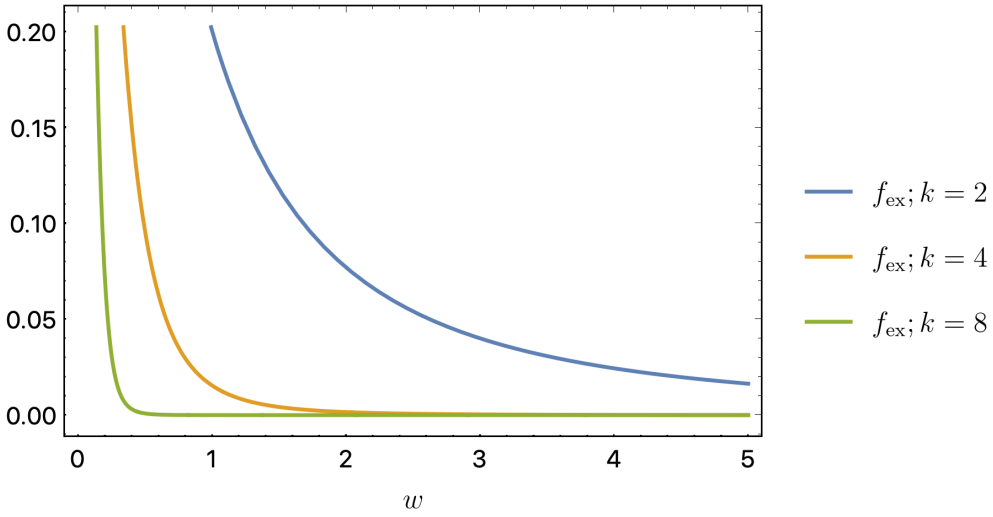

Figure 2.7: *Exact solution f_{ex} for some selected values of k.*

that lead to the *ansatz*:

$$\left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} \approx \exp(-c_k w)$$

Since the largest part of the relevant domain lies on $0 \leq w \leq 1$, we opted for interpolate the ansatz in that region; in other words, we chose the parameter c_k to be such that $\left[\sum_{i=0}^{k}\frac{k!}{i!}\right]^{-1} = \exp(-c_k)$, resulting $c_k = 1 + \log(\Gamma(k+1,1))$ and, consequently:

$$\left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1} = \exp(-w[1+\log(\Gamma(k+1,1))]) \tag{2.19}$$

Figure 2.8 represents the logarithm of both functions (the exact solution $f_{\text{ex}} = \left[\sum_{i=0}^{k} w^{k-i}\frac{k!}{i!}\right]^{-1}$ and the approximated $f_{\text{app}} = \exp(-w[1+\log(\Gamma(k+1,1))])$), for some values of k.

Figure 2.9 represents the absolute difference between the functions, for some values of k and p.

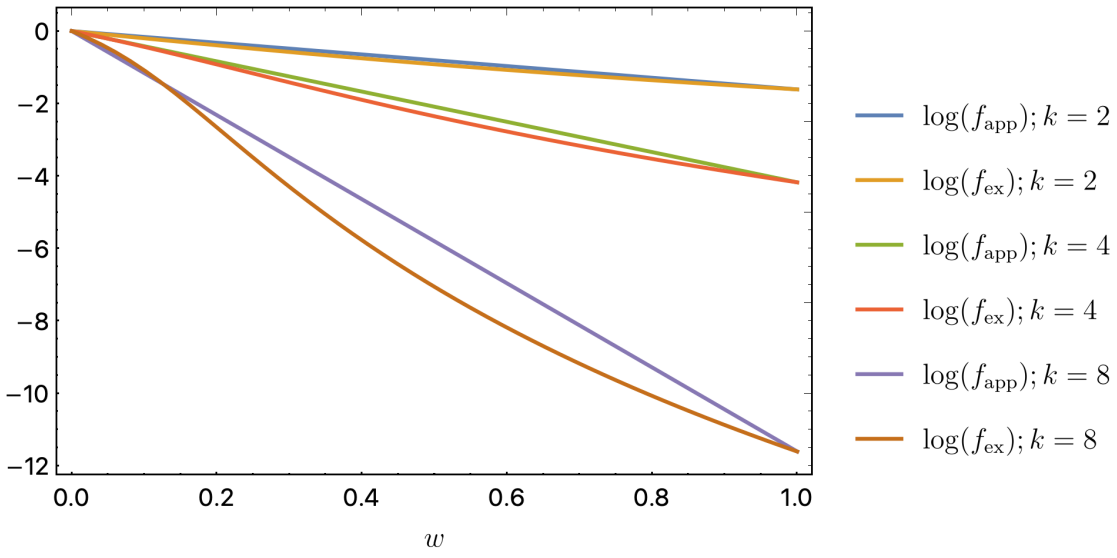

Figure 2.8: *Logarithm of the exact and empirically approximated solutions for some selected values of k*

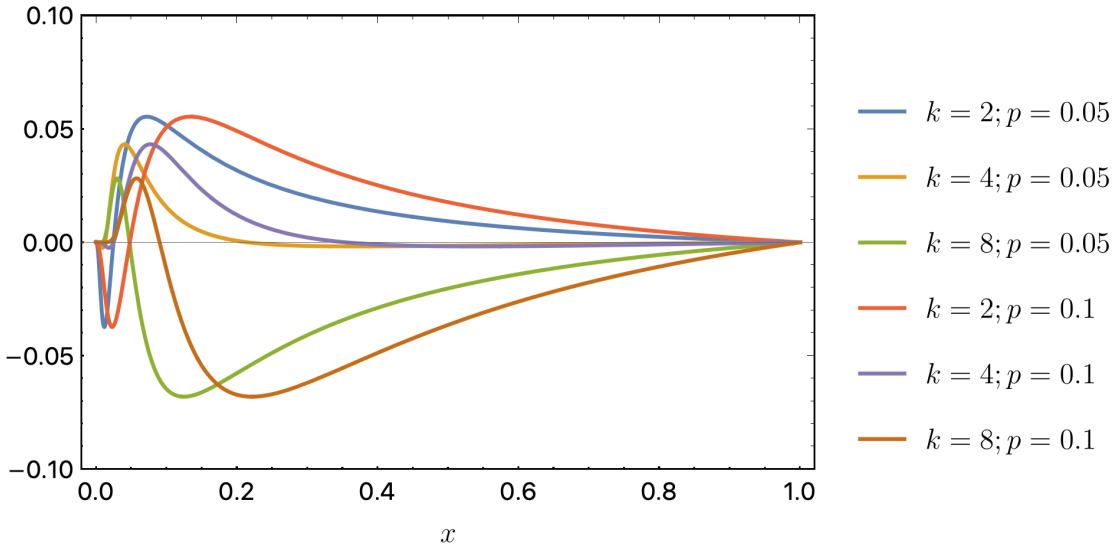

Figure 2.9: *Absolute difference between the exact and empirically approximated solutions for some selected values of k and p*

Calculation of the integral

Integrating the approximation found in the last section, given by equation (2.19), we finally find:

$$\begin{aligned} P\{U[0,1] < d_k^S\} &\approx 1 - \int_0^1 \exp\left(-\frac{p(1-x)}{kx}[1 + \log(\Gamma(k+1,1))]\right) dx \\ &= -\frac{p(\log(\Gamma(k+1,1)) + 1)e^{\frac{p(\log(\Gamma(k+1,1))+1)}{k}} \mathrm{Ei}\left(-\frac{p(\log(\Gamma(k+1,1))+1)}{k}\right)}{k} \\ &= z\exp(z)\mathrm{Ei}(-z) \end{aligned}$$

for $z = p(\log(\Gamma(k+1,1)) + 1)/k$ and where Ei stands for the exponential integral function, defined as $\mathrm{Ei}(z) = \int_{-z}^{\infty} \exp(-t)/t dt$.

Chapter 3

Results

In this final chapter of the first part we present answers to the questions introduced in the first chapter; in other words, we analyse if the presented algorithm provides:

1. a negligible probability of successfully being attacked by a malicious actor
2. a reasonable message overhead
3. a reasonable distribution of the numbers of peers of a node

We start by proving the following lemma:

Lemma 3.0.1. *Let $\{p_i\}_{i\in\mathbb{N}}$ be any sequence of i.i.d random variables with values over the interval $[0,1]$ and let $X_i \sim Ber(p_i)$. Then, the distribution of the number of trials needed to find the first success of the sequence $\{X_i\}_{i\in\mathbb{N}}$ will be $Geo(E(p_1))$.*

Proof. If $\{p_i\}_{i\in\mathbb{N}}$ are i.i.d random variables with values over the interval $[0,1]$, then the joint distribution of $\{p_i\}_{i\in\mathbb{N}}$ will be

$$f_{p_1,p_2,\ldots,p_n}(x_1,x_2,\ldots,x_n) = \prod_{i=1}^{n} f_{p_i}(x_i)$$

Now let $X_i \sim \text{Ber}(p_i)$ and $j \in \mathbb{N}\cup\infty$ the random variable that denotes the number of Bernoulli trials until the first success (we say that $j=\infty$ if all Bernoulli trials are failures). The variable j will have the following distribution:

$$\begin{aligned}
P\{j=n\} &= \int_{x_1,x_2,\ldots,x_n\in[0,1]^n} P\{j=n|p_i=x_i \text{ for } i=1,\ldots,n\} f_{p_1,p_2,\ldots,p_n}(x_1,x_2,\ldots,x_n) \\
&= \int_{x_1,x_2,\ldots,x_n\in[0,1]^n} (1-x_1)(1-x_2)\ldots(1-x_{n-1})x_n \prod_{i=1}^{n} f_{p_i}(x_i) \\
&= \int_{x_n\in[0,1]} x_n f_{p_n}(x_n) \prod_{i=1}^{n-1} \int_{x_i\in[0,1]} (1-x_i) f_{p_i}(x_i) \\
&= E(p_n)\prod_{i=1}^{n-1}(1-E(p_i)) = E(p_1)(1-E(p_1))^{n-1} = P\{\text{Geo}(E(p_1))=n\}
\end{aligned}$$

that completes our proof. □

3.1 The life cycle of a connection

Now we analyze the life cycle of a connection; in other words, we calculate how long it takes to a connection to being established, for how long this connection will last, among other properties of the present problem.

Proposition 3.1.1. *On the stationary regime, the number of required trials n_{acc} before acceptance behaves as a geometric distribution with parameter $p + (1-p)E(d_k^S)$.*

Proof. Suppose that, at each time step $i \in \mathbb{N}^+$, a node N requests a connection to one of the other nodes, let us say N_i. Also, let $\hat{d}_k(i)$, for $i \in \mathbb{N}^+$, be the acceptance metric of the farthest connection to N_i at each time step i. Recall that, on the stationary regime, all the connections' acceptance metrics distributions behave as the stationary distribution d_i^S calculated in the last chapter, thus $\hat{d}_k(i) \sim d_k^S$ for all $i \in \mathbb{N}^+$.

At each time step, the connection will be accepted with a probability $p+(1-p)\hat{d}_k(i)$, since this event will occur if and only if one of the two following disjoints events is true:

- A connection of the accepting node N_i is dropped by its requester (which has a probability p).
- None of the connections of the accepting node N_i is dropped by its requester but the acceptance metric X of the new request is smaller than $\hat{d}_k(i)$ (which has a probability $(1-p)\hat{d}_k(i)$).

Then, by Lemma 3.0.1, the number of requests needed until being accepted will be a random variable distributed as a geometric $\text{Geo}(p + (1-p)E(d_k^S))$. □

Corollary 3.1.2. *On average, a node will send $m = E(n_{acc}) = \frac{1}{p+(1-p)E(d_k^S)}$ messages to get a request accepted.*

Corollary 3.1.3. *If we assume that the nodes respond instantly to the requests (which means that a node will receive the answer to each request in $2h$ units of time after the query, where h is the network delay), then the time t_{acc} until acceptance will be given by $t_{acc} \sim 2hGeo(p+(1-p)E(d_k^S))$, that has an average value of $E(t_{acc}) = \frac{2h}{p+(1-p)E(d_k^S)} = 2hm$.*

Proposition 3.1.4. *The number of steps n_{sur} for which a connection will survive can be approximated by:*

$$n_{sur} \sim Y\,Geo\left(\frac{p+(1-p)E(d_k^S)}{k}\right) + (1-Y)\,Geo\left(\frac{p}{k} + (1-p)\frac{1+E(d_k^S)}{2}\right)$$

where $Y \sim Ber\left(\frac{E(d_k^S)}{E(d_k^S)+p(1-E(d_k^S))}\right)$.

Proof. In the last proposition, we could assume that $\hat{d}_k(i)$ was distributed as d_k^S because we were picking nodes at random and looking at its farthest connection. Nevertheless, when calculating the survival probability we assume that a connection has just been established, so we are not looking at the whole space of events anymore. Then, we cannot just say that a connection will not survive with probability $p + (1-p)d_k^S$, because d_k^S is not the distribution of the farthest connection after one acceptance.

To calculate the probability of not surviving after one trial, we divide the event "being accepted" into two disjoint events (here we call X the acceptance metric of the new connection and d_k^0 the acceptance metric of the farthest node just before the request has taken place):

- E_1: being accepted and $X < d_k^0$: This event has a probability d_k^0, since $X < d_k^0$ always implies being accepted.
- E_2: being accepted and $X > d_k^0$: In that case, we know that a connection has been dropped by its requester. Then, this event has a probability $p(1-d_k^0)$.

First, for the sake of simplicity, we assume that, given that a connection was just established, it will be the i^{th} closest connection to the node (for $i = 1, \dots, k$) with a constant probability[1] of $1/k$. Then, in the first case, we know that the probability of not surviving for the first trial will be around $\frac{p+(1-p)d_k^0}{k}$. In the second case, the probability of not surviving for the first trial will be $\frac{p}{k} + (1-p)X$.

Now, for the sake of simplification, we assume that the probability of survival for one trial is constant over time[2]. Then, the number of time steps n_1 for which the connection will survive given E_1 is given by

$$n_1 \sim \text{Geo}\left(\frac{p + (1-p)d_k^0}{k}\right) \sim \text{Geo}\left(\frac{p + (1-p)E(d_k^S)}{k}\right)$$

where the second equivalence is an implication of Lemma 3.0.1. Additionally (and analogously), the number of steps n_2 for which the connection will survive given E_2 is given by

$$n_2 \sim \text{Geo}\left(\frac{p}{k} + (1-p)X\right) \sim \text{Geo}\left(\frac{p}{k} + (1-p)E(X)\right) \sim \text{Geo}\left(\frac{p}{k} + (1-p)\frac{1 + E(d_k^S)}{2}\right)$$

Finally, the number of steps n_{sur} for which a connection will survive will be given by (here, we say $Z \sim \text{Ber}\left(\frac{d_k^0}{d_k^0 + p(1-d_k^0)}\right)$ and approximate Z by $Y \sim \text{Ber}\left(\frac{E(d_k^S)}{E(d_k^S) + p(1-E(d_k^S))}\right) \sim \text{Ber}(E(d_k^S)m)$):

$$\begin{aligned} n_{\text{sur}} &\sim Y\text{Geo}\left(\frac{p + (1-p)E(d_k^S)}{k}\right) + (1-Y)\text{Geo}\left(\frac{p}{k} + (1-p)\frac{1 + E(d_k^S)}{2}\right) \\ &\sim Y\text{Geo}\left(\frac{1}{km}\right) + (1-Y)\text{Geo}\left(\frac{1}{2m} + \frac{1}{2} + \frac{p(1-k)}{k}\right) \end{aligned}$$

□

Corollary 3.1.5. *On average, a connection will survive for a number $E(n_{sur})$ of time steps, with $E(n_{sur})$ given by:*

$$E(n_{sur}) = E(d_k^S)km^2 + \frac{2(1 - E(d_k^S))kpm}{(E(d_k^S) + 1)k(1-p) + 2p}$$

Corollary 3.1.6. *A connection will survive for $t_{sur} = \frac{n_{sur}}{\lambda}$ units of time, where λ is the average rate of incoming requests of a node (then, on average, the survival time will be $E(t_{sur}) = \frac{E(n_{sur})}{\lambda}$) units of time.*

Corollary 3.1.7. *The total lifetime LT of a connection will be*

$$LT = 2hn_{acc} + \frac{n_{sur}}{\lambda} \tag{3.1}$$

Corollary 3.1.8. *The average proportion of connected time $prop_{conn}$ of a connection will be given by*

$$\begin{aligned} prop_{conn} &= \frac{E(t_{surv})}{E(LT)} = \frac{E(n_{sur})}{2h\lambda E(n_{acc}) + E(n_{sur})} \\ &= \left(\frac{2h\lambda}{k}\frac{1}{E(d_k^S)m + (1 - E(d_k^S)m)\frac{2}{k+mk+2mp(1-k)}} + 1\right)^{-1} \end{aligned} \tag{3.2}$$

Proposition 3.1.9. *The average rate λ of messages received by a node will be*

$$\lambda = (1 - E(d_k^S)m)(1-p)\frac{k}{2h}\frac{(E(d_k^S)(k-2) + k)}{(E(d_k^S) + 1)k(1-p) + 2p} \tag{3.3}$$

[1] Note that this is not true in general, but it is only an assumption made to ease the calculations.

[2] Again, this is not true in general, but it is only an assumption made for the sake of simplicity.

Proof. The average rate of sent messages λ_S will be

$$\lambda_S = \frac{mk}{E(LT)} = \frac{1}{\frac{2h}{k} + \frac{E(n_{\text{sur}})}{mk\lambda}}$$

By symmetry, we have that $\lambda = \lambda_S$ and then (since we want the positive solution to the equation above):

$$\lambda = \frac{km - E(n_{\text{sur}})}{2hm} = (1 - E(d_k^S)m)(1-p)\frac{k}{2h}\frac{(E(d_k^S)(k-2)+k)}{(E(d_k^S)+1)k(1-p)+2p}$$

□

Proposition 3.1.10. *The probability p of one connection being dropped by its requester, at each time step, is given by the solution*[3] *to the equation:*

$$\frac{T}{h}p = \hat{f}(k, p, E(d_k^S)) \approx f(k,p) \tag{3.4}$$

where

$$\hat{f}(k,p,E(d_k^S)) = \frac{2h}{T}(p + (1-p)E(d_k^S))\frac{(E(d_k^S)+1)k(1-p)+2p}{(1-E(d_k^S))(1-p)p(E(d_k^S)(k-2)+k)}$$

and

$$f(k,p) = \frac{4(k+1)^4\left(\frac{\left(k^2+2k+2\right)(1-p)}{(k+1)^2} + 2\sqrt{p}\right)\left(\frac{k\left(k^2+2k+2\right)\sqrt{p}(1-p)}{2(k+1)^2} + k(1-p) + 2p\right)}{(p-1)\sqrt{p}\left(4\left(k^2+2k+2\right)(k+1)^2\sqrt{p} + (k-2)\left(k^2+2k+2\right)^2 p - 4k(k+1)^4\right)}$$

Proof. A node will always drop each of its requesting connection once at each period before expiration of T seconds, i.e., after an average of λT events of the Poisson process. Then, by symmetry, the probability of one of all the accepted connections of a node being dropped by its requester will $\frac{k}{T\lambda}$ at each time step. Then, the probability p will given by

$$\begin{aligned} p(k,h,T,E(d_k^S)) &= \frac{k}{T\lambda} = \frac{2h}{T}\frac{(E(d_k^S)+1)k(1-p)+2p}{(1-E(d_k^S)m)(1-p)(E(d_k^S)(k-2)+k)} \\ &= \frac{2h}{T}(p+(1-p)E(d_k^S))\frac{(E(d_k^S)+1)k(1-p)+2p}{(1-E(d_k^S))(1-p)p(E(d_k^S)(k-2)+k)} \end{aligned}$$

Substituting the approximation (2.18) (i.e., $E(d_k^S) \approx \frac{k^2+2k+2}{2(k+1)^2}\sqrt{p}$) found in the last chapter, we have:

$$p(k,h,T) \approx \frac{h}{T}f(k,p)$$

where:

$$f(k,p) = \frac{4(k+1)^4\left(\frac{\left(k^2+2k+2\right)(1-p)}{(k+1)^2} + 2\sqrt{p}\right)\left(\frac{k\left(k^2+2k+2\right)\sqrt{p}(1-p)}{2(k+1)^2} + k(1-p) + 2p\right)}{(p-1)\sqrt{p}\left(4\left(k^2+2k+2\right)(k+1)^2\sqrt{p} + (k-2)\left(k^2+2k+2\right)^2 p - 4k(k+1)^4\right)}$$

[3] If this equation does not have a solution in the interval $p \in [0,1]$, then the parameters of the problem do not define a realistic system.

Figure 3.1 depicts the function $f(k,p)$ for some selected values of k. Notice that the function $f(k,p)$

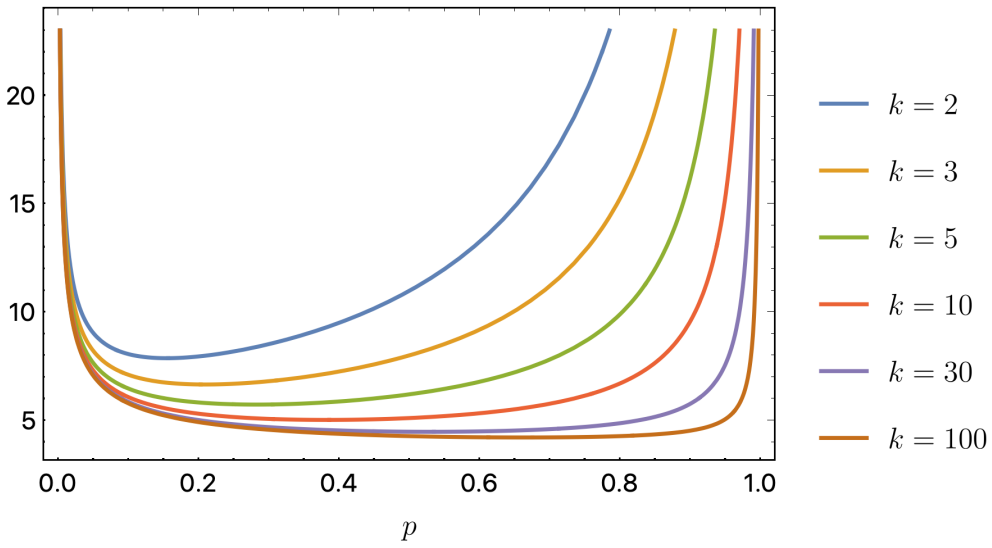

Figure 3.1: *Values of function $f(k,p)$ for some selected values of k*

is decreasing in k, for each fixed p. Also, recall that we assumed that the probability p was small; then, we are interested on the solutions for this equation on the left part of the Figure above. Since $f(k,p)$ is decreasing in k, by continuity, we know that all the possible solutions to equation (3.4) will be determined by the intersection of the line $\frac{T}{h}p$ and some curve in the region defined by the graphs of $f(2,p)$ and $\lim_{k\to+\infty} f(k,p)$, depicted in Figure 3.2.

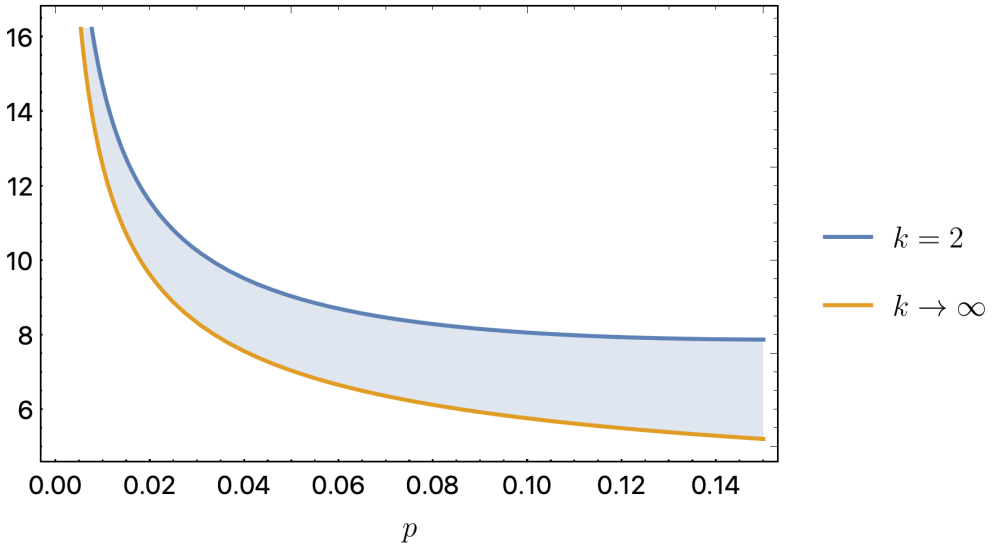

Figure 3.2: *Region defined by the graphs of the functions $f(2,p)$ and $\lim_{k\to+\infty} f(k,p)$, for small p*

Finally, Table 3.1 shows the solution to our model's equation, for $k = 2$, $k \to \infty$ and some values of T/h. For any k, the solution to equation (3.4) will be in the interval defined by the two values presented in the table. □

Proposition 3.1.11. *Each node will have a number of $N_c \sim Bin(k, prop_{conn})$ of connections, with an average value of:*

$$E(N_c) = k\left(\frac{2h\lambda}{k}\frac{1}{E(d_k^S)m + (1-E(d_k^S)m)\frac{2}{k+mk+2mp(1-k)}} + 1\right)^{-1} \tag{3.5}$$

T/h	Solution for $k=2$	Solution for $k\to\infty$	T/h	Solution for $k=2$	Solution for $k\to\infty$
10	-	0.427262	1000	0.013261	0.0117528
20	-	0.235953	2000	0.00797921	0.00716711
30	0.27873	0.168779	3000	0.00595767	0.00538295
40	0.198655	0.133674	4000	0.00485123	0.00439873
50	0.157395	0.111818	5000	0.00414045	0.00376335
60	0.131609	0.0967768	6000	0.00363974	0.00331421
70	0.113789	0.0857299	7000	0.00326512	0.00297728
80	0.100663	0.0772365	8000	0.00297267	0.00271371
90	0.0905524	0.0704807	9000	0.00273704	0.00250099
100	0.082502	0.0649643	10000	0.00254248	0.0023251
200	0.046002	0.038315	20000	0.00156988	0.00144213
300	0.0332415	0.0283007	30000	0.00118651	0.00109222
400	0.0265434	0.02288	40000	0.000973495	0.000897284
500	0.0223511	0.0194249	50000	0.000835312	0.000770607
600	0.0194516	0.0170055	60000	0.000737274	0.000680622
700	0.0173122	0.0152041	70000	0.000663522	0.000612863
800	0.0156604	0.0138034	80000	0.000605691	0.000559692
900	0.0143416	0.0126787	90000	0.000558924	0.000516666

Table 3.1: *Relevant solutions on the range* $[0,1]$ *for equation* (3.4) *and* $k=2$ *and* $k\to\infty$

Proof. Recall that each connection will not be realized all the time; in fact, a connection will only be realized for a fraction $\text{prop}_{\text{conn}}$ (given by equation (3.2)) of its life time. Then, when picking a (uniformly) random point in time, there will be a probability $\text{prop}_{\text{conn}}$ of this connection being realized. Since the connections of a node are all independent, the number of connections of a node will be given by $N_{\text{c}} \sim \text{Bin}(k, \text{prop}_{\text{conn}})$, that has an average as given by this proposition. □

Proposition 3.1.12. *Suppose an attacker can -by trial and error- obtain N non suspicious requests, i.e. their requests metrics are reasonably small. The probability of successfully initiating at least i connections with a certain node will approximately be:*

$$P_{EC}(N,i,k) = P\left\{Bin\left(N, \frac{k-i+1}{k}E(d_k)\right) \geq i\right\} \tag{3.6}$$

Proof. Suppose an attacker can mine N non suspicious requests. In order to successfully initiate at least i connections with a certain node, the requests will have to present at least i accepting metrics smaller than d_{k-i+1}. The probability of success will be approximately:

$$P_{\text{EC}}(N,i,k) = P\left\{\text{Bin}\left(N, E(d_{k-i+1})\right) \geq i\right\}$$

Since the expression above is not easily calculated, we approximate $E(d_{k-i+1}) \approx \frac{k-i+1}{k}E(d_k)$, resulting in the probability in this proposition. □

3.1.1 A Real Life Example

Finally, in this section we analyze the problem applied to a real world example. We base our parameters setting in the ones used in the IOTA[4] network.

We estimate the network delay in about 5 seconds, and the time until expiration of the connections in 30 minutes. Applying equations (3.4) and (2.18) we find some values for p and $E(d_k^S)$, for k between two and ten, which are shown in the following Table 3.2:

With the values above, we can find all other parameters, as m, λ (eq (3.3)), LT (eq (3.1)), $\text{prop}_{\text{conn}}$) (eq (3.2)) as N_{c} (eq (3.5)), whose values are represented in the following Table 3.3. In

[4] see www.iota.org

k	p	$E(d_k^S)$
2	0.02881	0.09429
3	0.02708	0.08742
4	0.02633	0.08437
5	0.02592	0.08273
6	0.02566	0.08172
7	0.02549	0.08107
8	0.02537	0.08062
9	0.02528	0.08029
10	0.0252	0.08002

Table 3.2: *Values of p and $E(d_k^S)$, for values of $k = 2, \ldots, 10$ and $\frac{T}{h} = 360$*

the IOTA network, we want to establish a total of eight connections (i.e., we want an effective k of four), so we want to aim at $k = 5$ connections.

k	m	λ	LT	$\text{prop}_{\text{conn}}$	N_{c}
2	8.30629	0.03856	430.74739	0.80716	1.61433
3	8.91781	0.06152	434.86393	0.79492	2.38478
4	9.21775	0.08437	436.96713	0.78905	3.15620
5	9.38878	0.10714	438.12675	0.78570	3.92853
6	9.49747	0.12983	438.89427	0.78360	4.70162
7	9.56950	0.15248	439.28760	0.78215	5.47511
8	9.62023	0.17510	439.51162	0.78111	6.24892
9	9.65776	0.19769	439.66058	0.78033	7.02302
10	9.68881	0.22023	439.92987	0.77976	7.79764

Table 3.3: *Values of m, λ, LT, $prop_{conn}$ and N_c, for values of $k = 2, \ldots, 10$ and $\frac{T}{h} = 360$*

3.2 Final remarks

In this work, we modeled the governing equations of the auto peering stochastic process. We found the solution to the equations for two different cases: to what we called the transient regime of the particular case with $p = 0$, and to the stationary regime of the general problem with $p \geq 0$. We also proved that the solution of the transient regime problem, for the particular case $p = 0$ will be a bound to the solution of the transient regime of the general problem. Nevertheless, even though the relation between the transient regime and the stationary regime can be intuitive, we did not prove the convergence to the stationary regime of the problem yet.

Let $\mathcal{S}$ be the space of possible events of the problem (i.e., each vector $\vec{d}(i) = (d_1(i), \ldots, d_k(i))$ belongs to $\mathcal{S}$). For any $\vec{d}(i)$, there is a strictly positive probability that the process will reach a certain open set $B \subset \sigma$ with positive Lebesgue measure, where σ is the Borel σ-algebra of $\mathcal{S}$. In fact, notice that any $B \subset \sigma$ will have a subset with positive measure $\hat{B} = (a_1, b_1) \times \cdots \times (a_k, b_k) \subset \sigma$ with (a_i, b_i), $i = 1, \ldots, k$ being disjoint intervals. The probability of reaching B in exactly k steps, starting from any $\vec{d}(i)$, will be never smaller than the probability of every connection being dropped by its requesters and the new connections belonging to $\hat{B}$; in other words, if $p > 0$:

$$P\{\vec{d}(i+k) \in B | \vec{d}(i)\} \geq P\{\vec{d}(i+k) \in \hat{B} | \vec{d}(i)\} \geq k! \left(\frac{p}{k}\right)^k \prod_{i=1}^{k} (b_i - a_i) > 0$$

Then, the probability of **not** reaching B in k steps will be smaller than one, and the probability of never reaching B will be zero. This way, the Markov chain defined by this problem will be

irreducible and recurrent. Also, the Markov chain cannot be periodic (with period larger than one), since, for any $p < 1$ there is a positive probability of returning to any state in only one step of time (recall that the connections can stay the same for some time steps). Last, for $p > 0$, the mean recurrence time M_B of B will be bounded by:

$$M_B \leq kE\left[\text{Geo}(P\{\vec{d}(i+k) \in B|\vec{d}(i)\})\right] \leq kE\left[\text{Geo}\left(k!\left(\frac{p}{k}\right)^k \prod_{i=1}^{k}(b_i - a_i)\right)\right]$$
$$\leq \frac{k}{k!\left(\frac{p}{k}\right)^k \prod_{i=1}^{k}(b_i - a_i)} < \infty$$

that implies the ergodicity of the chain. Then, from π-almost every initial condition $\vec{d}(0)$, the system will converge to the stationary distribution π given by the derivative of equation (2.14).

Let us suppose that the convergence velocity is fast enough, meaning that the model is indeed adequate. Then, we conclude that the nodes should aim at a number of connections not that larger than the expected ones (compare the first and last columns of Table 3.3). In the current practical case presented, the nodes should aim at only five connections instead of four to have a number of requested realized connections given by a $\text{Bin}(5, 0.78570)$.

The message overhead also presented a reasonable behaviour (see the second columns of Table 3.3), with a influx of messages at the nodes around 40 messages/LT.

Additionally, the probability of a node getting eclipsed in our practical scenario will be given by[5]:

$$P_{\text{EC}}(N, 4, 5) = P\left\{\text{Bin}\left(N, 0.0331\right) \geq 4\right\} \tag{3.7}$$

whose numerical values are shown in Figure 3.3. Notice that, to have a reasonably large probability, the attacker must be able to mine around a hundred non suspicious requests and still would be susceptible to being dropped by the target.

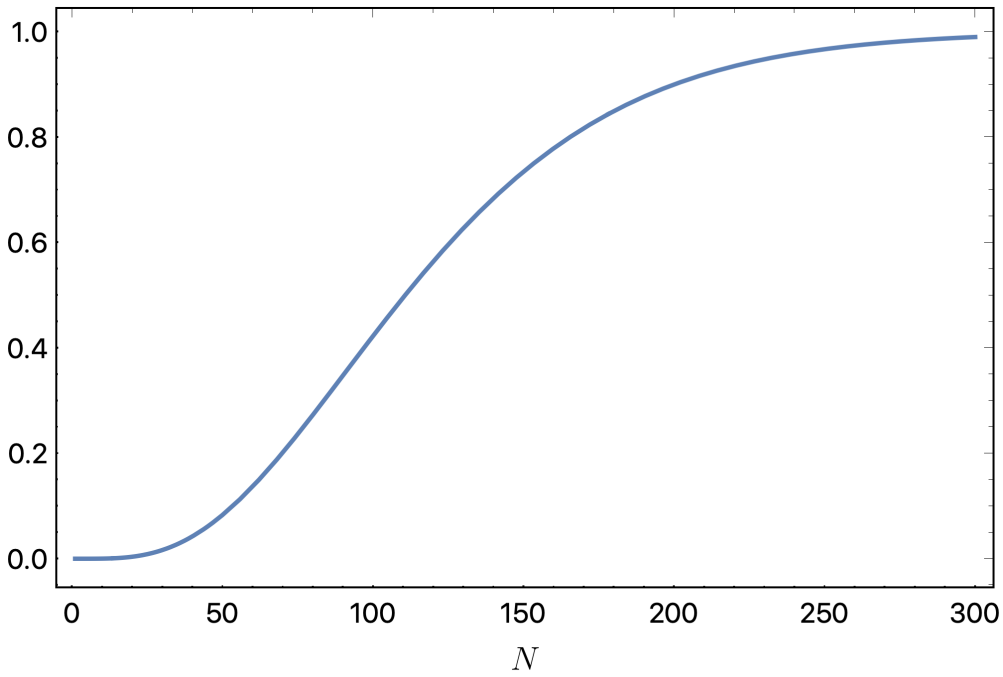

Figure 3.3: *Values of the eclipsing probability* $P_{EC}(N, 4, 5)$

[5]Notice that we need to aim at $k = 5$ to get an average value of four connections. To eclipse a node, i.e., to have more than half of all its connections (accepted and requested) malicious. Since the malicious actor does not have control about the connections that the target initiate, he must control at least four of the five accepted connections of the target.

Finally, we conclude that the system will be safe against eclipsing if and only if the price of mining one non suspicious request is reasonably high. If mining requests comes for free, any malicious node try can to eclipse a target with virtually no cost. Therefore -in the case where mining new requests is tied to some scarce (and hence valuable) resource-, the present auto peering scheme presents the desired properties introduced in the first chapter.

3.2.1 Open questions

Despite having proved that the problem indeed converges to the stationary regime problem solved, it is still not clear how long does it take to convergence. If after λT time steps the problem is still too far from the stationary regime solution, the model might not be that precise. This assumption might be addressed in future studies.

Additionally, future studies could investigate the effects on the properties of the system by changing the fixed value k for a random variable. More specifically, with a random variable k depending on the total number of connections of the system it could be possible to stabilize the variability over time of the total number of edges of the realized connection's graph.

Part II

Part Two - Nash Equilibria and Tip Selection Algorithms

Chapter 4

Equilibria in the Tangle

In this chapter, we present the paper "Equilibria in the Tangle" (from Serguei Popov, Olivia Saa and Paulo Finardi) [PSF19]. The objective of this article is to study the *Tangle* stochastic process from a game theoretical point of view. In other words, we define a game where each player pl_i attaches a new vertex (with some additional rules that are going to be later explained) to a certain subset of vertices of a graph, following independent Poisson processes of rate λ_i. We call the rules that these players use the *default strategy*. We also define a certain cost function to each new vertex, related to the probability of having future new vertices attached to it, when using the default strategy. Then, if a small subset of players (called *selfish players*) decide not to follow the default strategy, they will have a different average cost than the other players, depending on which strategy they choose. We prove the existence of "almost symmetric" Nash equilibria for this system and present simulations showing that the selfish players will not have sufficient incentives to use a strategy considerably different that the default one.

4.1 Article - Equilibria in the Tangle

We analyse the Tangle — a DAG-valued stochastic process where new vertices get attached to the graph at Poissonian times, and the attachment's locations are chosen by means of random walks on that graph. These new vertices, also thought of as "transactions", are issued by many players (which are the nodes of the network), independently. The main application of this model is that it is used as a base for the IOTA cryptocurrency system[1]. We prove existence of "almost symmetric" Nash equilibria for the system where a part of players tries to optimize their attachment strategies. Then, we also present simulations that show that the "selfish" players will nevertheless cooperate with the network by choosing attachment strategies that are similar to the "recommended" one.

Keywords: random walk, Nash equilibrium, directed acyclic graph, cryptocurrency, tip selection, IOTA

AMS 2010 subject classifications: Primary 91A15. Secondary 60J20, 68M14.

4.2 Introduction

In this paper we study *the Tangle*, a stochastic process on the space of (rooted) Directed Acyclic Graphs (DAGs). This process "grows" in time, in the sense that new vertices are attached to the graph according to a Poissonian clock, but no vertices/edges are ever deleted. When that clock rings, a new vertex appears and attaches itself to locations that are chosen with the help of certain random walks on the state of the process in the *recent past* (this is to model the network propagation delays); these random walks therefore play the key role in the model.

[1] `http://www.iota.org/`

Random walks on random graphs can be thought of as a particular case of Random Walks in Random Environments: here, the transition probabilities are functions of the graph only, i.e., there are no additional variables, such as conductances[2] etc., attached to the vertices and/or edges of the graph. Still, this subject is very broad, and one can find many related works in the literature. One can mention the internal DLA models (e.g. [JLS14] and references therein), random walks on Erdös-Rényi graphs ([CFP17, JLS14]), or random walks on the preferential attachment graphs ([CF07]), which most closely resembles the model of this paper.

The motivation for studying the particular model presented in this paper stems from the fact that it is applied in the IOTA cryptocurrency ([Pop15]). The IOTA is an ambitious project started in 2015, it aims to provide a globally scalable system capable of processing payments and storing data. One of its distinguishing features is that it uses (nontrivial) DAGs as the primary ledger for the transactions' data[3]. This is different from "traditional" cryptocurrencies such as the Bitcoin, where that data is stored in a sequence of blocks[4], also known as *blockchain*. An important observation, which motivates the use of more general DAGs instead of blockchains is that the latter *scale* poorly. Indeed, it is not hard to see that the chain of blocks of finite size, which can only be produced at regular discrete time intervals, produces a throughput bottleneck and leads to high transaction fees that need to be paid to the miners (which is by design). Also, when the network is large, it is difficult for it to achieve consensus on which blocks are "valid" in the situations when the new blocks come too frequently. If one wants to remove the fees and allow the system to scale, the natural idea would thus be to eliminate the bottleneck and the miners.

This is, of course, easier said than done — it raises all sorts of new questions. Where should the next block/transaction/vertex be attached? Who will vet the transactions for consistency and why? How can it be secure against possible attacks? How will consensus be achieved? These questions do not have trivial answers. The paper [Pop15] presented an idea for an architecture which could potentially resolve these issues. In that system, each transaction, represented by a vertex in the graph, would approve two previous transactions it selects using a particular class of random walks. To eliminate the transaction fees, it was necessary to first eliminate the miners — after all, if one wants to design a feeless system, there cannot be a dichotomy of "miners" who serve the "simple users". This bifurcation of roles between the miners and transactors naturally leads to transaction fees because the miners have some kind of resource that others do not have and they will use this monopoly power to extract rents, in the form of transaction fees, or block rewards, or both. Therefore, to eliminate the fees, all the users would have to fend for themselves. The main principle of such a system would be "help others, and others will help you".

You can help others by approving their transactions; others can help you by approving your transactions. Let us call "tips" the transactions which do not yet have any approvals; all new-coming transactions are tips at first. The idea is that, by approving a transaction, you also indirectly approve all its "predecessors". It is intuitively clear that, to help the system progress, the incoming transactions must approve tips because this adds new information to the system. However, due to the network delays, it is not practical to *impose* that this must happen — how can one be sure that what one believes to be a tip has not already been approved by someone else maybe 0.1 seconds ago?

In any case, if everybody collaborates with everybody — only approving recent and "good" (non-contradicting) transactions, then we are in a good shape. On the other hand, for someone who only cares about themself, a natural strategy would just be to choose a couple of old transactions and approve them all the time without having to do the more cumbersome work of checking new transactions for consistency thereby adding new information to the system. If everybody behaves in this way, then no new transactions will be approved, and the network will effectively come to a halt. Thus, if we want it to work, we need to incentivize the participants to collaborate and approve each other's recent transactions. Therefore, in some sense, it is all about the incentives.

[2] this refers to the well-known relation between reversible Markov chains and electric networks, see e.g. the classical book [DS84].

[3] we also cite [Bai16, Chu16, Ler15, SLZ16] which deal with other approaches to using DAGs as distributed ledgers

[4] that is, the underlying graph is essentially $\mathbb{Z}_+$ (after discarding finite forks)

Everybody wants to be helped by others, but, not everybody cares about helping others themselves. To resolve this without having to introduce monetary rewards, we could instead think of a reward as simply not being punished by others. So, we need to slightly amend the above main principle — it now reads: "Help others, and the others will help you; however, if you choose not to help others, others will not help you either". When a new transaction references two previous transactions, it is a statement of "I vouch for these transactions which have not been vouched for before, as well as all their predecessors, and their success is tied to my success". It was suggested in [Pop15] that the Markov Chain Monte Carlo (MCMC) tip selection algorithm (more precisely, the family of tip selection algorithms) would have these properties.

This paper mainly deals with the following question: what if some participants of the network are trying to minimize their *costs* by adopting a behavior different from the "default" one? How will the system behave in such circumstances? In other words, are there enough incentives for the participants to "behave well"? To address these kinds of questions, we first provide general arguments to prove existence of "almost symmetric" Nash equilibria for the system, see Section 4.4. Although one can hardly access the explicit form of these equilibria in a purely analytical way, simulations presented in Section 4.5 show that the "selfish" players will typically still choose attachment strategies that are similar to the default one, meaning that they would prefer *cooperating* with the network rather than simply *using* it).

Let us stress also that, in this paper, we consider only "selfish" players, i.e., those who only care about their own costs but still want to use the network in a legitimate way[5]. We do not consider the case when there are "malicious" ones, i.e., those who want to disrupt the network even at a cost to themselves. We are going to treat several types of attacks against the network in the subsequent papers.

This paper is organized in the following way. In Section 4.3 we first introduce some notations and define the objects we are working with; then, in Section 4.3.1 we describe the "recommended" algorithm of how the nodes choose where to attach a new transaction, and then discuss some basic properties of it, also formulating an open problem about the asymptotic behavior of the total number of tips. Section 4.4 contains the main "theoretical" advances of this paper. There, we first discuss what is a "strategy" that could be used by a selfish player, and then (Section 4.4.1) make some further assumptions necessary to formulate our main results (which are placed in Section 4.4.2). Then, we prove these results in Section 4.4.3.

Section 4.5 discusses some simulation results, mainly in the case where the selfish players try to use a very natural "greedy" attachment strategy (Section 4.5.1). In Section 4.6 one will find conclusions and some final remarks.

4.3 Description of the model

In the following we introduce the mathematical model describing the Tangle ([Pop15]).

Let $\mathrm{card}(A)$ stand for the cardinality of (multi)set A. Consider an oriented multigraph $\mathcal{T} = (V, E)$, where V is the set of vertices[6] and E is the multiset of edges. For $u, v \in V$, we say that u *approves* v, if $(u, v) \in E$. For a vertex $v \in V$, let us denote by

$$\deg_{\mathrm{in}}(v) = \mathrm{card}\{e = (u_1, u_2) \in E : u_2 = v\},$$
$$\deg_{\mathrm{out}}(v) = \mathrm{card}\{e = (u_1, u_2) \in E : u_1 = v\}$$

the "incoming" and "outgoing" degrees of the vertex v (counting the multiple edges). In the following, we refer to multigraphs simply as graphs. We use the notation $\mathcal{A}(u)$ for the set of the vertices approved by u. We say that $u \in V$ *references* $v \in V$ if there is a sequence of sites $u = x_0, x_1, \ldots, x_k = v$ such that $x_j \in \mathcal{A}(x_{j-1})$ for all $j = 1, \ldots, k$, i.e., there is a directed path from u to v. If $\deg_{\mathrm{in}}(w) = 0$ (i.e., there are no edges pointing to w), then we say that $w \in V$ is a *tip*.

[5] i.e., want to issue valid transactions and have them confirmed by the rest of the network

[6] one can think of vertices as transactions

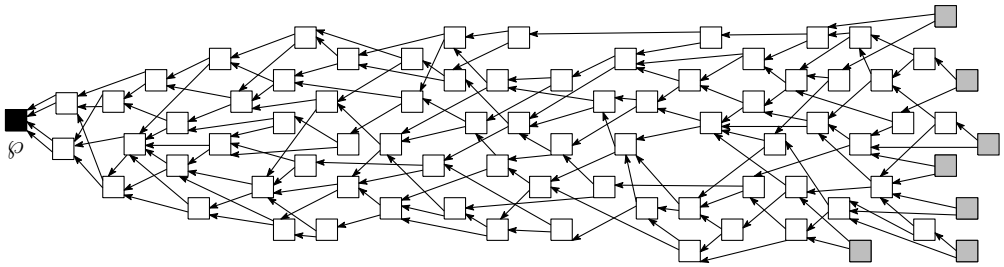

Figure 4.1: *On the DAGs we are considering: the genesis vertex is on the left, and the tips are grey*

Let $\mathcal{G}$ be the set of all directed acyclic graphs (also known as DAGs, that is, oriented graphs without cycles) $G = (V, E)$ with the following properties (see Figure 4.1).

- The graph G is finite and the multiplicity of any edge is at most two (i.e., there are at most two edges linking the same vertices).
- There is a distinguished vertex $\wp \in V$ such that $\deg_{\text{out}}(v) = 2$ for all $v \in V \setminus \{\wp\}$, and $\deg_{\text{out}}(\wp) = 0$. This vertex $\wp$ is called *the genesis*.
- Any $v \in V$ such that $v \neq \wp$ references $\wp$; that is, there is an oriented path[7] from v to $\wp$.

We now describe the tangle as a continuous-time Markov process on the space $\mathcal{G}$. The state of the tangle at time $t \geq 0$ is a DAG $\mathcal{T}(t) = (V_{\mathcal{T}}(t), E_{\mathcal{T}}(t))$, where $V_{\mathcal{T}}(t)$ is the set of vertices and $E_{\mathcal{T}}(t)$ is the multiset of directed edges at time t. The process's dynamics are described in the following way:

- The initial state of the process is defined by $V_{\mathcal{T}}(0) = \wp$, $E_{\mathcal{T}}(0) = \emptyset$.
- The tangle *grows with time*, that is, $V_{\mathcal{T}}(t_1) \subset V_{\mathcal{T}}(t_2)$ and $E_{\mathcal{T}}(t_1) \subset E_{\mathcal{T}}(t_2)$ whenever $0 \leq t_1 < t_2$.
- For a fixed parameter $\lambda > 0$, there is a Poisson process of incoming *transactions*; these transactions then become the vertices of the tangle.
- Each incoming transaction chooses[8] two vertices v' and v'' (which, in general, may coincide), and we add the edges (v, v') and (v, v''). We say in this case that this new transaction was *attached* to v' and v'' (equivalently, v *approves* v' and v'').
- Specifically, if a new transaction v arrived at time t', then $V_{\mathcal{T}}(t'+) = V_{\mathcal{T}}(t') \cup \{v\}$, and $E_{\mathcal{T}}(t'+) = E_{\mathcal{T}}(t) \cup \{(v, v'), (v, v'')\}$.

Let us write

$$\mathcal{P}^{(t)}(x) = \big\{y \in \mathcal{T}(t) : y \text{ is referenced by } x\big\},$$
$$\mathcal{F}^{(t)}(x) = \big\{z \in \mathcal{T}(t) : z \text{ references } x\big\}$$

for the "past" and the "future" with respect to x (at time t). Note that these introduce a *partial order* structure on the tangle. Observe that, if t_0 is the time moment when x was attached to the tangle, then $\mathcal{P}^{(t)}(x) = \mathcal{P}^{(t_0)}(x)$ for all $t \geq t_0$. We also define the *cumulative weight* $\mathcal{H}_x^{(t)}$ of the vertex x at time t by

$$\mathcal{H}_x^{(t)} = 1 + \operatorname{card}\big(\mathcal{F}^{(t)}(x)\big); \tag{4.1}$$

that is, the cumulative weight of x is one[9] plus the number of vertices that reference it. Observe that, for any $t > 0$, if y approves x then $\mathcal{H}_x^{(t)} - \mathcal{H}_y^{(t)} \geq 1$, and the inequality is strict if and only if

[7] not necessarily unique
[8] the precise selection mechanism will be described below
[9] its "own weight"

there are vertices different from y which also approve x. Also note that the cumulative weight of any tip is equal to 1.

There is some data associated to each vertex (transaction), created at the moment when that transaction was attached to the tangle. The precise nature of that data is not relevant for the purposes of this paper, so we assume that it is an element of some (unspecified, but finite) set $\mathcal{D}$; what is important, however, is that there is a natural way to say if the set of vertices is *consistent* with respect to the data they contain[10]. When it is necessary to emphasize that the vertices of $G \in \mathcal{G}$ contain some data, we consider the *marked* DAG $G^{[\mathfrak{d}]}$ to be $(G, \mathfrak{d}) = (V, E, \mathfrak{d})$, where $\mathfrak{d}$ is a function $V \to \mathcal{D}$. We define $\mathcal{G}^{[\mathfrak{d}]}$ to be the set of all marked DAGs $(G, \mathfrak{d})$, where $G \in \mathcal{G}$.

A note on terminology: we reserve the term "node" for entities that participate in the system by issuing transactions (which are, by their turn, *vertices* of the tangle graph). That is, the "players" mentioned in Section 4.2 are nodes.

4.3.1 On attaching a new transaction to the Tangle

There is one very important detail that has not been explained, namely: how does a newly arrived transaction choose which two vertices in the tangle it will approve, i.e., what is the *attachment strategy*? Notice that, in principle, it would be good[11] for the whole system if the new transactions always prefer to select tips as attachment places, since this way more transactions would be "confirmed"[12]. In any case, it is quite clear that the appropriate choice of the attachment strategy is essential for the correct functioning of the system, whatever this could mean.

It is also important to comment that the attachment strategy of a network node is something "internal" to it; what others can see, are the *attachment choices* of the node, but the mechanism behind them need not be publicly known. For this reason, an attachment strategy cannot be *imposed* in the protocol.

We now describe a possible choice of the attachment strategy, used to determine where the incoming transaction will be attached. It is also known as the *recommended tip selection algorithm* ([Pop15]), since, due to reasons described above, the recommended nodes' behavior is always to try to approve tips. We stress again, however, that approving only tips is not imposed in the protocol, since there is usually no way to know if a node "knew" if the transaction it approved was already approved by someone else before (also, there is no way to know which approving transaction was the first).

Let us denote by $\mathcal{L}(t)$ the set of all vertices that are tips at time t, and let $L(t) = \mathrm{card}(\mathcal{L}(t))$. To model the network propagation delays, we introduce a parameter $h > 0$, and assume that at time t only $\mathcal{T}(t-h)$ is known to the entity that issued the incoming transaction. We then define the *tip-selecting random walk*, in the following way. It depends on a parameter q (the backtracking probability) and on a function f. The initial state of the random walk is the genesis $\wp$[13], and it is stopped upon hitting the set $\mathcal{L}(t-h)$. It is important to observe that $v \in \mathcal{L}(t-h)$ does not necessarily mean that v is still a tip at time t. Let $f : \mathbb{R}_+ \to \mathbb{R}_+$ be a monotone non-increasing function. The transition probabilities of the walkers are defined in the following way: the walk *backtracks* (i.e., jumps to a randomly chosen site it approves) with probability $q \in [0, 1/2)$; if y

[10] one may think that the data refers to value transactions between accounts, and consistency means that no account has negative balance as a result, and/or the total balance has not increased

[11] good in the sense described in Section 4.2

[12] we discuss the exact meaning of this later; for now, think that "confirmed" means "referenced by many other transactions"

[13] although in practical implementations one may start it in some place closer to the tips

approves $x \neq \wp$, then the transition probability $P_{xy}^{(f)}$ is proportional to $f(\mathcal{H}_x - \mathcal{H}_y)$, that is,

$$P_{xy}^{(f)} = \begin{cases} \dfrac{q}{\mathrm{card}(\mathcal{A}(x))}, & \text{if } y \in \mathcal{A}(x), \\ \dfrac{(1-q) f(\mathcal{H}_x^{(t-h)} - \mathcal{H}_y^{(t-h)})}{\sum_{z: x \in \mathcal{A}(z)} f(\mathcal{H}_x^{(t-h)} - \mathcal{H}_z^{(t-h)})}, & \text{if } x \in \mathcal{A}(y), \\ 0, & \text{otherwise;} \end{cases} \tag{4.2}$$

for $x = \wp$ we define the transition probabilities as above, but with $q = 0$. In words, the walker *backtracks* (i.e., moves one step away from the tips) with (total) probability q, and advances one step towards the tips with (total) probability $(1 - q)$ and relative weights as above. Note that the fact that $q < 1/2$ guarantees that the random walk eventually reaches a tip[14] almost surely. In what follows, we will mostly assume that $f(s) = \exp(-\alpha s)$ for some $\alpha \geq 0$. We use the notation $P^{(\alpha)}$ for the transition probabilities in this case. Intuitively, the smaller is the value of α, the more *random* the walk is[15]. It is worth observing that the case $q = 0$ and $\alpha \to \infty$ corresponds to the GHOST protocol of [SZ13] (more precisely, to the obvious generalization of the GHOST protocol to the case when a tree is substituted by a DAG).

Now, to select two tips w_1 and w_2 where our transaction will be attached, just run two independent random walks as above, and stop when on the first hit $\mathcal{L}(t-h)$. One can also require that w_1 should be different from w_2; for that, one may re-run the second random walk in the case its exit point happened to be the same as that of the first random walk. Observe that $(\mathcal{T}(t), t \geq 0)$ is a continuous-time transient Markov process on $\mathcal{G}$; since the state space is quite large, it is difficult to analyse this process. In particular, for a fixed time t, it is not easy to study the above random walk since it takes place on a *random* graph, e.g., can be viewed as a random walk in a random environment; it is common knowledge that random walks in random environments are notoriously hard to deal with.

Some motivation for choosing the attachment strategy in the above way is provided in [Pop15]. Very briefly, it encourages the nodes to choose *recent* transactions for approval (since a transaction which approved a couple of old transactions, also known as *lazy tip*, is unlikely to be chosen by the above random walk, due to the large difference in cumulative weights in the argument of f in (4.2)) and also gives protection against certain kinds of attacks (e.g., the double-spending attack).

Let $\gamma_0 \in (0, 1)$ be some number, typically close to 1. We say that a transaction is *confirmed with confidence* γ_0 if, with probability at least γ_0, the large-α random walk[16] ends in a tip which references that transaction. It may happen that a transaction does not get confirmed (even, possible, does not get approved a single time), and becomes orphaned forever. Let us define the event

$$\mathcal{U} = \{\text{every transaction eventually gets approved}\}.$$

We believe that the following statement holds true; however, we have only a heuristical argument in its favor, not a rigorous proof. In any case, it is mostly of theoretical interest, since, as explained below, in practice we will find ourselves in the situation where $\mathbb{P}[\mathcal{U}] = 0$. We therefore state it as

Conjecture 4.3.1. *It holds that*

$$\mathbb{P}[\mathcal{U}] = \begin{cases} 0, & \text{if } \int_0^{+\infty} f(s)\, ds < \infty, \\ 1, & \text{if } \int_0^{+\infty} f(s)\, ds = \infty. \end{cases} \tag{4.3}$$

[14] more precisely, reaches a vertex that the node *assumes* to be a tip

[15] physicists would call the case of small α *high temperature regime*, and the case of large α *low temperature regime* (that is, α stands for the inverse temperature)

[16] recall that the large-α random walk is "more deterministic"

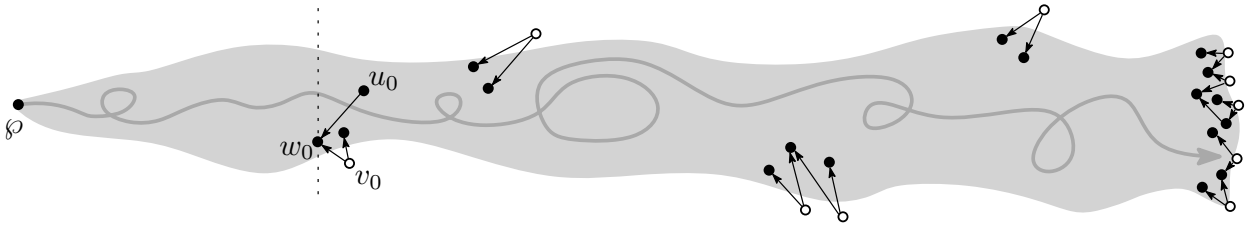

Figure 4.2: *The walk on the tangle and tip selection. Tips are circles, and transactions which were approved at least once are disks.*

Explanation. First of all, it should be true that $\mathbb{P}[\mathcal{U}] \in \{0, 1\}$ since $\mathcal{U}$ is a *tail event* with respect to the natural filtration; however, it does not seem to be very easy to prove the 0–1 law in this context – recall that we are dealing with a transient Markov process on an infinite state space. Next, consider a tip v_0 which got attached to the tangle at time t_0, and assume that it is still a tip at time $t \gg t_0$; also, assume that, among all tips, v_0 is "closest", in some suitable sense, to the genesis. Let us now think of the following question: what is the probability that v_0 will still be a tip at time $t + 1$?

Look at Figure 4.2: during the time interval $[t, t + 1)$, $O(1)$ new particles will arrive, and the corresponding walks will travel from the genesis $\wp$ looking for tips. Each of these walks will have to cross the dotted vertical segment on the picture, and with positive probability at least one of them will pass through w_0, one of the vertices approved by v_0. Assume that w_0 was already confirmed, i.e., it is connected to the right end of the tangle via some other transaction u_0 that approves w_0. Then, it is clear (but not easy to prove!) that the cumulative weight of both u_0 and w_0 should be $O(t)$, and so, when in w_0, the walk will jump to the tip v_0 with probability $f(O(t))$.

This suggests that the probability that $v_0 \in \mathcal{L}(t + 1)$ (i.e., that v_0 still is tip at time $t + 1$) is $f(O(t))$, and the Borel-Cantelli lemma[17] gives that the probability that v_0 will be eventually approved is less than 1 or equal to 1 depending on whether $\sum_n f(n)$ converges or diverges; the convergence (divergence) of the sum is equivalent to convergence (divergence) of the integral in (4.3) due to the monotonicity of the function f. A standard probabilistic argument[18] would then imply that if the probability that *a given* tip remains orphaned forever is uniformly positive, then the probability that *at least one* tip remains orphaned forever is equal to 1. □

One may naturally think that it would be better to choose the function f in such a way that, almost surely, every tip eventually gets confirmed. However, as explained in Section 4.1 of [Pop15], there is a good reason to choose a rapidly decreasing function f, because this defends the system against nodes' misbehavior and attacks. The idea is then to assume that a transaction which did not get confirmed during a sufficiently long period of time is "unlucky", and needs to be reattached[19] to the tangle. Let us fix some $K > 0$: it stands for the time when an unlucky transaction is reissued (because there is already very little hope that it would be confirmed "naturally"). We call a transaction issued less than K time units ago "unconfirmed", and if a transaction was issued more than K time units ago and was not confirmed, we call it "orphaned". In the following, we assume that the system is *stable*, in the sense that the "recent" unconfirmed transactions do not accumulate and the time until a transaction is confirmed does not depend a lot on the moment when it appeared in the system[20]. We prefer not to elaborate on the exact mathematical definition of stability here, since it requires considering a certain compactification of the space of DAGs (which essentially amounts to considering DAGs with "genesis at minus infinity"), but, hopefully, the idea is intuitively clear anyway.

[17] to be precise, a bit more refined argument is needed since the corresponding events are not independent

[18] which is also not so easy to formalize in these circumstances

[19] in fact, the nodes of the network may adopt a rule that instructs to delete the transactions that are older than K and still are tips from their databases

[20] simulations indicate that this is indeed the case when α is small ([KG18]); however, it is not guaranteed to happen for large values of α

In that stable regime, let p be the probability that a transaction is confirmed K time units after it was issued for the first time; the number of times a transaction should be issued to achieve confirmation is then a Geometric random variable with parameter p (and, therefore, with expected value p^{-1}); so, the mean time until the transaction is confirmed is K/p. Let us then recall the following remarkable fact belonging to the queuing theory, known as the Little's formula (sometimes also referred to as the Little's theorem or the Little's identity):

Proposition 4.3.2. *Suppose that λ_a is the arrival rate, μ is the mean number of customers in the system, and T is the mean time a customer spends in the system. Then $T = \mu/\lambda_a$.*

Proof. See e.g. Section 5.2 of [Coo81]. To understand intuitively why this fact holds true, one may reason in the following way: assume that, while in the system, each customer pays money to the system with rate 1. Then, at large time t, the total amount of money earned by the system would be (approximately) μt on one hand, and $T\lambda_a t$ on the other hand. Dividing by t and then sending t to infinity, we obtain $\mu = T\lambda_a$. □

Little's formula then implies[21] the following (recall that λ is the rate of the incoming transactions flow, not counting reattachments)

Proposition 4.3.3. *The average number of unconfirmed transactions[22] in the system is equal to $p^{-1}\lambda K$.*

Proof. Indeed, apply Proposition 4.3.2 with $\lambda_a = \lambda$ (think of a transaction which was reattached as a customer which returns to the server after an insuccessful service attempt; this way, the incoming flow of customers still has rate λ). As observed before, the mean time spent by a customer in the system is equal to K/p. □

When the tangle contains data, this, in principle, can make transactions incompatible between each other. In this case one may choose more sophisticated methods of tip selection. As we already mentioned[23], selecting tips with larger values of α provides better defense against attacks and misbehavior; however, smaller values of α make the system more stable with respect to the transactions' confirmation times. An example of "mixed-α" strategy is the following. Define the "model tip" w_0 as a result of the random walk with large α, then select two tips w_1 and w_2 with random walks with small α, but check that

$$\mathcal{P}^{(t-h)}(w_0) \cup \mathcal{P}^{(t-h)}(w_1) \cup \mathcal{P}^{(t-h)}(w_2)$$

is consistent.

4.4 Selfish nodes and Nash equilibria

Now, we are going to study the situation when some participants of the network are "selfish" and want to use a customized attachment strategy, in order to improve the confirmation time of their transactions (possibly at the expense of the others).

For a finite set A let us denote by $\mathcal{M}(A)$ the set of all probability measures on A, that is

$$\mathcal{M}(A) = \Big\{\mu : A \to \mathbb{R} \text{ such that } \mu(a) \geq 0 \text{ for all } a \in A \text{ and } \sum_{a\in A} \mu(a) = 1\Big\}.$$

Let

$$\mathfrak{M} = \bigcup_{G=(V,E)\in\mathcal{G}} \mathcal{M}(V \times V)$$

[21] in the language of queuing systems, a reissued transaction is a customer which goes back to the server after an unsuccessful service attempt

[22] we regard all reattachments as a single trasaction, and if one of the reattachments is confirmed, the transaction is considered confirmed

[23] recall the discussion around $f(s) = \exp(-\alpha s)$ right after (4.2)

be the union of the sets of all probability measures on the pairs of (not necessarily distinct) vertices of DAGs belonging to $\mathcal{G}$. Then, a *general mixed attachment strategy* $\mathcal{S}$ is a map

$$\mathcal{S} : \mathcal{G}^{[\mathfrak{d}]} \to \mathfrak{M} \tag{4.4}$$

with the property $\mathcal{S}(V, E, \mathfrak{d}) \in \mathcal{M}(V \times V)$ for any $G^{[\mathfrak{d}]} = (V, E, \mathfrak{d}) \in \mathcal{G}^{[\mathfrak{d}]}$; that is, for any $G \in \mathcal{G}$ with data attached to the vertices (which corresponds to the state of the tangle at a given time) there is a corresponding probability measure on the set of pairs of the vertices. Note also that in the above we considered *ordered* pairs of vertices, which, of course, does not restrict the generality.

Let $\kappa > 0$ be a fixed number. We now assume that, for a large N, there are κN nodes that follow the default tip selection algorithm, and N "selfish" nodes that try to minimize their "cost", whatever it could mean[24]. Assume that all nodes issue transactions with the same rate $\frac{\lambda}{(\kappa+1)N}$, independently. The overall rate of "honest" transactions in the system is then equal to $\frac{\lambda\kappa}{\kappa+1}$, and the overall rate of transactions issued by selfish nodes equals $\frac{\lambda}{\kappa+1}$. We also justify the assumption that the number of selfish nodes is large by observing that

- a small number of nodes that do not want to disrupt the system but just want to obtain some advantages for themselves (like e.g. faster confirmations times) are unlikely to "globally" influence the system in any considerable way, even if they do obtain those advantages for themselves;
- however, when it becomes known that it is possible to obtain advantages by deviating from the "recommended" behavior, it is reasonable to expect that a large number of independent entities would try to do it.

4.4.1 Some further assumptions and definitions

Let us now recall that, in practice, the nodes are computers running a specialized software, so they are selecting the places to attach their transactions in some algorithmic way, using limited physical resourses. In such situation, it is unrealistic to assume that a general strategy as in (4.4) could be implemented "directly", since the space $\mathcal{G}^{[\mathfrak{d}]}$ is infinite; for the same reason, even working with *simple* attachment strategies (which are maps that take an element of $\mathcal{G}^{[\mathfrak{d}]}$ as an input and produce a *deterministic* pair of its vertices as an output) is unrealistic.

Therefore, it looks like a good idea to *restrict* the strategy space we are working with. First, we consider the following simplifying assumption (which is, by the way, also quite reasonable, since, in practice, one would hardly use the genesis as the starting vertex for the random walks due to runtime issues):

Assumption L. There is $n_1 > 0$ such that the attachment strategies of all nodes (including those that use the default attachment strategy) only depend on the restriction of the tangle to the last n_1 transactions that they see.

Observe that, under the above assumption, the set of all such strategies can be thought of as a compact convex subset of $\mathbb{R}^d$, where $d = d(n_1)$ is sufficiently large.

In this section we use a different approach to model the network propagation delays: instead of assuming that an incoming transaction does not have information about the state of the tangle during last h units of time, we rather assume that it does not have information about the last n_0 transactions attached to the tangle, where $n_0 < n_1$ is some fixed positive number (so, effectively, the strategies would depend on subgraphs induced by $n_1 - n_0$ transactions, although the results of this section do not rely on this assumption). Clearly, these two approaches are quite similar in spirit; however, the second one permits us to avoid certain technical difficulties related to randomness of

[24] for example, the cost may be the expected confirmation time of a transaction (conditioned that it is eventually confirmed), the probability that it was not approved during certain (fixed) time interval, etc.; below in (4.6) we provide the exact definition of the cost function we are working with in this paper

the number of unseen transactions in the first case. Also, it will be more natural and convenient to pass from continuous to discrete time.

Now, even with the restrictions as above, it is still unrealistic to work with the simple strategies of the sort "choose a fixed pair of transactions for each possible restriction of the tangle to the set of last n_1 transactions", because implementing it in practive would require effectively dealing with sets indexed by all possible restrictions, and the size of the latter set clearly grows exponentially in n_1. Instead, as hinted in the beginning of this subsection, we think of different "attachment methods" as simple strategies. Formally, let $\mathcal{G}_{n_1}^{[\mathfrak{d}]}$ be the set of all possible sub-DAGs of $\mathcal{G}^{[\mathfrak{d}]}$ with n_1 vertices, and $\mathfrak{M}_{n_1}$ be the set of all probability measures on the vertices' pairs of elements of $\mathcal{G}_{n_1}^{[\mathfrak{d}]}$. Clearly, the set $\mathcal{G}_{n_1}^{[\mathfrak{d}]}$ is finite. An *attachment method* is then a map

$$\mathcal{J} : \mathcal{G}_{n_1}^{[\mathfrak{d}]} \to \mathfrak{M}_{n_1};$$

it is thought of as a (randomized) polynomial-time polynomial-memory algorithm which takes the last n_1 transactions and returns a pair of those transactions which would serve as attachment's locations. Then, the available simple strategies are attachment methods

$$\{\mathcal{J}_\beta, \beta \in \mathfrak{A}\},$$

where $\mathfrak{A}$ is some (unspecified) index set. It is also important to observe that this approach does not restricts generality. We then denote by $\mathcal{Q}$ the set of all *mixed* strategies of the form $\mathcal{J}_\Xi$, where Ξ is a random variable on $\mathfrak{A}$. Observe also that the set of simple strategies can be thought of as a subset of $\mathbb{R}^d$ (which we assume also to be compact), where $d = d(n_1)$ is sufficiently large, and $\mathcal{Q}$ would be then its convex hull.

Let $\mathcal{S}_1, \dots, \mathcal{S}_N \in \mathcal{Q}$ be the attachment strategies used by the selfish nodes. To evaluate the "goodness" of a strategy, one has to choose and then optimize some suitable observable (that stands for the "cost"); as usual, there are several "reasonable" ways to do this. We decided to choose the following one, for definiteness and also for technical reasons (to guarantee the continuity of a certain function used below); one can probably extend our arguments to other reasonable cost functions. Assume that a transaction v was attached to the tangle at time t_v, so $v \in V_{\mathcal{T}}(t)$ for all $t \geq t_v$. Fix some (typically large) $M_0 \in \mathbb{N}$. Let $t_1^{(v)}, \dots, t_{M_0}^{(v)}$ be the moments when the subsequent M_0 (after v) transactions were attached to the tangle. For $k = 1 \dots, M_0$ let $R_k^{(v)}$ be the event that the *default* tip-selecting walk[25] on $\mathcal{T}\left(t_k^{(v)}\right)$ stops in a tip that *does not* reference v. We then define the random variable

$$W(v) = \mathbf{1}_{R_1^{(v)}} + \dots + \mathbf{1}_{R_{M_0}^{(v)}} \tag{4.5}$$

to be the number of times that the M_0 "subsequent" tip selection random walks do not reference v (in the above, $\mathbf{1}_A$ is the indicator function of an event A). Intuitively, the smaller is the value of $W(v)/M_0$, the bigger is the chance that v is quickly confirmed.

Next, assume that $(v_j^{(k)}, j \geq 1)$ are the transactions issued by the kth (selfish) node. We define

$$\mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_N) = M_0^{-1} \lim_{n\to\infty} \frac{W(v_1^{(k)}) + \dots + W(v_n^{(k)})}{n}, \tag{4.6}$$

to be the *mean cost* of the kth node given that $\mathcal{S}_1, \dots, \mathcal{S}_N$ are the attachment strategies of the selfish nodes.

Definition 4.4.1. *We say that a set of strategies* $(\mathcal{S}_1, \dots, \mathcal{S}_N) \in \mathcal{Q}^N$ *is a* Nash equilibrium *if*

$$\mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_{k-1}, \mathcal{S}_k, \mathcal{S}_{k+1}, \dots, \mathcal{S}_N) \leq \mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_{k-1}, \mathcal{S}', \mathcal{S}_{k+1}, \dots, \mathcal{S}_N)$$

for any k *and any* $\mathcal{S}' \in \mathcal{Q}$.

[25] i.e., the one used by nodes following the default attachment strategy

Observe that, since the nodes are indistinguishable, the fact that $(\mathcal{S}_1, \dots, \mathcal{S}_N)$ is a Nash equilibrium implies that so is $(\mathcal{S}_{\sigma_1}, \dots, \mathcal{S}_{\sigma(N)})$ for any permutation σ.

4.4.2 Main results

From now on, we assume that vertices contain no data, i.e., the set $\mathcal{D}$ is empty; this is not absolutely necessary because, with the data, the proof will be essentially the same; however, the notations would become much more cumbersome. Also, there will be no reattachments; again, this would unnecessarily complicate the proofs (one would have to work with *decorated* Poisson processes). In fact, we are dealing with a so-called *random-turn game* here, see e.g. Chapter 9 of [KP17] for other examples.

Consider, for the moment, the situation when all nodes use the same attachment strategy (i.e., there are no selfish nodes). The restriction of the tangle on the last n_1 transactions then becomes a Markov chain on the state space $\mathcal{G}_{n_1}$. We now make the following technical assumption on that Markov chain:

Assumption D. The above Markov chain is irreducible and aperiodic.

It is important to observe that Assumption D is *not* guaranteed to hold for *every* natural attachment strategy; however, still, this is not a very restrictive assumption in practice because every finite Markov chain may be turned into an irreducible and aperiodic one by an arbitrarily small perturbation of the transition matrix.

Then, we are able to prove the following

Theorem 4.4.2. *Under Assumptions L and D, the system has at least one Nash equilibrium.*

Symmetric games do not always have symmetric Nash equilibria, as shown in [Fey12]. Also, even when such equilibria exist in the class of mixed strategies, they may be "inferior" to asymmetric pure equilibria; for example, this happens in the classical "Battle of the sexes" game (see e.g. Section 7.2 of [KP17]).

Now, the goal is to prove that, if the number of selfish nodes N is large, then for *any* equilibrium state the costs of distinct nodes cannot be significantly different. Let us recall the notations we use: $\mathcal{S}_1, \dots, \mathcal{S}_N$ are the strategies of the N selfish nodes, and $\mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_N)$, $k = 1, \dots, N$, are the mean costs of the selfish nodes, defined in (4.6). Now, we have the following

Theorem 4.4.3. *For any $\varepsilon > 0$ there exists N_0 (depending on the default attachment strategy) such that, for all $N \geq N_0$ and any Nash equilibrium $(\mathcal{S}_1, \dots, \mathcal{S}_N)$ it holds that*

$$\left|\mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_N) - \mathfrak{C}^{(j)}(\mathcal{S}_1, \dots, \mathcal{S}_N)\right| < \varepsilon \tag{4.7}$$

for all $k, j \in \{1, \dots, N\}$.

Now, let us define the notion of *approximate* Nash equilibrium:

Definition 4.4.4. *For a fixed $\varepsilon > 0$, we say that a set of strategies $(\mathcal{S}_1, \dots, \mathcal{S}_N) \in \mathcal{Q}^N$ is an ε-equilibrium if*

$$\mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_{k-1}, \mathcal{S}_k, \mathcal{S}_{k+1}, \dots, \mathcal{S}_N) \leq \mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_{k-1}, \mathcal{S}', \mathcal{S}_{k+1}, \dots, \mathcal{S}_N) + \varepsilon$$

for any k and any $\mathcal{S}' \in \mathcal{Q}$.

The motivation for introducing this notion is that, if ε is very small, then, in practice, ε-equilibria are essentially indistinguishable from the "true" Nash equilibria.

Theorem 4.4.5. *For any $\varepsilon > 0$ there exists N_0 (depending on the default attachment strategy) such that, for all $N \geq N_0$ and any Nash equilibrium $(\mathcal{S}_1, \dots, \mathcal{S}_N)$ it holds that $(\mathcal{S}, \dots, \mathcal{S})$ is an ε-equilibrium, where*

$$\mathcal{S} = \frac{1}{N} \sum_{k=1}^{N} \mathcal{S}^{(k)} \tag{4.8}$$

(that is, all selfish nodes use the same "averaged" strategy defined above). The costs of all selfish nodes are then equal to

$$\frac{1}{N}\sum_{k=1}^{N}\mathfrak{C}^{(k)}(\mathcal{S}_1,\dots,\mathcal{S}_N),$$

that is, the average cost in the Nash equilibrium.

In other words, for large N one can essentially assume that all selfish nodes follow the same attachment strategy. This result will be important in Section 4.5, because it makes it possible to use (practical) simulations in order to find the Nash equilibria of systems with large number of selfish players.

4.4.3 Proofs

First, we need the following technical result:

Lemma 4.4.6. *Let P be the transition matrix of an irreducible and aperiodic discrete-time Markov chain on a finite state space E. Let $\widehat{P}$ be a continuous map from a compact set $F \subset \mathbb{R}^d$ to the set of all stochastic matrices on E (equipped by the distance inherited from the usual matrix norm on the space of all matrices on E). Fix $\theta \in (0,1)$, denote $\widetilde{P}(s) = \theta P + (1-\theta)\widehat{P}(s)$, and let π_s be the (unique) stationary measure of $\widetilde{P}(s)$. Then π_s is also continuous (as a function of s).*

Proof. In the following we give a (rather) probabilistic proof of this fact via the Kac's lemma, although, of course, a purely analytic proof is also possible. Irreducibility and aperiodicity of P imply that, for some $m_0 \in \mathbb{N}$ and $\varepsilon_0 > 0$

$$P_{xy}^{m_0} \geq \varepsilon_0 \tag{4.9}$$

for all $x, y \in E$, where $P^{m_0} = (P_{xy}^{m_0}, x, y \in E)$ is the transition matrix in m_0 steps. Now, (4.9) implies that

$$\widetilde{P}_{xy}^{m_0}(s) \geq \theta^{m_0}\varepsilon_0 \tag{4.10}$$

for all $x, y \in E$ and all $s \in F$.

Being $(X_n, n \geq 0)$ a stochastic process on E, let us define

$$\tau(x) = \min\{k \geq 1 : X_k = x\}$$

(with the convention $\min\emptyset = \infty$) to be the *hitting time* of the site $x \in E$ by the stochastic process X. Now, let $\mathbb{P}_x^{(s)}$ and $\mathbb{E}_x^{(s)}$ be the probability and the expectation with respect to the Markov chain with transition matrix $\widetilde{P}(s)$ starting from $x \in E$. We now recall the Kac's lemma (cf. e.g. Theorem 1.22 of [Dur12]): for all $x \in E$ it holds that

$$\pi_s(x) = \frac{1}{\mathbb{E}_x^{(s)}\tau(x)}. \tag{4.11}$$

Now, (4.10) readily implies that, for all $x \in E$ and $n \in \mathbb{N}$,

$$\mathbb{P}_x^{(s)}[\tau(x) \geq n] \leq c_1 e^{-c_2 n} \tag{4.12}$$

for some positive constants $c_{1,2}$ which do not depend on s. This in its turn implies that the series

$$\mathbb{E}_x^{(s)}\tau(x) = \sum_{n=1}^{\infty}\mathbb{P}_x^{(s)}[\tau(x) \geq n]$$

converges uniformly in s and so $\mathbb{E}_x^{(s)}\tau(x)$ is uniformly bounded from above[26]; also, the Uniform Limit

[26] and, of course, it is also bounded from below by 1

Theorem (see e.g. Section D.6.2 of [Ok07]) implies that $\mathbb{E}_x^{(s)}\tau(x)$ is continuous in s. Therefore, for any $x \in E$, (4.11) implies that $\pi_s(x)$ is also a continuous function of s. □

Proof of Theorem 4.4.2. The authors were unable to find a result available in the literature that implies Theorem 4.4.2 directly; nevertheless, its proof is quite standard and essentially follows Nash's original paper ([Nas50]) (see also [Fin64]). There is only one technical difficulty, which we intend to address via the above preparatory steps: one needs to prove the continuity of the cost function.

Denote by $\pi_{\mathcal{S}}$ the invariant measure of the Markov chain given that the (selfish) nodes use the "strategy vector" $\mathbf{s} = (\mathcal{S}_1, \dots, \mathcal{S}_N)$. Then, the idea is to use Lemma 4.4.6 with $\theta = \frac{\kappa}{\kappa+1}$, P the transition matrix obtained from the default attachment strategy, and $\widehat{P}(s)$ is the transition matrix obtained from the strategy $\mathcal{S}' = N^{-1}\sum_{k=1}^{N}\mathcal{S}_k$ (observe that N nodes using the strategies $\mathcal{S}_1, \dots, \mathcal{S}_N$, is the same as one node with strategy $\mathcal{S}'$ issuing transactions N times faster). Assumption D together with Lemma 4.4.6 then imply that $\pi_{\mathbf{s}} := \pi_{\mathcal{S}'}$ is a continuous function of $\mathbf{s}$.

Let $\mathbb{E}_{\pi_{\mathcal{S}'}}^{\mathcal{S},\hat{\mathcal{S}}}$ be the expectation with respect to the following procedure: take the "starting" graph according to $\pi_{\mathcal{S}'}$, then attach to it a transaction according to the strategy $\mathcal{S}$, and then keep attaching subsequent transactions according to the strategy $\hat{\mathcal{S}}$ (instead of $\mathcal{S}'$ and $\hat{\mathcal{S}}$ we may also use the strategy vectors; $\mathcal{S}'$ and $\hat{\mathcal{S}}$ would be then their averages). Let also $W^{(k)}$ be the random variable defined as in (4.5) for an arbitrary transaction v issued by the kth node. Then, the Ergodic Theorem for Markov chains (see e.g. Theorem 1.23 of [Dur12]) implies that

$$\mathfrak{C}^{(k)}(\mathcal{S}) = \mathbb{E}_{\pi_{\mathcal{S}'}}^{\mathcal{S}_k,\mathcal{S}'} W^{(k)}. \tag{4.13}$$

It is not difficult to see that the above expression is a polynomial of the $\mathcal{S}$'s coefficients (i.e., the corresponding probabilities) and $\pi_{\mathcal{S}'}$-values, and hence it is a continuous function on the space of strategies $\mathfrak{M}_{n_1}$. Using this, the rest of the proof is standard, it is obtained as a consequence of the Kakutani's fixed point theorem ([Kak41]), also with the help of the Berge's Maximum Theorem (see e.g. Chapter E.3 of [Ok07]). □

Proof of Theorem 4.4.3. Without restricting generality we may assume that

$$\mathfrak{C}^{(1)}(\mathcal{S}_1, \dots, \mathcal{S}_N) = \max_{k=1,\dots,N} \mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_N),$$
$$\mathfrak{C}^{(2)}(\mathcal{S}_1, \dots, \mathcal{S}_N) = \min_{k=1,\dots,N} \mathfrak{C}^{(k)}(\mathcal{S}_1, \dots, \mathcal{S}_N),$$

so we then need to proof that $\mathfrak{C}^{(1)}(\mathbf{s}) - \mathfrak{C}^{(2)}(\mathbf{s}) < \varepsilon$, where $\mathbf{s} = (\mathcal{S}_1, \dots, \mathcal{S}_N)$. Now, the main idea of the proof is the following: if $\mathfrak{C}^{(1)}(\mathbf{s})$ is considerably larger than $\mathfrak{C}^{(2)}(\mathbf{s})$, then the owner of the first node may decide to adopt the strategy used by the second one. This would not necessarily decrease his costs to the former costs of the second node since a change in an individual strategy leads to changes in *all* costs; however, when N is large, the effects of changing the strategy of only one node would be small, and (if the difference of $\mathfrak{C}^{(1)}(\mathbf{s})$ and $\mathfrak{C}^{(2)}(\mathbf{s})$ were not small) this would lead to a contradiction to the assumption that $\mathbf{s}$ was a Nash equilibrium.

So, let us denote $\mathbf{s}' = (\mathcal{S}_2, \mathcal{S}_2, \mathcal{S}_3, \dots, \mathcal{S}_N)$, the strategy vector after the first node adopted the strategy of its "more successful" colleague, see Figure 4.3. Let

$$\mathcal{S} = \frac{1}{N}\big(\mathcal{S}_1 + \dots + \mathcal{S}_N\big) \text{ and } \mathcal{S}' = \frac{1}{N}\big(2\mathcal{S}_2 + \mathcal{S}_3 + \dots + \mathcal{S}_N\big)$$

be the two "averaged" strategies. In the following, we are going to compare $\mathfrak{C}^{(2)}(\mathbf{s}) = \mathbb{E}_{\pi_{\mathcal{S}}}^{(\mathcal{S}_2,\mathcal{S})} W^{(2)}$ (the "old" cost of the second node) with $\mathfrak{C}^{(1)}(\mathbf{s}') = \mathbb{E}_{\pi_{\mathcal{S}'}}^{(\mathcal{S}_2,\mathcal{S}')} W^{(1)}$ (the "new" cost of the first node, after it adopted the second node's strategy). We need the following

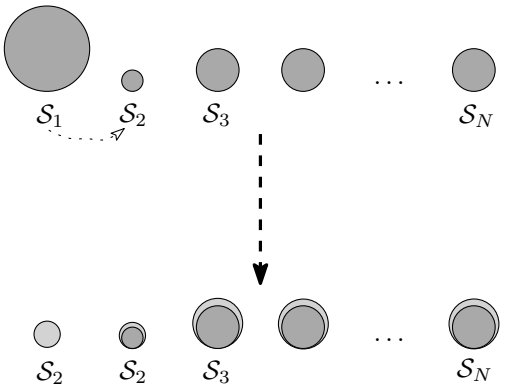

Figure 4.3: *On the main idea of the proof of Theorem 4.4.3. The node with the highest cost will switch to the strategy of the node with the lowest cost. That will not guarantee exactly that same cost to the former node, but the difference will be rather small since N is large (so the change in one component of the strategy vector will not influence a lot the outcome).*

Lemma 4.4.7. *For any measure π on $\mathcal{G}_{n_1}$ and any strategy vectors $\mathbf{s} = (\mathcal{S}_1, \dots, \mathcal{S}_N)$ and $\mathbf{s}' = (\mathcal{S}'_1, \dots, \mathcal{S}'_N)$ such that $\mathcal{S}_k = \mathcal{S}'_k$ for all $k = 2, \dots, N$, we have*

$$\left|\mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S})} W^{(j)} - \mathbb{E}_\pi^{(\mathcal{S}'_j,\mathcal{S}')} W^{(j)}\right| \le \frac{M_0}{N} \tag{4.14}$$

for all $j = 2, \dots, N$.

Proof. Let us define the event

$$A = \{\text{among the } M_0 \text{ transactions there is at least one issued by the first node}\},$$

and observe that, by the union bound, the probability that it occurs is at most M_0/N. Then, using the fact that $\mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S})}(W^{(j)}\mathbf{1}_{A^c}) = \mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S}')}(W^{(j)}\mathbf{1}_{A^c})$ (since, on A^c, the first node does not "contribute" to $W^{(j)}$), write

$$\begin{aligned}
&\left|\mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S})} W^{(j)} - \mathbb{E}_\pi^{(\mathcal{S}'_j,\mathcal{S}')} W^{(j)}\right| \\
&= \left|\mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S})}(W^{(j)}\mathbf{1}_A) + \mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S})}(W^{(j)}\mathbf{1}_{A^c}) - \mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S}')}(W^{(j)}\mathbf{1}_A) - \mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S}')}(W^{(j)}\mathbf{1}_{A^c})\right| \\
&= \left|\mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S})}(W^{(j)}\mathbf{1}_A) - \mathbb{E}_\pi^{(\mathcal{S}_j,\mathcal{S}')}(W^{(j)}\mathbf{1}_A)\right| \\
&\le \frac{M_0}{N},
\end{aligned}$$

where we also used that $W^{(j)} \le 1$. This concludes the proof of Lemma 4.4.7. □

We continue proving Theorem 4.4.3. First, by symmetry, we have

$$\mathbb{E}_{\pi_{\mathcal{S}'}}^{(\mathcal{S}_2,\mathcal{S}')} W^{(1)} = \mathbb{E}_{\pi_{\mathcal{S}'}}^{(\mathcal{S}_2,\mathcal{S}')} W^{(2)}. \tag{4.15}$$

Also, it holds that

$$\left|\mathbb{E}_{\pi_{\mathcal{S}'}}^{(\mathcal{S}_2,\mathcal{S}')} W^{(2)} - \mathbb{E}_{\pi_{\mathcal{S}'}}^{(\mathcal{S}_2,\mathcal{S})} W^{(2)}\right| \le \frac{M_0}{N} \tag{4.16}$$

by Lemma 4.4.7. Then, similarly to the proof of Theorem 4.4.2, we can obtain that the function

$$(\mathcal{S}, \mathcal{S}', \mathcal{S}'') \mapsto \mathbb{E}_{\pi_{\mathcal{S}''}}^{(\mathcal{S},\mathcal{S}')} W^{(2)}$$

is continuous; since it is defined on a compact, it is also uniformly continuous. That is, for any $\varepsilon' > 0$

there exist $\delta' > 0$ such that if $\|(\mathcal{S}, \mathcal{S}', \mathcal{S}'') - (\tilde{\mathcal{S}}, \tilde{\mathcal{S}}', \tilde{\mathcal{S}}'')\| < \delta'$, then

$$\left|\mathbb{E}^{(\mathcal{S},\mathcal{S}')}_{\pi_{\mathcal{S}''}} W^{(2)} - \mathbb{E}^{(\tilde{\mathcal{S}},\tilde{\mathcal{S}}')}_{\pi_{\tilde{\mathcal{S}}''}} W^{(2)}\right| < \varepsilon'.$$

Choose $N_0 = \lceil 1/\delta' \rceil$. We then obtain from the above that

$$\left|\mathbb{E}^{(\mathcal{S}_2,\mathcal{S})}_{\pi_{\mathcal{S}'}} W^{(2)} - \mathbb{E}^{(\mathcal{S}_2,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(2)}\right| < \varepsilon'. \tag{4.17}$$

The relations (4.15), (4.16), and (4.17) imply that

$$\left|\mathbb{E}^{(\mathcal{S}_2,\mathcal{S}')}_{\pi_{\mathcal{S}'}} W^{(1)} - \mathbb{E}^{(\mathcal{S}_2,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(2)}\right| \le \varepsilon' + \frac{M_0}{N}.$$

On the other hand, since we assumed that $\mathbf{s}$ is a Nash equilibrium, it holds that

$$\mathbb{E}^{(\mathcal{S}_2,\mathcal{S}')}_{\pi_{\mathcal{S}'}} W^{(1)} = \mathfrak{C}^{(1)}(\mathbf{s}') \ge \mathfrak{C}^{(1)}(\mathbf{s}) = \mathbb{E}^{(\mathcal{S}_1,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)}, \tag{4.18}$$

which implies that

$$\mathbb{E}^{(\mathcal{S}_1,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)} - \mathbb{E}^{(\mathcal{S}_2,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(2)} \le \varepsilon' + \frac{M_0}{N}.$$

This concludes the proof of Theorem 4.4.3. □

Proof of Theorem 4.4.5. To begin, we observe that the proof of the second part is immediate, since, as already noted before, for an external observer, the situation where there are N nodes with strategies $(\mathcal{S}_1, \dots, \mathcal{S}_N)$ is indistinguishable from the situation with one node with averaged strategy.

Now, we need to prove that, for any fixed $\varepsilon' > 0$ it holds that

$$\mathfrak{C}^{(1)}(\mathcal{S}, \dots, \mathcal{S}) \le \mathfrak{C}^{(1)}(\tilde{\mathcal{S}}, \mathcal{S}, \dots, \mathcal{S}) + \varepsilon' \tag{4.19}$$

for all large enough N (the claim would then follow by symmetry). Recall that we have

$$\mathfrak{C}^{(1)}(\mathcal{S}, \dots, \mathcal{S}) = \mathbb{E}^{(\mathcal{S},\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)}, \tag{4.20}$$

$$\mathfrak{C}^{(1)}(\mathcal{S}_1, \dots, \mathcal{S}_N) = \mathbb{E}^{(\mathcal{S}_1,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)}, \tag{4.21}$$

and

$$\mathfrak{C}^{(1)}(\tilde{\mathcal{S}}, \mathcal{S}, \dots, \mathcal{S}) = \mathbb{E}^{(\tilde{\mathcal{S}},\mathcal{S}')}_{\pi_{\mathcal{S}'}} W^{(1)}, \tag{4.22}$$

where

$$\mathcal{S}' = \frac{1}{N}\big(\tilde{\mathcal{S}} + (N-1)\mathcal{S}\big) = \frac{1}{N}\Big(\tilde{\mathcal{S}} + \frac{N-1}{N}(\mathcal{S}_1 + \dots + \mathcal{S}_N)\Big).$$

Now, the second part of this theorem together with Theorem 4.4.3 imply[27] that, for any fixed $\varepsilon > 0$

$$\left|\mathbb{E}^{(\mathcal{S},\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)} - \mathbb{E}^{(\mathcal{S}_1,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)}\right| < \varepsilon \tag{4.23}$$

for all large enough N.

Next, let us denote

$$\mathcal{S}'' = \frac{1}{N}(\tilde{\mathcal{S}} + \mathcal{S}_2 + \dots + \mathcal{S}_N).$$

Then, again using the uniform continuity argument (as in the proof of Theorem 4.4.3), we obtain that, for any $\varepsilon'' > 0$

$$\left|\mathbb{E}^{(\tilde{\mathcal{S}},\mathcal{S}')}_{\pi_{\mathcal{S}'}} W^{(1)} - \mathbb{E}^{(\tilde{\mathcal{S}},\mathcal{S}'')}_{\pi_{\mathcal{S}''}} W^{(1)}\right| < \varepsilon'' \tag{4.24}$$

[27] note that Theorem 4.4.3 implies that, when N is large, the nodes already have "almost" the same cost in the Nash equilibrium $(\mathcal{S}_1, \dots, \mathcal{S}_N)$

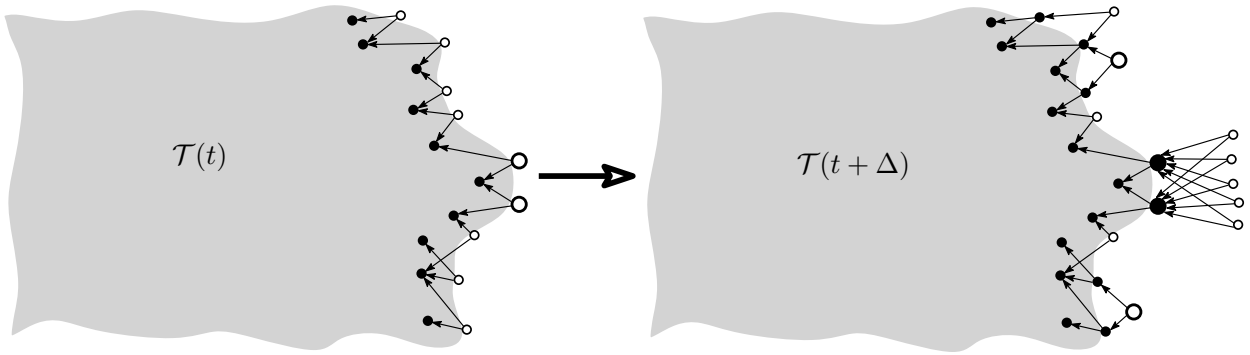

Figure 4.4: *Why the "greedy" tip selection strategy will not work (the two "best" tips are shown as larger circles).*

for all large enough N. However,

$$\mathbb{E}^{(\tilde{\mathcal{S}},\mathcal{S}'')}_{\pi_{\mathcal{S}''}} W^{(1)} = \mathfrak{C}^{(1)}(\tilde{\mathcal{S}}, \mathcal{S}_2, \dots, \mathcal{S}_N) \geq \mathfrak{C}^{(1)}(\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_N) = \mathbb{E}^{(\mathcal{S}_1,\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)},$$

since $(\mathcal{S}_1, \dots, \mathcal{S}_N)$ is a Nash equilibrium. Then, (4.23)–(4.24) imply that

$$\left|\mathbb{E}^{(\mathcal{S},\mathcal{S})}_{\pi_{\mathcal{S}}} W^{(1)} - \mathbb{E}^{(\tilde{\mathcal{S}},\mathcal{S}')}_{\pi_{\mathcal{S}'}} W^{(1)}\right| < \varepsilon + \varepsilon'',$$

and, recalling (4.20) and (4.22), we conclude the proof of Theorem 4.4.5. □

4.5 Simulations

In this section we investigate Nash equilibria between selfish nodes via simulations. As discussed in Section 4.2, this is motivated by the following important question: since the choice of an attachment strategy is not enforced, there may indeed be nodes which would prefer to "optimise" their strategies in order to decrease the mean confirmation time of their transactions. So, can this lead to a situation where the corresponding Nash equilibrium is "bad for everybody", effectively leading to the system's malfunctioning?

Due to Theorem 4.4.5 we may assume that all selfish nodes use the same attachment strategy. Even then, it is probably unfeasible to calculate that strategy exactly; instead, we resort to simulations, which indeed will show that the equilibrium strategy of the selfish nodes will not be much different from the (suitably chosen) default strategy, at least in the (very natural) situation below. But, before doing that, let us explain the intuition behind this fact. Naively, a reasonable strategy for a selfish node would be the following:

(1) Calculate the exit distribution of the tip-selecting random walk.

(2) Find the two tips where this distribution attains its "best"[28] values.

(3) Approve these two tips.

However, this strategy fails when other selfish nodes are present. To understand this, look at Figure 4.4: *many* selfish nodes attach their transactions to the two "best" tips. As a result, the "neighborhood" of these two tips becomes "overcrowded": there is so much competition between the transactions issued by the selfish nodes, that the chances of them being approved soon actually decrease[29].

To illustrate this fact, several simulations have been done. All the results depicted here were generated using (4.2) as the transition probabilities, with $q = 1/3$, and a network delay of $h = 1$ second. Also, a transaction will be reattached if the two following criteria are met:

[28] i.e., the maximum and the second-to-maximum

[29] the "new" best tips are not among them, as shown on Figure 4.4 on the right

(1) the transaction is older than 20 seconds

(2) the transaction is not referenced by the tip selected by a random walk with $\alpha = \infty$[30].

This way, we guarantee not only that the unconfirmed transactions will be eventually confirmed, but also that all transactions that were never reattached are referenced by most of the tips. Note that when the reattachment is allowed in the simulations, if a new transaction references an old, already reattached transaction together with its newly reissued counterpart, there will be a double spending. Even though the odds of that are low (since when a transaction is re-emitted, it will be old enough to be almost never chosen by the random walk algorithm), a specific procedure was included in the simulations in order to not allow double spendings.

The average costs were simulated as defined at equations (4.5) and (4.6), so a certain value of M_0 had to be chosen. Since the value of $W(v)/\lambda$ is related to the time of approval of v (whenever the transaction is indeed approved before $t^{(v)}_{M_0}$), we want M_0 to be sufficiently large, in order to capture the effect of most of the approvals. Figure 4.5 depicts the typical cumulative distribution of the time of the first approval, for several values of α and λ. Note that roughly 95% of the transactions will be approved before $t = 5$s, and almost its totality will be approved before $t = 10$s. For that reason, in both cases ($\lambda = 25$ and $\lambda = 50$), the mean cost was calculated over the transactions attached during a time interval of approximately 10s ($M_0 = 500$ for $\lambda = 50$ and $M_0 = 250$ for $\lambda = 25$), so almost the totality of approvals will be "seen" by the average cost.

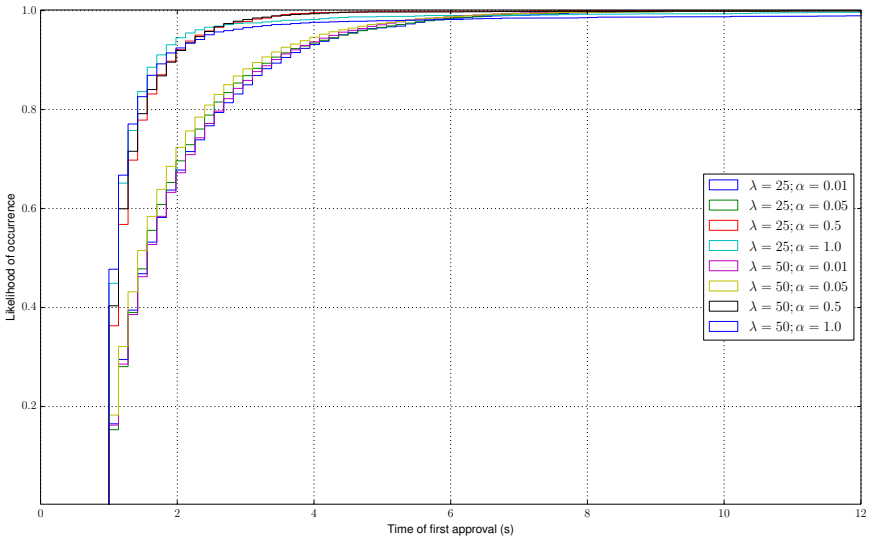

Figure 4.5: *Cumulative distribution of time of approvals for several values of α and λ*

4.5.1 One dimensional Nash equilibria

In this section, we will study the Nash equilibria $(\mathcal{S}_1, \ldots, \mathcal{S}_N)$ of the tangle problem, considering the following strategy space:

$$\left\{(1-\theta)\mathcal{S}^0 + \theta\mathcal{S}^1,\ 0 \leq \theta \leq 1\right\}$$

where the simple strategies $\mathcal{S}^0$ and $\mathcal{S}^1$ are the default tip selection strategy and the "greedy" strategy (defined in the beginning of this section) correspondingly; that is, $\mathcal{S}_i = (1-\theta_i)\mathcal{S}^0 + \theta_i\mathcal{S}^1$ where $\theta_i \in [0,1]$, $i = 1, \ldots, N$. The goal is to find the Nash equilibria relative to the costs defined in the last section (equations (4.6) and (4.5)). The selfish nodes will try to optimise their transaction cost with respect to θ_i.

[30]here, when the random walk must choose among n transactions with the same weight, it will choose randomly, with equal probabilities

By Theorem 4.4.5, each Nash equilibrium in this form will be equivalent to another Nash equilibrium with "averaged" strategies, i.e.:

$$\mathcal{S} = \left(1 - \frac{1}{N}\sum_{k=1}^{N}\theta_i\right)\mathcal{S}^0 + \frac{1}{N}\sum_{k=1}^{N}\theta_i\mathcal{S}^1 = (1-\theta)\mathcal{S}^0 + \theta\mathcal{S}^1 \qquad \text{for each } i = 1, \ldots, N,$$

Now, suppose that we have a fixed fraction γ of selfish nodes, that choose a strategy among the possible $\mathcal{S}$. The non-selfish nodes will not be able to choose their strategy, so they will be restricted, as expected, to $\mathcal{S}^0$. Note that, since they cannot choose their strategy, they will not "play" the game. Since the costs are linear over $\mathcal{S}$, such mixed strategy game will be equivalent[31] to a game where only a fraction $p = \gamma\theta \leq \gamma$ of the nodes chooses $\mathcal{S}^1$ over $\mathcal{S}^0$, and the rest of the nodes chooses $\mathcal{S}^0$ over $\mathcal{S}^1$. Note that this equivalence does not contradict the theorems proved in the last sections, that state:

- all the nodes will have the same average costs when the system is at a Nash equilibrium;
- any Nash equilibrium has an equivalent Nash equilibrium with "averaged" strategies, where all the nodes will have the same strategies.

From now on, we will refer (unless stated otherwise) to this second pure strategy game. Figure 4.6(a) represents a typical graph of average costs of transactions issued under $\mathcal{S}^0$ and $\mathcal{S}^1$, as a function of the fraction p, for a low α and two different values of λ. As already demonstrated, when in equilibrium, the selfish nodes should issue transactions with the same average costs. That means that the system should reach equilibrium in one of the following states:

(1) some selfish nodes choose $\mathcal{S}^0$ and the rest choose $\mathcal{S}^1$ $(0 < p < \gamma)$, all of them with the same average costs;

(2) all selfish nodes choose $\mathcal{S}^1$ $(p = \gamma)$;

(3) all selfish nodes choose $\mathcal{S}^0$ $(p = 0)$.

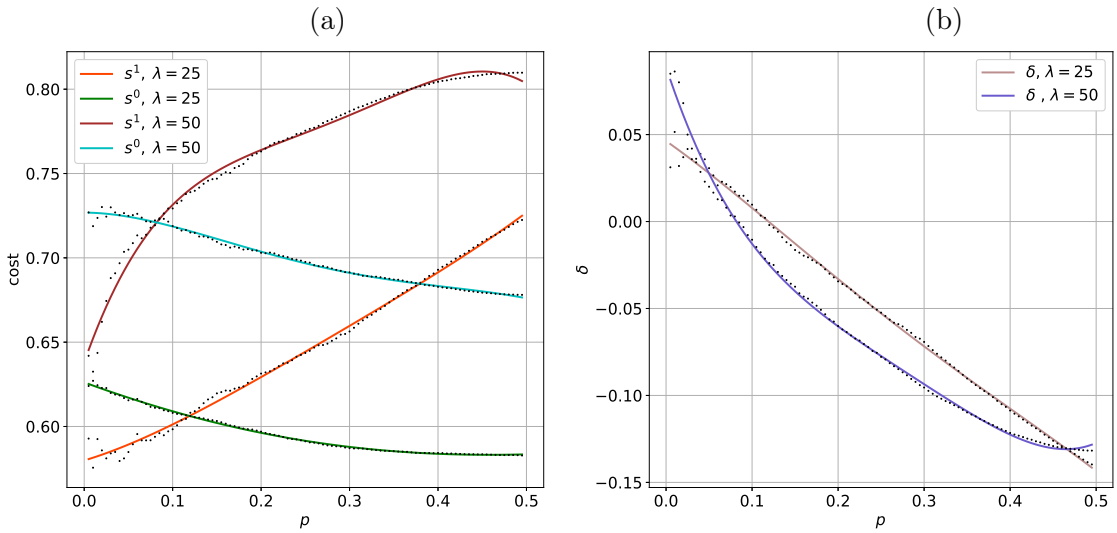

Figure 4.6: *Dotted lines are the raw data. Solid lines were fitted with least squares polynomials of four-degree. Costs (a) and gain of the strategy $\mathcal{S}^1$ over $\mathcal{S}^0$; (b) for $\alpha = 0.01$.*

If the two curves on the graphs do not intersect, the equilibrium should be clearly at state (2) or (3), depending on which of the average costs is larger. If the two curves on the graphs intercept

[31] this way, we deal with just one variable (p) instead of two (γ and θ) and none of the parameters of the system is lost

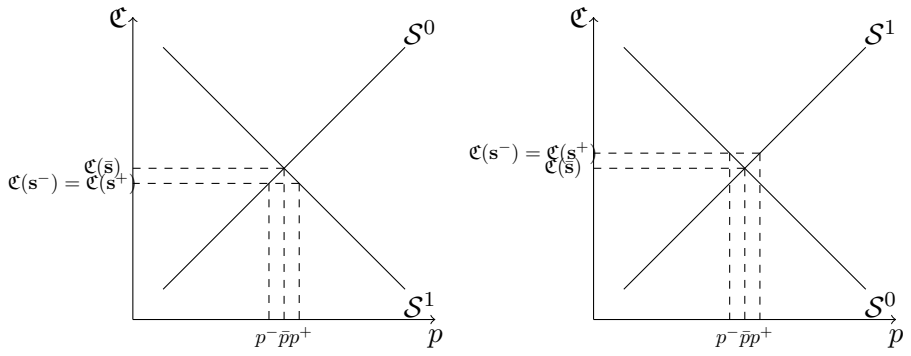

Figure 4.7: *Different Nash equilibrium points in systems with similar curves*

each other, we will also have the intersection point as a Nash equilibrium candidate. We call $\bar{\mathbf{s}}$ the vector of strategies on equilibrium and $\bar{p}$ the fraction of nodes that will issue transactions under $\mathcal{S}^1$ when the system is in $\bar{\mathbf{s}}$. We define $p^- = \bar{p} - \frac{\gamma}{N}$ and $p^+ = \bar{p} + \frac{\gamma}{N}$, meaning that p^- and p^+ will be deviations from $\bar{p}$, that result from one node switching strategies, from $\mathcal{S}^0$ to $\mathcal{S}^1$ and from $\mathcal{S}^1$ to $\mathcal{S}^0$, respectively. We also define $\bar{\mathbf{s}}^-$ and $\bar{\mathbf{s}}^+$ as strategy vectors related to p^- and p^+. Note on Figure 4.7 that this kind of Nash equilibrium candidate may not be a real equilibrium. In the first example (4.7(a)), when the system is at point $\bar{p}$ and a node switches strategies from $\mathcal{S}^0$ to $\mathcal{S}^1$ (moving from $\bar{p}$ to p^+), the cost actually decreases, so $\bar{p}$ cannot be a Nash equilibrium. On the other hand, the second example (4.7(b)) shows a Nash equilibrium at point $\bar{p}$, since deviations to p^- and p^+ will increase costs.

Now, let us re-examine Figure 4.6(a). Here, the Nash equilibrium will occur at the point $\bar{p}$, since we have a situation as on Figure 4.7(b). That point is easily found at Figure 4.6(b), when $\delta = 0$. Note that the Nash equilibrium for a larger λ will be at a smaller θ_0 than the Nash equilibrium for a smaller λ. This was already expected, since, for a larger λh, the tips will be naturally more "overcrowded", so the effect depicted at Figure 4.4 will be amplified. Thus, the Nash equilibrium for the higher λh cases must occur with a smaller proportion of transactions issued with the pure strategy $\mathcal{S}^1$.

Let us now again consider the mixed strategy game. In the case when all the nodes are allowed to choose between the two pure strategies ($\mathcal{S}^0$ and $\mathcal{S}^1$), the Nash equilibrium will be indeed at $\theta_0 = \bar{p}$ (as expected, since in this case $\gamma = 1$). If just a fraction $\gamma = p/\theta > \bar{p}$ of the nodes is selfish, then the Nash equilibrium will occur when $\theta_0 = \bar{p}/\gamma$. Now, if $\gamma \leq \bar{p}$, the costs of the nodes will not coincide[32]. In that case, the average cost of transactions under $\mathcal{S}^1$ will always be smaller than the average cost of transactions under $\mathcal{S}^0$, meaning that the Nash equilibrium will be met at $\theta_0 = 1$. Summing up, the Nash equilibrium θ_0, in these cases, will be met at:

$$\theta_0 = \min\{\bar{p}/\gamma, 1\}.$$

Figure 4.8(a) represents a typical graph of average costs of transactions under $\mathcal{S}^0$ and transactions under $\mathcal{S}^1$ as a function of fraction p, for a higher α. In that case, even though the average costs of transactions under $\mathcal{S}^0$ and transactions under $\mathcal{S}^1$ do not coincide for any reasonable p (meaning that, here, the Nash equilibrium will be met at $\theta = 1$), the typical difference between the possible pure strategies (that, from now on, we will call absolute gains) will be low, as depicted on Figure 4.8(b).

Figure 4.9 shows the average cost increase imposed on the nodes following the default strategy by the nodes issuing transactions under $\mathcal{S}^1$. Let $W(p)$ be the non-greedy nodes costs depicted in Figure 4.8(a). The cost increase is calculated as $(W(p) - W(0))/W(0)$, so it will be the relative difference of the cost of a non-selfish node in the presence of a fraction p of selfish transactions and the cost of a non-selfish node when there are no selfish transactions at all. This difference is low,

[32] that is the case for the range of studied parameters

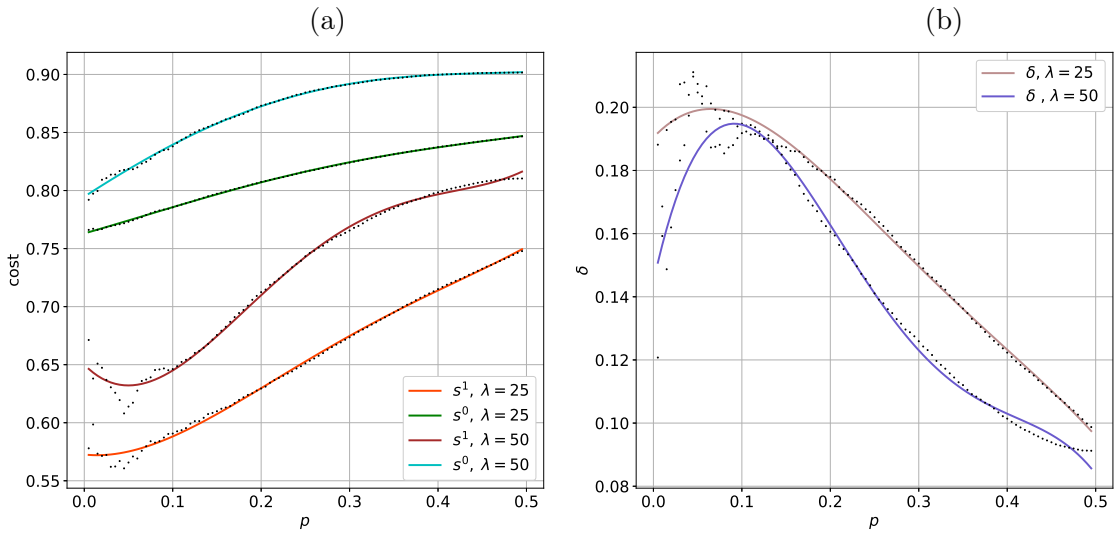

Figure 4.8: *Costs (a) and gain (b) of the strategy $\mathcal{S}^1$ over $\mathcal{S}^0$; for $\alpha = 0.5$.*

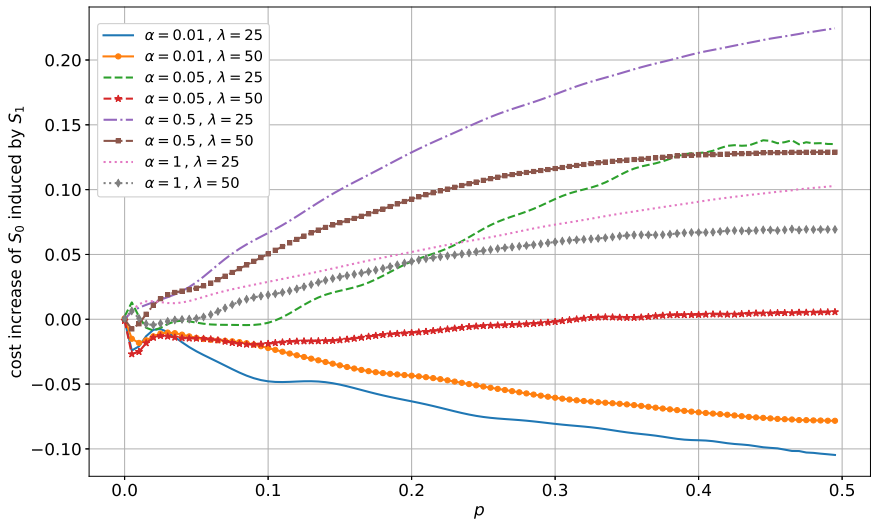

Figure 4.9: *Relative cost increase of the transactions issued by the strategy $\mathcal{S}^0$ induced by the presence of transactions emitted by the strategy $\mathcal{S}^1$.*

meaning that the presence of selfish nodes do not harm the efficiency of the non-selfish nodes. Note that this difference is small for all reasonable values of p, but even for the larger simulated values of p, the difference is still less than 25%. An interesting phenomenon, as shown in the same graph, is that the average cost increase imposed on the non-greedy nodes may actually be negative. For low values of α, just a small fraction of the transactions under $\mathcal{S}^0$ will share the approved tips with the transactions under $\mathcal{S}^1$. This fraction of transactions will approve overcrowded tips, and will have their costs increased. All the other transactions under $\mathcal{S}^0$ will have their sites less crowded, since an increase in $\mathcal{S}^1$ will mean a decrease in competition over these transactions. Finally, on average, the honest nodes will have their costs decreased.

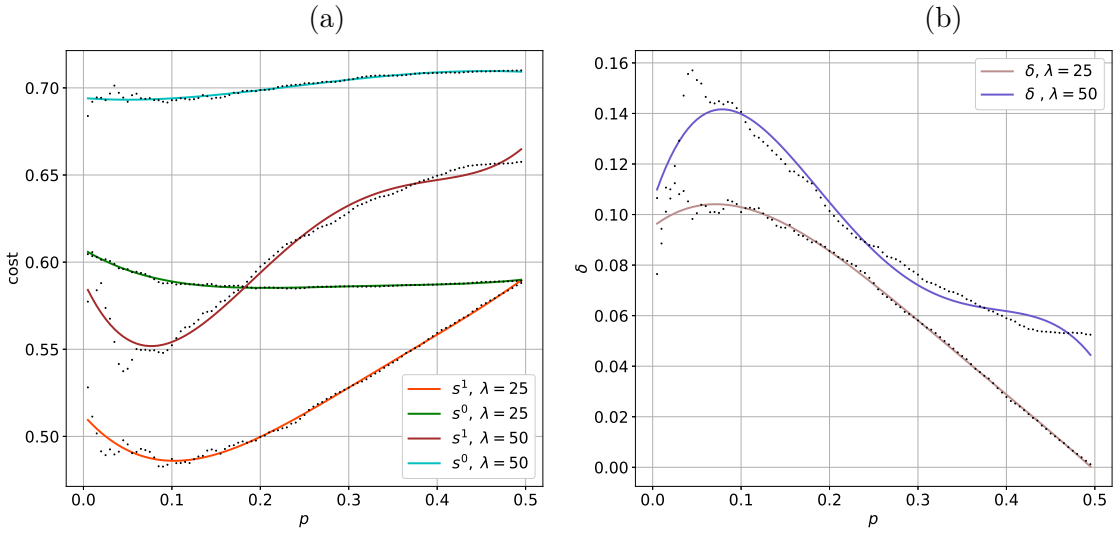

Figure 4.10: *Costs (a) and gain (b) of the strategy $\mathcal{S}^1$ over $\mathcal{S}^0$; for $\alpha = 0.05$.*

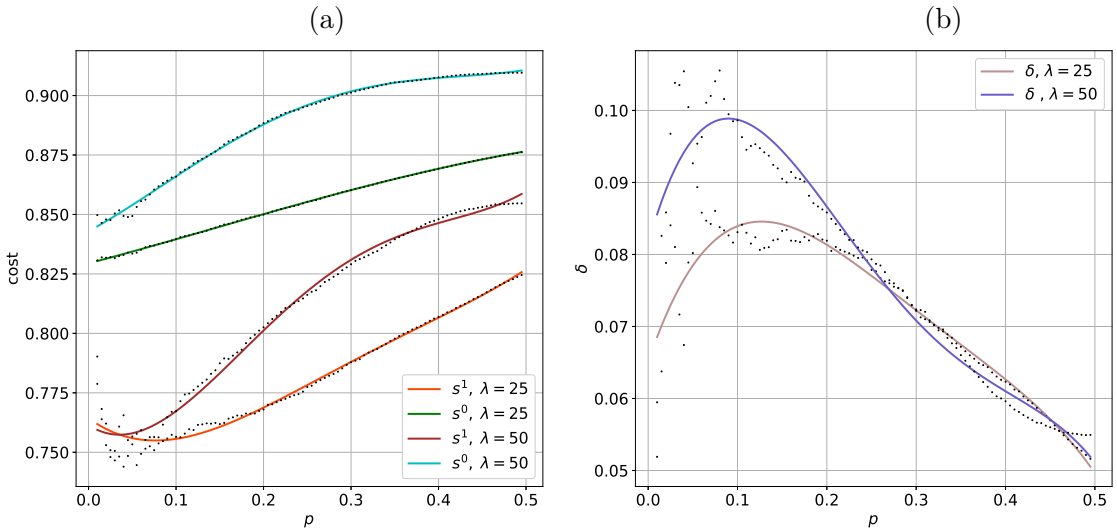

Figure 4.11: *Costs (a) and gain (b) of the strategy $\mathcal{S}^1$ over $\mathcal{S}^0$; for $\alpha = 1$.*

Figures 4.10 and 4.11 are analogous to the first figures, for other values of α and λh; part (a) of each figure represents average costs and part (b) absolute gains.

4.6 Conclusions and future work

In the first part of this paper, we prove the existence of ("almost symmetric") Nash equilibria for a game in the tangle where a part of players tries to optimise their attachment strategies. In the

second part of the paper, we numerically determine, for a simple space strategy and some range of parameters, where these equilibria are located.

Our results show that the studied selfish strategy outperform the non-selfish ones by a reasonable order of magnitude. The data show a 25% (in the most extreme scenario) difference in the nodes gains, which in some situations, may be large enough. Nevertheless, the computational cost of a selfish strategy is intrinsically larger than the computational cost of the non-selfish strategies, since the selfish strategy uses the probability distribution of the tips, which is costly to calculate for a random walk with backtracking. They will also have to monitor the tangle, to know its parameters (like λ, h etc) and act accordingly. Also, even a extreme scenario, where almost half of the transactions were issued by a selfish node, is not enough to harm the non-selfish ones in a meaningful way.

On the other hand, our results raise further questions. The obtained data exhibit a deep qualitative dependence on the parameter α of the simulation. This parameter is related to the randomness of the random walk: a low α implies a high randomness; a higher α implies a low randomness, meaning that the walk will be almost deterministic. Further simulations will be done in order to study the effect of that variable in the equilibria. Also, we only studied equilibria for a given cost, relative to the probability of confirmation of the transactions in a certain interval of time. Since this probability depends heavily on the interval of time chosen (because the probability distribution of the confirmations is far from uniform), another time intervals, that will have another practical meaning, must be analysed.

Finally, the equilibrium in the multidimensional strategy space should be studied in a more quantitative and analytic way, since it should depend strongly on α and p; and until now it was studied in just a narrow range of parameters. Further research will also be done in order to optimise the default tip selection strategy in a way that minimises this cost imposed by the selfish strategies. Through implementing research methods and techniques from the cross-reactive fields of measure theory, game theory, and graph theory, progress towards resolving the tangle-related open problems has been well under way and will continue to be under investigation.

As already mentioned, in this paper we consider only "selfish" players, i.e., those who only care about their own costs but still want to use the network in a legitimate way. We do not consider at all the case when there are "malicious" ones, i.e., those who want to disrupt the network even at a cost to themselves. We are going to treat several types of attacks against the network in the subsequent papers. Some preview of this ongoing work is available in [Pop18].

Bibliography

[Anh08] Nguyen Hoang Anh. Peer-to-peer systems : " a shared social network ". 2008. vii, 7

[Bai16] L. Baird. The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance. 2016.
http://www.swirlds.com/downloads/SWIRLDS-TR-2016-01.pdf. 46

[Bar01] David Barkai. *Peer-to-Peer Computing: Technologies for Sharing and Collaborating on the Net.* Intel Press, 2001. 5

[CF07] C. Cooper e A. Frieze. The cover time of the preferential attachment graph. *J. Comb. Theory B*, 97(2):269–290, 2007. 46

[CFP17] C. Cooper, A. Frieze e S. Pett. The covertime of a biased random walk on $g_{n,p}$. 2017. arXiv:1708.04908. 46

[Chu16] A. Churyumov. Byteball: a decentralized system for storage and transfer of value. 2016. https://byteball.org/Byteball.pdf. 46

[Coo81] R. B. Cooper. *Introduction to Queueing Theory (2nd Ed).* North Holland., 1981. 52

[DS84] P. G. Doyle e J. L. Snell. *Random Walks and Electric Networks.* 1984. Carus Mathematical Monographs **22**, Mathematical Association of America, Washington. 46

[Dur12] R. Durrett. *Essentials of Stochastic Processes.* Springer., 2012. 56, 57

[Fey12] M. Fey. Symmetric games with only asymmetric equilibria. *Games Econ. Behavior*, 75 (1):424–427, 2012. 55

[Fin64] A. M. Fink. Equilibrium in a stochastic n-person game. *J. Sci. Hiroshima Univ. Ser. A-I Math.*, 28 (1):89–93, 1964. 57

[JLS14] D. Jerison, L. Levine e S. Sheffield. Internal dla and the gaussian free field. *Duke Math. J.*, 163 (2):267–308, 2014. 46

[Kak41] S. Kakutani. A generalization of brouwer's fixed point theorem. *Duke Math. J.*, 8 (3):457–459, 1941. 57

[KG18] B. Kuśmierz e A. Gal. Probability of being left behind and probability of becoming permanent tip in the tangle. 2018.
https://www.iota.org/research/academic-papers. 51

[KP17] A. R. Karlin, e Y. Peres. *Game Theory, Alive.* American Mathematical Society., 2017. 55

[Ler15] S. D. Lerner. Dagcoin: a cryptocurrency without blocks. 2015.
https://bitslog.wordpress.com//09/11/dagcoin/. 46

[Nas50] J. F. Nash. Equilibrium points in n-person games. *Proc. Natl. Acad. Sci.*, 36(1):48–49, 1950. 57

[Ok07] E. A. Ok. *Real Analysis with Economics Applications*. Princeton University Press., 2007. 57

[Pop15] S. Popov. The tangle. 2015. `https://iota.org/IOTA_Whitepaper.pdf`. 46, 47, 49, 50, 51

[Pop18] S. Popov. Local modifiers in the tangle. 2018. `https://www.iota.org/research/academic-papers`. 66

[PSF19] Serguei Popov, Olivia Saa e Paulo Finardi. Equilibria in the tangle. *Computers Industrial Engineering*, 136:160–172, Oct 2019. 45

[Sch01] R. Schollmeier. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. Em *Proceedings First International Conference on Peer-to-Peer Computing*, páginas 101–102, 2001. 5

[SLZ16] Y. Sompolinsky, Y. Lewenberg e A. Zohar. Spectre: Serialization of proof-of-work events: Confirming transactions via recursive elections. 2016. `https://eprint.iacr.org//1159.pdf`. 46

[SZ13] Y. Sompolinsky e A. Zohar. Secure high-rate transaction processing in bitcoin. 2013. `https://eprint.iacr.org//881.pdf`. 50

[You91] Robert M. Young. 75.9 euler's constant. *The Mathematical Gazette*, 75, 06 1991. 20