Influence of short cycles on the PageRank distribution in scale-free random graphs

MSc Thesis

Hamed Ghasemieh

Exam committee:

Dr. Nelly Litvak

Prof. Richard Boucherie

Dr. Wilbert Rossi

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)

ABSTRACT

PageRank algorithms are widely used for ranking/scoring the nodes of a network, e.g for world wide web or citation networks. In this project we will consider directed networks, generated using configuration models, with fixed in- and out-degree distributions. We will investigate how the number short cycles including self-loops will affect the distribution of PageRank in such networks. For this, we firstly propose an algorithm to control the number of short cycles in the artificially generated networks with fixed in and out-degree distribution. Afterward we examine how the PageRank distribution is affected by varying the number of short cycles in such networks. As we will see, by increasing the fraction of self-loops, the number of nodes with high PageRank increases. However, the effect of cycles of size larger than one, is in contrast with this observation, i.e., the fraction of nodes with high PageRank decreases by increasing the number of these cycles. We will explain and prove the former observation by asymptotic analysis of stochastic equations as a representation of PageRanks, and for the latter we discuss an analytical example to provide an insight about what is happening.

A FEW WORDS OF THANK

For this project I should be thankful to Dr. Nelly Litvak more than anyone. She had a very good understanding the entire field of work, and for this she guided me very well in the process. I enjoyed discussing with her both scientific and non-scientific topics. But above all these, the most important thing I learned from her, was that one can be successful in any field of work if he is the really passionate about what he does.

I also would like to express my thankfulness to Prof. Richard Boucherie. This entire master program in applied mathematics was done as a parallel study to my PhD in computer science. Richard and Nelly both had a good understanding of my special situation and helped me a lot to make it to the end. I also have to thank Dr. Wilbert Rossi for accepting to be part of my graduate committee.

At the end, nothing could have been possible for me to cope with if Maryam, my wife and mother my little girl, was not there for me shoulder to shoulder.

CONTENTS

1

INTRODUCTION

The age that we are living in, is called the age of information by many thinkers and scientists [1]. This is indeed growing especially because now, almost everyone has the possibility to generate and share contents with others. According to Erik Schmidt, Executive Chairman of Alphabet Inc. (the parent company of Google), every two days now we create as much information as we did from the dawn of civilization up until 2003!

Lots of information around us encompass some sorts of relations among themselves, therefore one can assume a network structure for plenty of data around us. Examples are ubiquitous: astronomical social networks like Facebook, networks of citations, networks of sellers and buyers in a market competing each other for offering their products [2], networks of webpages, and etc. With such a growth, demands for analysis, and developing new tools for comprehending or even controlling the behaviors of such networks, is of substantial value.

One of the most important problems in analysis of networks is the problem of *centrality*, i.e., finding a way to see how central or important is a node in a network. This knowledge can be used in many different ways. For instance if one is interested in influencing behaviors of a members of a network, or in order to spread certain information in the network, it is totally reasonable to find and invest on central nodes. There are many methods introduced for measuring centrality of a node in a network. We refer to [3] for a good covering survey.

Among all these centrality measures, PageRank is one of the most famous ones. PageRank is first introduced by founders of Google, Page and Brin in the seminal paper "The anatomy of a large-scale hypertextual web search engine" [4]. This measure was specifically designed for the networks of webpages, and precisely formulated to implement search algorithms for webpages. This was indeed the grounds for Google.

Many real networks exhibit *clustering* property [5–7]. This means, if two nodes are connected via a short path, it is probable that they are connected directly as well. In informal terms, this means, for instance, if two persons have friends in a same community, the probability of them being friends increases. This probability indeed increases if the

length of connecting path between two nodes become shorter. like for instance, two persons having a common friend, probably know each other as well. The measure of this probability in network terminology is known as *clustering coefficient*. This measure, specifically known to be quite high in social networks [8].

Having said the above, and knowing that PageRank is among the most important measures of network centrality, it is interesting to examine the impact of clusters on PageRank distribution in a network. This is the main aim of this thesis. More specifically, we consider directed scale-free graph, i.e, graphs which their in- and out-degree sequences are distributed according to power law [9]. On such graphs we will consider effects of short cycles, on the distribution of PageRank. For generation of such graphs we will use *configuration models* [10], with fixed in- and out-degree distributions [11]. We will introduce an algorithm to modify this generated network, to change the fraction of short cycle, while keeping the in- and out- degree sequences untouched. After this we will experiment how PageRank distribution is going to be affected with different size of short cycles for some randomly generated networks. At the end we try to prove some of the observed results for a general case.

CONTRIBUTIONS

This thesis consists of three main contributions:

1. We introduce an algorithm using which one can generate directed graphs with different clustring scructure, but fixed in- and out-degree sequences. This algorithm is built on top of the algorithm given in [11], for generating directed random graph, with given distribution for in- and out-degree sequences. The proposed algorithm in this thesis, receives a control parameter as input for fraction of short cycles in the network, along with a directed graph. Then it restructures this graph such that the in- and out-degree sequences remain unchanged, but the fraction of the short cycles are altered according to the given control parameter. Using this algorithm one can generated graphs with different clustering properties, but same in- and out degree sequences.

2. Having devised the above algorithm to generate the desired graph, we investigate how the PageRank distribution is altered by changing the control parameter, i.e., by changing the amount of short cycles, while the in-and out-degrees are kept fixed. As we will see we have two contrasting numerical observations regarding cycles of length one, i.e., self-loops, and cycles of larger length. As we will see, by increasing the number of self-loops, the number of nodes with high PageRank increases, and vice

versa. However, the effect of cycles of size larger than one, is in contrast with this observation, i.e., the fraction of nodes with high PageRank decreases by increasing the number of these cycles.

3. For the observation regarding slef-loops, we employ stochastic equation representation of PageRanks [12, 13] to prove it in the general case. On the other hand, the observation related to cycles of size larger than one, turns out to be more difficult to be proved in general case. This is because, the proposed algorithm for controlling the fraction of cycles, although keeps the in- and out-degrees fixed, however, it completely restructures the graph. This makes it hard to track and compare two different networks with fixed in- and out-degrees, but different fraction of cycles. Having said this, for this observation we provide an analytical example, to supply it with some insight about what is actually happening.

OUTLINE

This thesis consists of four more chapters. In the following we briefly mention what is discussed in each of the following chapters.

Chapter 2 provides the grounds for theoretical developments of PageRank, and briefly discusses an algorithm for computation of PageRank in a graph. At the end of this chapter, PageRank is discussed as the solution of a stochastic equation. The developments of this part is later used to prove numerical observations in this thesis.

Chapter 3 discusses the well-known configuration models in general, and configuration models for generation of directed graphs with given distribution for in- and out-degree sequences, in particular. After this we introduce our proposed algorithm built on top of these models, for controlling the fraction of short cycles in a given graph, while its in- and out-degrees remain unchanged.

In *Chapter 4* we investigate the influence of fraction of self loops on the distribution of PageRank. Moreover, we use stochastic equations to prove our numerical observations, for the case of graphs with fixed out-degrees.

Chapter 5 investigates the effect of cycles of length larger than one on PageRank distribution. For numerical results in this chapter, we provide an analytical example.

2

PAGERANK AND NETWORKS' STRUCTURE

In this chapter we briefly discuss the famous PageRank algorithm first introduced by founders of Google, Page and Brin [4]. In a broad sense, this algorithm is dealing with the problem of centrality: given a directed graph, how we can rank its nodes according to their centrality or importance? A 'given graph' can be model of any phenomena, such as graph of web pages linking to each other, or network of paper citations, or graph of friendship in a social network.

In the Section 2.1, we briefly discuss the theoretical ideas behind the original PageRank algorithm, and Section 2.2.1, discusses the power method as one of the most effective methods for computation of PageRank. Finally, in Section 2.3, we discuss specifically, PageRank distribution as a solution of a stochastic equation, following the same idea in [12]. The advantage of this stochastic equation is that it provides the possibility to capture PageRank distribution of the entire network in one equation.

2.1 THEORETICAL DEVELOPMENT

Plenty of academic papers concerning PageRank have been published since Page and Brin's original paper. One can see the PageRank score as the stationary distribution of a random walk process with some additional jumping probabilities on a given directed graph. Consider a directed graph $G_n = (V_n, E_n)$ with n nodes, and let $N_i^+ = \{j : (j,i) \in E_n\}$ and $N_i^- = \{j : (i,j) \in E_n\}$ be the set of incoming and outgoing neighbours of node i, respectively. Moreover, let (d_i^+, d_i^-) be the bi-degree sequence, i.e., $d_i^+ = |N_i^+|$ and $d_i^- = |N_i^-|$. The PageRank of node i is defined as the following:

$$\pi_i = c\left(\sum_{j\in N_i^+}\frac{1}{d_j^-}\pi_j + \frac{1}{n}\sum_{j\in\mathcal{D}_0^-}\pi_j\right) + \frac{1-c}{n}, \quad i = 1,\cdots,n. \tag{2.1}$$

Where, $c \in (0,1)$, is a damping factor, and $\mathcal{D}_0^-$ is the set of nodes with out-degree zero.

Here is the physical description of the process. A random walker in a node of the network with probability c may decide to follow an outgoing link, or with probability $(1-c)$ may jump to any randomly chosen node in the entire network. The outgoing links are chosen

uniformly, if any. In case the current node does not have a successor, i.e., it is a *dangling node*, the walker will jump randomly to any other node.

The Equation (2.1) is the convex combination of two terms, describing this physical process. The first term is the summation of PageRank of incoming nodes weighted by the inverse of their out-degrees, and the second term is incorporating the possibility of jumping to the node i from any other node in the network. The latter term, in the context of network of web pages, can be interpreted as the possibility that a user just typing the address of a page in the browser, instead of following a link in another page. From technical point of view, this consideration makes sure that the underlying Markov chain is irreducible, and hence the stationary distribution, i.e., the PageRank exists and is unique.

Let $\pi = (\pi_1, \cdots, \pi_n)^T$ be the vector of PageRanks, then the Equation (2.1) can be written in the vector form as the following:

$$\pi^T = c\pi^T A + \frac{1-c}{n}\mathbb{1}_n^T. \tag{2.2}$$

Where $\mathbb{1}_n$, is the unit column vector of n, 1's, and A is the transition matrix defined as:

$$A_{ij} = \begin{cases} 1/d_i^- & \text{if } d_i^- > 0 \text{ and } (i,j) \in E_n \\ 1/n & \text{if } d_i^- = 0 \\ 0 & \text{otherwise.} \end{cases}$$

One can easily notice that matrix A is a stochastic matrix, since the sum of the elements of each row adds up to 1.

Using Equation (2.2), the PageRank vector π can be computed as follows:

$$\pi^T = \frac{1-c}{n}\mathbb{1}_n^T(I - cA)^{-1}.$$

From the above equation, and the fact that matrix A is stochastic we can check that the computed PageRank vector is normalized:

$$\begin{aligned} \pi^T\mathbb{1}_n &= \frac{1-c}{n}\,\mathbb{1}_n^T(I-cA)^{-1}\mathbb{1}_n \\ &= \frac{1-c}{n}\,\mathbb{1}_n^T\sum_{k=0}^{\infty} c^k A^k \mathbb{1}_n \\ &= \frac{1-c}{n}\,\mathbb{1}_n^T\,\frac{1}{1-c}\,\mathbb{1}_n \\ &= 1. \end{aligned} \tag{2.3}$$

2.2 PAGERANK COMPUTATION

As mentioned before, in a broad sense, PageRank computation can be seen as computation of stationary distribution of a Markov chain,

which in turn can be done by either solving an eigenvector problem, or a system of linear equations. Knowing that $\pi^T \mathbb{1}_n = 1$, one can rewrite Equation (2.2) as the following:

$$\pi^T = \pi^T \left(cA + \frac{1-c}{n} \mathbb{1}_n \mathbb{1}_n^T \right) = \pi^T P. \tag{2.4}$$

Where matrix P is again stochastic (since it is a convex combination of two matrices with rows adding up to 1) with eigenvalues:

$$1 > c\lambda_2(A) \geq c\lambda_3(A) \geq \cdots,$$

in which, $\lambda_i(A)$ is the ith largest eigenvalue of matrix A (in absolute value) [14].

Therefore, the PageRank vector π is the left eigenvector of the matrix P associated with largest eigenvalue which is 1. However, computation of this eigenvector is practically impossible by just solving equations. This is because although P is usually a sparse matrix, but it is of huge dimension. For instance for the network of webpages for which Google needs to perform the search algorithm, the dimension is of tens of billions, which is growing on daily basis [15].

There are plenty of methods for computing PageRank, among which we briefly discuss the power method introduced in the original paper of Brin and Page [16]. For survey of other numerical methods for computing PageRank we refer to [17, 18].

2.2.1 *The Power Method*

The goal is to find vector π such that:

$$\pi^T = \pi^T P,$$

where

$$P = cA + \frac{1-c}{n} \mathbb{1}_n \mathbb{1}_n^T.$$

The power method is as follows:

Algorithm 2.1 Power Method

1: Pick an initial vector $\pi_{(0)}$
2: **repeat**
3: $\quad \pi_{(k+1)}^T = \pi_{(k)}^T P$
4: **until** Termination criterion is satisfied.

We first compute the error for one iteration:

$$\begin{aligned} \pi_{(k+1)}^T - \pi^T &= \pi_{(k)}^T P - \pi^T P \\ &= c\pi_{(k)}^T A + \frac{1-c}{n} \mathbb{1}_n^T - c\pi^T A - \frac{1-c}{n} \mathbb{1}_n^T \\ &= c \left(\pi_{(k)}^T - \pi^T \right) A. \end{aligned}$$

Therefore, we have:

$$||\pi^T_{(k+1)} - \pi^T||_1 \leq c||\pi^T_{(k)} - \pi^T||_1,$$

and after k iterations [19]:

$$||\pi^T_{(k+1)} - \pi^T||_1 \leq c^k||\pi^T_{(0)} - \pi^T||_1 \leq 2c^k.$$

This means that, independent of the dimension of the matrix, we can lower the error as we want by repeating the iteration.

The main advantage of Power method is that it is easy to implement, and since the matrix at hand is usually sparse then each iteration is not costly. In addition to this, power method has robust convergence behaviour, it is known that the convergence rate of iterative methods is proportional to the second eigenvalue of the matrix involved in iteration. However, one should have in mind that still for some cases the convergence can be quite slow. For accelerarion of power method one can refer to [20, 21].

2.3 PAGERANK AS A SOLUTION TO A STOCHASTIC EQUATION

One of the main goals in this thesis is to investigate distribution of PageRank of a randomly chosen node, in the graph. For this, following the idea in [12, 13], we will model PageRank as the solution of a distributional identity, i.e., a stochastic equation. Note that PageRank values in Equation (2.1), are scaled by number of nodes, i.e. by $1/n$. However, in analysis of distributions, it is more convenient to deal with scale-free PageRanks:

$$R_i = n\pi_i, \quad i = 1, \cdots, n.$$

In this case the Equation (2.1), with the extra assumption that there is no dangling node, is simplified as

$$R_i = c \sum_{j \in N_i^+} \frac{1}{d_j^-} R_j + (1 - c), \quad i = 1, \cdots, n. \tag{2.5}$$

For consideration of networks with dangling nodes one can refer to [12, 22].

In the following we consider the random variable R as the PageRank of a randomly chosen node. Note that, as a result of Equation (2.3), we have $\mathbb{E}(R) = 1$. One of the main goal in this thesis is to compare the tail properties of probability distribution of R, i.e., $P(R > x)$ as x is large enough, for different graph structures. To this end PageRank R can be modelled as the solution of stochastic equation involving random variables associated with in- and out-degrees.

Let R and D^+ be the random variables representing the PageRank and the in-degree of a randomly chosen node. Moreover, let D_j^- be

the random variable associated with the out-degree of a randomly chosen node, say j, which has a link to the node we want to compute its PageRank. Apparently, D_j^-'s are not the same random variable as the out-degree of a randomly chosen node, this is because of the extra assumtion that they all have a link to a specific node. Hence D_j^-'s have different distributions. Now the stochastic equation can be written as [12]:

$$R \stackrel{d}{=} c \sum_{j=0}^{D^+} \frac{1}{D_j^-} R_j + (1-c), \tag{2.6}$$

where, $\stackrel{d}{=}$ means that the two sides are following the same probability distribution. In the above equation we assume R_j's are identically and independently distributed (i.i.d) as R. In what follows, to simplify the analysis, we add an extra assumption that D_j^-'s are also i.i.d and distributed as random variable D^-. This assumption is acceptable since the out-degree is not extensively influential on the value of PageRank. Now the goal is to find the probability distribution of R that satisfies the above equation.

By introducing new random variables $A \stackrel{d}{=} c/D^-$ and $B \stackrel{d}{=} (1-c)$, one reaches the general stochastic equation

$$R \stackrel{d}{=} \sum_{j=0}^{D^+} A_j R_j + B. \tag{2.7}$$

In the remainder of this section we briefly discuss the existence of solution to the general stochastic equation, and provide theorems for its asymptotic solution. One can refer to [23] for detailed analysis. We assume the following assumption is valid for the remainder of this section.

Assumption 2.1. *R_j's are i.i.d and distributed as R, A_j's are i.i.d and distributed as A, and R_j's, A_j and D^+ are mutually independent.*

2.3.1 *Existence and uniqueness*

The solution strategy is based on the following iteration, with initial known distribution of $R^{(0)}$:

$$R^{(k)} \stackrel{d}{=} \sum_{j=0}^{D^+} A_j R_j^{(k-1)} + B, \tag{2.8}$$

where $R_j^{(k-1)}$'s and A_j's are independent and distributed as $R^{(k)}$ and A, receptively. The following theorem is proved in [12].

Theorem 2.1 (Existence and uniqueness)**.** *Equation (2.7) has a unique and non-trivial solution R with mean 1. Moreover, the iteration in Equation (2.8) converges to this solution:*

$$R = R^{(\infty)} = \lim_{k \to \infty} R^{(k)}.$$

2.3.2 *Asymptotic behaviour*

The asymptotic, i.e., tail solution of R depends on relation of distributions of D^+ and B. However, in our case we have assumed B is a constant $1-c$, therefore we can assume that B is a random variable with lighter tail, i.e., $P(B > x) = o(P(D^+ > x))$ as $x \to \infty$. For other cases we refer to [12]. Moreover, as we will see in Chapter 3, Section 3.1, in-degree distributions that we will be dealing with are regularly varying random variable, i.e., we have:

$$P(D^+ > x) \sim x^{-\alpha} L(x) \text{ as } x \to \infty, \tag{2.9}$$

where, $L(x)$ is a slowly varying function.

The following theorem, proved in [12], summarizes the the asymptotic behavior of distribution of R:

Theorem 2.2. *If $P(B > x) = o(P(D^+ > x))$ and $P(R^{(0)} > x) = o(P(D^+ > x))$, then for all $k \geq 1$:*

$$P(R^{(k)} > x) \sim C^{(k)} P(D^+ > x) \text{ as } x \to \infty,$$

in which, $C^{(k)} = (\mathbb{E}(A))^\alpha \sum_{i=0}^{k-1} [\mathbb{E}(D^+)\mathbb{E}(A^\alpha)]^i$.

Note, that in the above theorem it is assumed that the initial distribution of $R^{(0)}$ have a lighter tail than in-degree distribution. This is a reasonable assumption as usually the iteration starts with $R^{(0)} \equiv 1$.

Using Theorem 2.1, one can write:

$$P(R > x) = \lim_{k \to \infty} P(R^{(k)} > x) \sim C^{(\infty)} P(D^+ > x) \text{ as } x \to \infty,$$

where, $C^{(\infty)}$ is given by:

$$C^{(\infty)} = \lim_{k \to \infty} C^{(k)} = \frac{(\mathbb{E}(A))^\alpha}{1 - \mathbb{E}(D^+)\mathbb{E}(A^\alpha)}. \tag{2.10}$$

3

GENERATING RANDOM GRAPHS WITH SHORT CYCLES

In this chapter we develop a model of network for which we can control the number of triangles, while the in- and out-degree of the nodes remain untouched. In order to do this, we first need to be able to generate a graph based on given in- and out-degree sequence, which from now on we call bi-degree sequence. For this we use the *directed configuration models* [10].

Configuration model for undirected graphs is basically a way of generating a graph such that the degree of each of its nodes is a random variable samples from a probability distribution with integer support [24]. More specifically, let $v_1, \cdots, v_n$ be the vertices of a graph, and P be a probability distribution with positive integer support. The configuration model is constructed as follows. For each vertex v_i independently sample a value d_i, with probability of $P(D_i = d_i)$. Then attach d_i *half edges* or *stubs* to v_i. After this randomly attach all the half edges together, i.e., at each time choose two random stubs, and connect them together, until no stub is remained.

A minor problem in the above process is that, $\sum d_i$ may be odd. However, this can be easily solved by either repeating the sampling process till the summation adds up to be even, or just by removing a stub. A more significant problem shows itself when we want to generate simple graph. The above described process apparently allows creation of multiple edges and loops, as e.g., the two chosen stubs can be connected to a same vertex. We will consider this problem in the next section for the directed configuration models.

The process of generating bi-degree sequence for directed graphs will involve more technicalities. This is mainly due to the more complex requirements for a bi-degree sequence to be *graphical*, i.e., if it is possible to actualize or draw a graph with that sequence. So one need to come up with procedure which guarantees that the generated sequence is graphical. This is the content of Section 3.1, which is based on [11].

Section 3.2, discusses the modification of configuration model for directed graphs. In other words given a graphical bi-degree sequence how one can draw graph for that. This method resembles in some aspects to the process described above for undirected graphs.

Finally, in Section 3.3 we introduce an algorithm to control amount of cycles in the networks. This algorithm receives as input a bi-degree sequence, and a parameter to control the number of loops in the graph, so one can easily alter the total number of loops by changing this parameter.

3.1 GRAPHICAL BI-DEGREE SEQUENCE

The goal in this section is to generate a simple random directed graph from given distributions for both in- and out-degrees. For this we need to specify two probability distributions with non-negative integer support, indicating distribution of in- and out-degree of each node. Let us name these distributions F and G for in- and out-degrees, respectively:

$$F(x) = \sum_{i=1}^{x} f_i, \quad \text{and} \quad G(x) = \sum_{i=1}^{x} g_i,$$

where f_i, and g_i are the probability of having in- and out-degree of i, respectively. Like the undirected case we assume that both in- and out-degree elements are drawn independently from F and G. However, here we have more serious criteria for the drawn bi-degree sequence to be graphical. The corresponding condition to the undirected case, for which summation of degree sequence has to be even, for the directed case is that the sum of in-degrees should be equal to sum of out-degrees.

This condition turns out to be more hard to meet comparing to the case of undirected graphs. This is because in general, the probability that two i.i.d sequences have the same sum, as the length of the sequence goes to infinity, converges to zero, even if they have equal means. In order to deal with this, the authors in [11], have provided an algorithm which modifies an i.i.d bi-degree sequence $\mathbf{D} = (\mathbf{D}^+, \mathbf{D}^-)$, to get another sequence $\hat{\mathbf{D}} = (\hat{\mathbf{D}}^+, \hat{\mathbf{D}}^-)$, which will be graphical with probability one, as the the number of nodes goes to infinity. Moreover, they prove by way of their construction, the sequence $\hat{\mathbf{D}}$, converges in distribution to the sequence $\mathbf{D}$. In the following we briefly discuss the algorithm and the results from [11].

In order for algorithm to work we need to make an extra assumption on distributions F and G. Particularly we need to assume that there exist *slowly varying functions* L_F and L_G such that:

$$1 - F(x) \leq x^{-\alpha} L_F(x) \quad \text{and} \quad 1 - G(x) \leq x^{-\beta} L_G(x),$$

for all $x \geq 0$ and $\alpha, \beta > 1$. A function L is said to be slowly varying if $\lim_{x\to\infty} L(tx)/L(x) = 1$ for all fixed $t > 0$. In this case we say F and G are *regularly varying distributions*. The above conditions make sure that F and G have finite moments of order r and s, for $0 < r < \alpha$ and

$0 < s < \beta$, respectively. The following constant is important in the asymptotic analysis of the algorithm:

$$\kappa = \min\{1 - \alpha^{-1}, 1 - \beta^{-1}, 1/2\}.$$

The algorithm for generating the bi-degree sequence is given in Algorithm 3.1.

Algorithm 3.1 GENERATEBIDEGREE(F, G)

Require: Probability distributions F and G for in- and out-degrees.
Ensure: The graphical bi-degree sequence.

1: Fix $0 < \delta_0 < \kappa$
2: **repeat**
3: Sample i.i.d sequence $\{\hat{d}_1^+, \cdots, \hat{d}_n^+\}$ from F
4: Sample i.i.d sequence $\{\hat{d}_1^-, \cdots, \hat{d}_n^-\}$ from G
5: $\Delta_n \leftarrow \sum_{i=1}^n \hat{d}_1^+ - \sum_{i=1}^n \hat{d}_1^-$
6: **until** $|\Delta_n| \leq n^{1-\kappa+\delta_0}$
7: $S_{|\Delta_n|} \leftarrow |\Delta_n|$ randomly chosen nodes (without replacement)
8: **if** $\Delta_n > 0$ **then**
9: $\forall i \in S_{|\Delta_n|} : d_i^- \leftarrow \hat{d}_i^- + 1$
10: **if** $\Delta_n < 0$ **then**
11: $\forall i \in S_{|\Delta_n|} : d_i^+ \leftarrow \hat{d}_i^+ + 1$

The algorithm is based on repeated generation of samples of in- and out-degrees until their summation difference satisfies the condition, $|\Delta_n| \leq n^{1-\kappa+\delta_0}$, given in line 6. After this the degree sequences in lines 7-11 are modified such that the new bi-degree sequence satisfies the condition that the summation of in and out-degrees are equal.

The first issue about the above algorithm is proof of termination. In other words, one needs to prove the condition $|\Delta_n| \leq n^{1-\kappa+\delta_0}$, is satisfied after a reasonable number of samplings. In [11], it is proven that this event occurs with probability one as n goes to infinity, i.e.,

$$\lim_{n\to\infty} P(\{|\Delta_n| \leq n^{1-\kappa+\delta_0}\}) = 1.$$

The next point to consider is whether the generated bi-degree sequence is *graphical*, i.e., if there is an actual graph with the generated bi-degree sequence. The necessary and sufficient conditions for a bi-degree sequence to be graphical is given in the following Theoerm taken from [25].

Theorem 3.1. *Given vertices set $V = \{v_1, \cdots, v_n\}$, the bi-degree sequence $(\boldsymbol{D}^+, \boldsymbol{D}^-) = (\{d_1^+, \cdots, d_n^+\}, \{d_1^-, \cdots, d_n^-\})$, is graphical if and only if:*

(i) $\sum_{i=0}^n d_i^+ = \sum_{i=0}^n d_i^-$, *and*

(ii) $\sum_{i=1}^n \min\{d_i^-, |A - \{v_i\}|\} \geq \sum_{v_i \in A} d_i^+$ *for all* $A \subset V$.

Having the above characterization for graphical bi-degree sequences, in [11], it is proven that, bi-degree sequence generated by Algorithm 3.1, is asymptotically graphical with probability one, i.e.,

$$\lim_{n\to\infty} P((\{d_i^+\},\{d_i^-\}) \text{ is graphical}) = 1.$$

Regarding the generated bi-degree, the other property to be discussed, is that the modification of sampled sequences in Algorithm 3.1, is negligible, so to speak. Particularly, it has to be shown that although the generated bi-degree sequence is no longer i.i.d, and may have different distributions than of the original F and G, however, asymptotically they assimilate these properties. Intuitively this is because the modification needed to be done to the degree sequences ($|\Delta_n|$) is small proportionate to n. Therefore the following theorem is proven in [11].

Theorem 3.2. *The bi-degree sequence, $(\{d_i^+\},\{d_i^-\})$, generated in Algorithm 3.1, for any fixed $s, r \in \mathbb{N}$, satisfies*

$$(d_{i_1}^+,\cdots,d_{i_r}^+,d_{j_1}^-,\cdots,d_{j_s}^-) \xrightarrow{D} (\hat{d}_1^+,\cdots,\hat{d}_r^+,\hat{d}_1^-,\cdots,\hat{d}_s^-),$$

as $n\to\infty$, where $\{\hat{d}_i^+\}$, $\{\hat{d}_i^-\}$ are samples from original distributions F and G, and $\xrightarrow{D}$ is convergence in distribution.

3.2 DIRECTED CONFIGURATION MODEL

Having a graphical bi-degree sequence one can use the idea of configuration models, as in undirected graphs, to realize a directed graph. Here instead of half-edges we consider half directed edges. Let $v_1,\ldots,v_n$ be nodes of the graph, and $(\{d_i^+\},\{d_i^-\})$ be the bi-degree sequence. We attach to each node v_i, d_i^+ inbound half-edges, d_i^- outbound half-edges. Then we randomly choose one inbound half-edge and connect it to a randomly chosen outbound half-edge (all the selection processes are uniform). This process is continued until no half-edge remains.

Like the case of undirected graphs here is also possible to have multiple edges, and self-loops. Hence we may not have simple graphs as a result. In order to solve this two methods are proposed, *repeated directed configuration models* and *erased directed configuration models*. In the former method, as the naming suggests, the process is repeated until a simple graph is realized. For this to happen, one has to prove that the probability of realization of a simple graph is bounded away from zero, hence a repeated performance of the method guarantees that the probability of drawing a simple graph is one. In [11], authors have proved that this is indeed the case, when certain reasonable conditions on the bi-degree sequence is satisfied (e.g., they are sampled from a distributions with finite moments), and the number of nodes goes to infinity.

Erased configuration model is simply based on creation of the possible multi-graph first, and then removing self-loops, and merging multiple edges on the same direction into one edge. This is useful in two ways. First when the conditions on bi-degree sequence for the probability of realization of simple graph to be positive, are not satisfied, hence the repeated configuration model will not work. Second it is more efficient, since the graph is generated only once, and no repetition is needed [26]. However, we should note that in this method the bi-degree sequence is modified and we may be violating the obligation that the bi-degree sequence should follow a certain probability distributions.

Let $(\{d(e)_i^+\}, \{d(e)_i^-\})$ be the bi-degree sequence of the simple graph obtained using erased directed configuration method, i.e., $d(e)_i^+$ and $d(e)_i^-$ are the in-degree and out-degree of node i after erasing multiple edges and self-loops. We define the joint distribution as:

$$h^{(n)}(i,j) = \frac{1}{n}\sum_{k=1}^{n} P(d(e)_k^+ = i, d(e)_k^- = j),$$

Moreover, define the empirical distribution of the realised erased model as:

$$\hat{f}_k^{(n)} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{d(e)_i^+ = k\} \text{ and } \hat{g}_k^{(n)} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{d(e)_i^- = k\}.$$

The following is proven in [11].

Theorem 3.3. *For the erased configuration model as described above, from the bi-degree obtained from Algorithm 3.1, we have:*

(i) $h^{(n)}(i,j) \to f_i g_j$ *as* $n \to \infty$, *and*

(ii) $\forall k = 0, 1, \cdots : \hat{f}_k^{(n)} \xrightarrow{P} f_k$ *and* $\hat{g}_k^{(n)} \xrightarrow{P} g_k$ *as* $n \to \infty$,

where $\xrightarrow{P}$ *is convergence in probability.*

3.3 DIRECTED CONFIGURATION MODEL WITH CYCLES

In this section we provide an algorithm to control the number of loops in directed graph with fixed in- and out-degrees. For this algorithm we use the method discussed in Section 3.1 for generation of in- and out-degrees. Moreover, we use the same idea as in [27], for adding loops. This is based on the modification of the standard configuration model as discussed in Section 3.2 as a model for generation of random graphs with a given bi-degree distribution. The idea is that, instead of just indicating in and out (half) edges for each node, we also indicate the number of *loop corners* for each node.

Let $\{d_i^+\}$ and $\{d_i^-\}$ be a generated sequence of graphical bi-degrees as provided in Algorithm 3.1. Let $0 \leq \alpha \leq 1$ be the parameter which

controls the number of loops in the network, such that the number of loops present in the network increases by increasing α. We split the in- and out-degree for each node into single edges and loops corners as follow:

$$\begin{aligned} t_i &= \alpha \cdot \min(d_i^+, d_i^-) \\ s_i^+ &= d_i^+ - t_i \\ s_i^- &= d_i^- - t_i \end{aligned}$$

The sequences $\{s_i^+\}$ and $\{s_i^-\}$, are the sequence of incoming and outgoing single half-edges, respectively. The sequence $\{t_i\}$ defines the number of loop corner for each node, i.e., the number of loops the node i is going to be part of. While for single half-edge of a node we are choosing another node for forming a connection, for loop corners we need to choose a set of other nodes equal to the provided size of loop, and connect them to each other in a circular way.

Algorithm 3.2 CREATGRAPHWITHLOOPS($\{d_i^+\}, \{d_i^-\}, \alpha, l$)

Require: Bi-degree sequence $\{d_i^+\}$ and $\{d_i^-\}$, α as the parameter of controlling the number of loops, and the loop size l.
Ensure: The graph with the given bi-degree with loops.

1: $t_i \leftarrow \alpha.\min(d_i^+, d_i^-)$
2: $s_i^+ \leftarrow d_i^+ - t_i$
3: $s_i^- \leftarrow d_i^- - t_i$
4: Build the graph with $\{s_i^+\}$ and $\{s_i^-\}$ as bi-degree sequence as discussed in Section 3.2.
5: $T = \sum_i t_i$
6: **while** $T > l$ **do**
7: Randomly choose the set of l nodes $S_l = \{n_1, \cdots, n_l\}$ s.t. $\forall n_i \in S_l : \ t_{n_i} > 0$
8: Add edges $n_1 \rightarrow n_2 \rightarrow \cdots n_l \rightarrow n_1$
9: $\forall n_i \in S_l : \ t_{n_i} \leftarrow t_{n_i} - 1$
10: $T \leftarrow T - l$

Algorithm 3.2 provides the procedure for this idea. In lines 1-3 the single edge and loops corner sequences are created as discussed above. In line 4 using the configuration model of Section 3.2, a graph is created only using single half-edges. In line 5 the total number of available loop corners are stored. Through line 6-10, an iteration takes place until no other loops can be formed among available loop corners, i.e., until when the number of available loop corners are less than the given loop size. In line 7, l nodes that can contribute to a loop are randomly chosen. Note that a node n_i can be part of a loop if $t_{n_i} > 0$, i.e., when it has at least one incoming and one outgoing edge. In line 8 a loop is created among the chosen nodes. In line 9 the value of loop corner indicator for each chosen node is updated. Finally, in

line 10 the total number of available loop corners are reduced by the loop size l.

Note that in the above algorithm for generating the graph we can use both so-called repeated directed and erased configuration models discussed in Section 3.2. More specifically, in line 4 we use either of repeated or erased configuration models, therefore the resulting graph is simple. Note that, after connecting loop corners through lines 6-10, the resulting graph may not be simple any more, therefore another pass of multiple edge removal is needed in case of requirement for non-existence of multiple edges. However, if we allow multiple edges, with the above algorithm the bi-degree of graph remains unchanged, and only wiring is done in such a way that the number of loops varies with α.

In the following proposition we compute and relate the expected number of added cycles to the control parameter α.

Proposition 3.1. *By the given process of the Algorithm 3.2, the expected number of cycles increases linearly by α. Indeed,*

$$\mathbb{E}(M_l) = \alpha \cdot \frac{n}{l} \sum_{x=0}^{\infty} \bar{F}(x)\bar{G}(x),$$

where M_l is the random variable representing the number of added cycles of size l, and $\bar{F}(x) = 1 - F(x)$ and $\bar{G}(x) = 1 - G(x)$.

Proof. Let T_i be the random variable representing the number of a loop corners for node i. We have

$$\begin{aligned} P(T_i/\alpha > x) &= P(\min(D^+, D^-) > x) \\ &= P(D^+ > x)P(D^- > x) \\ &= \bar{F}(x)\bar{G}(x). \end{aligned}$$

Next we have

$$\mathbb{E}(T_i/\alpha) = \sum_{x=0}^{\infty} P(T/\alpha > x) = \sum_{x=0}^{\infty} \bar{F}(x)\bar{G}(x).$$

Therefore,

$$\mathbb{E}(T_i) = \alpha \cdot \sum_{x=0}^{\infty} \bar{F}(x)\bar{G}(x).$$

Moreover, we have

$$\mathbb{E}(M_l) = \frac{1}{l} \sum_{i=1}^{n} \mathbb{E}(T_i),$$

which leads to the proof of the proposition. □

4

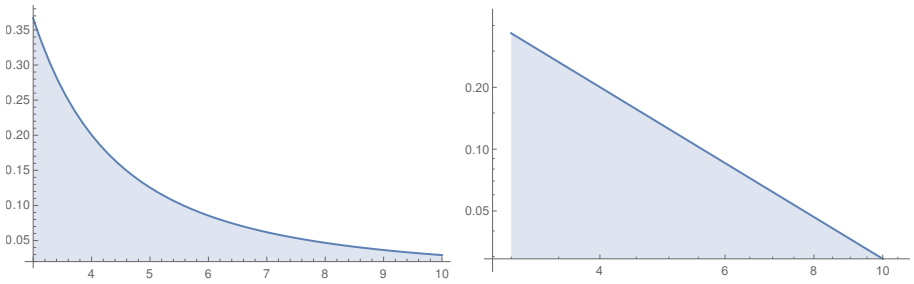
INFLUENCE OF SELF-LOOPS ON PAGERANK DISTRIBUTIONS

In this chapter we use the Algorithm 3.2 in Chapter 3 to see how self-loops can influence PageRank distribution. We will use the tail analysis results in Theorem 2.2 to prove the observations we have for self-loops in the general sense.

In this chapter (and the next) for the generation of in- and out-degrees, we use Pareto distribution, with parameters β and x_m, as shape and scale parameters, receptively. Probability density function of Pareto distribution is given as:

$$P(X = x) = \frac{\beta x_m^{\beta}}{x^{\beta+1}}, \quad x \in [x_m, +\infty]. \tag{4.1}$$

In order to conduct observations, we use $\beta = 1.1$, $x_m = 3$. This means that the minimum possible in- or out-degree of a node would be 3. Figure 4.1, shows the probability density functions of Pareto distribution, in ordinary and log-log scale for these parameters.



(a) PDF.

(b) PDF in log-log scale.

Figure 4.1: PDF and log-log PDF of Pareto distribution for scale and shape parameters of 3, and 1.1, respectively.

This chapter is organized as follows. In Section 4.1 we illustrate the observations and experiments done by addition of slef-loops to the graph. In Section 5.2, we prove the observation for the general case.

4.1 EXPERIMENTS

In the following we specifically investigate the influence of addition of self-loops to the distribution of PageRank. We use Algorithm 3.2 with loop size 1, to control the amount of self-loops. Figure 4.2 illustrate the empirical distribution of PageRank computed on a graph of ten thousands nodes. Each curve in this figure is associated with a value of α, controlling the amount of self loops. As can be seen by increasing the value of α, i.e., increasing self-loops, the fraction nodes with high PageRanks increases, while it decreases for the case of having nodes with lower PageRanks. This is indeed more visible in Figure 4.2b, in which the value of probabilities are shown in log-scale. Note that we have shown the curves of this figure in a longer domain range, to show the effect of self-loops on the tail of the probability distribution.

Figure 4.3, shows the same measures for a graph of half a million nodes. One can easily see as the number of nodes increases we have smoother curves. This clearly shows the importance of asymptotic analysis for proving the observations in general case.

Given the above experiments we can formulate the following observation. Let G_α be a randomly realized graph according to the Algorithm 3.2, for a fixed value of α, and let $\pi_i(G_\alpha)$ be random variable representing the PageRank of a randomly chosen node i, in the graph G_α.

Observation 4.1. *For $\alpha_1 < \alpha_2$, there exists the threshold γ, such that for all $x > \gamma$, we have:*

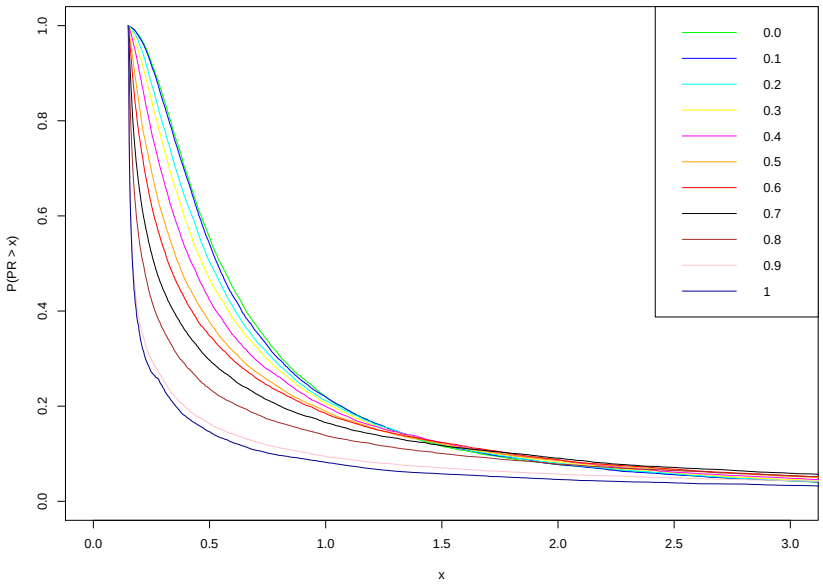
$$\sum_{i=1}^{n} \mathbb{1}\{\pi_i(G_{\alpha_1}) > x\} < \sum_{i=1}^{n} \mathbb{1}\{\pi_i(G_{\alpha_2}) > x\}, \tag{4.2}$$

as $n \to \infty$.

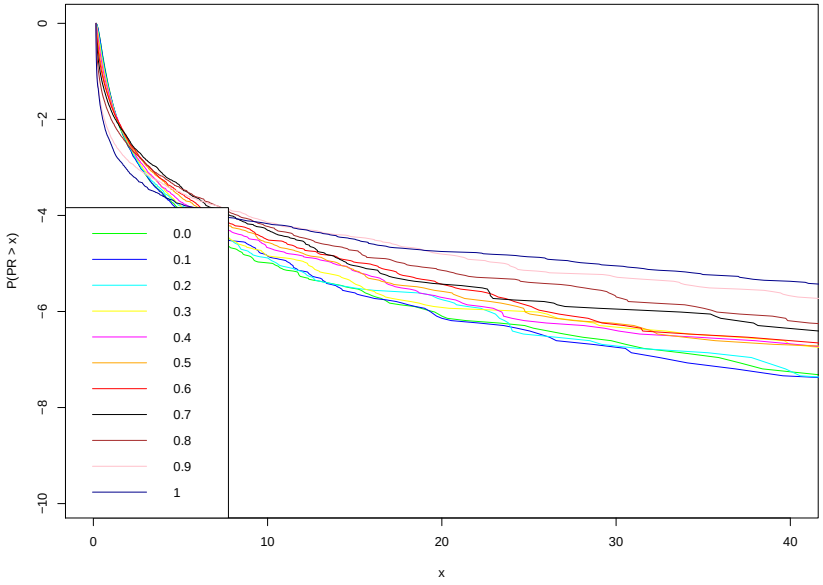
In plain words, the above observation states that the number of nodes with high PageRank increases as the number of self-loops increases. In Section 4.2, we prove the above observation for the special case where all the nodes have the same out-degrees, using tail analysis in Theorem 2.2.

4.2 TAIL ANALYSIS OF OBSERVATIONS

In this section we use tail analysis given in Theorem 2.2 to prove that the observation in Section 4.1 generally holds. In order for Assumption 2.1 to hold, we follow the idea in [12, 22] to assume that the out-degrees are constant. This assumption is admissible since it is widely known that the influence of out-degrees on PageRanks is not extensive. Figure 4.4, depicts empirical distribution of PageRanks in a graph of half a million nodes, with constant out-degree of 20.

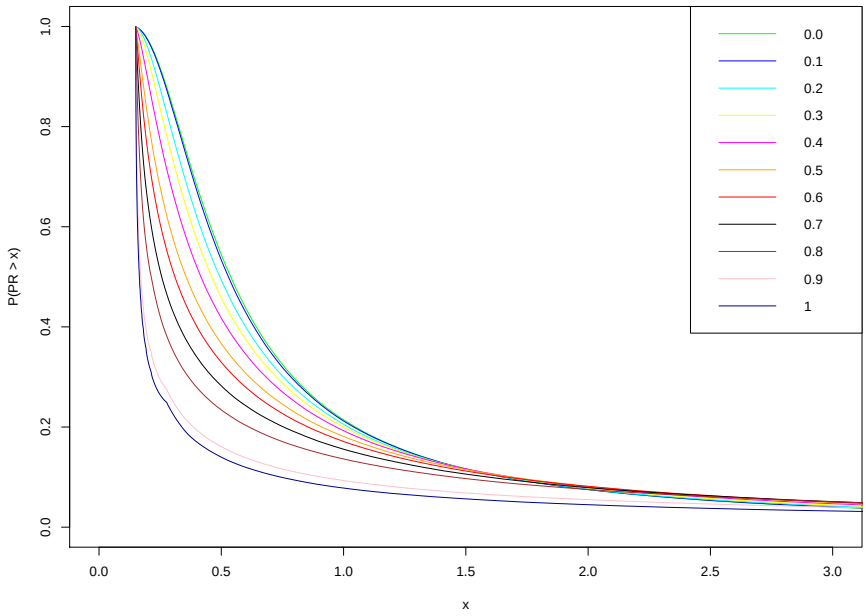


(a) $P\left(\pi_i(G_\alpha) > x\right)$.

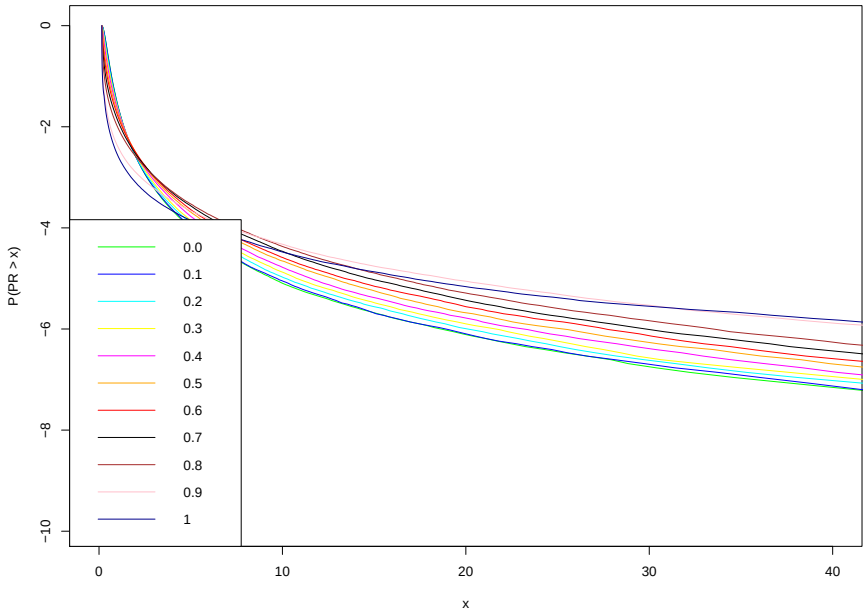


(b) $log\left(P\left(\pi_i(G_\alpha) > x\right)\right)$.

Figure 4.2: Empirical PageRank distribution for a graph of 10,000 nodes, for different values of α, and loop size 1, i.e., self-loops. Note that $\pi_i(G_\alpha)$ is random variable representing PageRank of a realized random graph G_α.

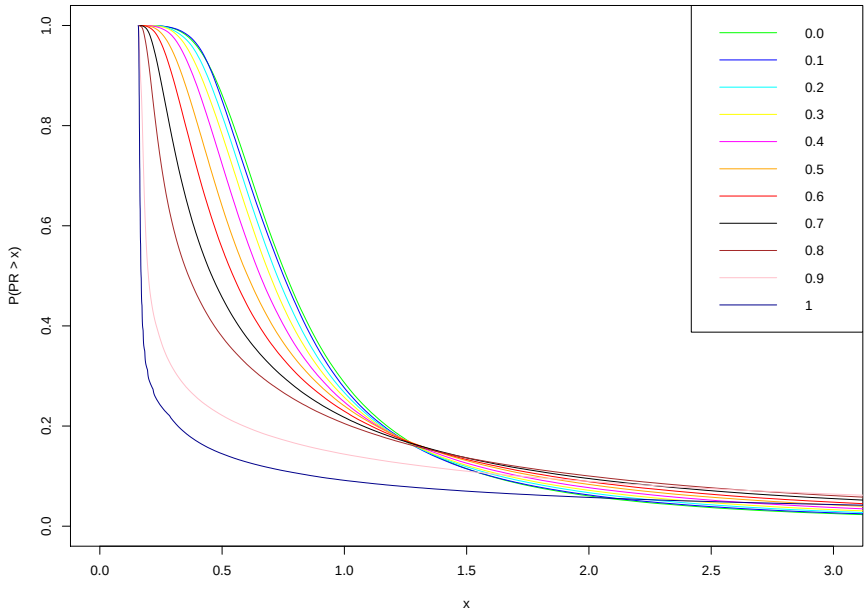


(a) $P\left(\pi_i(G_\alpha) > x\right)$.

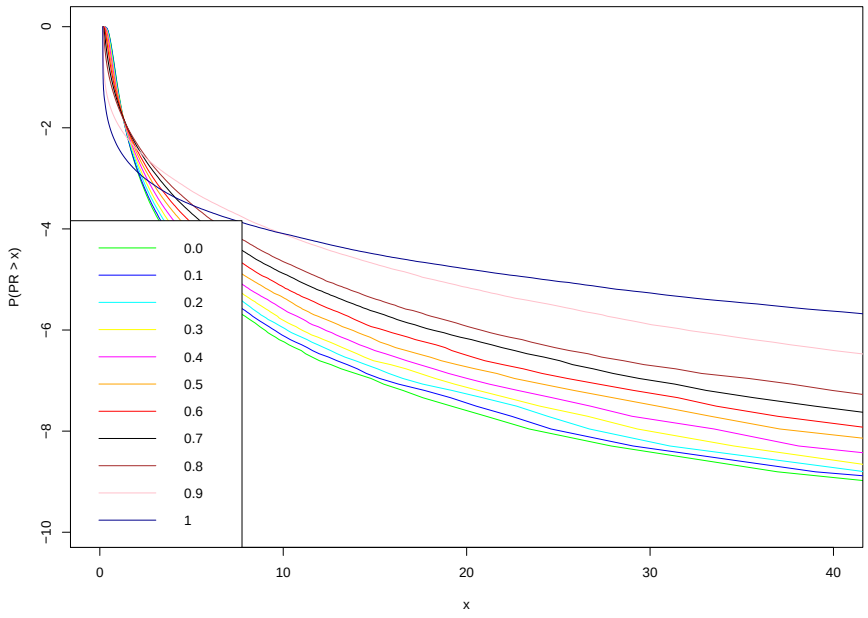


(b) $log\left(P\left(\pi_i(G_\alpha) > x\right)\right)$.

Figure 4.3: Empirical PageRank distribution for a graph of half a million nodes, for different values of α, and loop size 1, i.e., self-loops. Note that $\pi_i(G_\alpha)$ is random variable representing PageRank of a realized random graph G_α.



(a) $P\left(\pi_i(G_\alpha) > x\right)$.



(b) $log\left(P\left(\pi_i(G_\alpha) > x\right)\right)$.

Figure 4.4: Empirical PageRank distribution for a graph of half a million nodes with constant out-degree of 20, for different values of α, and **loop size 1**. Note that $\pi_i(G_\alpha)$ is random variable representing PageRank of a realized random graph G_α.

In the following we use the stochastic equation for PageRank to prove the Observation 4.1, indeed holds asymptotically. For the original graph we can repeat the stochastic Equation (2.6) as follows:

$$R \overset{d}{=} c \sum_{j=0}^{D^+} \frac{1}{D_j^-} R_j + (1-c)$$

Now let each node destroy one outgoing and incoming link and form a self-loop. The above equation will be altered, and the new PageRank can be computed as follows:

$$\begin{aligned}
\hat{R} &\overset{d}{=} c \sum_{j=0}^{D^+-1} \frac{1}{D_j^-} \hat{R}_j + \frac{c}{D^-} \hat{R} + (1-c) \\
\Rightarrow \hat{R}(1 - \frac{c}{D^-}) &\overset{d}{=} c \sum_{j=0}^{D^+-1} \frac{1}{D_j^-} \hat{R}_j + (1-c) \\
\Rightarrow \hat{R} &\overset{d}{=} \frac{cD^-}{D^- - c} \sum_{j=0}^{D^+-1} \frac{1}{D_j^-} \hat{R}_j + \frac{(1-c)D^-}{D^- - c}.
\end{aligned}$$

Note that the above distributional identity captures the entire modification in the network. If we add the extra assumption that all nodes have the constant out-degree, say d^-, we have the following stochastic equations for the original and altered PageRanks:

$$R \overset{d}{=} \frac{c}{d^-} \sum_{j=0}^{D^+} R_j + (1-c) \tag{4.3}$$

$$\hat{R} \overset{d}{=} \frac{c}{d^- - c} \sum_{j=0}^{D^+-1} \hat{R}_j + \frac{(1-c)d^-}{d^- - c}. \tag{4.4}$$

Both above equations can be seen as instances of the general stochastic equation given in (2.7). Moreover, since the Assumption 2.1 holds, Theorem 2.2 is immediately applicable for their tail analysis. Knowing that,

$$\begin{aligned}
&\mathbb{E}(A) = c/d^- \text{ and } \mathbb{E}(D^+) = \mathbb{E}(D^-) = d^-, \\
&\mathbb{E}(\hat{A}) = c/(d^- - c) \text{ and } \mathbb{E}(\hat{D}^+) = \mathbb{E}(D^+ - 1) = \mathbb{E}(D^+) - 1,
\end{aligned}$$

we have:

$$\begin{aligned}
P(R > x) &\sim C^{(\infty)} P(D^+ > x) \text{ as } x \to \infty, \\
P(\hat{R} > x) &\sim \hat{C}^{(\infty)} P(D^+ > x) \text{ as } x \to \infty,
\end{aligned}$$

where, $C^{(\infty)}$ and $\hat{C}^{(\infty)}$ are given by:

$$\begin{aligned}
C^{(\infty)} &= \frac{(c/d^-)^\alpha}{1 - d^-(c/d^-)^\alpha} \\
\hat{C}^{(\infty)} &= \frac{(c/(d^- - c))^\alpha}{1 - d^-(c/(d^- - c))^\alpha}.
\end{aligned}$$

After simplification, the above identities become:

$$C^{(\infty)} = \frac{c^{\alpha}}{(d^{-})^{\alpha} - c^{\alpha} d^{-}} \tag{4.5}$$

$$\hat{C}^{(\infty)} = \frac{c^{\alpha}}{(d^{-} - c)^{\alpha} - c^{\alpha}(d^{-} - 1)}. \tag{4.6}$$

In general we have:

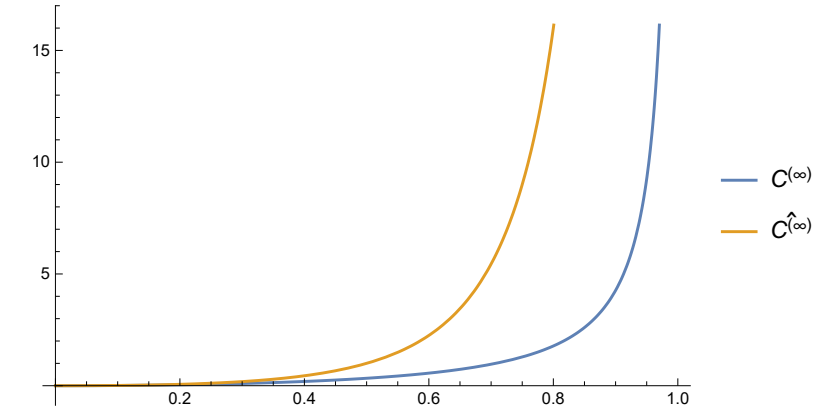
$$\hat{C}^{(\infty)} > C^{(\infty)},$$

for all $c \in (0,1)$ and $d \in \{1, \cdots, n\}$. This proves the observations in Section 4.1, for the general case. Figure 4.5, is illustrating both $C^{(\infty)}$ and $\hat{C}^{(\infty)}$ for $\alpha = 2$ and for all $c \in (0,1)$ and different values of d^{-}.

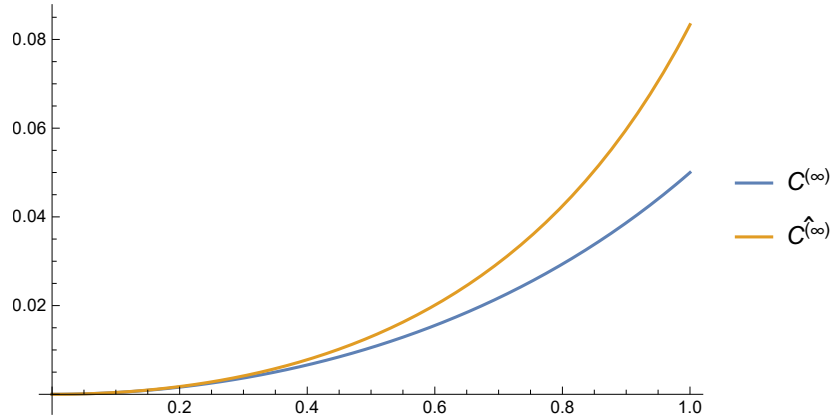
Therefore we have proved the following theorem:

Theorem 4.1. *For $\alpha_1 < \alpha_2$, the exists the threshold γ, such that for all $x > \gamma$, we have:*

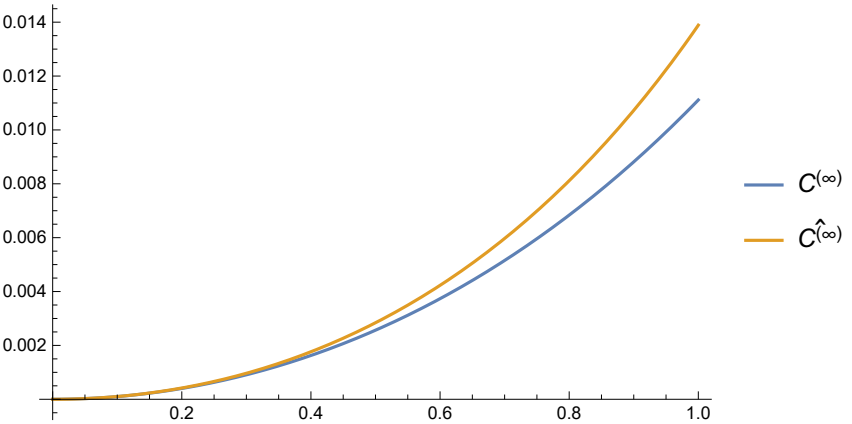
$$\lim_{n \to \infty} \mathbb{P}\left(\pi_i(G_{\alpha_1}) > x\right) < \lim_{n \to \infty} \mathbb{P}\left(\pi_i(G_{\alpha_2}) > x\right). \tag{4.7}$$



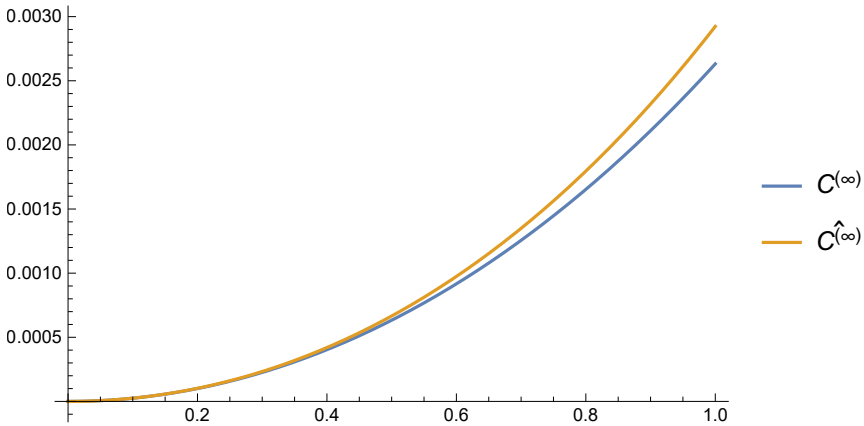
(a) $d^- = 1$.



(b) $d^- = 5$.



(c) $d^- = 10$.



(d) $d^- = 20$.

Figure 4.5: Comparison of $C^{(\infty)}$ and $\hat{C}^{(\infty)}$ for different values of d^-, for all $c \in (0, 1)$ on x-axis, and $\alpha = 2$.

5

INFLUENCE OF CYCLES ON PAGERANK DISTRIBUTIONS

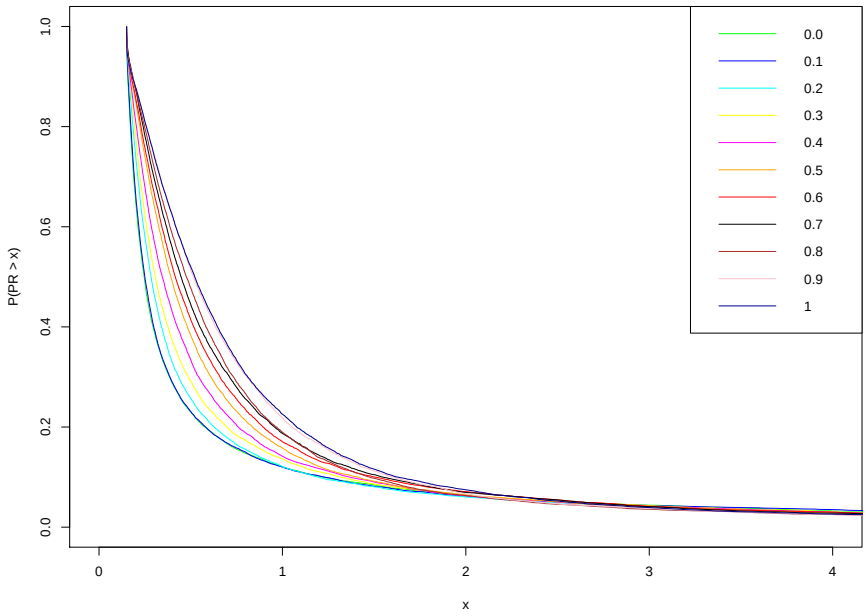
In this chapter we follow the same lines as in Chapter 4, but for cycle sizes of length larger than one. Again we use the Algorithm 3.2 in Chapter 3 to see how different loop sizes can influence PageRank distribution. This chapter is further organized as follows. In Section 5.1 we illustrate the observations and experiments done by addition of loops of different size to the graph. And in Section 5.2, we provide an analytical examples in which one can see why the numerical observations are happening.

5.1 EXPERIMENTS

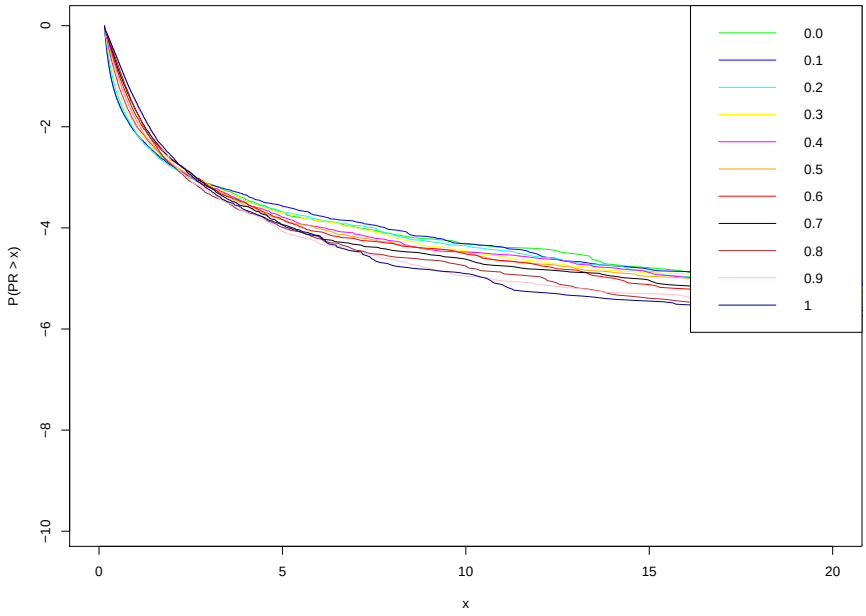
In this section we mainly discuss the observations and experiments on the influence of loops on PageRank distribution. For the generation of in- and out-degree, like the previous chapter, we use Pareto distribution (cf. Equation 4.1). In this section we investigate the influence of short cycles, i.e., loops of size greater than one on the distribution of PageRanks. As we will see, the effect in this case is exactly the reverse of self-loops.

Figure 5.1 shows the empirical distribution of PageRank computed on a graph of 10,000 nodes, for different values of α, and loop size 3, which means we are adding triangles. As one can see the probability of having nodes with lower PageRank increases by increasing value of α, i.e., having more triangles. So one may deduce that by increasing α PageRank of a node is likely to be increased, specifically for low ranked nodes. Figure 5.2, depicts the same results for half a million nodes. As can be seen we have more smooth curves, especially for log-scale diagram. This, again, shows the importance of asymptotic analysis.

One can see that the results for addition of loops of size larger than 1, e.g., triangles, are the reverse of what we saw in the previous section for addition of self-loops. Given the above experiments we can can formulate the following observation. Recall from previous section that G_α is a randomly realized graph according to Algorithm 2.1, for a value of α, and $\pi_i(G_\alpha)$ is the PageRank of node i, in the graph G_α.

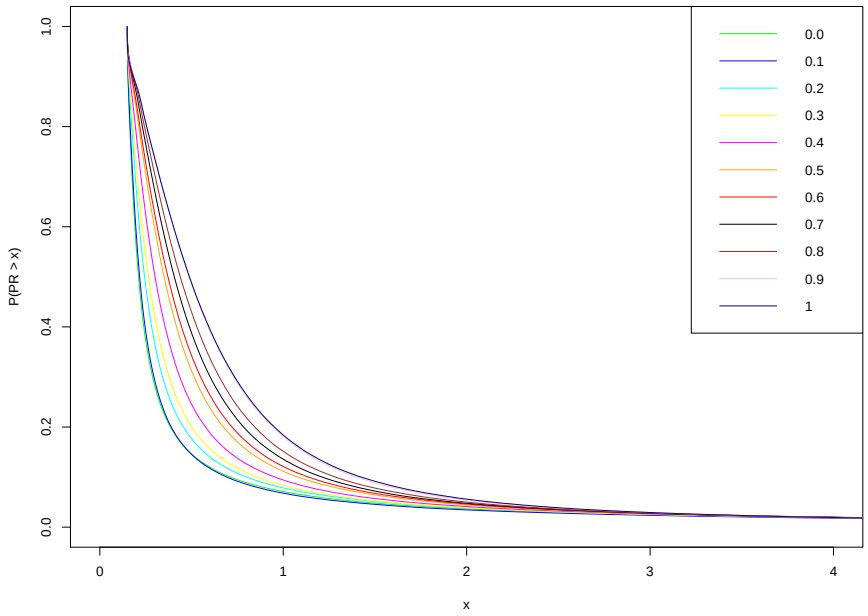


(a) $P\left(\pi_i(G_\alpha) > x\right)$.

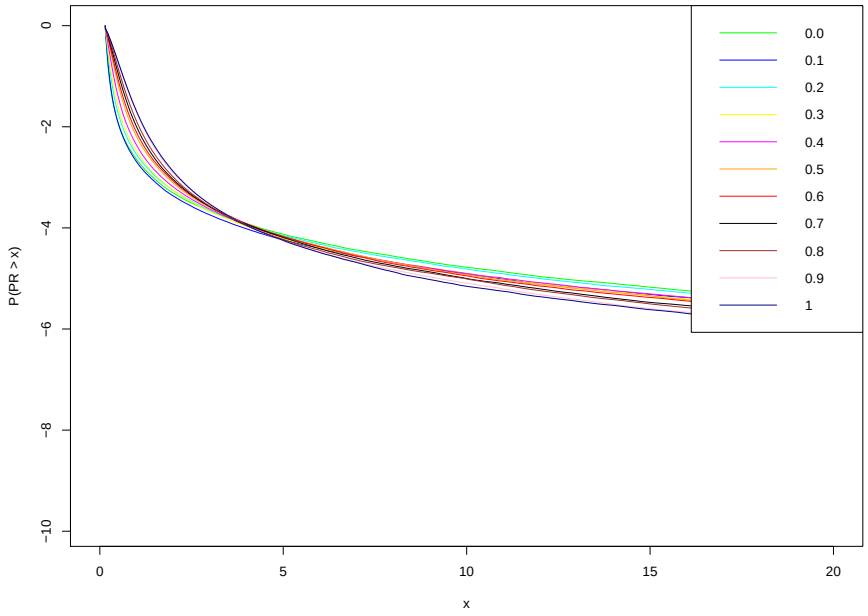


(b) $log\left(P\left(\pi_i(G_\alpha) > x\right)\right)$.

Figure 5.1: Empirical PageRank distribution for a graph of 10.000 nodes, for different values of α, and **loop size 3**, i.e., triangles. Note that $\pi_i(G_\alpha)$ is random variable representing PageRank of a realized random graph G_α.



(a) $P\left(\pi_i(G_\alpha) > x\right)$.



(b) $log\left(P\left(\pi_i(G_\alpha) > x\right)\right)$.

Figure 5.2: Empirical PageRank distribution for a graph of half a million nodes, for different values of α, and **loop size 3**, i.e., triangles. Note that $\pi_i(G_\alpha)$ is random variable representing PageRank of a realized random graph G_α.

Observation 5.1. *For $\alpha_1 < \alpha_2$, there exists the threshold γ, such that for all $x > \gamma$, we have:*

$$\sum_{i=1}^{n} \mathbb{1}\{\pi_i(G_{\alpha_1}) > x\} > \sum_{i=1}^{n} \mathbb{1}\{\pi_i(G_{\alpha_2}) > x\}, \tag{5.1}$$

as $n \to \infty$.

The above observation in plain words means that the fraction of nodes with high PageRank decreases by increasing α. On the other hands due to normalization, the fraction of nodes with lower PageRank increases as α is increased.

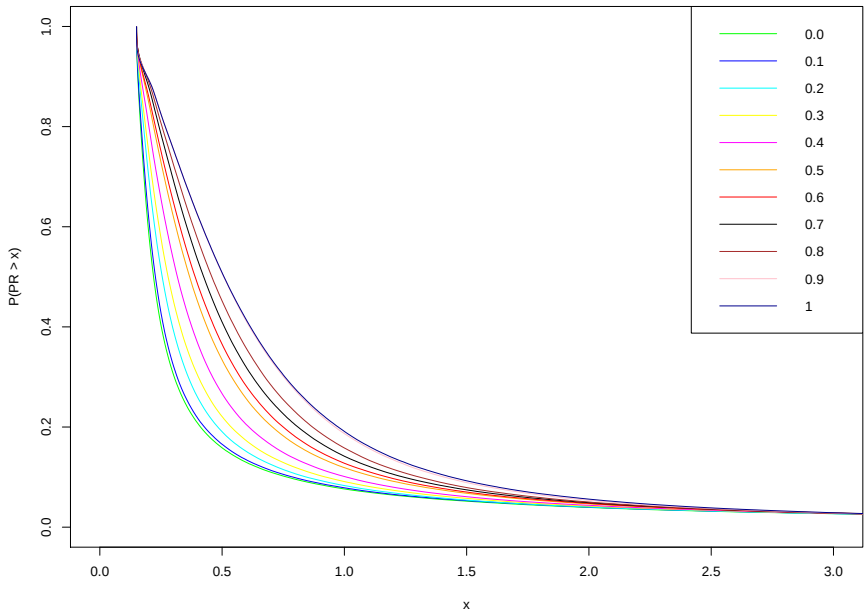
We also consider addition of loops of greater size. Figure 5.3 illustrates the PageRank distribution in a graph of half a million nodes for loops of size 15, for different values of α. As one can see the effect is the same as for triangles, but with less intensity, i.e., the curves for different values of α are closer to each other for lower values of PageRanks, however, they follow the same order as for triangles. This can be interpreted as follows: loops of smaller size have a greater effect on nodes with lower PageRanks. Hence we have the following observation.

Observation 5.2. *Addition of loops of larger size, in comparison with small size loops, causes more decrease in fraction of nodes with lower PageRank. As a result of this, since the PageRank is normalized, loops of larger size will cause more increase in fraction of nodes with higher PageRank.*

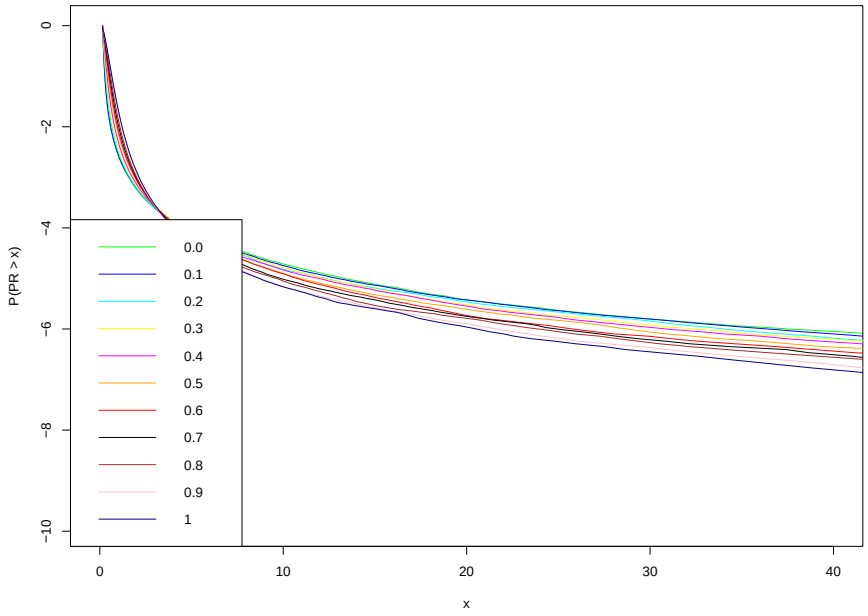
5.2 AN ANALYTICAL EXAMPLE

In this section in order to illustrate the observed results for case of general loops, we investigate some toy examples for addition of triangles, and will calculate the PageRank before and after addition of triangles.

Figure 5.4, demonstrates a network with central node 1, which is connected to other nodes via triangles or cycles. We assume there are m_1 and m_2, triangles and cycles connected to the node 1 respectively. Therefore the total number of nodes in this network is $n = 2m_1 + \sum_{k=1}^{m_2} L_k + 1$, where L_k is the number of nodes in the kth cycle, without counting node 1. Since all the nodes in triangles have the same PageRanks we refer to them with the same label, i.e., the node connected to 1 via an out going edge from 1 is labeled a and the other is labeled b. Moreover, we label the nodes in the kth cycle ($1 \leq k \leq m_2$) as v_i^k, where $2 \leq i \leq L_k$. We first analytically com-



(a) $P\left(\pi_i(G_\alpha) > x\right)$.



(b) $log\left(P\left(\pi_i(G_\alpha) > x\right)\right)$.

Figure 5.3: Empirical PageRank distribution for a graph of half a million nodes, for different values of α, and **loop size 15**. Note that $\pi_i(G_\alpha)$ is random variable representing PageRank of a realized random graph G_α.

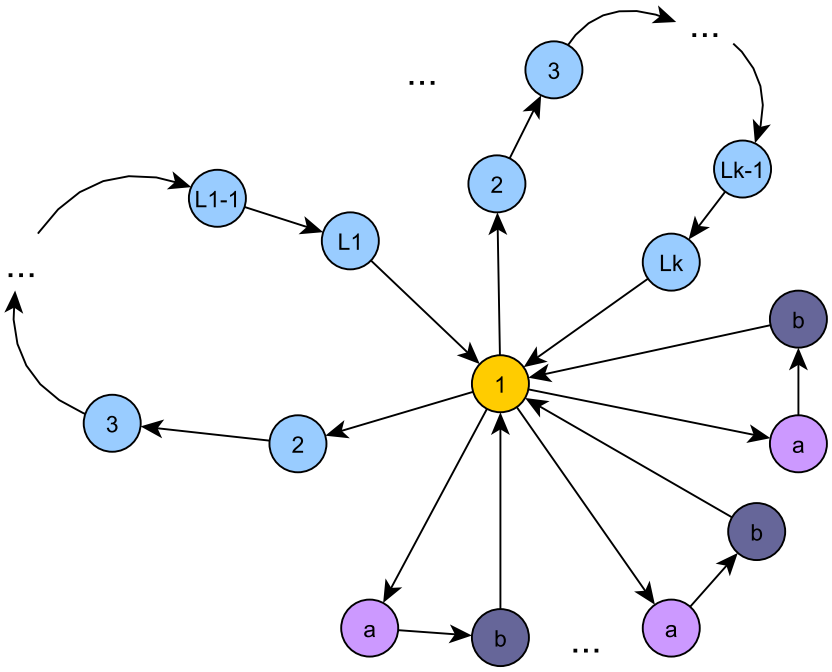


Figure 5.4: Structure of the original network.

pute PageRanks of all the nodes in this network. The equations for PageRanks are as follows:

$$\pi_1 = c\left(m_1\pi_b + \sum_{k=1}^{m_2}\pi_{L_k}^k\right) + \frac{1-c}{n}$$

$$\pi_a = \frac{c}{m_1+m_2}\pi_1 + \frac{1-c}{n}$$

$$\pi_b = c\pi_a + \frac{1-c}{n}$$

$$\pi_i^k = \begin{cases} c\pi_{i-1}^k + \frac{1-c}{n} & 3 \le i \le L^k \quad \text{and} \quad 1 \le k \le m_2 \\ \frac{c}{m_1+m_2}\pi_1 + \frac{1-c}{n} & i = 2 \quad \text{and} \quad 1 \le k \le m_2. \end{cases}$$

Where π_i^k $(2 \le i \le L^k)$ is the PageRank of the ith node in the kth cycle.

As one can see PageRank of node 1 is central in the above equations. By playing around with the above equations one can reach the following value for π_1, as the function of damping factor c:

$$A(c) = 1 - \frac{1}{m_1+m_2}\left(c^3 m_1 - \sum_{k=1}^{m_2} c^{L_k}\right) \tag{5.2}$$

$$B(c) = \frac{1}{n}\left((1-c)(m_1c^2 + m_1c + 1) - \sum_{k=1}^{m_2} c^{L_k} + m_2\right) \tag{5.3}$$

$$\pi_1(c) = \frac{B(c)}{A(c)}. \tag{5.4}$$

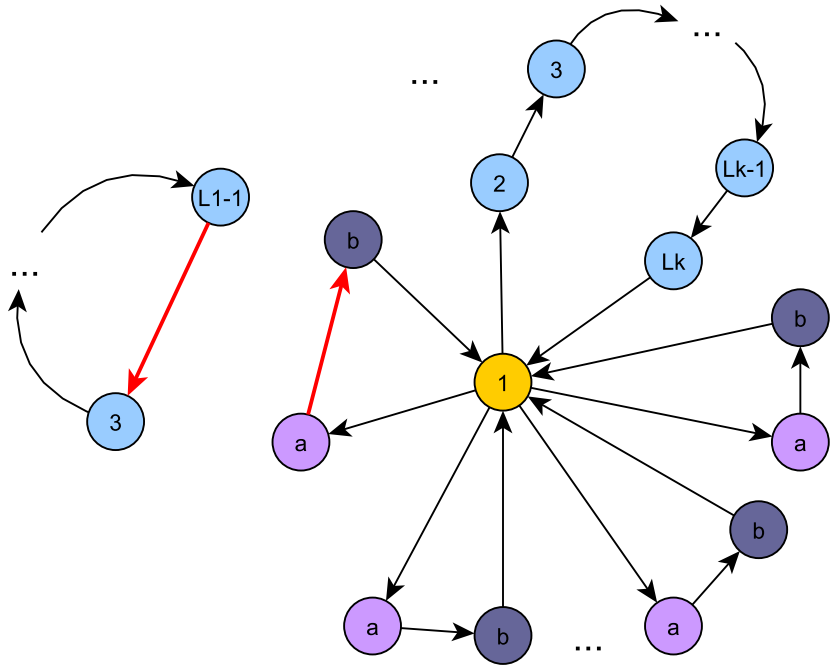


Figure 5.5: Addition of a triangle to the network.

Now assume among m_2 cycle we choose l of them and convert them to triangles connected to node 1. This process is shown in Figure 5.5 for one cycle. More specifically, in cycle lth the edges $(3,2)$ and $(L_l - 1, L_l)$ are removed and edges $(L_l - 1, 3)$ and $(2, L_l)$ are added. Note that with this process the bi-degrees of nodes remain unchanged. Now one can see that the PageRanks of nodes separated from cycles, and added to triangles connected to node 1, are the same as the (modified) PageRanks of other nodes in triangles, namely, a and b. This is illustrated in Figure 5.5 by using the same colors and labels. Without loss of generality assume that the last l cycles are converted into triangles. Now one can write the PageRank equations as follows:

$$\hat{\pi}_1 = c\left((m_1 + l)\hat{\pi}_b + \sum_{k=1}^{m_2 - l} \hat{\pi}^k_{L_k}\right) + \frac{1-c}{n}$$

$$\hat{\pi}_a = \frac{c}{m_1 + m_2}\hat{\pi}_1 + \frac{1-c}{n}$$

$$\hat{\pi}_b = c\hat{\pi}_a + \frac{1-c}{n}$$

$$\hat{\pi}^k_i = \begin{cases} c\hat{\pi}^k_{i-1} + \frac{1-c}{n} & 1 \le k \le l \quad \text{and} \quad 3 \le i \le L^k \\ \frac{c}{m_1+m_2}\hat{\pi}_1 + \frac{1-c}{n} & 1 \le k \le l \quad \text{and} \quad i = 2 \\ 1/n & l \le k \le m_2 \quad \text{and} \quad 3 \le i \le L^k - 2. \end{cases}$$

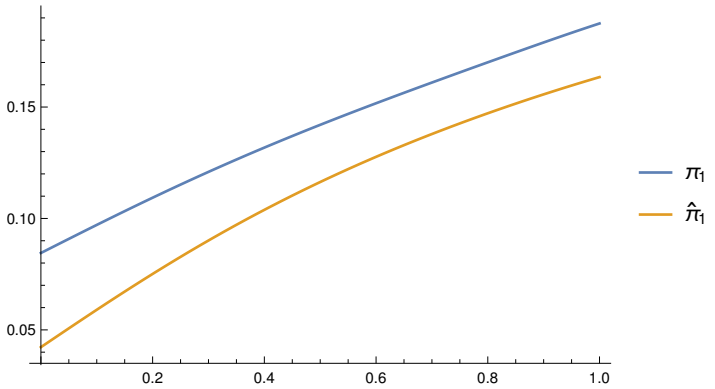


Figure 5.6: Comparison of PageRank of node 1, by introduction of triangles.

Which result in the following equation for $\hat{\pi}_1$:

$$\hat{A}(c) = A(c) - \frac{1}{m_1 + m_2}\left(kc^3 - \sum_{k=m_2-l}^{m_2} c^{L_k}\right) \tag{5.5}$$

$$\hat{B}(c) = B(c) + \frac{1}{n}\left((1-c)(lc^2 + lc + 1) + \sum_{k=m_2-l}^{m_2} c^{L_k} - l\right) \tag{5.6}$$

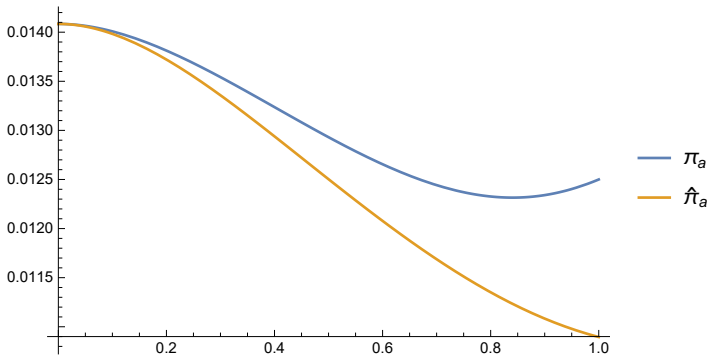
$$\hat{\pi}_1(c) = \frac{\hat{B}(c)}{\hat{A}(c)}. \tag{5.7}$$

Where $A(c)$ and $B(c)$ are given in Equations (5.2) and (5.3).

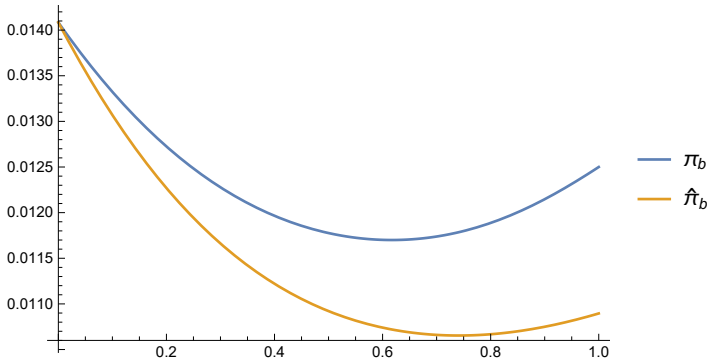
Now we can compare the change of PageRank for node 1, by changing the number of triangles connected to it. By the observations of previous section we expect a decrease in PageRank of nodes which already have high PageRank, such as node 1. This is because this node is apparently the central node, hence having the highest PageRank in the network. In order to do the comparison we consider a network with 10 triangles, and 5 cycles connected to node 1. The length of the cycles are $6, 8, 10, 12, 14$. For modification and adding more triangles, we isolates cycles of length,$10, 12, 14$, and keep the rest. Figure 5.6, illustrates the values of $\pi_1(c)$ and $\hat{\pi}_1(c)$ for all possible values of $c \in [0, 1]$. As can be seen the PageRank of node 1, as the central node with the highest PageRank in the entire network, is decreased for all values of damping factor.

Figure 5.7, shows the same comparison for nodes labeled a and b. Again one can see that PageRanks of these nodes are decreased. Note that these nodes have the highest PageRanks after node 1. Moreover, one can see how for lower values of damping factor c, PageRank of node a is more than PageRank of b. this is because this node is closer to node 1. However, for higher values of c, this node suffer more decrease in PageRank.

On the other hand, for the nodes in the cycles we have different results. Figure 5.8, demonstrates PageRanks of nodes who are closer



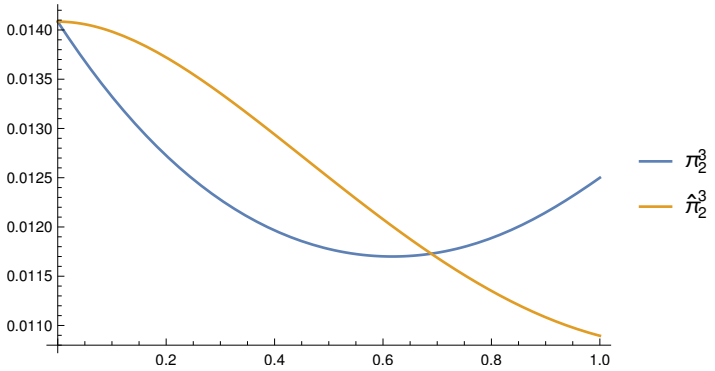
(a) Node a.



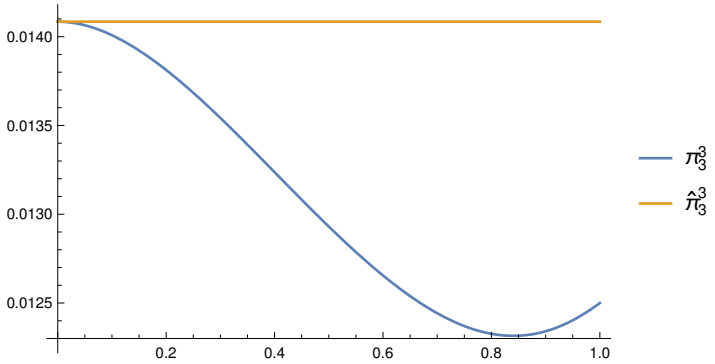
(b) Node b.

Figure 5.7: Comparison of PageRanks of different nodes a and b, by introduction of triangles.

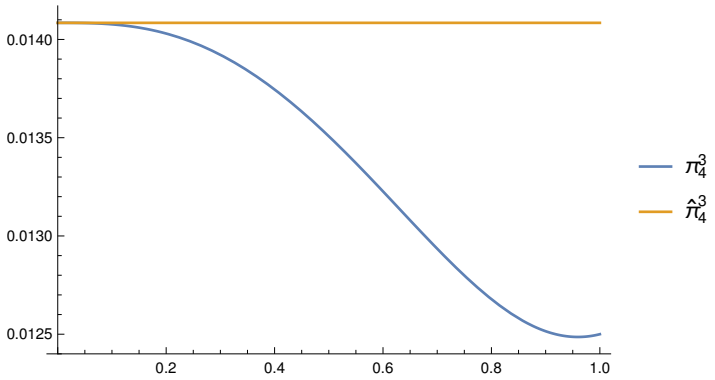
to node 1 in an arbitrary cycle, namely nodes, 2, 3, and 4. As one can see PageRanks of these nodes are increased. Note that in a cycle, nodes which are closer to the node 1, have higher PageRanks, and as we go further from node 1 PageRanks are decreased. This is the result of damping factor c. Specifically, in Figure 5.8a one can see the effect of damping factor c, and how it can cause an increase or decrease in PageRank of a node. For this case, for high values of c instead of increase we have decrease in PageRank of node 2.



(a) Node 2.



(b) Node 3.



(c) Node 4.

Figure 5.8: Comparison of PageRanks of different nodes 2, 3, and 4 of an arbitrary cycle, by introduction of triangles.

BIBLIOGRAPHY

[1] Manuel Castells. The information age, volumes 1–3: Economy, society and culture, 1999.

[2] Hamed Ghasemieh, Mohammad Ghodsi, Hamid Mahini, and Mohammad Ali Safari. Pricing in population games with semi-rational agents. *Operations Research Letters*, 41(3):226–231, 2013.

[3] Paolo Boldi and Sebastiano Vigna. Axioms for centrality. *Internet Mathematics*, 10(3-4):222–262, 2014.

[4] Sergey Brin and Lawrence Page. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks*, 56(18):3825–3833, 2012.

[5] Duncan J Watts and Steven H Strogatz. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440–442, 1998.

[6] Gábor Szabó, Mikko Alava, and János Kertész. Clustering in complex networks. In *Complex networks*, pages 139–162. Springer, 2004.

[7] Anatol Rapoport. Cycle distributions in random nets. *The bulletin of mathematical biophysics*, 10(3):145–157, 1948.

[8] Mark EJ Newman and Juyong Park. Why social networks are different from other types of networks. *Physical Review E*, 68(3):036122, 2003.

[9] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[10] Remco Van Der Hofstad. Random graphs and complex networks. *Available on http://www. win. tue. nl/rhofstad/NotesRGCN. pdf*, page 11, 2009.

[11] Ningyuan Chen, Mariana Olvera-Cravioto, et al. Directed random graphs with given degree distributions. *Stochastic Systems*, 3(1):147–186, 2013.

[12] Yana Volkovich. *Stochastic analysis of web page ranking*. PhD thesis, Enschede, April 2009.

[13] N Litvak, WRW Scheinhardt, and Y Volkovich. In-degree and pagerank: Why do they follow similar power laws. *Internet Mathematics*, 4(2-3):129–298, 2007.

[14] Taher Haveliwala and Sepandar Kamvar. The second eigenvalue of the google matrix. Technical Report 2003-20, Stanford InfoLab, 2003.

[15] Cleve Moler. The worlds largest matrix computation, 2002.

[16] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[17] Amy N Langville and Carl D Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2004.

[18] Pavel Berkhin. A survey on pagerank computing. *Internet Mathematics*, 2(1):73–120, 2005.

[19] Monica Bianchini, Marco Gori, and Franco Scarselli. Pagerank and web communities. In *Web Intelligence*, pages 365–371, 2003.

[20] Sepandar D Kamvar, Taher H Haveliwala, Christopher D Manning, and Gene H Golub. Extrapolation methods for accelerating pagerank computations. In *Proceedings of the 12th international conference on World Wide Web*, pages 261–270. ACM, 2003.

[21] Claude Brezinski and Michela Redivo-Zaglia. The pagerank vector: properties, computation, approximation, and acceleration. *SIAM Journal on Matrix Analysis and Applications*, 28(2):551–575, 2006.

[22] Nelly Litvak, Werner RW Scheinhardt, and Yana Volkovich. Probabilistic relation between in-degree and pagerank. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 72–83. Springer, 2006.

[23] Quansheng Liu. Asymptotic properties and absolute continuity of laws stable by random weighted mean. *Stochastic processes and their applications*, 95(1):83–107, 2001.

[24] Tom Britton, Maria Deijfen, and Anders Martin-Löf. Generating simple random graphs with prescribed degree distribution. *Journal of Statistical Physics*, 124(6):1377–1397, 2006.

[25] Claude Berge and Edward Minieka. *Graphs and hypergraphs*, volume 7. North-Holland publishing company Amsterdam, 1973.

[26] Pim van der Hoorn. *Asymptotic analysis of network structures: degree-degree correlations and directed paths*. PhD thesis, 2016.

[27] Mark EJ Newman. Random graphs with clustering. *Physical review letters*, 103(5):058701, 2009.