University of Nice - Sophia Antipolis – UFR Sciences
École Doctorale STIC

THESIS

Presented to obtain the title of :

Doctor of Philosophy of University of Nice - Sophia Antipolis (France)

Speciality : INFORMATICS

by

Danil NEMIROVSKY

Team : MAESTRO – INRIA Sophia Antipolis - Méditerranée

MONTE CARLO METHODS AND MARKOV CHAIN BASED APPROACHES FOR PAGERANK COMPUTATION

Thesis is directed by Konstantin AVRACHENKOV

Defence in INRIA on 2nd July 2010, at 16:00 in front of the council including :

President :	Philippe	NAIN	INRIA Sophia Antipolis - Méditerranée
Advisor :	Konstantin	AVRACHENKOV	INRIA Sophia Antipolis - Méditerranée
Reviewers :	Ilse	IPSEN	North Carolina State University
	Paolo	BOLDI	University of Milano
Examinators :	Vladimir	DOBRYNIN	St.Petersburg State University
	Jerzy	FILAR	University of South Australia
	Bruno	GAUJAL	INRIA Grenoble - Rhône-Alpes

Université de Nice - Sophia Antipolis – UFR Sciences
École Doctorale STIC

THÈSE

Présentée pour obtenir le titre de :

Docteur en Sciences de l'Université de Nice - Sophia Antipolis (France)

Spécialité : INFORMATIQUE

par

Danil NEMIROVSKY

Équipe d'accueil : MAESTRO – INRIA Sophia Antipolis - Méditerranée

DES APPROCHES POUR LE CALCUL DU PAGERANK FONDÉES SUR LES MÉTHODES DE MONTE CARLO ET CHAÎNES DE MARKOV

Thèse dirigée par Konstantin AVRACHENKOV

Soutenance à l'INRIA le 2 juillet 2010, à 16h00 devant le jury composé de :

Président :	Philippe	NAIN	INRIA Sophia Antipolis - Méditerranée
Directeur :	Konstantin	AVRACHENKOV	INRIA Sophia Antipolis - Méditerranée
Rapporteurs :	Ilse	IPSEN	North Carolina State University
	Paolo	BOLDI	University of Milano
Examinateurs :	Vladimir	DOBRYNIN	St.Petersburg State University
	Jerzy	FILAR	University of South Australia
	Bruno	GAUJAL	INRIA Grenoble - Rhône-Alpes

Thèse

Des Approches pour le Calcul du PageRank Fondées sur les Méthodes de Monte Carlo et Chaînes de Markov

Monte Carlo Methods and Markov Chain Based Approaches for PageRank Computation

Danil Nemirovsky

July 2010

MONTE CARLO METHODS AND MARKOV CHAIN BASED APPROACHES FOR PAGERANK COMPUTATION

by
Danil Nemirovsky
Advisor of the thesis: Konstantin Avrachenkov
MAESTRO, INRIA Sophia Antipolis, France

ABSTRACT

Nowadays a lot of data has become readily available due to the fast development of storage devices and telecommunication systems. Now the problem of effective search of required information has arisen but because of the huge volume of information, even a filtered result reflecting only requested data is still large. In such situation, the results should be sorted according to some criteria, one of which can be authoritativeness. In the case of the World Wide Web, when a search over Web pages is performed, authoritativeness can be measured by the PageRank algorithm. The main idea of the PageRank algorithm is that the authoritativeness of a page depends on the number and quality of in-coming links to that page. The PageRank algorithm is based on the model of a random surfer that moves from the current page with some probability to an arbitrary page on the Web, or follows the out-going links of the current page with a complementary probability. This complementary probability is called the damping factor. The behaviour of the random surfer can be viewed as a random walk on the Web graph, a directed graph whose nodes are the Web pages and arcs are hyper-links between them. If the surfer chooses an arbitrary page from all the Web pages with some particular probability distribution reflecting her preferences, PageRank is called Personalized PageRank. The behaviour of the random surfer is captured analytically by an ergodic Markov chain, and PageRank is the stationary distribution of the Markov chain.

As for any ergodic Markov chain and its stationary distribution, the Power method can be used to calculate PageRank, but the size of the Web is so big that the speed with which the computation performed by the Power Iteration method, is not satisfactory. Some accelerating methods of the PageRank computation are considered, and, in particular, aggregation-disaggregation methods. Full aggregation-disaggregation method and a partial aggregation-disaggregation methods are discussed in details. Equivalence conditions of the two methods, in the sense that the two methods give an identical sequence of intermediate results, are discovered. New mixed aggregation-disaggregation algorithm is proposed that possesses better convergence of the partial aggregation-disaggregation method and is less computationally consuming than the full aggregation-disaggregation method.

The choice of the damping factor is not evident, although a lot of attention was attracted to this problem in the literature. Parameter-free measures as alternative to PageRank are proposed and analyzed. The measures are based on quasi-stationary distributions over the pages belonging to the Extended Strongly Connected Component (ESCC) of the Web graph, assuming that the dangling pages link to all the pages of the Web, as it is usually done in the Markov chain model applied to PageRank. Four quasi-stationary distributions having intuitive implications are considered. It is concluded that the quasi-stationary distributions are close to each other and to the PageRank values of ESCC according to Kendall tau and angular measures.

Although iterative methods of the PageRank calculation are extensively developed, asides from them, there are other probabilistic methods aiming for this purpose. Monte Carlo methods applied to the PageRank computation are discussed. Two versions of the Monte Carlo methods proposed by other authors and three other versions proposed in the thesis are compared. In general, one run of a Monte Carlo method simulates a random walk on the Web graph sampling the path of the random surfer. It is analytically demonstrated that Monte Carlo methods which keeps all the information about the pages visited by a random walk outperforms analogous methods that keep only the last visited page. It is

concluded that starting a random walk iteratively at each page is better than choosing the starting page randomly. It is shown by experiments that already after one iteration a good approximation of PageRank values can be obtained for popular pages.

In many cases actual values of PageRank are not important. Monte Carlo method applied to Personalized PageRank is analyzed with the aim to discover the ranking of the number of pages having high Personalized PageRank values.

Moments of an absorbing Markov chain are considered. First moments and non-mixed second moments of the number of visits are determined and can be easily expressed in a matrix form using the fundamental matrix of the absorbing Markov chain. Since the representation of the mixed moments of higher orders in a matrix form is not straightforward, if ever possible, tensor approach to the mixed high-order moments is proposed and compact closed-form expressions for the moments are discovered.

ACKNOWLEDGMENTS

I thank Konstantin Avrachenkov for his supervision which was exceptionally attentive. When I was in need to discuss my research, I knew that Konstantin would never refused to have a discussion. All the advice given me by Konstantin improved my writing, presentation and thinking skill. I am very grateful to Vladimir Dobrynin for a lot of very useful discussions which enriched my understanding of a general picture of my area of research. I am very thankful to my collaborators, Nelly Litvak, Natalia Osipova, Vivek Borkar and Konstantin Avrachenkov who did a huge contribution into the papers that became the chapters of the thesis. I thank Philippe Nain for the warm and collaborative atmosphere established in MAESTRO project. I am grateful to Ephie Deriche for her patience in solving problems with my travels and conference registrations which I have created quite a lot during my PhD program. I thank Giovanni Neglia as an inexhaustible source of jokes and good mood. I thank Abdulhalim Dandoush for being the best neighbour of the office I ever had. I am very grateful to Abdulhalim and Soheir for accommodating me when I was leaving France for Tokyo to do an internship there. I am very thankful to Corinne Touati for her help with translation of the summary of the thesis into the French language. I thank Sophie van Dommelen for her French lessons in FJT and MJC. Sophie opened me the pleasure of speaking in French, and I use each opportunity to talk in this language with sincere appreciation to Sophie.

I would like to thank Natalia Osipova for her support in my decision to change my master thesis advisor. If I had not done it, I would write a totally different (if any) dissertation. My master work became a chapter of this thesis. My heartfelt gratitude to Tatiana Selchenkova who was setting off a “dissertation” charge within me and it did not let me to give it up in the middle. I am very grateful to Vera Ryankel who helped me to relearn the filling of love, including the love to the science. It is important when one writes a thesis in the applied mathematics. I warmly thank Lora Vysotskaya for the wonderful week which changed me a lot. Lora helped me to fill the moment when falsification and sophistic evolve into a philosophy. The events of that week brought me to the idea that became the foundation of the last chapter of this thesis. I am very grateful to Marina Sokol who gave me evidences in sake of which I can continue my research. I thank Maria Babaeva and Julia Krasnokutskaya for spending time on-line and talking with me in Russian which hepled me to keep in touch with my homeland during my intership in Japan. Actually, I would like to thank Maria and Julia for absolutely different things, but I cannot find words to express the difference. I shall say it them personally, while I mention them together here. I am very thankful to Alexander Kremer for the nights spent together creating a trinary processor and adaptive control to TapMania. I really needed it to

refresh my mind. My warm gratitude to Yana Kremer for her point of view on the life which alter my way of thinking.

The last, but not least, who I would like to thank is my family: mother, father, grandmother, my brother Stepan, my sisters Zoya, Alla and Anya and other relatives. Thank you very much for your permanent support in my travelling through the World.

All the people mentioned (and somebody who I did not mention) above influenced on me a lot during the last years when I was writting my PhD thesis. Thank you very much! Please, accept my warmest gratitude and appreciation.

Danil Nemirovsky
Danil.Nemirovsky@gmail.com
Sophia Antipolis, France

To my muses

CONTENTS

FIGURES

TABLES

1

INTRODUCTION

"All men by nature desire to know" (by Aristotle) [8]. To survive, a prehistoric man [39] had to pick vegetables, fruits and mushrooms and to hunt after animals and birds, and he needed only limited amount of information to do these actions. He obtained the information from his predecessor, who taught him how to pick and how to hunt. During the primeval time there was no such a problem of accessing to the information if it was available in principle. The only issue was to learn it quickly, before the predecessor died, since, in that case, the source of information was lost and knowledge had to be rediscovered again, but developing the means to survive, a man needs to deal with the increasing volume of information. Constructing a bronze axe is more complicated task than constructing a stone one, but the bronze axe is lighter, sharper and, thus, more practical.

As centuries passed by, the amount of the discovered information became so large that a single human was not able to keep it in his/her mind. A means to retain the information on an external medium had to appear. A man invented, firstly, cuneiform, papyrus, writing on animal skin and, later, paper and ink, and, thus, the age of scrolls and books has arrived. Scrolls and books were collected in libraries alike one of the most famous ones - the Library of Alexandria [72]. Having from 400 up to 7000 thousands of scrolls and books, we call them items, the issue of effective access and quick search arise. Nobody could know the whole content of the items, but someone could keep references to the content that required classification of the items according to the areas of the knowledge which had to be performed manually at that time. One of the first famous classification was the division of all the knowledge into "Physics" and "Metaphysics" done by Aristotle [8, 9]. But one advantage of the limited amount of information is that once someone defined what he needed, he got a few items related to his needs. In that case, the order of the corresponding to the needs items in which the items are presented

to the interested person does not matter, since the number of the items is so small that they are all observable in a reasonable time.

However, after some more centuries passed by and electronic media came to our life, particularly with appearance of the Internet, the increase of the volume of information became significant. It was a revolution when the Internet and the World Wide Web appeared since it gave a birth to the enormous universe of information, but it does not necessarily mean that the information is accessible. The information became accessible for a lot of people only readily, but, in the same time, because of the lack of structure, a person can easily lose himself in this ocean of information. The information is there, you need just to stretch your hand to it, but you do not know what direction you should stretch your hand to. A kind of classification of the information is then required to lead a user, but the volume is so huge that it is unimaginable that it can be classified manually. To address this problem, some companies developed search engines, specially designed computer system including hardware and software components, which have ability to collect a huge amount of data from the World Wide Web, process it constructing indices to perform fast search and provide references to it to users upon a request that should carry user information needs [78, 84]. In contrast to ancient libraries with scrolls and books, search items became electronic documents, Web pages. As a request, one understands a list of key words that an item should contain, but a request can be ambiguous and may not reflect user information needs. On one hand, the request language is too poor to express perfectly the information needs of a user. On the other hand, the request language should be simple to enable more people to use it. Given that the request is the only data available to a search engine, it is not able to provide a perfect response. By the response, one understands a list of found items which are considered to be relevant to the request and able to meet the user information needs, from search engine point of view. Since a search engine is not able to reveal exact value of relevance, it gives just an estimation. The found items are sorted according to the decreasing order of the estimated relevance to firstly present more relevant results to a user. Relevance is a complex notion having several dimensions [38]; results are sorted according to values of one of them or a weighted sum of the values. There are a lot of factors having influence on the relevance value, and one of them is the authoritativeness of an item.

Authoritativeness of an item or a document is the quality of trustworthiness and reliability of the item which makes it dependent on the users of the item. The value of authoritativeness can be considered as an aggregated opinion of a community of users about the item. The way to determine the value of the authoritativeness depends on the community. For example, science citation index is one of the means to measure the authoritativeness of scientific publications.

We shall consider the problem of search in a global collection of documents (items) such as the Internet and the World Wide Web. In the contrast to traditional Information Retrieval in the traditional text collections, the Web is much more dynamic, massive and less coherent [6].

That is why a particular special means are required to determine the authoritativeness of a Web pages based on the hyper-text structure of the World Wide Web. As a means to calculate the authoritativeness, we consider the *PageRank* algorithm, a method invented by the founders of Google [71, 27]. The major idea of the method is that the authoritativeness of a Web page depends on the number and the quality of hyper-link to the page placed on other Web pages. A hyper-link to a Web page is called an in-coming link. Intuitively, the larger the number of in-coming links to a Web page is, the higher the authoritativeness of the page is. But the authoritativeness of a Web page where the in-coming link placed also plays an important role. In contrast to scientific citation index, PageRank algorithm takes it into account.

Let us introduce the PageRank algorithm formally. We consider the World Wide Web as a directed graph. A Web page is a node and a hyper-link is an arc where the tail of the arc is the Web page where the hyper-link is placed and the head of the arc is the Web page which the hyper-link refers to. This directed graph is called the *Web Graph*. We shall use term "page" and "node" interchangeably in the further discussion. Let us assume that there are n Web pages on the World Wide Web and let us denote by $\pi(\mathcal{P})$ the PageRank value of Web page $\mathcal{P}$. We calculate the PageRank value by the following formula:

$$\pi(\mathcal{P}) = \frac{(1-c)}{n} + c(\pi(\mathcal{T}_1)/l(\mathcal{T}_1) + \ldots + \pi(\mathcal{T}_m)/l(\mathcal{T}_m)), \tag{1.1}$$

where $0 < c < 1$, $\mathcal{T}_1, ..., \mathcal{T}_m$ are Web pages having out-going links to $\mathcal{P}$, $l(\mathcal{T}_i)$ is the number of out-going links from page $\mathcal{T}_i$, $\pi(\mathcal{T}_i)$ is the PageRank value of page $\mathcal{T}_i$. One can see that PageRank of a page depends on the PageRank value of pages linking to that page. We can take into account the quality of links by that way. A page having more in-coming links from pages with high PageRank values will have high PageRank. At the same time, a page gains only a fraction of PageRank values of pages which link to the page, and the fraction coefficient is the constant c. One can see that the minimum PageRank value that can be obtained by a page, is $\frac{1-c}{n}$ if the page has no in-coming links. That is why constant c is called a *damping factor*.

If someone sum up PageRank value for all the pages on the Web, he/she get unity, which means that PageRank values can be considered as a probability distribution on the set of Web pages. This allows us to give a probabilistic interpretation of equation (1.1) exploiting the following imaginary experiment. Let us imagine a surfer on the Web staying at one of the Web pages. With probability $1 - c$ she jumps to an arbitrary Web page and, with probability c, she chooses to follow one of the out-going links of the page at which she is at the moment. The particular out-going link is chosen uniformly from the set of out-going links the current Web page. If we imagine that a lot of surfers, spread uniformly on the Web, follow the behaviour described above, then the number of the surfers on a Web page will be proportional to its PageRank value after a while. It implies that the probability to find a surfer on a Web page is the PageRank value of the Web page. It is clear that the higher the PageRank value is, the

more visited the page is. Thus, PageRank can be considered as the result of a particular random walk on the Web Graph and the result is a centrality measure defined on the Web Graph that determines the relative importance of a node within the graph.

The behaviour of the imaginary surfer can be modelled by a Markov chain whose state space is the set of all the Web pages. We give formalization of the behaviour from [61, 58, 62]. Let us enumerate the Web pages by integer numbers from 1 to n and define the $n \times n$ hyperlink matrix H such that

$$h_{ij} = \begin{cases} 1/d_i, & \text{if page } i \text{ links to page } j, \\ 0, & \text{otherwise,} \end{cases} \tag{1.2}$$

for $i, j = \overline{1, n}$[1], where d_i is the number of out-going links from page i. If a page has no out-going links, then the page is called a *dangling page*. A row of matrix H corresponding to a dangling page is the zero row. Matrix H is not stochastic matrix and cannot be used in modeling by a Markov chain. We fix it by assuming that a dangling page refers to all the pages on the Web. These imaginary links are called *artificial links*. Let us introduce a column vector a which element $a_i = 1$ if the i^{th} row of matrix H corresponds to a dangling page and 0 otherwise. Let us define a stochastic matrix P by

$$P = H + \frac{1}{n} a \mathbf{1}^T. \tag{1.3}$$

The matrix corresponding to the surfer behaviour is called the *Google matrix* and expresses as

$$G = cP + \frac{1-c}{n} E, \tag{1.4}$$

where E is the matrix of unities. The summand cP corresponds to the surfing from one page to another by out-going links including artificial links, and summand $\frac{1-c}{n}E$ corresponds to jumping to an arbitrary Web page. It is easy to check that equality (1.1) can be written in matrix form:

$$\pi = \pi G, \tag{1.5a}$$

$$\pi \mathbf{1} = 1, \tag{1.5b}$$

where $\mathbf{1}$ is a column vector of appropriate dimension whose all the entries are equal to one. The row vector π consists of the PageRank values of the enumerated Web pages and is called PageRank vector, or simply PageRank. The PageRank vector is the eigenvector of the Google matrix corresponding to its principal eigenvalue $\lambda_1 = 1$. One can see that if we consider the Google matrix as a transition matrix of a Markov chain, then PageRank is the stationary probability distribution of the Markov chain. The PageRank vector can be found from (1.5) as [21, 58, 68]

$$\pi = \frac{1-c}{n} \mathbf{1}^T (I - cP)^{-1}, \tag{1.6}$$

[1] By $i = \overline{1, n}$, we mean $i \in \{1, \ldots, n\}$. I know that such a notation is not a standard for English-language scientific literature, but the intensive usage of the notion in Chapter 6 makes me to choose as short notation as possible.

where I is an identity matrix.

Above we assumed that the surfer has not any preference in choosing a Web page when she jumps onto an arbitrary page, but this model is not adequate. One user may prefer sport, another - news, and a third one - arts. We can take into account such preference by a dangling node vector w and a personalization vector v. The vectors w and v are probability distributions over all the Web pages, and each entry is a probability to choose a Web page according to the user preferences. Stochastic matrix P is written, then, in the following way:

$$P = H + aw. \tag{1.7}$$

We write the Google matrix as

$$G = cP + (1 - c)\mathbf{1}v, \tag{1.8}$$

and the PageRank vector, which is called Personalized PageRank in that case, can be calculated as

$$\pi = v(1 - c)(I - cP)^{-1}. \tag{1.9}$$

One can see that the personalization vector and the dangling node vector in (1.4), (1.6) are the uniform distributions. Personalized PageRank is linear in its personalization vector, if the dangling node vectors stay the same, as it stated in the following theorem [46].

Theorem 1.1 (Linearity) *Given two arbitrary Personalized PageRanks π_1, π_2 and v_1, v_2 are their corresponding personalization vectors. Then, for any constants $\alpha_1, \alpha_2 > 0$ such that $\alpha_1 + \alpha_2 = 1$*

$$\alpha_1\pi_1 + \alpha_2\pi_2 = c\,(\alpha_1\pi_1 + \alpha_2\pi_2)\,P + (1 - c)\,(\alpha_1 v_1 + \alpha_2 v_2) \tag{1.10}$$

The theorem let us consider Personalized PageRank for basis personalization vectors e_i^T, where e_i is a column vector of zero which i^{th} entry is equal to one, and one can construct any Personalized PageRank using these basis Personalized PageRank vectors.

We shall use term PageRank meaning both usual PageRank defined in (1.6) and Personalized PageRank defined in (1.9). If we speak about usual PageRank or Personalized PageRank particularly, we mention it in the cases when it is not evident.

Calculation of PageRank is very computationally consuming since the dimension of the PageRank vector is huge. The dimension of PageRank is the number of the indexed Web pages on the World Wide Web. Authors [20] conjecture that the size of the static public Web as of November 1997 was at least 200 million documents. In [40] authors updated the estimated size of the indexable Web to at least 11.5 billion pages (in English) as of the end of January 2005. Site [34] reports about 25 billion indexed Web pages on the World Wide Web on 18 march 2009. It is evident that an index of a particular search engine is just a part of the mentioned values, but still Google index covers about 68.2% of the indexed Web [40]. The dimension size of

PageRank makes it impractical to use direct methods, such as Gauss elimination, to determine PageRank. Some approximating methods have to be used.

According to information which is available publicly, Google is using the Power method to calculate PageRank [71]. Starting from the initial approximation as the uniform distribution vector $\pi^{(0)} = (1/n)\mathbf{1}^T$, the k^{th} approximation vector is calculated by

$$\pi^{(k)} = \pi^{(k-1)}G, \quad k \geqslant 1. \tag{1.11}$$

The method stops when the required precision ε is achieved. The number of flops needed for the method to converge is of the order $\frac{\log \varepsilon}{\log c}\, nnz(P)$, where $nnz(P)$ is the number of non-zero elements of the matrix P [58].

Although, the Power method is proven to converge, it can still be too slow for practical use [57]. Some accelerating methods were proposed in [49,50,44,59]. Acceleration in [49,50] is achieved by the adaptive computation of the PageRank values and using extrapolation methods to adjust intermediate results. Another technique, exploited in [44, 59], is the reduction of the dimension of vectors and matrices taking part in the computation followed by reconstruction the full-dimensional PageRank. These methods are discussed below in Chapter 2. The contribution of Chapter 2 is the equivalence condition of two aggregation-disaggregation methods that allows to compose a novel method possessing advantages of the mentioned methods and avoiding their drawbacks.

The choice of the damping factor is not evident. The value of the damping factor attracted a lot of attention in the literature [14,23,30], but no single value is agreed to be optional. Google chose damping factor c equal to 0.85 claiming that it relates to a surfer behaviour [71]. In that case, the surfer does about 6 clicks on hyper-links before jumping to an arbitrary page and with it the damping factor has an influence on the convergence rate of the iterative methods used to calculate PageRank [61]. For example, the Power method converges faster for the smaller values of the damping factor. On the other hand, the less the damping factor is, the less information is kept in the Google matrix, which leads to doubts if PageRank reflects Web structure with small values of the damping factor. At the same time, PageRank is more robust with smaller c, that is, one can bound the effect of out-going links of a page on PageRank of other groups [21] and on its own PageRank [13]. Thus, the value of damping factor can have drastic influence on ranking produced by the PageRank algorithm. All this leads to the idea to avoid a specific choice of the damping factor [18, 22, 31, 81]. In Chapter 3 we propose damping-factor-free centrality measures based on quasi-stationary distributions as alternative to PageRank.

Although iterative methods of the PageRank calculation are highly developed, aside from them, there are other probabilistic methods aiming for this purpose. One of the feature of the iterative methods is that the components of the PageRank vector converge slowly for pages with

high PageRank [49]. But we are interested often only in pages having high PageRank values. In that case, using probabilistic algorithms, such as Monte Carlo methods, is preferable, since, as it is shown in Chapter 4, one iteration is enough to obtain good approximation of PageRank values for the pages having high PageRank. Applied to the PageRank computation, the Monte Carlo methods model the behaviour of a surfer and approximate PageRank values by simulation of the user traverse of the Web. To the best of our knowledge, there are two works devoted to Monte Carlo methods applied to PageRank [25, 37].

All the previous discussion was devoted to approximating PageRank vector, and, as a more general case, Personalized PageRank, using a number of methods, but there are a lot of applications where only the nodes having high value of Personalized PageRank matter. The applications include search for related entities [29, 85], finding local cuts in graphs [4, 5] and clustering large hyper-text document collections [11]. Chapter 5 is devoted to the analysis of Monte Carlo method applied to Personalized PageRank with the aim to discover the ranking of the number of pages having high Personalized PageRank values.

In the mathematical analysis, used in the analysis of Monte Carlo methods for Personalized PageRank, mixed high-order moments of the number of visits are exploited. Since mixed high-order moments are hard to express in matrix form, we used a tensor approach to calculate them. Compact close-form formule are obtained and presented in Chapter 6.

2

AGGREGATION-DISAGGREGATION METHODS FOR PAGERANK CALCULATION

2.1 Summary

The calculation of PageRank is a computationally very expensive operation. Since high dimension vectors and matrices are used in the computation, direct methods are impractical. Google uses the Power method to determine PageRank, but it may converge slowly. To accelerate the calculation of PageRank, some methods were developed using adaptive computation of the PageRank values and extrapolation to adjust intermediate results. Another technique is to reduce the dimension of the vectors and matrices taking part in the computation and to reconstruct the full-dimensional PageRank which is used in aggregation-disaggregation methods. Two aggregation-disaggregation methods to find PageRank and (more generally) the stationary distribution of a Markov chain are considered. An estimation of the convergence rate of one of them is discovered. The conditions of the equivalence of the methods are proposed and it allows to reduce the computational costs in the calculation of the stationary distribution.

2.2 Introduction

The PageRank calculation is a computationally very expensive operation. Because direct methods are very time consuming [76, §2], Google uses the Power method to compute the PageRank vector [71], but the convergence rate can be low [57]. Some accelerating methods were proposed in [49,50,44,59,48,64]. The authors of [49,50] accelerated the computation of the PageRank vector by modifications of the Power method, while the authors of [44,59,48,64] used an aggregation-disaggregation approach.

In [49], authors observed that the convergence rate of the PageRank values of individual pages during application of the Power method is not uniform, and, moreover, the rate is generally lower for pages having a high PageRank. They designed a simple adaptive algorithm in which once a PageRank value converged, it is fixed and is not recomputed in further iterations. This speeds up PageRank computation by nearly 30% in large scale empirical studies.

In [50], the authors extrapolated new approximation of the PageRank vector using two previous approximations in Aitken Extrapolation and three previous approximations in Quadratic Extrapolation. The authors claimed that the time saving is about 38% for Aitken Extrapolation and about 23% for Quadratic Extrapolation.

We consider aggregation-disaggregation methods below. Aggregation-disaggregation methods (A/D methods) for the computation of the PageRank vector use the decomposition of the set of pages which we denote by $\mathcal{I}$. Let us assume that the set $\mathcal{I}$ is decomposed into $N \leqslant n$ non-intersecting sets $\mathcal{I}^{(i)}$, $i = \overline{1,N}$, such that

$$\begin{aligned} &\mathcal{I}^{(1)} = \{1,\dots,n_1\},\\ &\mathcal{I}^{(2)} = \{n_1+1,\dots,n_1+n_2\},\\ &\vdots\\ &\mathcal{I}^{(N)} = \left\{\textstyle\sum_{i=1}^{N-1} n_i + 1,\dots,\sum_{i=1}^{N} n_i\right\}, \end{aligned} \tag{2.1}$$

where $\sum_{i=1}^{N} n_i = n$.

According to the decomposition of the set of pages the transition matrix can also be partitioned as follows:

$$P = \begin{pmatrix} P_{11} & P_{12} & \dots & P_{1N}\\ P_{21} & P_{22} & \dots & P_{2N}\\ \vdots & \vdots & \ddots & \vdots\\ P_{N1} & P_{N2} & \dots & P_{NN} \end{pmatrix}, \tag{2.2}$$

where P_{ij} is a block with dimension $n_i \times n_j$. In the same manner the Google matrix G can be

presented in blocks,

$$G = \begin{pmatrix} G_{11} & G_{12} & \dots & G_{1N} \\ G_{21} & G_{22} & \dots & G_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ G_{N1} & G_{N2} & \dots & G_{NN} \end{pmatrix}. \tag{2.3}$$

Following the partitioning of the Google matrix, the PageRank vector is partitioned into components:

$$\pi = (\pi_1, \pi_2, \dots, \pi_N), \tag{2.4}$$

where π_i is a row vector with $dim(\pi_i) = n_i$.

All aggregation methods use an *aggregated matrix* A. Each element of matrix A corresponds to a block of matrix G, i.e. $a_{ij} \leftrightarrow G_{ij}$. Typically, the elements of the matrix A are formed as $a_{ij} = \zeta_i G_{ij} \mathbf{1}$, where ζ_i is a probability distribution vector and $\mathbf{1}$ is a vector of unities. We call the vector ζ_i the *aggregation vector*. Each aggregation method forms the aggregated matrix in its own way using different probability distributions as the aggregation vectors and different partitioning. One can consider the aggregated matrix as a transition matrix of a Markov chain with a state space formed by the sets of pages.

The convergence rate of an aggregation method depends on the choice of the decomposition. Typically, an aggregation method converges faster than the Power method if the off-diagonal blocks P_{ij} are close to a zero matrix. It means that the random walk performed by the transition matrix G most likely stays inside sets $\mathcal{I}^{(i)}$ and leaves with small probability.

In the following discussion, the aggregation methods are applied to the Google matrix (1.4) and the PageRank vector (1.6), but some of them can be applied to a general (irreducible or primitive) stochastic matrix P and its stationary probability distribution. Thus, we consider the vector π as a stationary distribution of matrix P if we do analysis of a general Markov chain and its transition matrix, and we consider vector π as PageRank vector (1.6) if we exploit the particular structure of the Google matrix (1.4).

The chapter is organized as follows. In the next Section 2.3, we review a number of aggregation-disaggregation algorithms, and, particularly, the full aggregation-disaggregation method (FAM) and the partial aggregation-disaggregation method (PAM). In Section 2.4, we analyze the relationship between the stationary distributions of aggregated matrices of the FAM and the PAM algorithms, and, in Section 2.5, we discover the equivalence conditions of the FAM and PAM algorithms and give the description of a new mixed aggregation-disaggregation method. Then, in Section 2.6 we consider some properties of the mixed aggregation-disaggregation method and discuss related works. Section 2.7 concludes the chapter.

2.2.1 Notation

Let A and B be $n \times n$ real matrices, where $A = (a_{ij})_{i,j=\overline{1,n}}$ and $B = (b_{ij})_{i,j=\overline{1,n}}$. If $a_{ij} \geqslant b_{ij}$, $\forall i,j = \overline{1,n}$, we write $A \geqslant B$. If $A \neq B$, we write $A > B$. If $a_{ij} > b_{ij}$, $\forall i,j = \overline{1,n}$, we write $A \gg B$. Besides these notations, we need to introduce a notation for normalization. Let us denote by $[a]$ a normalization of a positive row vector a, i.e. $[a] = \frac{a}{a\mathbf{1}}$, where $\mathbf{1}$ is a column vector of unities of an appropriate dimension.

2.3 Aggregation-disaggregation algorithms

Let us review some aggregation-disaggregation algorithms to give a picture of the area that we discuss. We start by the simplest but not trivial case.

2.3.1 Block-diagonal case

Let us consider the case when all the blocks excluding the diagonal ones are zero [12], i.e.

$$P = \begin{pmatrix} P_{11} & 0 & \dots & 0 \\ 0 & P_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & P_{NN} \end{pmatrix}.$$

Since P is a stochastic matrix, then all P_{ii} are stochastic. For the i^{th} block define the Google matrix

$$G_{ii} = cP_{ii} + (1-c)(1/n_i)\mathbf{1}\mathbf{1}^T,$$

where the vector $\mathbf{1}$ has an appropriate dimension. Let vector π_i be the PageRank vector of G_{ii},

$$\pi_i = \pi_i G_{ii}.$$

Then the PageRank vector π for Google matrix (1.4) is expressed by

$$\pi = \left(\frac{n_1}{n}\pi_1, \frac{n_2}{n}\pi_2, \dots, \frac{n_N}{n}\pi_N\right).$$

The block-diagonal structure of the matrix P allows us to produce the computation of each component of the PageRank vector in an absolutely independent way from the other components, although, the revealing such structure of matrix P will require some time and computational resources. We presented this case to illustrate the idea that knowledge of particular structure of Web graph can be useful and lead to more efficient computations. The aggregation-disaggregation algorithms presented below exploits the idea of partitioning further.

2.3.2 BlockRank Algorithm

The next method exploits the site structure of the Web. It is assumed that Google matrix defined by (1.4), i.e. the method is used for ordinary unpersonalized PageRank. According to the experiments made by the authors of [48], the majority of links are links between pages inside Web sites. Hence, we can decompose the set of pages $\mathcal{I}$ into subsets according to the Web sites they belong to, i.e., $\mathcal{I}^{(i)}$ is the set of the pages of site i. Then, the Google matrix is partitioned in (2.3) according to the decomposition of $\mathcal{I}$. We give a sketchy description of the algorithm which consists of three stages. At the first stage, PageRank is determined separately for each site. It can be done locally for each site. After that, at the second stage, BlockRank is calculated by composing an aggregated matrix with the local PageRank vectors of the sites used as the aggregation vectors. And, at the last stage, the global PageRank is composed using the local PageRank vectors taken with the corresponding BlockRank values as weights. The formal description of the BlockRank algorithm is given below.

Algorithm 2.1 (BlockRank algorithm) *Determine an approximation $\pi^{(k)}$ to PageRank vector π of Google matrix G in k iterations.*

1. *Determine the local PageRank vector for each diagonal block P_i*
   (a) *Normalize P_{ii}, i.e.* $\left(\bar{P}_{ii}\right)_{jk} = \frac{(P_{ii})_{jk}}{(P_{ii})_j \mathbf{1}}$.
   (b) *Form G_{ii},* $G_{ii} = c\bar{P}_{ii} + (1-c)(1/n)E$.
   (c) *Approximately determine $\bar{\pi}_i$*
      i. *Select a vector $\pi_i^{(0)}$.*
      ii. *Do $k = 1, 2 \ldots$*
$$\pi_i^{(k)} = \pi_i^{(k-1)} G_{ii}.$$
2. *Determine BlockRank β*
   (a) *Form aggregated matrix A*
$$a_{ij} = \bar{\pi}_i P_{ij} \mathbf{1}.$$
   (b) *Form B,* $B = cA + (1-c)(1/n)E$.
   (c) *Approximately determine β*
      i. *Select a vector $\beta^{(0)}$.*
      ii. *Do $k = 1, 2 \ldots$*
$$\beta^{(k)} = \beta^{(k-1)} B.$$
3. *Determine the global PageRank vector*

(a) Form the vector $\pi^{(0)} = (\beta_1\bar{\pi}_1, \beta_2\bar{\pi}_2, \ldots, \beta_N\bar{\pi}_N)$.

(b) Do $k = 1, 2 \ldots$

$$\pi^{(k)} = \pi^{(k-1)}G.$$

It was empirically shown that the BlockRank algorithm is faster than the Power method by at least a factor of two [48]. This example shows that a particular choice of the partitioning of the Web pages can leads to a gain in computations.

2.3.3 Exploiting dangling pages

In this section, we consider two algorithms which exploit dangling pages to boost PageRank computation. The first algorith is the *Fast Two-Stage Algorithm (FTSA)* [64]. The main idea of the method is, at the first stage, to lump the dangling nodes into one state and, at the second stage, to find the PageRank vector of the new aggregated matrix and to aggregate the non-dangling pages into one state. Therefore, the set of pages is decomposed into two sets $\mathcal{I}_1$ and $\mathcal{I}_2$, where $\mathcal{I}_1 \cup \mathcal{I}_2 = \mathcal{I}$, and $\mathcal{I}_1$ contains all the non-dangling pages and $\mathcal{I}_2$ contains all the dangling pages. Hence, the hyperlink matrix H (1.2) is partitioned as following:

$$H = \begin{pmatrix} H_{11} & H_{12} \\ 0 & 0 \end{pmatrix},$$

and Google matrix G from (1.8) is represented in the following way:

$$G = \begin{pmatrix} G_{11} & G_{12} \\ \mathbf{1}_{n_1} u_{n_1} & \mathbf{1}_{n_2} u_{n_2} \end{pmatrix},$$

where $\mathbf{1} = \left(\mathbf{1}_{n_1}^T, \mathbf{1}_{n_2}^T\right)^T$ and $u = (u_{n_1}, u_{n_2})$, $u = cw + (1-c)v$. Let us denote by $G^{(1)}$ the lumped matrix:

$$G^{(1)} = \begin{pmatrix} G_{11} & G_{12}\mathbf{1}_{n_2} \\ u_{n_1} & u_{n_2}\mathbf{1}_{n_2} \end{pmatrix}. \tag{2.5}$$

The formal description of the algorithm is given below.

Algorithm 2.2 (Fast Two-Stage Algorithm) *Determine an approximation* $\hat{\pi}$ *to the PageRank vector* π *of the Google matrix* G *in* k *iterations.*

1. *The first stage: lump dangling pages*

   (a) Form the lumped matrix $G^{(1)}$ (2.5).

   (b) Approximately determine $\bar{\pi}_1 = (\pi_1, \alpha)$*, where* $\dim(\pi_1) = n_1$

      i. Select a vector $\bar{\pi}_1^{(0)}$.

ii. Do $k = 1, 2 \ldots$

$$\bar{\pi}_1^{(k)} = \bar{\pi}_1^{(k-1)} G^{(1)}.$$

(c) Determine aggregation weights of the second stage

$$\eta(k) = \frac{\pi_1(k)}{\sum_{i=1}^{n_1} \pi_1(i)}, \qquad k = \overline{1, n_1}.$$

2. The second stage: aggregate non-dangling pages

(a) Form aggregated matrix $G^{(2)}$

$$G^{(2)} = \begin{pmatrix} \eta G_{11} \mathbf{1}_{n_1} & \eta G_{12} \\ (u_{n_1} \mathbf{1}_{n_1}) \mathbf{1}_{n_1} & \mathbf{1}_{n_2} u_{n_2} \end{pmatrix}.$$

(b) Approximately determine $\bar{\pi}_2 = (\beta, \pi_2)$*, where* $\dim(\pi_2) = n_2$

i. Select a vector $\bar{\pi}_2^{(0)}$.

ii. Do $k = 1, 2 \ldots$

$$\bar{\pi}_2^{(k)} = \bar{\pi}_2^{(k-1)} G^{(2)}.$$

3. Form the approximation of the PageRank vector

$$\hat{\pi} = (\pi_1, \pi_2).$$

The first stage requires less computational work than the Power method does, roughly $O(n_1)$ as opposed to $O(n)$ per iteration, and converges at least as fast as the Power method. The second stage usually converges after about three iterations. Although it is not proven that FSTA results into PageRank vector, FSTA is one of the first attepts to exploit the dangling pages for PageRank computation.

The second algorithm which we discuss is very similar to FSTA excepting the last stage where authors use exact computation [45].

Algorithm 2.3 *Determine an approximation* $\hat{\pi}$ *to the PageRank vector* π *of the Google matrix* G *in* k *iterations.*

1. The first stage: lump dangling pages

(a) Form the lumped matrix $G^{(1)}$ (2.5).

(b) Choose starting vector $\sigma^{(0)} = (\bar{\pi}^{(0)}, \alpha^{(0)})$ *with* $\sigma^{(0)} \geqslant 0$ *and* $\sigma^{(0)} \mathbf{1} = 1$.

(c) Do $k = 0, 1, 2 \ldots$

i. $\bar{\pi}^{(k+1)} = c\bar{\pi}^{(k)} H_{11} + (1 - c) v_1 + c\alpha^{(k)} w_1$.

ii. $\alpha^{(k+1)} = 1 - \bar{\pi}^{(k+1)}$.

2. The second stage: recover PageRank

(a) $\hat{\pi} = \left(\bar{\pi}^{(k)}, c\bar{\pi}^{(k)}H_{12} + (1-c)v_2 + c\alpha^{(k)}w_2\right)$.

This method has the same convergence rate equal to c as the Power method since matrix $G^{(1)}$ has the same nonzero eigenvalues as Google matrix (1.8), but this method is less computationally consuming because it deals with a small matrix.

2.3.4 Full aggregation-disaggregation method

We introduce the *full aggregation-disaggregation method (FAM)* in application to a general transition matrix P and give an estimation of its convergence rate when it is applied to the Google matrix G.The method is based on the theory of the stochastic complement and the coupling theorem [67]. Here we introduce it for the completeness.

Definition 1 (Stochastic complement) *For a given index* i*, let* P_i *denote the principal block submatrix of* P *obtained by deleting the* i^{th} *row and* i^{th} *column of blocks from* P*, and let* P_{i*} *and* P_{*i} *designate*

$$P_{i*} = (P_{i1}P_{i2}\cdots P_{i,i-1}P_{i,i+1}\cdots P_{iN})$$

and

$$P_{*i} = \begin{pmatrix} P_{1i} \\ \vdots \\ P_{i-1,i} \\ P_{i+1,i} \\ \vdots \\ P_{Ni} \end{pmatrix}.$$

That is, P_{i*} *is the* i^{th} *row of blocks with* P_{ii} *removed, and* P_{*i} *is the* i^{th} *column of blocks with* P_{ii} *removed. The stochastic complement of* P_{ii} *in* P *is defined to be the matrix*

$$S_i = P_{ii} + P_{i*}(I - P_i)^{-1}P_{*i}.$$

Theorem 2.1 ([67, Theorem 4.1] Coupling theorem) *If matrix* P *is an irreducible stochastic matrix partitioned as* (2.2)*, then the stationary distribution of the matrix is given by*

$$\pi = (\nu_1\sigma_1, \nu_2\sigma_2, \ldots, \nu_N\sigma_N),$$

where σ_i *is the unique stationary distribution vector for the stochastic complement*

$$S_i = P_{ii} + P_{i*}(I - P_i)^{-1}P_{*i}$$

and where

$$\nu = (\nu_1, \nu_2, \ldots, \nu_N)$$

is the unique stationary distribution vector for the aggregated matrix A *whose entries are defined by*

$$a_{ij} = \sigma_i P_{ij} \mathbf{1}.$$

Theorem 2.1 implies that the stationary distribution can be found by the exact aggregation, but it forces to compute the stochastic complements of diagonal blocks and their stationary distributions. One can avoid it by using approximate iterative aggregation method.

Algorithm 2.4 (Full aggregation-disaggregation method) *Determine an approximation* $\pi^{(k)}$ *to stationary distribution* π *of stochastic matrix* P *in* k *iterations.*

1. *Select a vector* $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}, \ldots, \pi_N^{(0)}\right)$ *with* $\pi^{(0)}\mathbf{1} = 1$.
2. *Do* $k = 0, 1, 2 \ldots$
   (a) *Normalize* $\sigma_i^{(k)} = [\pi_i^{(k)}]$, $i = \overline{1, N}$.
   (b) *Form aggregated matrix* $A^{(k)}$ *with elements*
   $$a_{ij} = \sigma_i^{(k)} P_{ij} \mathbf{1}.$$
   (c) *Determine stationary distribution* $\nu^{(k)}$ *of* $A^{(k)}$
   $$\nu^{(k)} = \nu^{(k)} A^{(k)}.$$
   (d) *Determine disaggregated vector* $\widetilde{\pi}^{(k)}$
   $$\widetilde{\pi}^{(k)} = \left(\nu_1^{(k)}\sigma_1^{(k)}, \nu_2^{(k)}\sigma_2^{(k)}, \ldots, \nu_N^{(k)}\sigma_N^{(k)}\right).$$
   (e) *Do* l *steps of the Power method*
   $$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^l.$$

The iteration of a aggregation-disaggregation method can contain several iterations of the Power method. That is why we increase the superscript on the vector $\pi^{(k)}$ in the last step of the above algorithm only by unity.

If matrix P is irreducible, it can be shown that the stationary distribution π is the fix point of FAM. Indeed, if $\pi^{(k)} = \pi$, then $A^{(k)} = A$ and $\nu^{(k)} = \nu$. Therefore, $\widetilde{\pi}^{(k)} = \pi$, and $\pi^{(k+1)} = \pi$.

The FAM converges only for particular choice of the initial vector $\pi^{(0)}$, namely only for vectors which are close to the stationary distribution π. This type of convergence is called the local convergence. For the local convergence of the FAM, it is necessary to fulfill one of the conditions [65, Theorem 1]:

1. $P \gg 0$ and $P \geqslant \delta_1 Q$, where $Q = \mathbf{1}\pi$ and δ_1 is a positive real number, or

2. $P \geqslant \mathbf{1}b$, where b is a row vector, $b\mathbf{1} = \delta_2$, where b is a positive row vector and δ_2 is a positive real number.

In the further discussion, we assume the above conditions individually in the corresponding items below.

Let us consider the application of the above algorithm to the Google matrix (1.4). Since the Google matrix satisfies both the conditions, as it shown below, FAM converges locally. FAM also converges globally if l is large enough [65].

Let us provide an estimation of the rate of convergence of FAM [69] assuming that matrix G defined by (1.4).

1. Consider the condition $G \geqslant \delta_1 Q$. Let us find δ_1. Denote by g_{min} the minimum entry of the matrix G. If $p_{ij} = 0$ then $g_{ij} = g_{min}$. Hence,

$$g_{min} = \frac{1-c}{n}. \tag{2.6}$$

   The maximum of the elements of PageRank vector π is achieved when all the other elements achieve a minimum, because of $\pi > 0$ and $\pi\mathbf{1} = 1$. The minimum entry of the PageRank vector for a page is realized if there is no other page referring to it. The minimum entry of the PageRank vector is $\frac{1-c}{n}$. Therefore, the maximum of one of the elements of the PageRank vector is less or equal to a value γ

$$\gamma = 1 - \frac{1-c}{n}(n-1) = \frac{1+c(n-1)}{n}. \tag{2.7}$$

   Hence, if we find δ_1 from the constraint

$$g_{min} \geqslant \delta_1 \gamma, \tag{2.8}$$

   it ensures that $G \geqslant \delta_1 Q$. From the equalities (2.6), (2.7) and (2.8) we get

$$\delta_1 \leqslant \frac{1-c}{1+c(n-1)}.$$

2. Consider the condition $G \geqslant \mathbf{1}b$, where $b\mathbf{1} = \delta_2$. Let us determine δ_2. From the equalities (1.4) we obtain that

3. Normalize $\sigma_i^{(k)} = [\pi_i^{(k)}]$, . $G \geqslant \frac{1-c}{n}E$. The equality can be rewritten as $G \geqslant \mathbf{1}\left(\frac{1-c}{n}\mathbf{1}^T\right)$. Therefore, as vector b one can take $\left(\frac{1-c}{n}\mathbf{1}^T\right)$. Hence,

$$\delta_2 = 1 - c.$$

The error vector of the method at the k^{th} iteration is given by

$$\pi^{(k+1)} - \pi = \left(\pi^{(k)} - \pi\right) J\left(\pi^{(k)}\right),$$

where $J\left(\pi^{(k)}\right) = \left(I - RT\left(\pi^{(k)}\right)\right)\left(I - RT\left(\pi^{(k)}\right)Z\right)^{-1} G^{l}$, $Z = P - \mathbf{1}\pi$, restriction matrix $R = (r_{ij})_{i=\overline{1,n},j=\overline{1,N}}$ defined by

$$r_{ij} = \begin{cases} 1, & \text{if page } i \in \mathcal{I}^{(j)}, \\ 0, & \text{otherwise,} \end{cases}$$

prolongation matrix $T(x) = (t_{ij})_{i=\overline{1,N},j=\overline{1,n}}$ defined by,

$$t_{ij} = \begin{cases} \frac{x_j}{\sum_{m\in\mathcal{I}^{(i)}} x_m}, & \text{if page } j \in \mathcal{I}^{(i)}, \\ 0, & \text{otherwise.} \end{cases}$$

From the above estimation and [65, Proposition 2] we can conclude that the spectral radius of matrix $J(\pi)$:

1. is less than $1 - \delta_1 = \frac{cn}{1+c(n-1)} < 1$,
2. is less than $\sqrt{1-\delta_2} = \sqrt{c} < 1$.

The second estimation becomes better than the first one for sufficiently large n. The second estimation ensures that the convergence rate of the method is not less than $\sqrt{c}$. Unfortunately, the estimation does not ensure that the method converges faster than the Power method. Nevertheless, for the partial aggregation method which is discussed in the next subsection and which is actually a particular case of the full aggregation method, it was shown that there exists such a partitioning of the Google matrix which provides faster convergence than the convergence of the Power method.

We introduced the full aggregation-disaggregation method and estimated its convergence rate when it is applied to the Google matrix.

Since we need 2×2 case of partitioning for the further discussion, we present a special version of FAM algorithm below.

Algorithm 2.5 (Full aggregation-disaggregation method for 2×2 case) *Determine an approximation $\pi^{(k)}$ to stationary distribution π of stochastic matrix P in k iterations if the state set is partitioned into two disjoint sets.*

1. *Select a vector $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}\right)$ with $\pi^{(0)}\mathbf{1} = 1$.*
2. *Do $k = 0, 1, 2 \ldots$*
   (a) *Normalize $\sigma_i^{(k)} = [\pi_i^{(k)}]$, $i = \overline{1, N}$.*

(b) Form aggregated matrix $A^{(k)}$

$$A^{(k)} = \begin{pmatrix} \sigma_1^{(k)} P_{11}\mathbf{1} & \sigma_1^{(k)} P_{12}\mathbf{1} \\ \sigma_2^{(k)} P_{21}\mathbf{1} & \sigma_2^{(k)} P_{22}\mathbf{1} \end{pmatrix}.$$

(c) Determine stationary distribution $\nu^{(k)}$ *of* $A^{(k)}$

$$\nu^{(k)} = \nu^{(k)} A^{(k)}.$$

(d) Determine disaggregated vector $\widetilde{\pi}^{(k)}$

$$\widetilde{\pi}^{(k)} = \left(\nu_1^{(k)}\sigma_1^{(k)}, \nu_2^{(k)}\sigma_2^{(k)}\right).$$

(e) Do l *steps of the Power method*

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^l.$$

We shall analyze this algorithm along with the partial aggregation-disaggregation method presented in the following section.

2.3.5 Partial aggregation-disaggregation method

We introduce the *partial aggregation-disaggregation method (PAM)* and mention its properties. The method is considered in detail in [44] and is used for updating of the stationary distribution in [60, 59].The method is applied to the 2×2 case, i.e. $N = 2$, and the irreducible matrix P is partitioned as follows

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}.$$

The matrix $I - P$ is singular, but the matrix $I - P_{11}$ is nonsingular [19]. Hence, we can factor $I - P = LDU$ [67, proof of Theorem 2.3], where

$$\begin{aligned} L &= \begin{pmatrix} I & 0 \\ -P_{21}(I - P_{11})^{-1} & I \end{pmatrix}, \\ D &= \begin{pmatrix} I - P_{11} & 0 \\ 0 & I - S_2 \end{pmatrix}, \\ U &= \begin{pmatrix} I & -(I - P_{11})^{-1} P_{12} \\ 0 & I \end{pmatrix}, \end{aligned}$$

where S_2 is the stochastic complement of the block P_{22}.

We consider the case when matrix P is irreducible. Since the matrix U is nonsingular, we have $\pi(I - P) = 0$ if and only if $\pi LD = 0$. Hence,

$$\pi_2 S_2 = \pi_2, \qquad \pi_1 = \pi_2 P_{21}(I - P_{11})^{-1}, \tag{2.9}$$

which means that π_2 is the principal eigenvector of the matrix S_2. The expression (2.9) represents a particular case of Theorem 2.1 for the 2×2 decomposition of the transition matrix [67, Colorary 4.1]. The matrix S_2 has the unique stationary distribution

$$\sigma_2 S_2 = \sigma_2, \qquad \sigma_2 \mathbf{1} = 1.$$

And we can find π_2 as $\pi_2 = \rho\sigma_2$, where the factor ρ is chosen to satisfy the normalization condition $\pi\mathbf{1} = 1$.

The component π_1 and the factor ρ can be expressed as the components of the stationary distribution of the aggregated matrix

$$A_1 = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ \sigma_2 P_{21} & \sigma_2 P_{22}\mathbf{1} \end{pmatrix}.$$

From (2.9), $\pi_2 = \rho\sigma_2$ and $\sigma_2\mathbf{1} = 1$ we get

$$(\pi_1, \rho)(I - A_1) = 0, \qquad (\pi_1, \rho)\mathbf{1} = 1.$$

Since A_1 is stochastic and irreducible [67, Teorem 4.1], it has the unique stationary distribution α,

$$\alpha A_1 = \alpha, \qquad \alpha\mathbf{1} = 1.$$

By the uniqueness, we get $\alpha = (\pi_1, \rho)$.

The above analysis implies that the stationary distribution can be found by the partial exact aggregation, but it forces to compute the stochastic complement of P_{22} block of matrix P and its stationary distribution. One can avoid it by using the approximate iterative partial aggregation method.

Algorithm 2.6 (Partial aggregation-disaggregation method) *Determine an approximation $\pi^{(k)}$ to stationary distribution π of stochastic matrix P in k iterations if the state set is partitioned into two disjoint sets.*

1. *Select a vector $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}\right)$ with $\pi^{(0)}\mathbf{1} = 1$.*
2. *Do $k = 0, 1, 2 \ldots$*
   (a) *Normalize $\sigma_2^{(k)} = [\pi_2^{(k)}]$.*

(b) Form aggregated matrix $A_1^{(k)}$

$$A_1^{(k)} = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ \sigma_2^{(k)} P_{21} & \sigma_2^{(k)} P_{22}\mathbf{1} \end{pmatrix}.$$

(c) Determine stationary distribution $\alpha^{(k)}$ *of* $A_1^{(k)}$

$$\alpha^{(k)} = \alpha^{(k)} A_1^{(k)}.$$

(d) Partition $\alpha^{(k)}$

$$\alpha^{(k)} = (\omega_1^{(k)}, \rho^{(k)}).$$

(e) Determine disaggregated vector $\widetilde{\pi}^{(k)}$

$$\widetilde{\pi}^{(k)} = \left(\omega_1^{(k)}, \rho^{(k)} \sigma_2^{(k)}\right).$$

(f) Do l *steps of the Power method*

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^l.$$

Let us consider the case when $l = 1$. The PAM method is the Power method with matrix $\widetilde{P}$ [44, Proposition 5.1, Theorem 5.2], where

$$\widetilde{P} = \begin{pmatrix} 0 & 0 \\ P_{21}(I - P_{11})^{-1} & S_2 \end{pmatrix}.$$

Therefore, the rate of convergence of PAM is equal to $|\lambda_2(S_2)|$ [44, Theorem 5.2], where $\lambda_2(S_2)$ is the second eigenvalue of matrix S_2. If the Power method converges for matrix P, then PAM converges, too [44, Proposition 7.1]. PAM can converge slower than the Power method [44, Example 6.3], but if the algorithms are applied to the Google matrix, then there always exists such a decomposition which ensures that PAM converges faster than the Power method.

2.4 Stationary distribution of aggregated matrices

Let us analyse the stationary distributions of the aggregated matrices of FAM and PAM. The analysis was presented in [69]. In this section, we do not compare FAM and PAM algorithm neigher by their convergence rate nor computational performance. We just show a relation between the stationary distributions of the aggregated matrices of FAM and PAM under some particular conditions.

Partition the stochastic matrix

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{21} \end{pmatrix}, \qquad \pi = (\pi_1, \pi_2), \tag{2.10}$$

where P_{ij} is $n_i \times n_j$ block and $n_1 + n_2 = n$. Note that [67]

$$P_{11}\mathbf{1} + P_{12}\mathbf{1} = \mathbf{1}, \tag{2.11a}$$
$$P_{21}\mathbf{1} + P_{22}\mathbf{1} = \mathbf{1}. \tag{2.11b}$$

We consider the case when matrix P is irreducible. Since $I - P$ is M-matrix [19] as well as irreducible and singular, the submatrix $I - P_{ii}$ is non-singular [19].

Let S_1 be the stochastic complement of block P_{11} of matrix P,

$$S_1 = P_{11} + P_{12}(I - P_{22})^{-1}P_{21}. \tag{2.12}$$

Let σ_1 be the stationary distribution of matrix S_1,

$$\sigma_1 S_1 = \sigma_1, \qquad \sigma_1\mathbf{1} = 1.$$

Let $\widetilde{\sigma}_2$ be an arbitrary row vector such that

$$\widetilde{\sigma}_2 > 0, \qquad \widetilde{\sigma}_2\mathbf{1} = 1.$$

Let us define aggregated matrix A^f (superscript "f" marks the relation to FAM) as

$$A^f = \begin{pmatrix} \sigma_1 P_{11}\mathbf{1} & \sigma_1 P_{12}\mathbf{1} \\ \widetilde{\sigma}_2 P_{21}\mathbf{1} & \widetilde{\sigma}_2 P_{22}\mathbf{1} \end{pmatrix}.$$

Matrix A^f is the aggregated matrix of FAM with a special choice of the aggregation vector. Let $\nu^f = (\nu_1^f, \nu_2^f)$ be the stationary distribution of matrix A^f,

$$\nu^f A^f = \nu^f, \qquad \nu^f\mathbf{1} = 1.$$

Then

$$\nu_1^f = \nu_1^f \sigma_1 P_{11}\mathbf{1} + \nu_2^f \widetilde{\sigma}_2 P_{21}\mathbf{1}, \tag{2.13a}$$
$$\nu_2^f = \nu_1^f \sigma_1 P_{12}\mathbf{1} + \nu_2^f \widetilde{\sigma}_2 P_{22}\mathbf{1}. \tag{2.13b}$$

Using (2.11b), we can derive

$$A^f = \begin{pmatrix} \sigma_1 P_{11}\mathbf{1} & \sigma_1 P_{12}\mathbf{1} \\ \widetilde{\sigma}_2 P_{21}\mathbf{1} & 1 - \widetilde{\sigma}_2 P_{21}\mathbf{1} \end{pmatrix}. \tag{2.14}$$

Let us define matrix A^p (superscript "p" marks the relation to PAM) as

$$A^p = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ \widetilde{\sigma}_2 P_{21} & \widetilde{\sigma}_2 P_{22}\mathbf{1} \end{pmatrix}.$$

Matrix A^p is the aggregated matrix of PAM. Let $\alpha^p = (\omega^p, \rho^p)$ be the stationary distribution of matrix A^p, where $\dim(\omega^p) = n_1$ and $\dim(\rho^p) = 1$. Then

$$\omega^p = \omega^p P_{11} + \rho^p \widetilde{\sigma}_2 P_{21}, \tag{2.15a}$$

$$\rho^p = \omega^p P_{12}\mathbf{1} + \rho^p \widetilde{\sigma}_2 P_{22}\mathbf{1}, \tag{2.15b}$$

Using (2.11b), we can derive

$$A^p = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ \widetilde{\sigma}_2 P_{21} & 1 - \widetilde{\sigma}_2 P_{21}\mathbf{1} \end{pmatrix}. \tag{2.16}$$

We look for the conditions when $\omega^p = \nu_1^f \sigma_1$ and $\nu_2^f = \rho^p$. Let us consider the case when $\mathrm{rank} P_{21} = 1$. Then P_{21} is expressed as follows:

$$P_{21} = \overline{b}\,\overline{a},$$

where a and b are real vector such as $\dim(\overline{b}) = n_2 \times 1$, and $\dim(\overline{a}) = 1 \times n_1$. We note that $\overline{b} \neq 0$ and $\overline{a} \neq 0$ because otherwise $\mathrm{rank} P_{21} = 0$. Because $P_{21} > 0$, there are two options: $\overline{b} > 0$ and $\overline{a} > 0$, or $\overline{b} < 0$ and $\overline{a} < 0$. Let us choose the first one without loss of generality.

Let $\widetilde{\sigma}_2 \overline{b} = h$. Because $\widetilde{\sigma}_2$ is a row vector, $\overline{b}$ is a column vector and $\widetilde{\sigma}_2 > 0$, $\overline{b} > 0$, then h is a number which is not equal to zero. Denote $b = \frac{\overline{b}}{h}$ and $a = h\overline{a}$. Then, it is easy to see that $P_{21} = \overline{b}\,\overline{a} = hb\frac{a}{h} = ba$. Note that $\widetilde{\sigma}_2 b = 1$. When $\mathrm{rank} P_{21} = 1$, the aggregated matrices and their stationary distribution can be simplified. Thus, (2.14)

$$A^f = \begin{pmatrix} \sigma_1 P_{11}\mathbf{1} & \sigma_1 P_{12}\mathbf{1} \\ a\mathbf{1} & 1 - a\mathbf{1} \end{pmatrix},$$

then ν^f is determined by

$$\nu_1^f = \frac{a\mathbf{1}}{\sigma_1 P_{12}\mathbf{1} + a\mathbf{1}}, \qquad \nu_2^f = \frac{\sigma_1 P_{12}\mathbf{1}}{\sigma_1 P_{12}\mathbf{1} + a\mathbf{1}}. \tag{2.17}$$

And from (2.16)

$$A^p = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ a & 1 - a\mathbf{1} \end{pmatrix},$$

and from (2.15)

$$\omega^p = \omega^p P_{11} + \rho^p a, \tag{2.18a}$$

$$\rho^p = \omega^p P_{12}\mathbf{1} + \rho^p (1 - a\mathbf{1}). \tag{2.18b}$$

Let us prove an auxiliary lemma which we need for the main result of this section.

Lemma 2.1 *Let matrix* P *be an irreducible stochastic matrix partitioned as in* (2.10). *Let* $\mathrm{rank} P_{21} = 1$, *and* $P_{21} = ba$, *where* b *such that* $\widetilde{\sigma}_2 b = 1$. *Then*

$$\nu_1^f \sigma_1 P_{11} + \nu_2^f a = \nu_1^f \sigma_1. \tag{2.19}$$

Proof Let us use (2.17) for the right part of (2.19).

$$\nu_1^f \sigma_1 = \frac{a\mathbf{1}\sigma_1}{\sigma_1 P_{12}\mathbf{1} + a\mathbf{1}}. \tag{2.20}$$

Use (2.17) for the left part of (2.19).

$$\begin{aligned} \nu_1^f \sigma_1 P_{11} + \nu_2^f a &= \frac{a\mathbf{1}\sigma_1 P_{11}}{\sigma_1 P_{12}\mathbf{1} + a\mathbf{1}} + \frac{\sigma_1 P_{12}\mathbf{1}a}{\sigma_1 P_{12}\mathbf{1} + a\mathbf{1}} = \\ &= \frac{a\mathbf{1}\sigma_1 P_{11} + \sigma_1 P_{12}\mathbf{1}a}{\sigma_1 P_{12}\mathbf{1} + a\mathbf{1}}. \end{aligned} \tag{2.21}$$

And we need to show equality of numerators (2.21) and (2.20). Note that, from (2.12), $P_{11} = S_1 - P_{12}(I - P_{22})^{-1}P_{21}$ and $\sigma_1 S_1 = \sigma_1$, $\sigma_1\mathbf{1} = 1$, then

$$\begin{aligned} \sigma_1 P_{11} &= \sigma_1(S_1 - P_{12}(I - P_{22})^{-1}P_{21}) = \\ &= \sigma_1 S_1 - \sigma_1 P_{12}(I - P_{22})^{-1}P_{21} = \\ &= \sigma_1 - \sigma_1 P_{12}(I - P_{22})^{-1}P_{21}. \end{aligned}$$

If we continue (2.21) for numerators using (2.11a), we derive that

$$\begin{aligned} & a\mathbf{1}\sigma_1 P_{11} + \sigma_1 P_{12}\mathbf{1}a = \\ = \;& a\mathbf{1}\sigma_1 P_{11} + \sigma_1(\mathbf{1} - P_{11}\mathbf{1})a = \\ = \;& a\mathbf{1}\sigma_1 P_{11} + (1 - \sigma_1 P_{11}\mathbf{1})a = \\ = \;& a\mathbf{1}(\sigma_1 - \sigma_1 P_{12}(I - P_{22})^{-1}P_{21}) + \\ + \;& (1 - (\sigma_1 - \sigma_1 P_{12}(I - P_{22})^{-1}P_{21})\mathbf{1})a = \\ = \;& a\mathbf{1}\sigma_1 - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1}P_{21} + \\ + \;& a - \sigma_1\mathbf{1}a + \sigma_1 P_{12}(I - P_{22})^{-1}P_{21}\mathbf{1}a = \\ = \;& a\mathbf{1}\sigma_1 - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1}P_{21} + \\ + \;& a - a + \sigma_1 P_{12}(I - P_{22})^{-1}P_{21}\mathbf{1}a = \\ = \;& a\mathbf{1}\sigma_1 - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1}P_{21} + \\ + \;& \sigma_1 P_{12}(I - P_{22})^{-1}P_{21}\mathbf{1}a. \end{aligned}$$

If we compare the received expression and (2.20) we can conclude that we need to show that

$$\sigma_1 P_{12}(I - P_{22})^{-1}P_{21}\mathbf{1}a - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1}P_{21} = 0.$$

Since $P_{21} = ba$ and $a\mathbf{1}$ is a number, we obtain

$$\begin{aligned} & \sigma_1 P_{12}(I - P_{22})^{-1} P_{21}\mathbf{1}a - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1} P_{21} = \\ = \; & \sigma_1 P_{12}(I - P_{22})^{-1} baea - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1} ba = \\ = \; & \sigma_1 P_{12}(I - P_{22})^{-1} b(a\mathbf{1})a - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1} ba = \\ = \; & a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1} ba - a\mathbf{1}\sigma_1 P_{12}(I - P_{22})^{-1} ba = 0. \end{aligned}$$

Therefore the equality (2.19) is proven. □

Example 2.1 *The example illustrates that Lemma 2.1 doesn't fulfill when* $\mathrm{rank} P_{21} > 1$.

Let us consider the matrix

$$P = \left(\begin{array}{ccc|ccc} \frac{1}{6} & \frac{1}{12} & \frac{1}{12} & \frac{1}{6} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{8} & \frac{1}{12} & \frac{1}{6} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{4} & \frac{1}{12} & \frac{1}{12} & \frac{1}{6} & \frac{1}{6} & \frac{1}{4} \\ \hline \frac{1}{12} & \frac{1}{6} & \frac{1}{3} & \frac{1}{6} & \frac{1}{12} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{12} & \frac{1}{4} & \frac{1}{6} & \frac{1}{9} & \frac{2}{9} \\ \frac{1}{12} & \frac{1}{24} & \frac{1}{8} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{array}\right).$$

Vertical and horizontal lines define the partitioning of the matrix. The case is $\mathrm{rank} P_{21} = 2$. Let us check Lemma 2.1 without condition that $\mathrm{rank} P_{21} = 1$,

$$\nu_1^f \sigma_1 P_{11} + \nu_2^f \widetilde{\sigma}_2 P_{21} = \nu_1^f \sigma_1.$$

Let us use that

$$S_1 = P_{11} + P_{12}(I - P_{22})^{-1} P_{21}$$

and σ_1 is the stationary distribution of matrix S_1.

$$\begin{aligned} & \nu_1^f \sigma_1 P_{11} + \nu_2^f \widetilde{\sigma}_2 P_{21} = \\ = \; & \nu_1^f \sigma_1 (S_1 - P_{12}(I - P_{22})^{-1} P_{21}) + \nu_2^f \widetilde{\sigma}_2 P_{21} = \\ = \; & \nu_1^f \sigma_1 S_1 - \nu_1^f \sigma_1 P_{12}(I - P_{22})^{-1} P_{21} + \nu_2^f \widetilde{\sigma}_2 P_{21} = \\ = \; & \nu_1^f \sigma_1 - \nu_1^f \sigma_1 P_{12}(I - P_{22})^{-1} P_{21} + \nu_2^f \widetilde{\sigma}_2 P_{21}. \end{aligned}$$

And we need to check that

$$\nu_2^f \widetilde{\sigma}_2 P_{21} - \nu_1^f \sigma_1 P_{12}(I - P_{22})^{-1} P_{21} = 0. \tag{2.22}$$

For the matrix and the partition we obtain that

$$S_1 = \begin{pmatrix} 0.3363 & 0.2273 & 0.4364 \\ 0.3861 & 0.2430 & 0.3708 \\ 0.3956 & 0.2111 & 0.3933 \end{pmatrix},$$

and

$$\sigma_1 = (0.3715, 0.2243, 0.4043)\,.$$

Let $\widetilde{\sigma}_2$ be

$$\widetilde{\sigma}_2 = \left(\frac{1}{3}, \frac{1}{6}, \frac{1}{2}\right).$$

Let us form the aggregated matrix (2.14)

$$A^f = \begin{pmatrix} 0.3951 & 0.6049 \\ 0.4028 & 0.5972 \end{pmatrix},$$

The stationary distribution of matrix A^f is

$$\nu^f = (0.3997, 0.6003)\,.$$

Have calculated the left part of (2.22), we obtain

$$\nu_2^f \widetilde{\sigma}_2 P_{21} - \nu_1^f \sigma_1 P_{12}(I - P_{22})^{-1} P_{21} = \\ (-0.0026, 0.0016, 0.0010) \neq 0.$$

So Lemma 2.1 is not correct without the condition that $\mathrm{rank} P_{21} = 1$.

Theorem 2.2 (About stationary distribution) *Let matrix* P *be an irreducible stochastic matrix partitioned as in* (2.10). *Let* $\mathrm{rank} P_{21} = 1$*, and* $P_{21} = ba$*, where* b *is such that* $\widetilde{\sigma}_2 b = 1$. *Then*

$$\nu_1^f \sigma_1 = \omega^p, \tag{2.23a}$$

$$\nu_2^f = \rho^p. \tag{2.23b}$$

Proof Vector (ω^p, ρ^p) is the stationary distribution of matrix A^p. Because the stationary distribution is unique, to prove the theorem, we need to show that $(\nu_1^f \sigma_1, \nu_2^f)$ is stationary distribution of matrix A^p,

$$(\nu_1^f \sigma_1, \nu_2^f) A^p = (\nu_1^f \sigma_1, \nu_2^f).$$

Let us multiply vector $(\nu_1^f \sigma_1, \nu_2^f)$ by A^p

$$\left(\nu_1^f \sigma_1, \nu_2^f\right) A^p = \left(\nu_1^f \sigma_1 P_{11} + \nu_2^f a,\ \nu_1^f \sigma_1 P_{12}\mathbf{1} + \nu_2^f (1 - a\mathbf{1})\right).$$

The truth of equality (2.23a) follows from Lemma 2.1. If we consider (2.18b) and (2.23a), then the equality (2.23b) is correct. Therefore (2.23) are proved. □

When $\mathrm{rank}P_{21} = 1$, Theorem 2.2 simplifies finding of stationary distribution σ_1 of stochastic complement S_1. To find σ_1, it is needed to determine stochastic complement S_1 itself. It requires the inversion of matrix $I - P_{22}$ with dimension $n_2 \times n_2$. After that, the inversion of matrix with dimension $n_1 \times n_1$ is necessary to find σ_1. Theorem 2.2 let us calculate σ_1 easier. Stationary distribution σ_1 is the part of stationary distribution α^p of matrix A^p. To determine it, the inversion of one matrix with dimension $(n_1 + 1) \times (n_1 + 1)$ is needed. After that, only the normalization of ω^p is necessary.

2.5 Equivalence conditions of A/D methods and mixed algorithm

Let us discover the conditions under which FAM and PAM are equivalent. The conditions were initially revealed in [69].

Lemma 2.2 *Let matrix* P *be an irreducible stochastic matrix partitioned as in* (2.10). *Let* $\mathrm{rank}P_{21} = 1$, *and* $P_{21} = ba$, *where* b *is such that* $\widetilde{\sigma}_2 b = 1$. *Then*

$$(\nu_1^f \sigma_1, \nu_2^f \widetilde{\sigma}_2)P = (\nu_1^f \sigma_1, \xi_2),$$

where $\xi_2 = \nu_1^f \sigma_1 P_{12} + \nu_2^f \widetilde{\sigma}_2 P_{22}$.

Proof

$$\begin{aligned}(\nu_1^f \sigma_1, \nu_2^f \widetilde{\sigma}_2)P &= (\nu_1^f \sigma_1, \nu_2^f \widetilde{\sigma}_2)\begin{pmatrix} P_{11} & P_{12} \\ ba & P_{22} \end{pmatrix} = \\ &= (\nu_1^f \sigma_1 P_{11} + \nu_2^f \widetilde{\sigma}_2 ba, \quad \nu_1^f \sigma_1 P_{12} + \nu_2^f \widetilde{\sigma}_2 P_{22}) = \\ &= (\nu_1^f \sigma_1 P_{11} + \nu_2^f a, \quad \xi_2).\end{aligned}$$

And, using Lemma 2.1,

$$(\nu_1^f \sigma_1, \nu_2^f \widetilde{\sigma}_2)P = (\nu_1^f \sigma_1, \quad \xi_2).$$

□

Theorem 2.3 (Equivalence conditions) *Let matrix* P *be an irreducible stochastic matrix partitioned as in* (2.10). *Let* $\mathrm{rank}P_{21} = 1$. *Then for initial vector* $\pi^{(0)} = (\nu_1^{(0)} \sigma_1, \pi_2^{(0)})$, *where* $\pi_2^{(0)}$ *is an arbitrary vector with* $\pi_2^{(0)} \mathbf{1} = \nu_2^{(0)}$, *FAM and PAM are equivalent.*

Proof Theorem 2.2 ensures that $\widetilde{\pi}^{(0)}$ is the same for both the methods. If we apply Lemma 2.2 to $\widetilde{\pi}^{(0)}$ l times, we conclude that we can repeat the reasoning on the next iteration. Repeating the reasoning k times for $\forall k \in \mathbb{N}$, we derive that $\pi^{(k)}$ is the same for both the methods. It was to be proven. □

When $\mathrm{rank}P_{21} = 1$, Theorem 2.2 and Theorem 2.3 allow us to formulate a new method of the finding stationary distribution π.

Algorithm 2.7 (Mixed aggregation-disaggregation method) *Determine an approximation $\pi^{(k)}$ to stationary distribution π of stochastic irreducible matrix P in k iterations if the state set is partitioned into two disjoint sets and $\mathrm{rank}P_{21} = 1$.*

1. *Select an arbitrary vector $\pi^{(0)}$ with $\pi^{(0)}\mathbf{1} = 1$.*

2. *Do $k = 0$*

   (a) *Normalize $\sigma_2^{(0)} = [\pi_2^{(0)}]$.*

   (b) *Form aggregated matrix $A_1^{(0)}$*

$$A_1^{(0)} = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ \sigma_2^{(0)}P_{21} & \sigma_2^{(0)}P_{22}\mathbf{1} \end{pmatrix}.$$

   (c) *Determine stationary distribution $\alpha^{(0)}$ of matrix $A_1^{(0)}$*

$$\alpha^{(0)} = \alpha^{(0)}A_1^{(0)}.$$

   (d) *Partition $\alpha^{(0)}$*

$$\alpha^{(0)} = (\omega_1^{(0)}, \rho^{(0)}).$$

   (e) *Form disaggregated vector*

$$\widetilde{\pi}^{(0)} = \left(\omega_1^{(0)}, \rho^{(0)}\sigma_2^{(0)}\right).$$

   (f) *Do l step by the Power method*

$$\pi^{(1)} = \widetilde{\pi}^{(0)}P^l.$$

   (g) *Normalize $\sigma_1 = [\omega_1^{(0)}] = [\pi_1^{(1)}]$.*

3. *Do $k = 1, 2, 3 \ldots$*

   (a) *Normalize $\sigma_2^{(k)} = [\pi_2^{(k)}]$.*

(b) Form aggregated matrix $A^{(k)}$

$$A^{(k)} = \begin{pmatrix} \sigma_1 P_{11}\mathbf{1} & \sigma_1 P_{12}\mathbf{1} \\ \sigma_2^{(k)} P_{21}\mathbf{1} & \sigma_2^{(k)} P_{22}\mathbf{1} \end{pmatrix}.$$

(c) Determine stationary distribution $\nu^{(k)}$ *of matrix* $A^{(k)}$

$$\nu^{(k)} = \nu^{(k)} A^{(k)}.$$

(d) Form disaggregated vector

$$\widetilde{\pi}^{(k)} = \left(\nu_1^{(k)}\sigma_1, \nu_2^{(k)}\sigma_2^{(k)}\right).$$

(e) Do l *step by the Power method*

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^l.$$

The mixed aggregation-disaggregation method devised in the thesis is computationally cheaper than PAM. It is computationally equivalent to FAM, excepting the first iteration, since after the first iteration we use procedure which is equivalent to iteration of FAM. In the same time, it possesses the same convergence rate as PAM which is better than FAM, since the sequence of intermediate vectors of the method is the same as in PAM, which is proven in Theorem 2.3.

2.6 Discussion and related works

Let us formulate the PageRank problem as a linear problem instead of the eigenvector problem as in (1.9). Assuming that the dangling node vector and the personalization vector are equal, Google matrix G defined by (1.8) can be written as [62]

$$G = cP + (1-c)\mathbf{1}v = cH + (ca + (1-c)\mathbf{1})v.$$

Then PageRank vector can be found as solution of the linear system [62]

$$\bar{\pi}(I - cH) = v$$

and normalization $\pi = \frac{\bar{\pi}}{\bar{\pi}\mathbf{1}}$. After a simetric reordering, matrix H has the following structure [62]:

$$H = \begin{pmatrix} H_{11} & H_{12} & H_{13} & \dots & H_{1N} \\ & 0 & H_{23} & \dots & H_{2N} \\ & & 0 & \dots & H_{3N} \\ & & & \ddots & \\ & & & & 0 \end{pmatrix}. \tag{2.24}$$

The last block row of the decomposition corresponds to the dangling pages on the Web. The matrix in the linear system is then written as

$$(I - cH) = \begin{pmatrix} I - cH_{11} & -cH_{12} & -cH_{13} & \dots & -cH_{1N} \\ & I & -cH_{23} & \dots & -cH_{2N} \\ & & I & \dots & -cH_{3N} \\ & & & \ddots & \\ & & & & I \end{pmatrix}.$$

Let PageRank vector be decomposed as in (2.4) and the solution of the linear system and personalisation vector v are decomposed in the same way:

$$\begin{aligned} \bar{\pi} &= (\bar{\pi}_1, \bar{\pi}_2, \dots, \bar{\pi}_N), \\ v &= (v_1, v_2, \dots, v_N). \end{aligned}$$

The only system that should be solved directly is the first system, $\bar{\pi}_1 (I - cH_{11}) = v_1$. The remaining subvectors of $\bar{\pi}$ are computed quickly and efficiently by forward substitution. We formalize the process in the following algorithm.

Algorithm 2.8 *Determine exact PageRank vector π by reordering Web pages and substitution.*

1. *Reorder the states of the Markov chain so that reordered matrix H has the structure of* (2.24).
2. *Solve for $\bar{\pi}_1$ in $\bar{\pi}_1 (I - cH_{11}) = v_1$.*
3. *For $i = \overline{2, N}$ compute $\bar{\pi}_i = c \sum_{j=1}^{i-1} \bar{\pi}_j H_{ji} + v_i$.*
4. *Normalize $\pi = \frac{\bar{\pi}}{\bar{\pi}\mathbf{1}}$.*

Although, the above algorithms makes an restrictive assumption that the dangling node vector is equal to the personalization vector, this algorithm exploits deeper the specific structure of hyper-link matrix H than FTSA algorithm that takes into account only the dangling pages. Dimension n_1 of matrix H_{11} depends on the number of zero rows that can be successively squeezed out of the submatrices. In practical experiments, authors [62] report that n_1 is about $30 - 50\%$ of the number of the part of the Web collected by them. In the scale of the whole Web it can be such a large number that it makes useless the direct methods used in step 2 of the above algorithm. In that case, either an iterative algorithm should be used or matrix H_{11} should be decomposed further. Since all the possible zero rows were already squeezed, we cannot reduce the dimension of matrix H_{11} as it proposed above [62].

The equivalence conditions of FAM and PAM, discussed in the previous section, lead to the exploiting of dangling pages when the conditions are applied to PageRank. Then, the above

algorithm exploits the structure of the Google matrix more extensively than the equivalence conditions. At the same time, in general case of the stationary distribution, the equivalence condition cover wider range of transition matrices since the condition that $\mathrm{rank}P_{21} = 1$ is weaker than the condition that $H_{21} = 0$ as it is required by the above algorithm.

To use the mixed aggregation-disaggregation algorithm, it is required to discover a particular structure of the Web, but if somebody would like to use either FAM or PAM algorithms then he/she still needs to find a partitioning of the set of the Web pages that posses some properties since otherwise the methods can converge slower than or as quick as, in case of PageRank, the Power method. Hence, the need to find a particular partitioning is not a drawback of the mixed aggregation-disaggregation algorithm only, but it is the property of all the aggregation-disaggregation algorithms.

2.7 Conclusions

In this chapter, we reviewed several approaches to the acceleration of the computation of the PageRank vector. We considered adaptive approach when the converged PageRank values are fixed in the further iteration. Also we discussed extrapolation approaches when the intermediate results are used to adjust the PageRank vector, but the most of our attention we devoted to the aggregation-disaggregation approach. We reviewed the BlockRank algorithm which exploits the site structure of the Web and also the Fast Two-Stage Algorithm which uses the presence of dangling pages in order to reduce computation. We presented the full aggregation-disaggregation algorithm and revealed the estimation of its convergence rate when the algorithm is applied to the Google matrix. Also we discussed the partial aggregation-disaggregation algorithm and discovered the equivalence conditions of the last two algorithms which let us formulate a novel method possessing the best properties of both the algorithms as the smaller computational cost of the full aggregation-disaggregation algorithm and the better convergence rate of the partial aggregation-disaggregation algorithm.

3

Quasi-Stationary Distributions as Centrality Measures for the Giant Strongly Connected Component of a Reducible Graph

3.1 Summary

The choice of the damping factor which is an essential input parameter in the PageRank algorithm is not evident, but the value of the damping factor has a crucial influence on the ranking produced by the PageRank algorithm and the properties of a number of methods developed for the PageRank computation. Due to the importance of the damping factor, a significant attention was attracted to the consideration of the problem. Some authors proposed specific fixed values of the damping factor supporting them by a number of arguments; others came to the idea to avoid the particular choice of the value. In this chapter, we propose to use a parameter-free centrality measure which is based on the notion of a quasi-stationary distribution. Specifically, we suggest four quasi-stationary based centrality measures, analyze them and conclude that they produce approximately the same ranking.

3.2 Introduction

The choice of the value of the damping factor which is an essential input parameter in the PageRank algorithm is an important problem having no evident solution so far. If someone fixes the damping factor equal to 0, she gets the uniform distribution as the ranking of the Web pages. Obviously, it does not make any sense. In the same time, as the other extreme case, the choosing the damping factor be equal to unity is at the same level of rationality. If the damping factor equals to one, the rank tends to concentrate at few pages called *rank sinks*. Google has chosen 0.85 as the damping factor arguing the choice by the behaviour of surfers [71]. In that case, a surfer does about 6 clicks on hyper-links before jumping to a new topic using, for example, a search engine.

The authors of several papers [18, 22, 31, 81] have suggested methods to overcome the problem of a particular choice of the damping factor. In [18], authors proposed to use damping functions instead of the damping factor. In the standard PageRank method, the values of PageRank exponentially decay from one page to another, and, hence, can be described as an exponentially decreasing function of the length of the path between the pages, which is called a damping function. They consider the other damping functions leading to the linear and hyperbolic decay on the length of the paths. However, there are also input parameters for the damping functions. In [22], the author considered TotalRank which is the averaged PageRank over all the values of the damping factor. The authors of [31] considered the damping factor as a random variable which brings PageRank itself being a random variable. They argued that using a particular value of the damping factor failed to model the behaviour of all the Web surfers. Uniform and beta distributions are considered as the distribution of the damping factor as a random variable, and, in particular, the approach is equivalent to TotalRank in the case of the uniform distribution of the damping factor. Inspite of making the damping factor to be a random variabel, one needs to define a shape of its probability distribution.

Other authors suggested different values of the damping factor arguing it in different ways [14, 23, 30]. An application of the PageRank algorithm on the Physical Review citation network is studied in [30]. The authors arrived to the conclusion that, in contrast to the Web surfers, a scientist unlikely follows more than two levels of the references from a paper. The observation leads to 0.5 as the value of the damping factor. In [14], authors discuss the distribution of PageRank among the principal components of the Web, and they concluded that the best choice of the damping factor is 0.5 to keep the enough proportion of PageRank in the central component which contains the most authoritative pages. The representation of the Web with its principal components will be discussed below in this chapter. In [23], authors considered PageRank as a function of the damping factor revealing the dependence of PageRank on it.

The goal of the chapter is to explore parameter-free centrality measures, i.e. measures

that determine the relative importance of the Web pages. Here we suggest centrality measures which take as an input only the adjacency list of a graph.

In [14, 28, 56], the authors have studied the graph structure of the Web. In particular, in [28, 56], it was shown that the Web Graph can be divided into three principal components: the Giant Strongly Connected Component, to which we simply refer as the SCC component, the IN component and the OUT component. The SCC component is the largest strongly connected component in the Web Graph. In fact, it is larger than the second largest strongly connected component by several orders of magnitude. Following hyper-links one can come from the IN component to the SCC component, but it is not possible to return back. Then, from the SCC component one can come to the OUT component, and it is not possible to return to SCC from the OUT component. In [28, 56], the analysis of the structure of the Web was made assuming that dangling nodes have no outgoing links. However, according to corrections made in hyper-link matrix in (1.3), there is a nonzero probability to jump from a dangling node to an arbitrary node. This can be viewed as a link between the nodes, and we call such a link an *artificial link*. As was shown in [14], these artificial links significantly change the graph structure of the Web. In particular, the artificial links of dangling nodes in the OUT component connect some parts of the OUT component with IN and SCC components. Thus, the size of the Giant Strongly Connected Component increases further. If the artificial links from dangling nodes are taken into account, it is shown in [14] that the Web Graph can be divided into two components: the Extended Strongly Connected Component (ESCC) and the Pure OUT component (POUT). POUT is small in size, but if the damping factor c is chosen equal to one, the random walk is absorbed with probability one into POUT. As we show in the numerical experiments section, a large majority of pages and nearly all the important pages are in the ESCC. We also note that even if the damping factor is chosen close to one, the random walk can spend a significant amount of time in the ESCC before the absorption. Therefore, for ranking Web pages from the ESCC we suggest the use of quasi-stationary distributions [33, 73] since they represent the dynamics of the random walk before it leaves the ESCC. We would like to note that our analysis based on quasi-stationarity can also be applied to rank nodes in the Giant Strongly Connected Component of the original graph if we assume that the artificial links from the dangling nodes point only back to the dangling nodes.

It turns out that there are several versions of the quasi-stationary distribution. Here we study four versions of the quasi-stationary distribution. Our main conclusion is that the rankings provided by them are very similar. Therefore, one can choose the version of the stationary distribution which is easier for computation.

The chapter is organized as follows. In the next Section 3.3 we discuss different notions of quasi-stationarity. The relation among them, and the relation between the quasi-stationary distributions and PageRank are considered in Section 3.4. Then, in Section 3.5 we present the

results of numerical experiments on the Web Graph which confirm our theoretical findings.

3.3 Quasi-stationary distributions as centrality measures

As noted in [14], by renumbering the nodes, the transition matrix P can be transformed to the following form:

$$P = \begin{pmatrix} Q & 0 \\ R & T \end{pmatrix},$$

where the block T corresponds to the ESCC, block Q corresponds to the part of the OUT component without dangling nodes and their predecessors, specifically, to POUT, and block R corresponds to the transitions from the ESCC to the nodes in block Q. Since matrix T corresponds to the ESCC, it is irreducible.

POUT is small in size, but if the damping factor c is chosen equal to one, the random walk is absorbed with probability one into POUT. We are mostly interested in the nodes in the ESCC. We recall that the ESCC can be regarded as the Giant Strongly Connected Component of a graph modified by the addition of artificial links from dangling nodes. Denote by π_Q the part of the PageRank vector corresponding to POUT and denote by π_T the part of the PageRank vector corresponding to the ESCC. Using the following formula (1.6)

$$\pi(c) = \frac{1-c}{n}\mathbf{1}^T(I - cP)^{-1},$$

we conclude that

$$\pi_T(c) = \frac{1-c}{n}\mathbf{1}^T(I - cT)^{-1},$$

where $\mathbf{1}$ is a vector of ones of an appropriate dimension.

Let us define the renormalized part of the PageRank vector corresponding to the ESCC:

$$\hat{\pi}_T(c) = \frac{\pi_T(c)}{\|\pi_T(c)\|_1}. \tag{3.1}$$

We note that this renormalization does not alter the rank among the nodes inside the ESCC.

We define four quasi-stationary distributions and provide intuitive explanations clarifying their meaning.

Definition 2 *The pseudo-stationary distribution $\hat{\pi}_T$ is given by*

$$\hat{\pi}_T = \frac{\mathbf{1}^T[I - T]^{-1}}{\mathbf{1}^T[I - T]^{-1}\mathbf{1}}.$$

We recall that the i, j^{th} element of the matrix $[I - T]^{-1}$ gives the expected number of visits to state j starting from state i [51]. Hence, the i^{th} component of $\hat{\pi}_T$ can be interpreted as the

fraction of time the random walk (with $c = 1$) spends in node i prior to absorption. We recall that the random walk as defined above, particularly in the formula for PageRank (1.5), starts from the uniform distribution. If the random walk were initiated from another distribution, the pseudo-stationary distribution would change. More detailed discussion of the properties of this notion of quasi-stationarity can be found in [33, 36]. Let us give the following definition from [83].

Definition 3 *Let $x = (x_1, x_2, \ldots, x_n)$ be a real vector, and σ a permutation that orders the elements of x in decreasing order, $x_{\sigma(1)} \geqslant x_{\sigma(2)} \geqslant \cdots \geqslant x_{\sigma(n)}$. Then the ordinal rank of an individual element is $\mathrm{Orank}(x_i) = \sigma(i)$, $i = \overline{1,n}$, and the ordinal rank of the whole vector is $\mathrm{Orank}(x) = (\mathrm{Orank}(x_1), \mathrm{Orank}(x_2), \ldots, \mathrm{Orank}(x_n))$.*

Proposition 3.1 below shows a relation between the pseudo-stationary distribution and renormalized PageRank.

Proposition 3.1 *The following limit exists*

$$\hat{\pi}_T = \lim_{c \to 1} \hat{\pi}_T(c),$$

and the ranking of pages in the ESCC provided by the PageRank vector converges to the ranking provided by $\hat{\pi}_T$ as the damping factor goes to one. Moreover, these two ordinal rankings producted by $\hat{\pi}_T$ and $\hat{\pi}_T(c)$ coincide for all values of c above some value c^*.

Proof Let us prove first of all that $\hat{\pi}_T = \lim_{c \to 1} \hat{\pi}_T(c)$.

$$\hat{\pi}_T(c) \;=\; \frac{\pi_T(c)}{\|\pi_T(c)\|_1} = \frac{\pi_T(c)}{\pi_T(c)\mathbf{1}} = \frac{\frac{1-c}{n}\mathbf{1}^T[I - cT]^{-1}}{\frac{1-c}{n}\mathbf{1}^T[I - cT]^{-1}\mathbf{1}} = \frac{\mathbf{1}^T[I - cT]^{-1}}{\mathbf{1}^T[I - cT]^{-1}\mathbf{1}}.$$

Since $I - T$ is invertible, the limit is justified. This ends this part of the proof.

Let us now prove that two rankings $\hat{\pi}_T(c)$ and $\hat{\pi}_T$ coincide for all values of c above some value c^*. Assuming that

$$\hat{\pi}_T(c) = \left(\hat{\pi}_T^{(1)}(c), \hat{\pi}_T^{(2)}(c), \ldots, \hat{\pi}_T^{(n)}(c)\right), \qquad \hat{\pi}_T = \left(\hat{\pi}_T^{(1)}, \hat{\pi}_T^{(2)}, \ldots, \hat{\pi}_T^{(n)}\right),$$

this is equivalent to proving that $\exists c^* : 0 < c^* < 1$ so that $\forall c : 0 < c^* < c \leqslant 1$, $\forall i, j = \overline{1,n}$, $i \neq j$: $\hat{\pi}_T^{(i)}(c) - \hat{\pi}_T^{(j)}(c)$ keeps its sign.

Let us denote $\mathrm{adj}(I - cT)$ by $A(c)$ and $\det(I - cT)$ by $D(c)$. Hence, we have

$$(I - cT)^{-1} = D^{-1}(c)A(c).$$

If we denote $(I - cT)^{-1}$ by $M = \{m_{ij}\}$, where $i, j = \overline{1,n}$, we can write that

$$m_{ij} = \frac{A_{ij}(c)}{D(c)}.$$

According to (3.1), we can express $\hat{\pi}_T^{(k)}(c)$ as follows:

$$\hat{\pi}_T^{(k)}(c) = \frac{\sum_{i=1}^{n_T} m_{ik}}{\sum_{i=1}^{n_T}\sum_{j=1}^{n_T} m_{ij}} = \frac{\sum_{i=1}^{n_T} \frac{A_{ik}(c)}{D(c)}}{\sum_{i=1}^{n_T}\sum_{j=1}^{n_T} \frac{A_{ij}(c)}{D(c)}} = \frac{\sum_{i=1}^{n_T} A_{ik}(c)}{\sum_{i=1}^{n_T}\sum_{j=1}^{n_T} A_{ij}(c)}.$$

Because the inverse $(I - cT)^{-1}$ exists for all $c \in [0,1]$, $\sum_{i=1}^{n_T}\sum_{j=1}^{n_T} A_{ij}(c) \neq 0$ for all $c \in [0,1]$. Let us denote by $Z_{k_1k_2}(c)$ the following:

$$Z_{k_1k_2}(c) = \left(\hat{\pi}_T^{(k_1)}(c) - \hat{\pi}_T^{(k_2)}(c)\right) \sum_{i=1}^{n_T}\sum_{j=1}^{n_T} A_{ij}(c).$$

If the ranking provided by $\hat{\pi}_T$ changes at some value c, then $Z_{k_1k_2}(c)$ becomes equal to zero. If $Z_{k_1k_2}(c) = 0$ for all $0 \leqslant c \leqslant 1$, then $\hat{\pi}_T^{(k_1)}(c) = \hat{\pi}_T^{(k_2)}(c)$ for all $0 \leqslant c \leqslant 1$, and we can put them in any order. If there is such $0 \leqslant c \leqslant 1$ that $Z_{k_1k_2}(c) \neq 0$ then, since $Z_{k_1k_2}(c)$ is a polynomial of degree $n_T - 1$, $Z_{k_1k_2}(c)$ can have no more than $n_T - 1$ isolated roots. Let us denote by $z_{k_1k_2}$ the biggest root of $Z_{k_1k_2}(c)$ when $c \in [0,1]$, i.e.,

$$z_{k_1k_2} = \max\{c | c \in [0,1], Z_{k_1k_2}(c) = 0\}.$$

Let us denote by $z'_{k_1k_2}$ the biggest root of $Z_{k_1k_2}(c)$ when $c \in [0,1)$, i.e.,

$$z'_{k_1k_2} = \max\{c | c \in [0,1), Z_{k_1k_2}(c) = 0\}.$$

If $z_{k_1k_2} \neq 1$, then $\hat{\pi}_T^{(k_1)}(c) \neq \hat{\pi}_T^{(k_2)}(c)$ for all $c \in [z_{k_1k_2}, 1]$ and the ranking does not change for all $c \in [z_{k_1k_2}, 1]$. If $z_{k_1k_2} = 1$, then $\hat{\pi}_T^{(k_1)}(c) \neq \hat{\pi}_T^{(k_2)}(c)$ for all $c \in [z'_{k_1k_2}, 1)$, and the ranking does not change for all $c \in [z'_{k_1k_2}, 1)$. In the last case, since the limiting entries $\hat{\pi}_T^{(k_1)}$ and $\hat{\pi}_T^{(k_2)}$ are equal to each other, we can put them in the same order as $\hat{\pi}_T^{(k_1)}(c)$ and $\hat{\pi}_T^{(k_2)}(c)$ for some $c \in [z'_{k_1k_2}, 1)$. Choosing $c^* = \max_{k_1,k_2}\{z'_{k_1k_2}\}$ completes the proof. □

In more general setting, the existence of the limit of the PageRank vector when c approaches unity is considered in [26, 43].

Definition 4 *The quasi-stationary distribution $\tilde{\pi}_T$ is defined by the equation*

$$\tilde{\pi}_T T = \lambda_1 \tilde{\pi}_T, \tag{3.2}$$

and the normalization condition

$$\tilde{\pi}_T \mathbf{1} = 1, \tag{3.3}$$

where λ_1 is the Perron-Frobenius eigenvalue of matrix T.

The irreducibility of matrix T guarantees the uniqueness of its Perron-Frobenius eigenvalue and eigenvector [51]. The quasi-stationary distribution $\tilde{\pi}_T$ can be interpreted as a proper initial

distribution on the non-absorbing states (states in ESCC) which is such that the distribution of the random walk, conditioned on the non-absorption prior time t, is independent of t [35]. An interested reader can find more detailed discussion of the properties of the quasi-stationary distribution $\tilde{\pi}_T$ in [33, 73].

Denote by $\bar{T}$ the normalized hyper-link matrix associated with the ESCC when the links leading outside of the ESCC are neglected. Clearly, we have

$$\bar{T}_{ij} = \frac{T_{ij}}{[T\mathbf{1}]_i},$$

where $[T\mathbf{1}]_i$ denotes the i^{th} component of the vector $T\mathbf{1}$. In other words, $[T\mathbf{1}]_i$ is the sum of the elements in row i of matrix T. Now we can define the third quasi-stationary distribution.

Definition 5 *The quasi-stationary distribution $\bar{\pi}_T$ is defined by the equation*

$$\bar{\pi}_T \bar{T} = \bar{\pi}_T, \tag{3.4}$$

and the normalization condition

$$\bar{\pi}_T \mathbf{1} = 1. \tag{3.5}$$

The $\bar{T}_{ij}$ entry of the matrix $\bar{T}$ can be viewed as a conditional probability to jump from node i to node j under the condition that the random walk does not leave the ESCC at the jump. Then, $\bar{\pi}_T$ can be interpreted as a stationary distribution of the random walk under the above condition.

We can generalize the notion of $\bar{\pi}_T$. Namely, we consider the situation when the random walk stays inside the ESCC after some finite number of jumps. The probability of such an event can be formally expressed as follows:

$$P\left(X_1 = j | X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S\right),$$

where X_k is the state which visits the random walk at time step k, N is the number of jumps during which the random walk stays in the ESCC, and the ESCC is denoted by S for the sake of shortening the notation, and symbol $\wedge$ denotes "and" meaning that the events happen together.

Let us denote by $T_{ij}^{(N)}$ the i,j^{th} element of T^N (the N-th power of T) and by $T_i^{(N)}$ the i^{th} row of the matrix T^N. Then

$$T_i^{(N)} = (T^N)_i = (TT^{N-1})_i = T_i T^{N-1}.$$

Proposition 3.2 *The following expression holds:*

$$P\left(X_1 = j | X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S\right) = \frac{T_{ij} T_j^{(N-1)} \mathbf{1}}{T_i^{(N)} \mathbf{1}}. \tag{3.6}$$

Proof The proof is quite technical, exploiting a conditional probability formula and independence assumption.

$$P\left(X_1 = j | X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S\right) =$$
$$= \frac{P\left(X_0 = i \wedge X_1 = j \wedge \bigwedge_{m=2}^{N} X_m \in S\right)}{P\left(X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S\right)}.$$

First let us develop the denominator:

$$P\left(X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S\right) =$$

We write $X_m \in S$ in element-wise form.

$$= P\left(X_0 = i \wedge \bigwedge_{m=1}^{N} \bigvee_{k_m \in S} X_m = k_m\right) =$$

Extract the first step and add conditioning on $X_0 = i$ which does not matter since it is a Markov chain.

$$= P\left(X_0 = i \wedge \bigvee_{k_1 \in S} X_1 = k_1 \wedge \bigwedge_{m=2}^{N} \bigvee_{k_m \in S} X_m = k_m | X_0 = i\right) =$$

Get $P(X_0 = i)$ and summation $k_1 \in S$ over outside.

$$= P(X_0 = i) \sum_{k_1 \in S} P\left(X_1 = k_1 \wedge \bigwedge_{m=2}^{N} \bigvee_{k_m \in S} X_m = k_m | X_0 = i\right) =$$

Using conditional probability formula, we take out $X_1 = k_1$ and put conditioning on it.

$$= P(X_0 = i) \sum_{k_1 \in S} P(X_1 = k_1 | X_0 = i) P\left(\bigwedge_{m=2}^{N} \bigvee_{k_m \in S} X_m = k_m | X_1 = k_1 \wedge X_0 = i\right) =$$

Since we deal with a Markov chain, we do not need the conditioning on $X_0 = i$ any more.

$$= P(X_0 = i) \sum_{k_1 \in S} P(X_1 = k_1 | X_0 = i) P\left(\bigvee_{k_2 \in S} X_2 = k_2 \wedge \bigwedge_{m=3}^{N} \bigvee_{k_m \in S} X_m = k_m | X_1 = k_1\right) =$$

We repeat the above reasoning for the next step of the Markov chain.

$$
\begin{aligned}
&= \mathrm{P}(X_0 = i) \sum_{k_1 \in S} \mathrm{P}(X_1 = k_1 | X_0 = i) \sum_{k_2 \in S} \mathrm{P}\left(X_2 = k_2 \wedge \bigwedge_{m=3}^{N} \bigvee_{k_m \in S} X_m = k_m | X_1 = k_1 \right) = \\
&= \mathrm{P}(X_0 = i) \sum_{k_1 \in S} \mathrm{P}(X_1 = k_1 | X_0 = i) \times \\
&\times \sum_{k_2 \in S} \mathrm{P}\left(\bigwedge_{m=3}^{N} \bigvee_{k_m \in S} X_m = k_m | X_2 = k_2 \wedge X_1 = k_1 \right) \mathrm{P}(X_2 = k_2 | X_1 = k_1) = \\
&= \mathrm{P}(X_0 = i) \sum_{k_1 \in S} \mathrm{P}(X_1 = k_1 | X_0 = i) \times \\
&\times \sum_{k_2 \in S} \mathrm{P}\left(\bigwedge_{m=3}^{N} \bigvee_{k_m \in S} X_m = k_m | X_2 = k_2 \right) \mathrm{P}(X_2 = k_2 | X_1 = k_1) = \\
&= \mathrm{P}(X_0 = i) \sum_{k_2 \in S} \mathrm{P}\left(\bigwedge_{m=3}^{N} \bigvee_{k_m \in S} X_m = k_m | X_2 = k_2 \right) \times \\
&\times \sum_{k_1 \in S} \mathrm{P}(X_2 = k_2 | X_1 = k_1) \mathrm{P}(X_1 = k_1 | X_0 = i) = \\
&= \mathrm{P}(X_0 = i) \sum_{k_2 \in S} \mathrm{P}\left(\bigwedge_{m=3}^{N} \bigvee_{k_m \in S} X_m = k_m | X_2 = k_2 \right) \mathrm{P}(X_2 = k_2 | X_0 = i) =
\end{aligned}
$$

The last summation is actually probability for the Markov chain to jump on two steps.

We repeat the above reasoning for the next step of the Markov chain.

$$
\begin{aligned}
&= P(X_0 = i) \sum_{k_2 \in S} P\left(\bigvee_{k_3 \in S} X_3 = k_3 \wedge \bigwedge_{m=4}^{N} \bigvee_{k_m \in S} X_m = k_m | X_2 = k_2 \right) P(X_2 = k_2 | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_2 \in S} \sum_{k_3 \in S} P\left(X_3 = k_3 \wedge \bigwedge_{m=4}^{N} \bigvee_{k_m \in S} X_m = k_m | X_2 = k_2 \right) P(X_2 = k_2 | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_3 \in S} \sum_{k_2 \in S} P\left(\bigwedge_{m=4}^{N} \bigvee_{k_m \in S} X_m = k_m | X_3 = k_3 \wedge X_2 = k_2 \right) \times \\
&\times P(X_3 = k_3 | X_2 = k_2) P(X_2 = k_2 | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_3 \in S} \sum_{k_2 \in S} P\left(\bigwedge_{m=4}^{N} \bigvee_{k_m \in S} X_m = k_m | X_3 = k_3 \right) \times \\
&\times P(X_3 = k_3 | X_2 = k_2) P(X_2 = k_2 | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_3 \in S} P\left(\bigwedge_{m=4}^{N} \bigvee_{k_m \in S} X_m = k_m | X_3 = k_3 \right) \times \\
&\times \sum_{k_2 \in S} P(X_3 = k_3 | X_2 = k_2) P(X_2 = k_2 | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_3 \in S} P\left(\bigwedge_{m=4}^{N} \bigvee_{k_m \in S} X_m = k_m | X_3 = k_3 \right) P(X_3 = k_3 | X_0 = i) = \dots
\end{aligned}
$$

We repeat the above reasoning for all the following steps of the Markov chain.

$$
\begin{aligned}
\dots &= P(X_0 = i) \sum_{k_{N-2} \in S} P\left(\bigwedge_{m=N-1}^{N} \bigvee_{k_m \in S} X_m = k_m | X_{N-2} = k_{N-2} \right) P(X_{N-2} = k_{N-2} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_{N-2} \in S} P\left(\bigvee_{k_{N-1} \in S} X_{N-1} = k_{N-1} \wedge \bigvee_{k_N \in S} X_N = k_N | X_{N-2} = k_{N-2} \right) \times \\
&\times P(X_{N-2} = k_{N-2} | X_0 = i) =
\end{aligned}
$$

$$
\begin{aligned}
&= P(X_0 = i) \sum_{k_{N-2} \in S} \sum_{k_{N-1} \in S} P\left(X_{N-1} = k_{N-1} \wedge \bigvee_{k_N \in S} X_N = k_N | X_{N-2} = k_{N-2} \right) \times \\
&\times \quad P(X_{N-2} = k_{N-2} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_{N-2} \in S} \sum_{k_{N-1} \in S} P\left(\bigvee_{k_N \in S} X_N = k_N | X_{N-1} = k_{N-1} \wedge X_{N-2} = k_{N-2} \right) \times \\
&\times \quad P(X_{N-1} = k_{N-1} | X_{N-2} = k_{N-2}) \, P(X_{N-2} = k_{N-2} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_{N-2} \in S} \sum_{k_{N-1} \in S} P\left(\bigvee_{k_N \in S} X_N = k_N | X_{N-1} = k_{N-1} \right) \times \\
&\times \quad P(X_{N-1} = k_{N-1} | X_{N-2} = k_{N-2}) \, P(X_{N-2} = k_{N-2} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_{N-1} \in S} P\left(\bigvee_{k_N \in S} X_N = k_N | X_{N-1} = k_{N-1} \right) \times \\
&\times \quad \sum_{k_{N-2} \in S} P(X_{N-1} = k_{N-1} | X_{N-2} = k_{N-2}) \, P(X_{N-2} = k_{N-2} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_{N-1} \in S} P\left(\bigvee_{k_N \in S} X_N = k_N | X_{N-1} = k_{N-1} \right) P(X_{N-1} = k_{N-1} | X_0 = i) =
\end{aligned}
$$

We repeat the above reasoning for the last step of the Markov chain.

$$
\begin{aligned}
&= P(X_0 = i) \sum_{k_{N-1} \in S} \sum_{k_N \in S} P(X_N = k_N | X_{N-1} = k_{N-1}) \, P(X_{N-1} = k_{N-1} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_N \in S} \sum_{k_{N-1} \in S} P(X_N = k_N | X_{N-1} = k_{N-1}) \, P(X_{N-1} = k_{N-1} | X_0 = i) = \\
&= P(X_0 = i) \sum_{k_N \in S} P(X_N = k_N | X_0 = i) = \\
&= \sum_{k_N = 1}^{n_T} T_{i k_N}^{(N)} P(X_0 = i) = \\
&= T_i^{(N)} \mathbf{1} P(X_0 = i) .
\end{aligned}
$$

Hence, we obtain

$$
P\left(X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S \right) = T_i^{(N)} \mathbf{1} P(X_0 = i) .
$$

Now let us develop the numerator:

$$
P\left(X_0 = i \wedge X_1 = j \wedge \bigwedge_{m=2}^{N} X_m \in S \right) =
$$

$$= P\left(\bigwedge_{m=2}^{N}\bigvee_{k_m\in S} X_m = k_m\right) P(X_1 = j|X_0 = i)\, P(X_0 = i) =$$
$$= T_{ij}T_j^{(N-1)}\mathbf{1}P(X_0 = i).$$

Therefore, we get

$$P\left(X_0 = i \wedge X_1 = j \wedge \bigwedge_{m=2}^{N} X_m \in S\right) = T_{ij}T_j^{(N-1)}\mathbf{1}P(X_0 = i).$$

□

Then, if we denote

$$\check{T}_{ij}^{(N)} = P\left(X_1 = j|X_0 = i \wedge \bigwedge_{m=1}^{N} X_m \in S\right),$$

we will be able to find the stationary distribution of $\check{T}_{ij}^{(N)}$, which can be viewed as a generalization of $\bar{\pi}_T$. Let us now consider the limiting case, when N goes to infinity.

We shall refer to the following limit as the twisted kernel

$$\check{T}_{ij} = \lim_{N\to\infty} \frac{T_{ij}T_j^{(N-1)}\mathbf{1}}{T_i^{(N)}\mathbf{1}}. \tag{3.7}$$

The existence of the limit and its explicit expression are given in the following theorem.

Theorem 3.1 *The limit in (3.7) exists if* $|\lambda_1| > |\lambda_2|$*, where* λ_1 *is the Perron-Frobenius eigenvalue of* T*, and* λ_2 *is the second by magnitude eigenvalue of* T*. The twisted kernel can be expressed as follows:*

$$\check{T}_{ij} = \frac{T_{ij}u_j}{\lambda_1 u_i}, \tag{3.8}$$

where u *is the right Perron-Frobenius eigenvector of* T*, namely,* $Tu = \lambda_1 u$.

Proof Let us introduce here auxiliary normalization of the right Perron-Frobenius eigenvector of T:

$$T\tilde{u} = \lambda_1\tilde{u}, \qquad \tilde{\pi}_T\tilde{u} = 1.$$

Let us note that [66, Theorem 11.18]

$$\tilde{u}_i = \lim_{N\to\infty} \frac{T_i T^{N-1}\mathbf{1}}{\lambda_1^N}. \tag{3.9}$$

We can perform the following transformations:

$$\begin{aligned}\frac{T_{ij}T_j^{(N-1)}\mathbf{1}}{T_i^{(N)}\mathbf{1}} &= T_{ij}\frac{T_jT^{N-2}\mathbf{1}}{T_iT^{N-1}\mathbf{1}} = \frac{T_{ij}}{\lambda_1}\frac{\frac{T_jT^{N-2}\mathbf{1}}{\lambda_1^{N-1}}}{\frac{T_iT^{N-1}\mathbf{1}}{\lambda_1^N}}.\\ \lim_{N\to\infty}\frac{T_{ij}T_j^{(N-1)}\mathbf{1}}{T_i^{(N)}\mathbf{1}} &= \frac{T_{ij}}{\lambda_1}\lim_{N\to\infty}\frac{\frac{T_jT^{N-2}\mathbf{1}}{\lambda_1^{N-1}}}{\frac{T_iT^{N-1}\mathbf{1}}{\lambda_1^N}} = \frac{T_{ij}}{\lambda_1}\frac{\lim_{N\to\infty}\frac{T_jT^{N-2}\mathbf{1}}{\lambda_1^{N-1}}}{\lim_{N\to\infty}\frac{T_iT^{N-1}\mathbf{1}}{\lambda_1^N}}.\end{aligned}$$

Using (3.9), we can write

$$\lim_{N\to\infty}\frac{T_{ij}T_j^{(N-1)}\mathbf{1}}{T_i^{(N)}\mathbf{1}} = \frac{T_{ij}\tilde{u}_j}{\lambda_1\tilde{u}_i}.$$

After renormalization, we obtain

$$\lim_{N\to\infty}\frac{T_{ij}T_j^{(N-1)}\mathbf{1}}{T_i^{(N)}\mathbf{1}} = \frac{T_{ij}u_j}{\lambda_1 u_i}.$$

□

As one can see, the twisted kernel does not depend on the normalization of u. Hence, we can take any normalization.

The twisted kernel plays an important role in multiplicative ergodic theory and large deviations for Markov chains: see, e.g., [55]. The matrix $\check{T}$ is clearly a transition probability kernel: i.e., $\check{T}_{ij} \geqslant 0\ \forall i,j$, and $\sum_j \check{T}_{ij} = 1,\ \forall i$. Also, it is irreducible if there exists a path $i \to j$ under T for all i,j, which we assume to be the case. In particular, it will have a unique stationary distribution $\check{\pi}_T$, which gives the fourth notion of the quasi-stationary distribution.

Definition 6 *The quasi-stationary distribution $\check{\pi}_T$ is defined as the stationary distribution of the twisted kernel. Namely, it is the solution of the following eigenvector equation and normalization condition:*

$$\check{\pi}_T = \check{\pi}_T\check{T}, \tag{3.10}$$

$$\check{\pi}_T\mathbf{1} = 1. \tag{3.11}$$

If we assume aperiodicity in addition, $\check{T}_{ij}$ can be given the interpretation of the probability of transition from i to j in the ESCC for the chain, conditioned on the fact that it never leaves the ESCC.

Proposition 3.3 *The following expression for $\check{\pi}_T$ holds:*

$$\check{\pi}_{Ti} = \tilde{\pi}_{Ti}\tilde{u}_i, \tag{3.12}$$

where $\tilde{u}$ is the right Perron-Frobenius eigenvector of T, which is normalized as follows:

$$\tilde{u} = \frac{u}{\tilde{\pi}_T u}. \tag{3.13}$$

Proof The normalization condition (3.11) is satisfied due to (3.13). Let us show that (3.10) holds as well: i.e.,

$$\breve{\pi}_{\breve{T}j} = \sum_{i=1}^{n_T} \breve{\pi}_{\breve{T}i} \breve{T}_{ij},$$

where n_T is the dimension of $\breve{\pi}_T$. Using the expression (3.8), we can write

$$\sum_{i=1}^{n_T} \tilde{\pi}_{\tilde{T}i} \tilde{u}_i \breve{T}_{ij} = \sum_{i=1}^{n_T} \tilde{\pi}_{\tilde{T}i} \tilde{u}_i \frac{T_{ij}\tilde{u}_j}{\lambda_1 \tilde{u}_i} = \sum_{i=1}^{n_T} \tilde{\pi}_{\tilde{T}i} \tilde{u}_i \frac{T_{ij}\tilde{u}_j}{\lambda_1 \tilde{u}_i} = \frac{\tilde{u}_j}{\lambda_1} \lambda_1 \tilde{\pi}_{\tilde{T}j} = \tilde{\pi}_{\tilde{T}j} \tilde{u}_j,$$

which proves the proposition. □

3.4 Relationships among quasi-stationary distributions

After the introduction of the four quasi-stationary distributions, let us now establish the relationships among them.

Let us now consider the substochastic matrix T as a perturbation of the stochastic matrix $\bar{T}$. We introduce the perturbation term

$$\varepsilon D = \bar{T} - T,$$

where the parameter ε is the perturbation parameter, which is typically small. Let us proof an auxiliary lemma before we proceed.

Lemma 3.1 *Let $\bar{T}$ be an irreducible stochastic matrix. And let $T(\varepsilon) = \bar{T} - \varepsilon D$ be a perturbation of $\bar{T}$ such that $T(\varepsilon)$ is a substochastic matrix. Then, for sufficiently small ε, the following Laurent series expansion holds*

$$[I - T(\varepsilon)]^{-1} = \frac{1}{\varepsilon} X_{-1} + X_0 + \varepsilon X_1 + \ldots, \tag{3.14}$$

with

$$X_{-1} = \frac{1}{\bar{\pi} D \mathbf{1}} \mathbf{1}\bar{\pi}, \tag{3.15}$$

$$X_0 = (I - X_{-1}D)H(I - DX_{-1}), \tag{3.16}$$

where $\bar{\pi}$ is the stationary distribution of $\bar{T}$ and $H = (I - \bar{T} + \mathbf{1}\bar{\pi})^{-1} - \mathbf{1}\bar{\pi}$ is the deviation matrix.

Proof The proof of this result is based on the approach developed in [16, 17]. The existence of the Laurent series (3.14) is a particular case of more general results of [17]. To calculate the terms of the Laurent series, let us equate the terms with the same powers of ε in the following identity:

$$(I - \bar{T} + \varepsilon D)(\frac{1}{\varepsilon} X_{-1} + X_0 + \varepsilon X_1 + \ldots) = I,$$

which results in

$$(I - \bar{T})X_{-1} = 0, \tag{3.17}$$

$$(I - \bar{T})X_0 + DX_{-1} = I, \tag{3.18}$$

$$(I - \bar{T})X_1 + DX_0 = 0. \tag{3.19}$$

From equation (3.17), we conclude that

$$X_{-1} = \mathbf{1}\mu_{-1}, \tag{3.20}$$

where μ_{-1} is some vector. We find this vector from the condition that equation (3.18) has a solution. In particular, equation (3.18) has a solution if and only if

$$\bar{\pi}(I - DX_{-1}) = 0.$$

By substituting the expression (3.20) into the above equation, we obtain

$$\bar{\pi} - \bar{\pi}D\mathbf{1}\mu_{-1} = 0,$$

and, consequently,

$$\mu_{-1} = \frac{1}{\bar{\pi}D\mathbf{1}}\bar{\pi},$$

which, together with (3.20), gives (3.15).

Since the deviation matrix H is a Moore-Penrose generalized inverse of $I - \bar{T}$, the general solution of equation (3.18) with respect to X_0 is given by

$$X_0 = H(I - DX_{-1}) + \mathbf{1}\mu_0, \tag{3.21}$$

where μ_0 is some vector. The vector μ_0 can be found from the condition that equation (3.19) has a solution. In particular, equation (3.19) has a solution if and only if

$$\bar{\pi}DX_0 = 0.$$

By substituting the expression for the general solution (3.21) into the above equation, we obtain

$$\bar{\pi}DH(I - DX_{-1}) + \bar{\pi}D\mathbf{1}\mu_0 = 0.$$

Consequently, we have

$$\mu_0 = -\frac{1}{\bar{\pi}D\mathbf{1}}\bar{\pi}DH(I - DX_{-1})$$

and we obtain (3.16). □

The following result holds.

Proposition 3.4 *The vector $\hat{\pi}_T$ can be expanded as follows:*

$$\hat{\pi}_T = \bar{\pi}_T - \bar{\pi}_T \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1} + \mathbf{1}^T X_0 \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1}) + o(\varepsilon), \tag{3.22}$$

where n_T is the number of nodes in the ESCC and X_0 is given in Lemma 3.1.

Proof We substitute $T = \bar{T} - \varepsilon D$ into $[I - T]^{-1}$ and use Lemma 3.1, to get

$$[I - T]^{-1} = \frac{1}{\bar{\pi}_T \varepsilon D\mathbf{1}} \mathbf{1}\bar{\pi}_T + X_0 + O(\varepsilon).$$

Using the above expression, we can write

$$\hat{\pi}_T = \frac{\mathbf{1}^T[I-T]^{-1}}{\mathbf{1}^T[I-T]^{-1}\mathbf{1}} = \frac{\frac{1}{\bar{\pi}_T \varepsilon D\mathbf{1}} n_T \bar{\pi}_T + \mathbf{1}^T X_0 + O(\varepsilon)}{\frac{1}{\bar{\pi}_T \varepsilon D\mathbf{1}} n_T + \mathbf{1}^T X_0 \mathbf{1} + O(\varepsilon)} = \frac{\bar{\pi}_T + \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 + o(\varepsilon)}{1 + \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1} + o(\varepsilon)} =$$

$$= \left(\bar{\pi}_T + \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 + o(\varepsilon)\right)\left(1 - \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1} + o(\varepsilon)\right) =$$

$$= \bar{\pi}_T - \bar{\pi}_T \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1} + \mathbf{1}^T X_0 \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1}) + o(\varepsilon).$$

□

Remark 3.1 *Since $R\mathbf{1} + T\mathbf{1} = \mathbf{1}$ and $\bar{T}\mathbf{1} = \mathbf{1}$, in lieu of $\bar{\pi}_T \varepsilon D\mathbf{1}$, we can write $\bar{\pi}_T R\mathbf{1}$. The latter expression has a clear probabilistic interpretation. It is the probability of exiting the ESCC in one step starting from the distribution $\bar{\pi}_T$. Later we shall demonstrate that this probability is indeed small. We note that not only $\bar{\pi}_T R\mathbf{1}$ is small but also the factor $1/n_T$ is small, as the number of states in the ESCC is large. Thus, we expect that $\hat{\pi}_T$ is very close to $\bar{\pi}_T$.*

In the next Proposition 3.5 we provide an alternative expression for the first-order terms of $\hat{\pi}_T$.

Proposition 3.5 *The vector $\hat{\pi}_T$ can be expanded as follows:*

$$\hat{\pi}_T = \bar{\pi}_T - \varepsilon \bar{\pi}_T D H + \varepsilon \mathbf{1}^T \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1})H + o(\varepsilon).$$

Proof Let us consider $\hat{\pi}_T$ as a power series

$$\hat{\pi}_T = \hat{\pi}_T^{(0)} + \varepsilon \hat{\pi}_T^{(1)} + \varepsilon^2 \hat{\pi}_T^{(2)} + \dots.$$

From (3.22), we obtain

$$\begin{aligned} \hat{\pi}_T &= \bar{\pi}_T - \bar{\pi}_T \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1} + \mathbf{1}^T X_0 \frac{1}{n_T}(\bar{\pi}_T \varepsilon D\mathbf{1}) + o(\varepsilon) = \\ &= \bar{\pi}_T + \varepsilon\left(\mathbf{1}^T X_0 \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1}) - \bar{\pi}_T \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1}\right) + o(\varepsilon), \end{aligned}$$

and hence

$$\hat{\pi}_T^{(1)} = \mathbf{1}^T X_0 \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1}) - \bar{\pi}_T \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1}, \tag{3.23}$$

where X_0 is given by (3.16). Before substituting (3.16) into (3.23), we make the transformations

$$\begin{aligned} X_0 &= (I - X_{-1}D)H(I - DX_{-1}) = \\ &= H - HDX_{-1} - X_{-1}DH + X_{-1}DHDX_{-1}, \end{aligned}$$

where X_{-1} is defined by (3.15). Pre-multiplying X_0 by $\mathbf{1}^T$ and then post-multiplying by $\mathbf{1}$, we obtain

$$\begin{aligned} \mathbf{1}^T X_0 &= \mathbf{1}^T H - \bar{\pi}_T(\mathbf{1}^T HD\mathbf{1})(\bar{\pi}_T D\mathbf{1})^{-1} - n_T\bar{\pi}_T(\bar{\pi}_T D\mathbf{1})^{-1}DH + \\ &+ n_T\bar{\pi}_T DHD\mathbf{1}\bar{\pi}_T(\bar{\pi}_T D\mathbf{1})^{-2}. \end{aligned} \tag{3.24}$$

and

$$\mathbf{1}^T X_0 \mathbf{1} = n_T\bar{\pi}_T DHD\mathbf{1}(\bar{\pi}_T D\mathbf{1})^{-2} - \mathbf{1}^T HD\mathbf{1}(\bar{\pi}_T D\mathbf{1})^{-1}. \tag{3.25}$$

Substituting (3.24) and (3.25) into (3.23), we get

$$\begin{aligned} \hat{\pi}_T^{(1)} &= \mathbf{1}^T X_0 \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1}) - \bar{\pi}_T \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1})\mathbf{1}^T X_0 \mathbf{1} = \\ &= \mathbf{1}^T H \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1}) - \frac{1}{n_T}\bar{\pi}_T \mathbf{1}_T HD\mathbf{1} - \bar{\pi}_T DH + \\ &+ \bar{\pi}_T(\bar{\pi}_T DHD\mathbf{1})(\bar{\pi}_T D\mathbf{1})^{-1} - \bar{\pi}_T(\bar{\pi}_T DHD\mathbf{1})(\bar{\pi}_T D\mathbf{1})^{-1} + \frac{1}{n_T}\bar{\pi}_T \mathbf{1}_T HD\mathbf{1} = \\ &= \mathbf{1}^T H \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1}) - \bar{\pi}_T DH. \end{aligned}$$

Thus, we have

$$\hat{\pi}_T^{(1)} = \mathbf{1}^T H \frac{1}{n_T}(\bar{\pi}_T D\mathbf{1}) - \bar{\pi}_T DH.$$

□

As in the analysis of the pseudo-stationary distribution, we take the matrix T in the form of the perturbation $T = \bar{T} - \varepsilon D$.

Proposition 3.6 *The vector $\tilde{\pi}_T$ can be expanded as follows:*

$$\tilde{\pi}_T = \bar{\pi}_T - \varepsilon\bar{\pi}_T DH + o(\varepsilon).$$

Proof We look for the quasi-stationary distribution and the Perron-Frobenius eigenvalue in the form of a power series,

$$\tilde{\pi}_T = \tilde{\pi}_T^{(0)} + \varepsilon\tilde{\pi}_T^{(1)} + \varepsilon^2\tilde{\pi}_T^{(2)} + \dots, \tag{3.26}$$

$$\lambda_1 = 1 + \varepsilon\lambda_1^{(1)} + \varepsilon^2\lambda_1^{(2)} + \dots.$$

Substituting $T = \bar{T} - \varepsilon D$ and the above series into (3.2), and equating terms with the same powers of ε, we obtain

$$\tilde{\pi}_T^{(0)}\bar{T} = \tilde{\pi}_T^{(0)}, \tag{3.27}$$

$$\tilde{\pi}_T^{(1)}\bar{T} - \tilde{\pi}_T^{(0)}D = 1\tilde{\pi}_T^{(1)} + \lambda_1^{(1)}\tilde{\pi}_T^{(0)}, \tag{3.28}$$

Substituting (3.26) into the normalization condition (3.3), we get

$$\tilde{\pi}_T^{(0)}\mathbf{1} = 1, \tag{3.29}$$

$$\tilde{\pi}_T^{(1)}\mathbf{1} = 0. \tag{3.30}$$

From (3.27) and (3.29), we conclude that $\tilde{\pi}_T^{(0)} = \bar{\pi}_T$. Thus, equation (3.28) takes the form

$$\tilde{\pi}_T^{(1)}\bar{T} - \bar{\pi}_T D = 1\tilde{\pi}_T^{(1)} + \lambda_1^{(1)}\bar{\pi}_T.$$

Post-multiplying this equation by $\mathbf{1}$, we get

$$\tilde{\pi}_T^{(1)}\bar{T}\mathbf{1} - \bar{\pi}_T D\mathbf{1} = 1\tilde{\pi}_T^{(1)}\mathbf{1} + \lambda_1^{(1)}\bar{\pi}_T\mathbf{1}.$$

Now, using $\bar{T}\mathbf{1} = \mathbf{1}$, (3.29) and (3.30), we conclude that

$$\lambda_1^{(1)} = -\bar{\pi}_T D\mathbf{1},$$

and, consequently,

$$\lambda_1 = 1 - \varepsilon\bar{\pi}_T D\mathbf{1} + o(\varepsilon). \tag{3.31}$$

Now, equation (3.28) can be rewritten as follows:

$$\tilde{\pi}_T^{(1)}[I - \bar{T}] = \bar{\pi}_T[(\bar{\pi}_T D\mathbf{1})I - D].$$

Its general solution is given by

$$\tilde{\pi}_T^{(1)} = \nu\bar{\pi}_T + \bar{\pi}_T[(\bar{\pi}_T D\mathbf{1})I - D]H,$$

where ν is some constant. To find the constant ν, we substitute the above general solution into condition (3.30):

$$\tilde{\pi}_T^{(1)}\mathbf{1} = \nu\bar{\pi}_T\mathbf{1} + \bar{\pi}_T[(\bar{\pi}_T D\mathbf{1})I - D]H\mathbf{1} = 0.$$

Since $\bar{\pi}_T \mathbf{1} = 1$ and $H\mathbf{1} = 0$, we get $\nu = 0$. Consequently, we have

$$\tilde{\pi}_T^{(1)} = \bar{\pi}_T[(\bar{\pi}_T D\mathbf{1})I - D]H = (\bar{\pi}_T D\mathbf{1})\bar{\pi}_T H - \bar{\pi}_T DH = -\bar{\pi}_T DH.$$

In the above, we have used the fact that $\bar{\pi}_T H = 0$. This completes the proof. □

If λ_1 is very close to one (which is indeed the case in practice: see Section 3.5), we conclude from (3.31) and the equality $\varepsilon\bar{\pi}_T D\mathbf{1} = \bar{\pi}_T R\mathbf{1}$ that indeed $\bar{\pi}_T R\mathbf{1}$ is typically very small, as we have conjectured in Remark 3.1.

Furthermore, there is also a simple relation between λ_1 and $\tilde{\pi}_T$.

Proposition 3.7 *The Perron-Frobenius eigenvalue* λ_1 *of matrix* T *is given by*

$$\lambda_1 = 1 - \tilde{\pi}_T R\mathbf{1}. \tag{3.32}$$

Proof Post-multiplying equation (3.2) by $\mathbf{1}$, we obtain

$$\lambda_1 = \tilde{\pi}_T T\mathbf{1}.$$

Then, using the fact that $T\mathbf{1} = \mathbf{1} - R\mathbf{1}$, we derive the formula (3.32). □

Proposition 3.7 indicates that if λ_1 is close to one then $\tilde{\pi}_T R\mathbf{1}$ is small.

Let us analyze the Perron-Frobenius right eigenvector u of the matrix T:

$$Tu = \lambda_1 u, \tag{3.33}$$

where λ_1 is the Perron-Frobenius eigenvalue, as in the previous section.

The vector u can be normalized in different ways. Let us define the main normalization for u as

$$\mathbf{1}^T u = n_T.$$

Let us also define $\bar{u}$ as

$$\bar{u} = \frac{u}{\bar{\pi}_T u}, \text{ so that } \bar{\pi}_T \bar{u} = 1. \tag{3.34}$$

Proposition 3.8 *The vector* $\bar{u}$ *can be expanded as follows:*

$$\bar{u} = \mathbf{1} - \varepsilon HD\mathbf{1} + o(\varepsilon).$$

Proof We look for the right eigenvector and the Perron-Frobenius eigenvalue in the form of a power series,

$$\bar{u} = \bar{u}^{(0)} + \varepsilon\bar{u}^{(1)} + \varepsilon^2\bar{u}^{(2)} + \dots. \tag{3.35}$$

$$\lambda_1 = 1 + \varepsilon\lambda_1^{(1)} + \varepsilon^2\lambda_1^{(2)} + \dots.$$

Substituting $T = \bar{T} - \varepsilon D$ and the above series into (3.33), and equating terms with the same powers of ε, we obtain

$$\bar{T}\bar{u}^{(0)} = \bar{u}^{(0)}, \tag{3.36}$$

$$\bar{T}\bar{u}^{(1)} - D\bar{u}^{(0)} = \bar{u}^{(1)} + \lambda_1^{(1)}\bar{u}^{(0)}. \tag{3.37}$$

Substituting (3.35) into the normalization condition (3.34), we obtain

$$\bar{\pi}_T\bar{u}^{(0)} = 1, \tag{3.38}$$

$$\bar{\pi}_T\bar{u}^{(1)} = 0. \tag{3.39}$$

From (3.36) and (3.38), we conclude that $\bar{u}^{(0)} = \mathbf{1}$. Thus, equation (3.37) takes the form

$$\bar{T}\bar{u}^{(1)} - D\mathbf{1} = \bar{u}^{(1)} + \lambda_1^{(1)}\mathbf{1}.$$

Pre-multiplying this equation by $\bar{\pi}_T$, we get

$$\bar{\pi}_T\bar{u}^{(1)} - \bar{\pi}_T D\mathbf{1} = \bar{\pi}_T\bar{u}^{(1)} + \bar{\pi}_T\lambda_1^{(1)}\mathbf{1}.$$

Now using $\bar{T}\mathbf{1} = \mathbf{1}$, (3.38) and (3.39), we conclude that

$$\lambda_1^{(1)} = -\bar{\pi}_T D\mathbf{1},$$

and, consequently,

$$\lambda_1 = 1 - \varepsilon\bar{\pi}_T D\mathbf{1} + o(\varepsilon).$$

Now, equation (3.37) can be rewritten as follows:

$$\left[I - \bar{T}\right]\bar{u}^{(1)} = \left[(\bar{\pi}_T D\mathbf{1})\, I - D\right]\mathbf{1}.$$

Its general solution is given by

$$\bar{u}^{(1)} = \nu\mathbf{1} + H\left[(\bar{\pi}_T D\mathbf{1})\, I - D\right]\mathbf{1},$$

where ν is some constant. To find the constant ν, we substitute the above general solution into condition (3.39)

$$\bar{\pi}_T\bar{u}^{(1)} = \nu\bar{\pi}_T\mathbf{1} + \bar{\pi}_T H\left[(\bar{\pi}_T D\mathbf{1})\, I - D\right]\mathbf{1}.$$

Since $\bar{\pi}_T\mathbf{1} = 1$ and $\bar{\pi}_T H = 0$, we get $\nu = 0$. Consequently, we have

$$\bar{u}^{(1)} = -HD\mathbf{1}.$$

In the above, we have used the fact that $H\mathbf{1} = 0$. This completes the proof. □

Since the substochastic matrix T is close to being stochastic, the vector $\mathfrak{u}$ will be very close to $\mathbf{1}$ (see Proposition 3.8). Consequently, the vector $\breve{\pi}_{\mathsf{T}}$ will be close to $\tilde{\pi}_{\mathsf{T}}$ and to $\bar{\pi}$ as well. This shows that in the case when the matrix T is close to the stochastic matrix all the alternative definitions of quasi-stationary distribution are quite close to each other. Moreover, from Proposition 3.1, we conclude that the PageRank ranking converges to the quasi-stationarity based ranking as the damping factor goes to one.

3.5 Numerical experiments and Applications

For our numerical experiments we have used the Web site of INRIA (http://www.inria.fr: the dataset is available from the author upon request). It is a typical Web site with about 300.000 Web pages and 2.200.000 hyper-links. Accordingly, datasets of similar or even smaller sizes have been extensively used in experimental studies of novel algorithms for PageRank computation [1, 58, 59]. To collect the Web Graph data, we construct our own Web crawler which works with an Oracle database. The crawler consists of two parts: the first part is realized in Java and is responsible for downloading pages from the Internet, parsing the pages, and inserting their hyper-links into the database; the second part is written in PL/SQL and is responsible for the data management. For a detailed description of the crawler reader is referred to [15].

As was shown in [28, 56], the Web Graph has three major distinct components: IN, OUT and SCC. However, if one takes into account the artificial links from the dangling nodes, the Web Graph has two major distinct components: ESCC and POUT [14]. We provide the statistics for the INRIA Web site in Table 3.1. In our experiments we consider the artificial links from

	INRIA
Total size	318585
Number of nodes in SCC	154142
Number of nodes in IN	0
Number of nodes in OUT	164443
Number of nodes in ESCC	300455
Number of nodes in POUT	18130

Table 3.1: Component sizes in the INRIA dataset

the dangling nodes and compute $\bar{\pi}_{\mathsf{T}}$, $\tilde{\pi}_{\mathsf{T}}$, $\hat{\pi}_{\mathsf{T}}$, and $\breve{\pi}_{\mathsf{T}}$ with 5-digit precision. Also we compute $\hat{\pi}_{\mathsf{T}}(0.85)$, which is the normalized PageRank vector of the ESCC with damping factor equal to 0.85. For each pair of these vectors we calculated Kendall's τ metric (see Table 3.2). Kendall's τ metric shows how two rankings are different in terms of the number of swaps which are

needed to transform one ranking to the other. Kendall's τ metric has the value of one if two rankings are identical and minus one if one ranking is the inverse of the other. In our case,

	$\hat{\pi}_T(0.85)$	$\bar{\pi}_T$	$\tilde{\pi}_T$	$\hat{\pi}_T$	$\check{\pi}_T$
$\hat{\pi}_T(0.85)$	1.0	0.87272	0.87275	0.87449	0.86462
$\bar{\pi}_T$		1.0	0.99390	0.99498	0.98228
$\tilde{\pi}_T$			1.0	0.99770	0.98786
$\hat{\pi}_T$				1.0	0.98597
$\check{\pi}_T$					1.0

Table 3.2: Kendall's τ comparison

Kendall's τ metric for all the pairs is very close to one. Thus, we can conclude that all the four quasi-stationarity based centrality measures produce very similar rankings.

We have also analyzed Kendall's τ metric between $\tilde{\pi}_T$ and PageRank of the ESCC as a function of the damping factor (see Figure 3.1). As c goes to one, Kendall's τ approaches one. This is in agreement with Proposition 3.1.

We have also compared the ranking produced by the quasi-stationary distributions and PageRank of the ESCC using the θ rank correlation measure. The measure is defined as follows:

$$\theta_i = \arctan(r_i^1/r_i^2),$$

where r_i^1 is the ranking of node i in a vector and r_i^2 is the ranking of the same node i in another vector. By the term ranking, we mean here the place of node i in a vector if we sort the entries of the vector in decreasing order. If the ranking of node i is the same in both vectors, θ_i is equal to $\pi/4$. Since there is a ratio of ranking in the expression, the θ rank correlation measure is more sensitive to changing of the ranking of highly ranked nodes. The interested reader is referred to [80] for further details on θ rank correlation measure. We calculated the cumulative distribution function over θ_i to see what fraction of nodes changed their ranking. As one can see from Figure 3.2 and 3.3, the cumulative distribution over θ_i corresponding to the PageRank vector of the ESCC and any quasi-stationary distribution is close to a vertical line at $\pi/4$, which means that rankings produced by the vectors are close to each other. The similarity in ranking between any two quasi-stationary distributions is even more pronounced: see Figure 3.4.

3.6 Conclusion

In this chapter we have proposed centrality measures which can be applied to the Giant Strongly Connected Component of a reducible graph to avoid the absorbtion problem. In Google PageRank the problem was solved by the introduction of uniform random jumps with

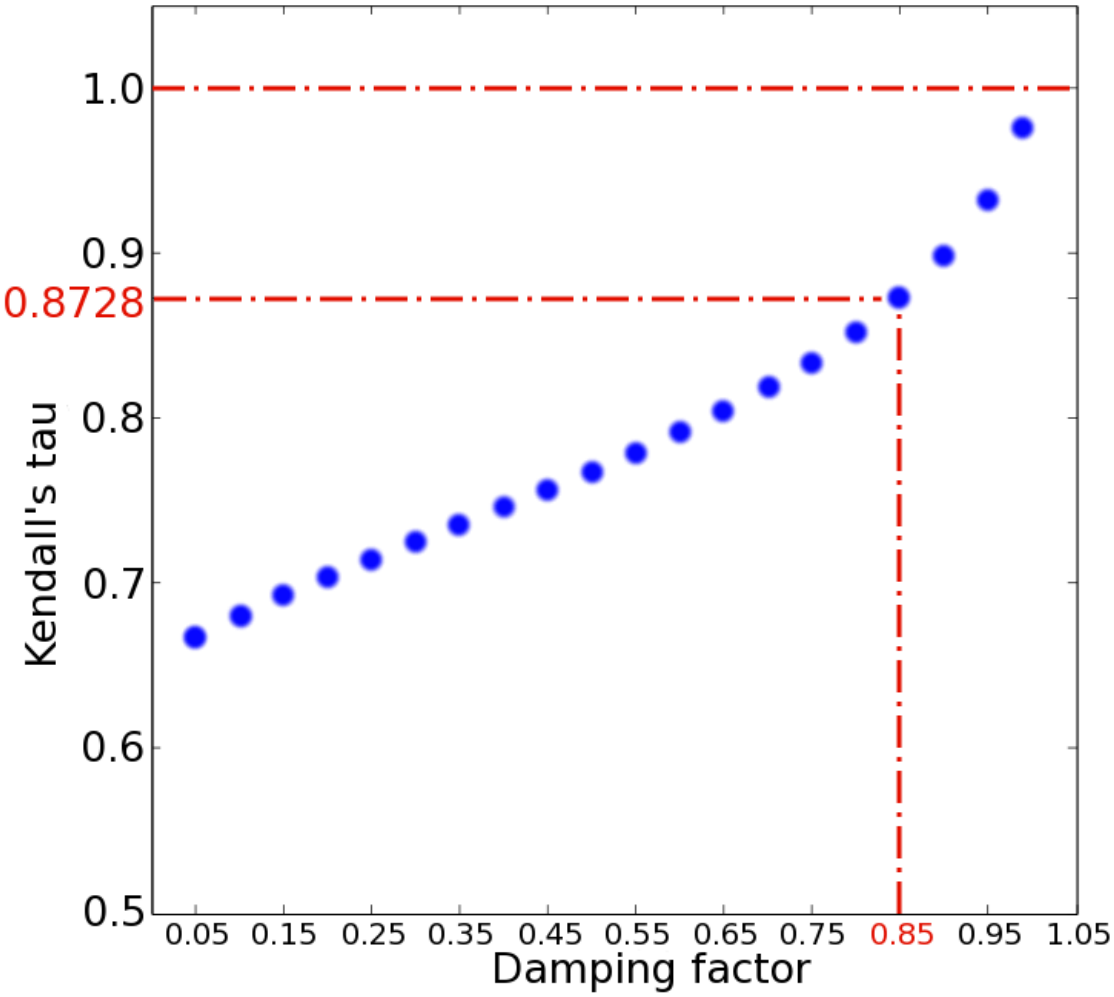
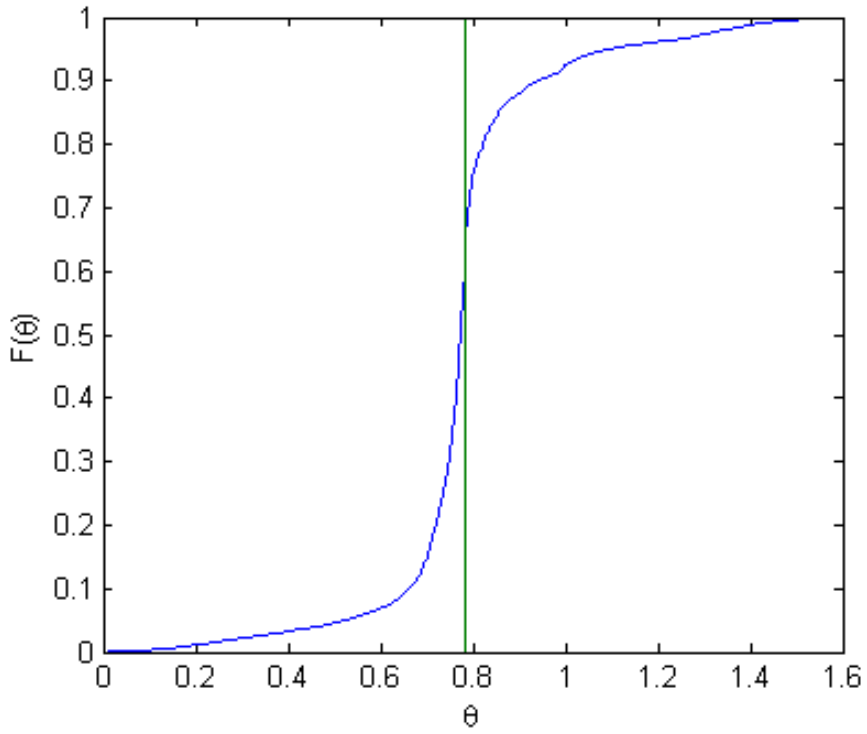
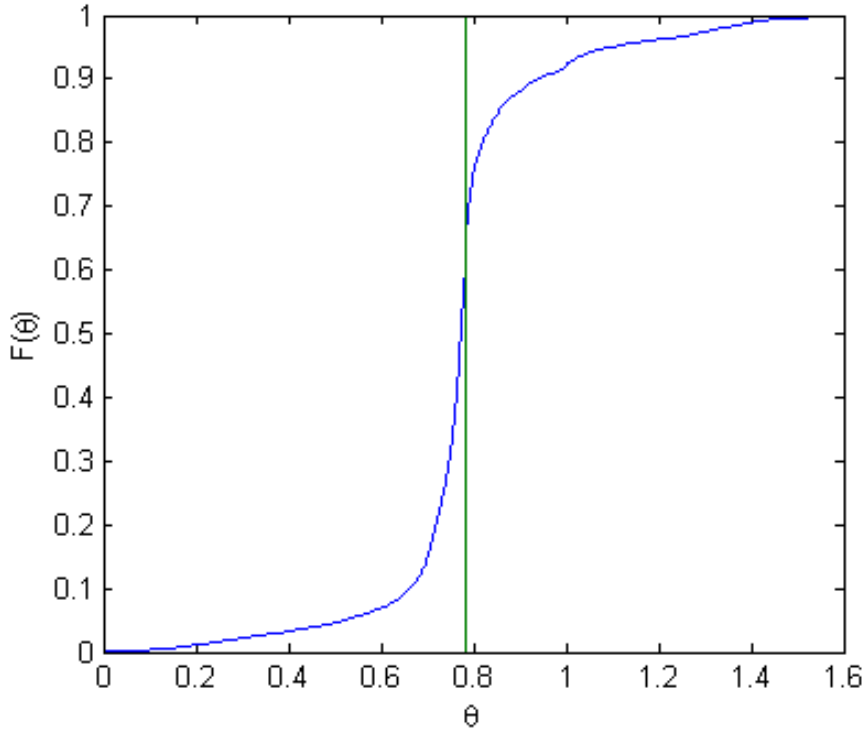


Figure 3.1: Kendall's τ metric between $\tilde{\pi}_T$ and PageRank of the ESCC $\hat{\pi}_T(c)$ as a function of the damping factor.

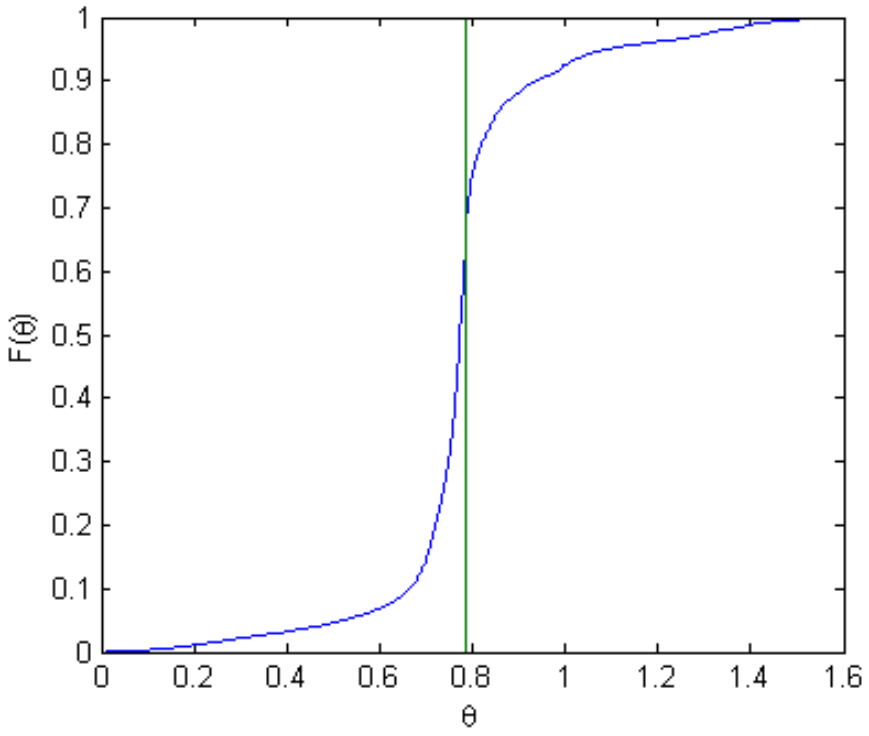


(a)

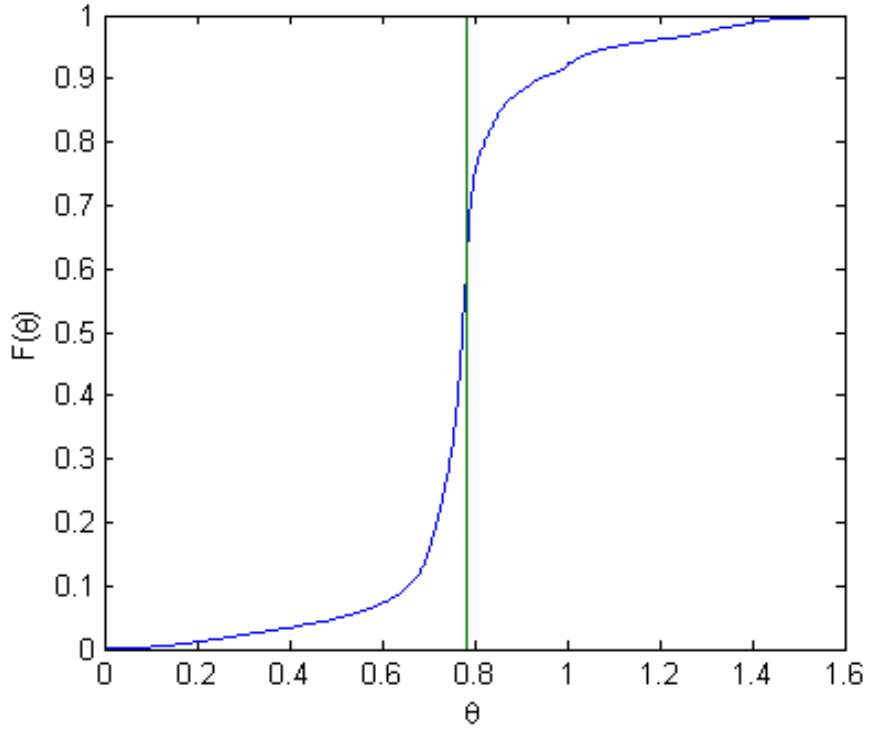


(b)

Figure 3.2: Cumulative distribution of the θ rank correlation measure: (a) $\hat{\pi}_T(0.85)$ and $\bar{\pi}_T$, (b) $\hat{\pi}_T(0.85)$ and $\tilde{\pi}_T$.

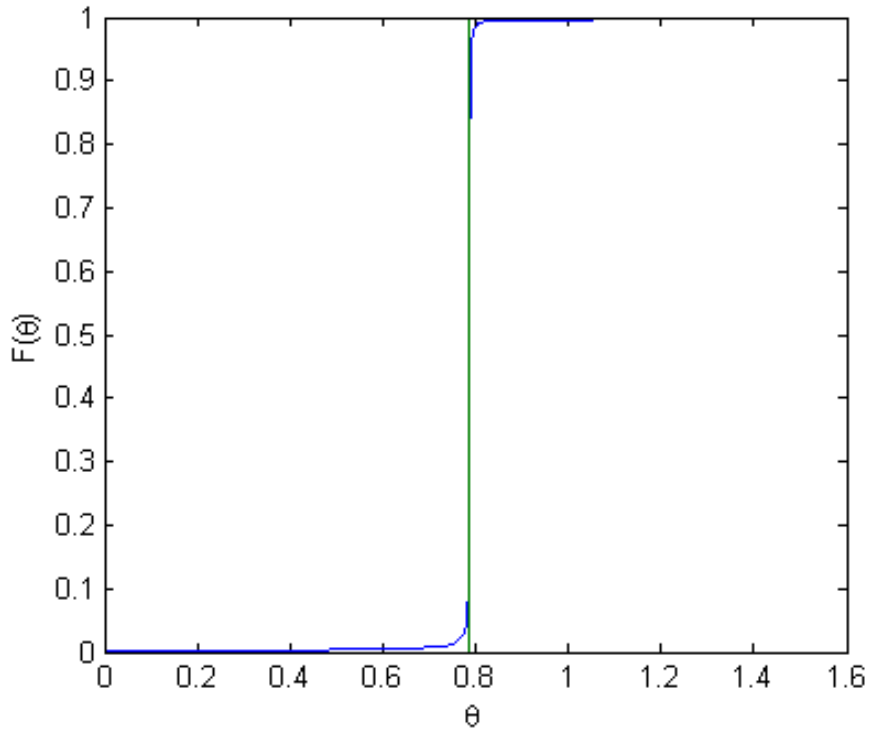


(c)

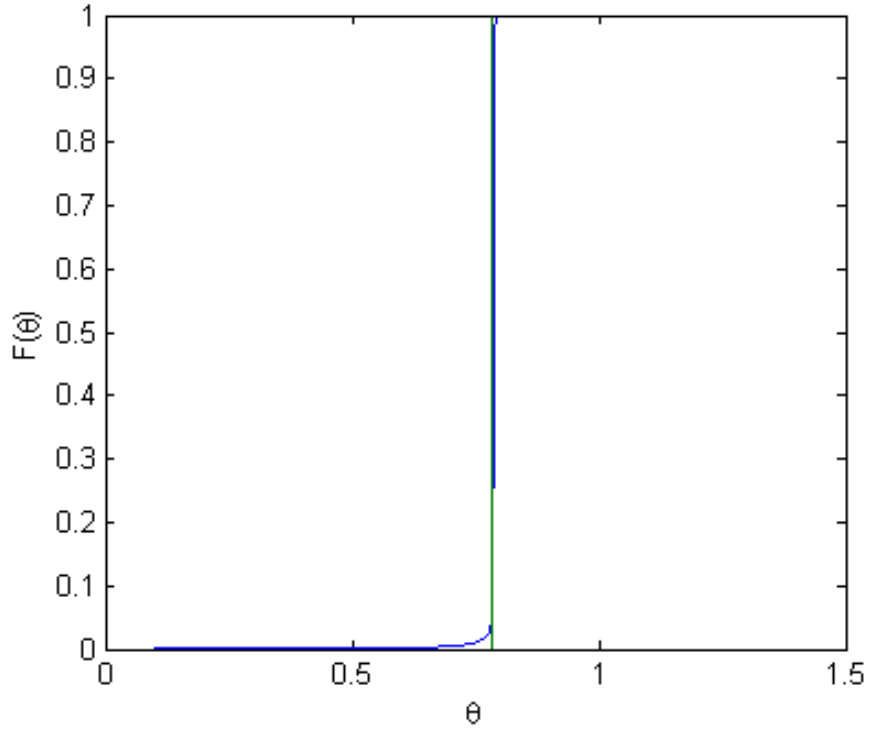


(d)

Figure 3.3: Cumulative distribution of the θ rank correlation measure: (c) $\hat{\pi}_T(0.85)$ and $\hat{\pi}_T$, (d) $\hat{\pi}_T(0.85)$ and $\check{\pi}_T$.

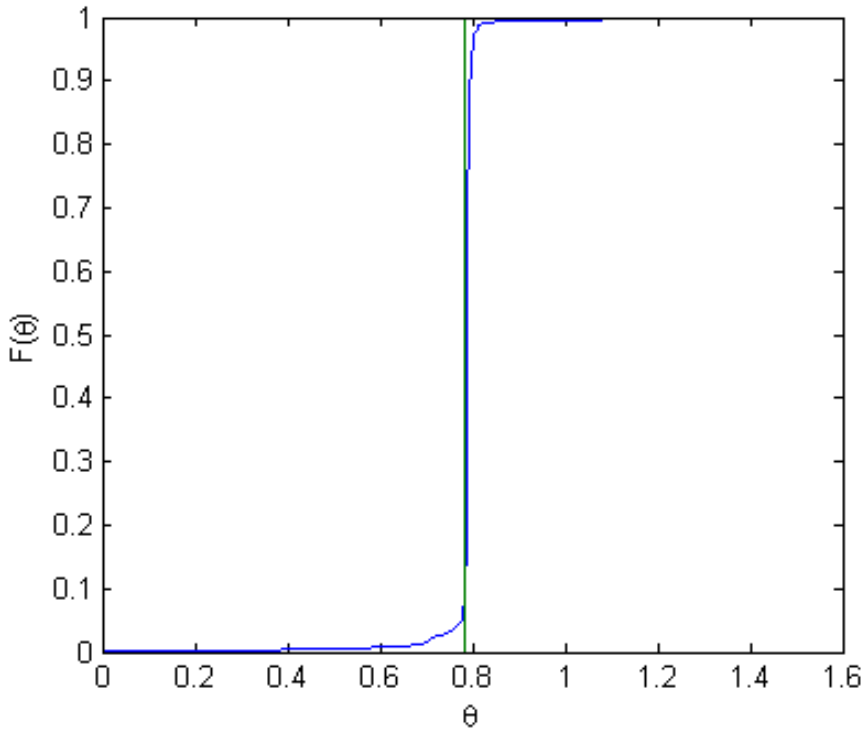


(a)

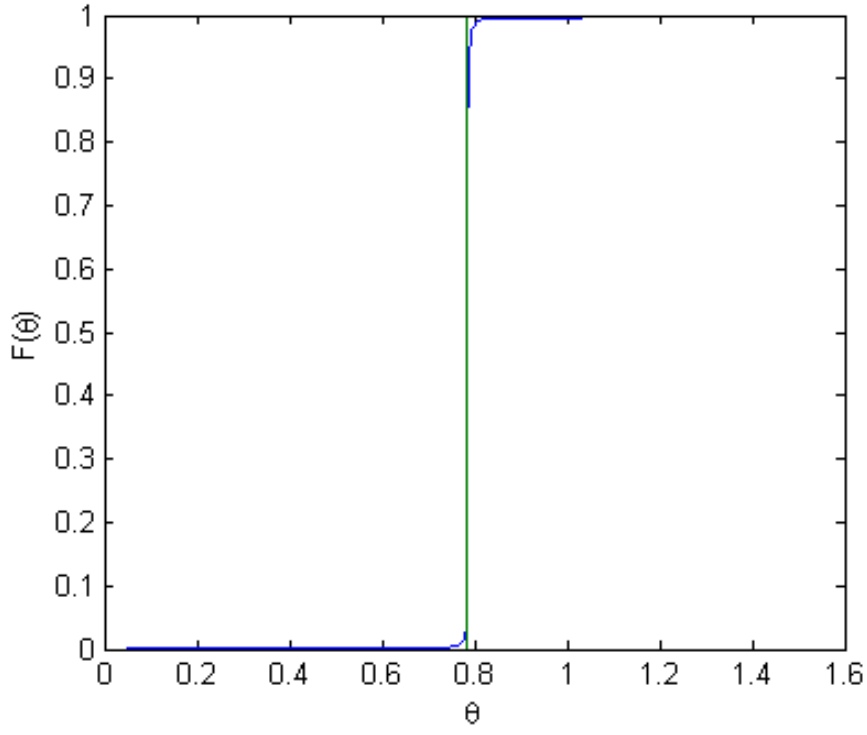


(b)

Figure 3.4: Cumulative distribution of the θ rank correlation measure: (a) $\bar{\pi}_T$ and $\tilde{\pi}_T$, (b) $\bar{\pi}_T$ and $\hat{\pi}_T$.

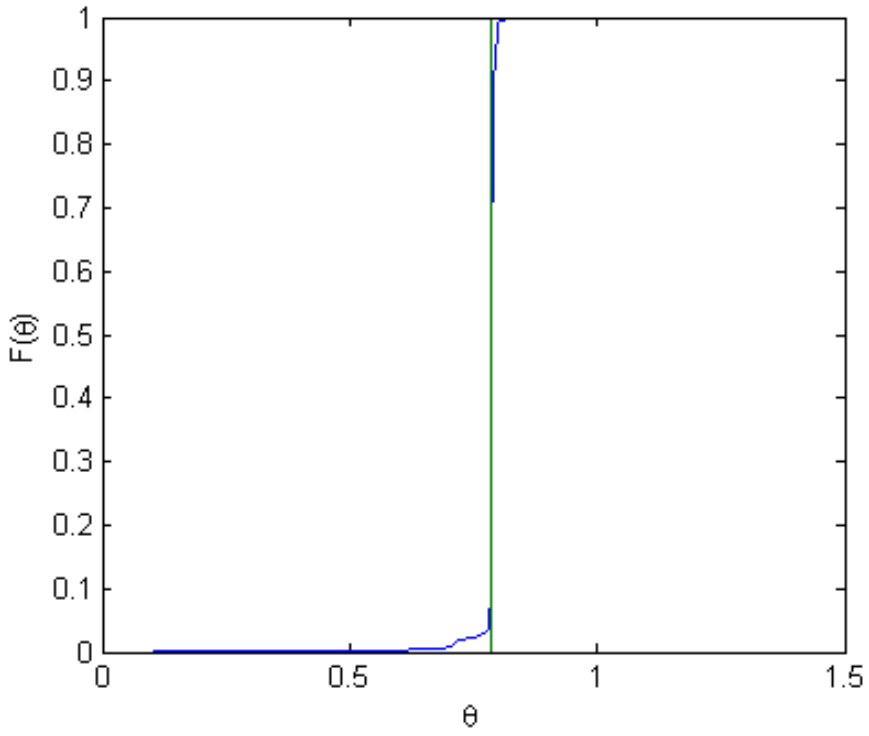


(c)

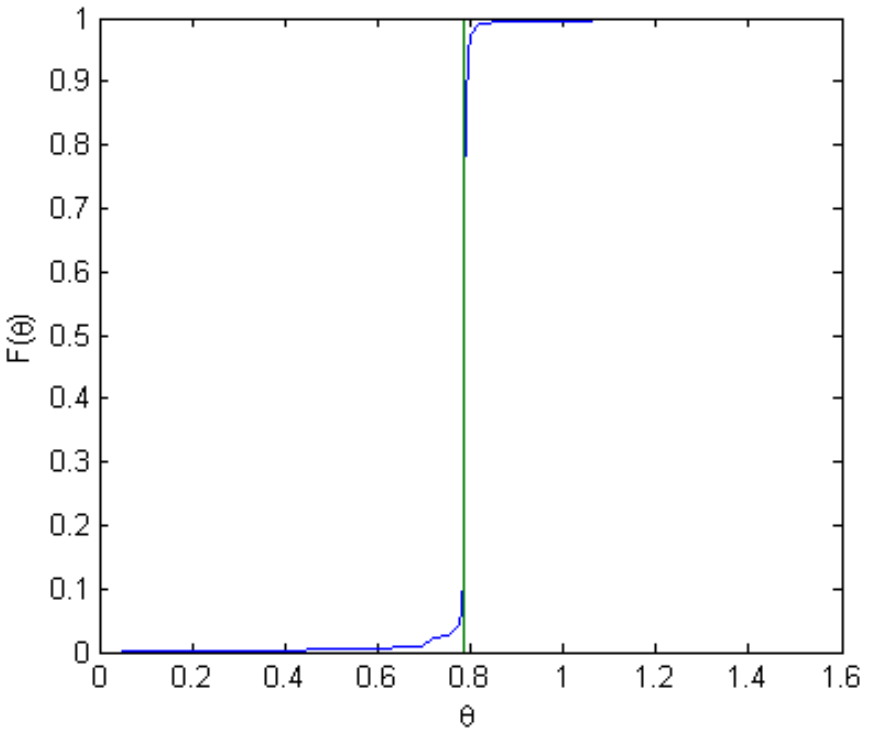


(d)

Figure 3.5: Cumulative distribution of the θ rank correlation measure: (c) $\bar{\pi}_T$ and $\check{\pi}_T$, (d) $\tilde{\pi}_T$ and $\hat{\pi}_T$.



(e)



(f)

Figure 3.6: Cumulative distribution of the θ rank correlation measure: (e) $\tilde{\pi}_T$ and $\check{\pi}_T$, (f) $\hat{\pi}_T$ and $\check{\pi}_T$.

some probability. Up to the present, there is no clear criterion for the choice of this parameter. In this chapter we have suggested four quasi-stationarity based parameter-free centrality measures, analyzed them and concluded that they produce approximately the same ranking. Therefore, in practice it is sufficient to compute only one quasi-stationarity based centrality measure. All our theoretical results are confirmed by numerical experiments.

4

MONTE CARLO METHODS IN PAGERANK COMPUTATION: WHEN ONE ITERATION IS SUFFICIENT

4.1 Summary

Although iterative methods of the PageRank calculation are highly developed, asides from them, there are other probabilistic methods aiming for this purpose. In this chapter we propose and analyze Monte Carlo type methods for the PageRank computation. There are several advantages of the probabilistic Monte Carlo methods over the deterministic Power method: Monte Carlo methods provide good estimation of the PageRank for relatively important pages already after one iteration; Monte Carlo methods have natural parallel implementation; and, finally, Monte Carlo methods allow to perform continuous update of the PageRank as the structure of the Web changes.

4.2 Introduction

Although iterative methods of the PageRank calculation are highly developed, asides from them, there are other probabilistic methods aiming for this purpose. Here we study Monte Carlo (MC) type methods for the PageRank computation. To the best of our knowledge, only in two works [25, 37] the Monte Carlo methods are applied to the PageRank computation. The principle advantages of the probabilistic Monte Carlo type methods over the deterministic methods are: the PageRank of important pages is determined with high accuracy already after the first iteration; MC methods have natural parallel implementation; and MC methods allow continuous update of the PageRank as the structure of the Web changes.

The structure and the contributions of the chapter are as follows. In Section 4.3, we describe different Monte Carlo algorithms. In particular, we propose an algorithm that takes into account not only the information about the last visited page (as in [25, 37]), but about all visited pages during the simulation run. In Section 4.4, we analyze and compare the convergence of Monte Carlo algorithms in terms of confidence intervals. We show that the PageRank of relatively important pages can be determined with high accuracy even after the first iteration. In Section 4.5, we show that experiments with real data from the Web confirm our theoretical analysis. Finally, we summarize the results of the present work in Section 4.6.

4.3 Monte Carlo algorithms

Monte Carlo algorithms are motivated by the following convenient formula that follows directly from the definition of the PageRank (1.6):

$$\pi = \frac{1-c}{n}\mathbf{1}^T[I - cP]^{-1} = \frac{1-c}{n}\mathbf{1}^T\sum_{k=0}^{\infty} c^k P^k. \tag{4.1}$$

This formula suggests a simple way of sampling from the PageRank distribution [24, 37, 46]. Consider a random walk $\{X_t\}_{t\geqslant 0}$ that starts from a randomly chosen page. Assume that at each step, the random walk terminates with probability $(1-c)$, and makes a transition according to the matrix P with probability c. It follows from (4.1) that the end-point of such random walk has a distribution π. Hence, one can suggest the following algorithm employed in [25].

Algorithm 4.1 MC end-point with random start. *Simulate* N *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at a randomly chosen page. Evaluate* π_j *as a fraction of* N *random walks which end at page* $j = \overline{1,n}$.

Let $\hat{\pi}_{j,N}$ be the estimator of π_j obtained by Algorithm 4.1. It is straightforward that

$$\mathbb{E}(\hat{\pi}_{j,N}) = \pi_j, \qquad \mathbb{V}ar(\hat{\pi}_{j,N}) = N^{-1}\pi_j(1-\pi_j).$$

A rough estimate $Var(\hat{\pi}_{j,N}) < 1/(4N)$ given in [25] results in a conclusion that the number of samples (random walks) needed to achieve a good relative accuracy with high probability, is of the order $O(n^2)$. In the ensuing Sections 3 and 4 we will show that this complexity evaluation is quite pessimistic. The number of required samples turns out to be linear in n. Moreover, a reasonable evaluation of the PageRank for popular pages can be obtain even with $N = n$, that is, one needs only as little as one run per page!

In order to improve the estimator $\hat{\pi}$, one can think of various ways of variance reduction. For instance, denoting $Z = [I - cP]^{-1}$ and writing π_j in (4.1) as

$$\pi_j = \frac{1-c}{n} \sum_{i=1}^{n} z_{ij}, \quad j = \overline{1,n},$$

we can view π_j as a given number $(1/n)$ multiplied by a sum of conditional probabilities $p_{ij} = (1-c)z_{ij}$ that the random walk ends at j given that it started at i. Since n is known, an unnecessary randomness in experiments can be avoided by taking $N = mn$ and initiating the random walk exactly m times from each page in a cyclic fashion, rather than jumping N times to a random page. This results in the following algorithm whose version was used in [37] for computing personalized PageRank.

Algorithm 4.2 (MC end-point with cyclic start) *Simulate* $N = mn$ *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at each page exactly* m *times. Evaluate* π_j *as a fraction of* N *random walks which end at page* $j = \overline{1,n}$.

Let $\hat{p}_{ij}$ be a fraction of m random walks initiated at i, that ended at j. Then the estimator for π_j suggested by Algorithm 4.2 can be expressed as

$$\hat{\hat{\pi}}_j = \frac{1}{n} \sum_{i=1} \hat{p}_{ij}.$$

For this estimator, we have

$$\mathbb{E}(\hat{\hat{\pi}}_j) = \pi_j,$$
$$Var(\hat{\hat{\pi}}_j) = (N)^{-1}[\pi_j - n^{-1} \sum_{i=1}^{n} p_{ij}^2] < Var(\hat{\pi}_j).$$

Besides the variance reduction, the estimator $\hat{\hat{\pi}}_i$ has important advantages in implementation because picking a page at random from a huge database is not a trivial problem [42]. This difficulty is completely avoided if the pages are visited in a cyclic fashion[1]. As the only advantage

[1]When referring to MC algorithms with cyclic start, we shall use the words "cycle" and "iteration" interchangeably.

of the Monte Carlo with random start, note that it does not require the number of samples N to be a multiple of n.

Another and probably more promising way of reducing the variance is to look at formula (4.1) from yet another angle. Note that for all $i, j = \overline{1,n}$, the element z_{ij} of the matrix

$$Z = [I - cP]^{-1} = \sum_{k=0}^{\infty} c^k P^k \tag{4.2}$$

can be regarded as the average number of times that the random walk $\{X_t\}_{t\geqslant 0}$ visits a page j given that this random walk started at page i. Thus, we can propose an estimator based on a complete path of the random walk $\{X_t\}_{t\geqslant 0}$ instead of taking into account only its end-point. The complete path version of the Monte Carlo method can be described as follows.

Algorithm 4.3 (MC complete path) *Simulate the random walk* $\{X_t\}_{t\geqslant 0}$ *exactly* m *times from each page. For any page* i*, evaluate* π_j *as the total number of visits to page* j *multiplied by* $(1-c)/(n*m)$.

MC complete path can be further improved by getting rid of artifacts in the matrix P related to pages without outgoing links, dangling pages. When a random walk reaches a dangling node, it jumps with the uniform probability to an arbitrary page. Clearly, it is more efficient just to terminate the random walk once it reaches a dangling node. Thus, we aim to rewrite (4.1) in terms of the original hyperlink matrix H defined in (1.2). Denote by $\mathcal{I}_0$ a set of dangling pages and by $\mathcal{I}_1 = \{1,\dots,n\}\backslash\mathcal{I}_0$ a set of pages which have at least one outgoing link. For all $j = \overline{1,n}$, it follows from (1.4) and (1.5) that

$$\pi_j = c\sum_{i=1}^{n} P_{ij}\pi_i + \frac{(1-c)}{n}\sum_{i=1}^{n}\pi_i = c\sum_{i=1}^{n} H_{ij}\pi_i + \gamma, \tag{4.3}$$

where γ is the same for each j:

$$\gamma = \frac{c}{n}\sum_{i\in\mathcal{I}_0}\pi_i + \frac{(1-c)}{n} < \frac{1}{n}. \tag{4.4}$$

Now, we rewrite equation (4.3) in the matrix form

$$\pi = \pi cH + \gamma \mathbf{1}^T,$$

which leads to the new expression for π:

$$\pi = \gamma \mathbf{1}^T[I - cH]^{-1}. \tag{4.5}$$

Note that the above equation is in accordance with the original definition of PageRank presented by Brin and Page [71]. The definition via the matrix P appeared later in order to develop the Markov chain formulation of the PageRank problem. The one-to-one correspondence

between (4.1) and (4.5) was noticed and proved in [21] but we find the proof presented above more insightful in our context.

Consider now a random walk $\{Y_t\}_{t\geqslant 0}$ which follows hyperlinks exactly as $\{X_t\}_{t\geqslant 0}$ except the transitions are governed by the matrix H instead of the matrix P. Thus, the random walk $\{Y_t\}_{t\geqslant 0}$ can be terminated at each step either with probability $(1-c)$ or when it reaches a dangling node. For all $i, j = \overline{1,n}$, the element w_{ij} of the matrix $W = [I - cH]^{-1}$, is the average number of visits of $\{Y_t\}_{t\geqslant 0}$ to page j given that the random walk started at page i. Denote

$$w_{\cdot j} = \sum_{i=1}^{n} w_{ij}.$$

Since the coordinates of π in (4.5) sum up to one, we have

$$\gamma = \left[\sum_{i,j=1}^{n} w_{ij}\right]^{-1} = \left[\sum_{j=1}^{n} w_{\cdot j}\right]^{-1} \tag{4.6}$$

and

$$\pi_j = w_{\cdot j}\left[\sum_{j=1}^{n} w_{\cdot j}\right]^{-1}. \tag{4.7}$$

This calls for another version of the complete path method.

Algorithm 4.4 (MC complete path stopping at dangling nodes) *Simulate the random walk $\{Y_t\}_{t\geqslant 0}$ starting exactly m times from each page. For any page j, evaluate π_j as the total number of visits to page j divided by the total number of visited pages.*

Let W_{ij} be a random variable distributed as a number of visits to page $j = \overline{1,n}$ by the random walk $\{Y_t\}_{t\geqslant 0}$ given that the random walk initiated at state $i = \overline{1,n}$. Formally,

$$\mathbb{P}(W_{ij} = x) = \mathbb{P}\left(\left[\sum_{t=0}^{\infty} \mathbf{1}_{\{Y_t = j\}}\right] = x | Y_0 = i\right), \quad x = 0, 1, \ldots,$$

where $\mathbf{1}_{\{\cdot\}}$ is the indicator function. Let $W_{ij}^{(l)}$, $l \geqslant 1$, be independent random variables distributed as W_{ij}. Then the estimator produced by Algorithm 4.4 can be written as

$$\bar{\pi}_j = \left[\sum_{l=1}^{m}\sum_{i=1}^{n} W_{ij}^{(l)}\right]\left[\sum_{l=1}^{m}\sum_{i,j=1}^{n} W_{ij}^{(l)}\right]^{-1}. \tag{4.8}$$

In the next section we present the analysis of this estimator.

We note that the complete path versions of the Monte Carlo methods also admit a random start. The corresponding algorithm is as follows.

Algorithm 4.5 (MC complete path with random start) *Simulate* N *samples of the random walk* $\{Y_t\}_{t\geqslant 0}$ *started at a random page. For any page* j*, evaluate* π_j *as the total number of visits to page* i *divided by the total number of visited pages.*

One can show however that Algorithm 4.4 provides an estimator with a smaller variance than Algorithm 4.5. Indeed, let W_{Uj} be the number of visits to page j from a randomly chosen page $U \in \{1,\dots,n\}$. Then, we have

$$\mathrm{Var}(W_{Uj}) = \frac{1}{n}\sum_{i=1}^{n}\mathrm{Var}(W_{ij}) + \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}^2(W_{ij}) - \left[\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}(W_{ij})\right]^2 > \frac{1}{n}\sum_{i=1}^{n}\mathrm{Var}(W_{ij}).$$

Now note that in n simulation runs, Algorithm 4.4 generates one sample of the sum $\sum_{i=1}^{n} W_{ij}$, whereas Algorithm 4.5 generates n samples of W_{Uj}. Hence, Algorithm 4.4 provides random variables with smaller variance in both numerator and denominator of (4.8).

4.4 Convergence Analysis

From the preliminary analysis of the previous section, we can already conclude that MC algorithms with cyclic start are preferable to the analogous MC algorithms with random start. In the present section we thoroughly analyze and compare MC complete path stopping at dangling nodes with MC end-point. We show that under natural conditions MC complete path stopping at dangling nodes outperforms MC end-point.

We start by studying the properties of W_{ij}'s. Denote by q_{ij} the probability that starting from page i, the random walk $\{Y_t\}_{t\geqslant 0}$ reaches page j:

$$q_{ij} = \mathbb{P}\left(\bigcup_{t\geqslant 1}\{Y_t = j\} | Y_0 = i\right), \quad i,j = \overline{1,n}.$$

Note that in this definition, $q_{jj} < 1$ is a probability to return to state j if the process started at j. It follows from the strong Markov property that W_{jj} has a geometric distribution with parameter $1 - q_{jj} \geqslant 1 - c$:

$$\mathbb{P}(W_{jj} = x) = q_{jj}^{x-1}(1-q_{jj}),\ x = 1,2,\dots,$$

which implies

$$\mathbb{E}(W_{jj}) = \frac{1}{1-q_{jj}}; \quad \mathrm{Var}(W_{jj}) = \frac{q_{jj}}{(1-q_{jj})^2};$$

Further, applying again the strong Markov property, one can show that for all $i, j = \overline{1,n}$, W_{ij} has a shifted geometric distribution:

$$\mathbb{P}(W_{ij} = x) = \begin{cases} 1 - q_{ij}, & x = 0, \\ q_{ij}\mathbb{P}(W_{jj} = x), & x = 1, 2, \dots. \end{cases}$$

Consequently,

$$\mathbb{E}(W_{ij}) = w_{ij} = q_{ij}\mathbb{E}(W_{jj}) = \frac{q_{ij}}{1 - q_{jj}} \tag{4.9}$$

and

$$\mathrm{Var}(W_{ij}) = \frac{1 + q_{jj}}{1 - q_{jj}} w_{ij} - w_{ij}^2. \tag{4.10}$$

Now, define

$$W_{\cdot j} = \sum_{i=1}^{n} W_{ij}, \quad j = \overline{1,n}, \quad W = \sum_{j=1}^{n} W_{\cdot j}.$$

Assuming that all W_{ij}'s are independent, we immediately obtain

$$\begin{aligned} \mathbb{E}(W_{\cdot j}) &= \sum_{i=1}^{n} w_{ij} = w_{\cdot i}, \\ \mathrm{Var}(W_{\cdot j}) &= \frac{1 + q_{jj}}{1 - q_{jj}} w_{\cdot j} - \sum_{i=1}^{n} w_{ij}^2 < \frac{1 + q_{jj}}{1 - q_{jj}} w_{\cdot j}, \\ \mathbb{E}(W) &= \sum_{j=1}^{n} w_{\cdot j} = \gamma^{-1}. \end{aligned}$$

For $i, j = \overline{1,n}$, let the empirical mean

$$\bar{W}_{ij} = \frac{1}{m} \sum_{l=1}^{m} W_{ij}^{(l)}$$

be the estimator of w_{ij}, and view

$$\bar{W}_{\cdot j} = \sum_{i=1} \bar{W}_{ij}, \quad j = \overline{1,n},$$

and

$$\bar{W} = \sum_{j=1} \bar{W}_{\cdot j}$$

as estimators of $w_{\cdot j}$ and γ^{-1}, respectively. The estimator (4.8) can be then written as

$$\bar{\pi}_j = \bar{W}_{\cdot j}\bar{W}^{-1}. \tag{4.11}$$

Since the second multiplier in (4.11) is the same for all $j = \overline{1,n}$, the estimator $\bar{\pi}_j$ is completely determined by $\bar{W}_{\cdot j}$. The following theorem states that the relative errors of $\bar{\pi}$ and $\bar{W}_{\cdot j}$ are similar. Let us to prove a auxiliary lemma before.

Lemma 4.1 *Let* $W_{i\cdot} = \sum_{j=1}^{n} W_{ij}$ *be the length of the random walk* $\{Y_t\}_{t\geqslant 0}$ *initiated at page* $i = 1, \ldots, n$. *Then for all dangling nodes* $i \in \mathcal{I}_0$, *it holds* $W_{i\cdot} \equiv 1$, *and for non-dangling nodes* $i \in \mathcal{I}_1$,

$$\mathbb{E}(W_{i\cdot}) \leqslant \frac{1}{1-c}, \qquad Var(W_{i\cdot}) \leqslant \frac{c(1+c^3)}{(1-c)^2}. \tag{4.12}$$

Proof The statement for dangling nodes is obvious. For non-dangling nodes, (4.12) essentially follows from the distributional identity

$$W_{i\cdot} \stackrel{d}{=} \min\{X, N_i\}, \quad i = 1, \ldots, n, \tag{4.13}$$

where N_i is a number of transitions needed to reach a dangling node from page i, and X has a geometric distribution with parameter $1-c$. The mean and variance of X are given by

$$\mathbb{E}(X) = \frac{1}{1-c}; \quad Var(X) = \frac{c}{(1-c)^2}.$$

The upper bound for the expectation of $W_{i\cdot}$ follows now directly from (4.13). For the variance, we write

$$Var(W_{i\cdot}) = \mathbb{E}[Var(W_{i\cdot}|N_i)] + Var[\mathbb{E}(W_{i\cdot}|N_i)].$$

Conditioning on events $[N_i = k]$ and computing $Var(W_i|k)$ for $k = 1, 2, \ldots$, one can show that

$$\mathbb{E}[Var(W_{i\cdot}|N_i)] < Var(X).$$

Furthermore, we derive

$$\mathbb{E}(W_{i\cdot}|N_i) = \sum_{k=1}^{N_i} \mathbb{P}(X \geqslant k) = \sum_{k=1}^{N_i} c^k = \frac{c(1-c^{N_i})}{1-c},$$

and thus the variance of $\mathbb{E}(W_{i\cdot}|N_i)$ satisfies

$$Var(\mathbb{E}(W_{i\cdot}|N_i)) = c^2 Var(c^{N_i})/(1-c)^2 \leqslant c^4/(1-c)^2,$$

because for non-dangling nodes, the random variable c^{N_i} takes values only in the interval $[0, c]$. This completes the proof of the lemma. □

Theorem 4.1 *Given the event that the estimator* $\bar{W}_{\cdot j}$ *satisfies*

$$|\bar{W}_{\cdot j} - w_{\cdot j}| \leqslant \varepsilon w_{\cdot j}, \tag{4.14}$$

the event

$$|\bar{\pi}_j - \pi_j| \leqslant \varepsilon_{n,\beta} \pi_j$$

occurs with probability at least $1-\beta$ *for any* $\beta > 0$ *and* $\varepsilon_{n,\beta}$ *satisfying*

$$|\varepsilon - \varepsilon_{n,\beta}| < \frac{C(\beta)(1+\varepsilon)}{\sqrt{nm}}.$$

The factor $C(\beta)$ *can be approximated as*

$$C(\beta) \approx x_{1-\beta/2}\sqrt{\frac{n-n_0}{n}(1+c^3)}\frac{c}{1-c},$$

where $x_{1-\beta/2}$ *is a* $(1-\beta/2)$*-quantile of the standard normal distribution and* n_0 *is the number of dangling nodes.*

Proof Using (4.6) and (4.7), we derive

$$\begin{aligned}\bar{\pi}_j - \pi_j &= \bar{W}_{\cdot j}\bar{W}^{-1} - \pi_j \\ &= \gamma(\bar{W}_{\cdot j} - w_{\cdot j})(\gamma\bar{W})^{-1} + \left((\gamma\bar{W})^{-1} - 1\right)\pi_j.\end{aligned}$$

Given the event (4.14), the last equation together with (4.6) and (4.7) yields

$$|\bar{\pi}_j - \pi_j| \leqslant \varepsilon\pi_j + \left|(\gamma\bar{W})^{-1} - 1\right|(1+\varepsilon)\pi_j. \tag{4.15}$$

Let us now investigate the magnitude of the term $(\gamma\bar{W})^{-1}$. First, note that the random variables

$$\bar{W}_{i\cdot} = \sum_{j=1}^{n}\bar{W}_{ij}, \quad i \in \mathcal{I}_1,$$

are independent because they are determined by simulation runs initiated at different pages. Further, for a non-dangling node i, using Lemma 4.1, we find

$$\begin{aligned}\mathbb{E}(\bar{W}_{i\cdot}) &= \sum_{j=1}^{n} w_{ij}, \\ \mathrm{Var}(\bar{W}_{i\cdot}) &= \frac{1}{m}\mathrm{Var}(W_{i\cdot}) \leqslant \frac{1}{m}\frac{c(1+c^3)}{(1-c)^2}.\end{aligned}$$

Thus, $\bar{W}$ equals the number of dangling nodes n_0 plus the sum of $n-n_0$ independent random variables $\hat{W}_{i\cdot}$, $i \in \mathcal{I}_1$. Since the number $n-n_0$ is obviously very large, $\bar{W}$ is approximately normally distributed with mean γ^{-1} and variance

$$\mathrm{Var}(\bar{W}) = \sum_{i\in\mathcal{I}_1}\mathrm{Var}(\hat{W}_{i\cdot}) \leqslant (n-n_0)\frac{c(1+c^3)}{m(1-c)^2}.$$

Hence, $\gamma\bar{W}$ is approximately normally distributed with mean 1 and variance

$$\mathrm{Var}(\gamma\bar{W}) \leqslant \gamma^2(n-n_0)\frac{c(1+c^3)}{m(1-c)^2} < \frac{n-n_0}{n^2}\frac{c(1+c^3)}{m(1-c)^2}, \tag{4.16}$$

which is a value of the order $(nm)^{-1}$. Now, let us consider a $(1-\beta)$-confidence interval defined as

$$\mathbb{P}\left(\left|(\gamma\bar{W})^{-1}-1\right|<\epsilon\right)>1-\beta \tag{4.17}$$

for some small positive β and ϵ. If ϵ is small enough so that $1/(1-\epsilon)\approx 1+\epsilon$ and $1/(1+\epsilon)\approx 1-\epsilon$, then the above probability approximately equals $\mathbb{P}\left(\left|\gamma\bar{W}-1\right|<\epsilon\right)$, and because of (4.16), the inequality (4.17) holds for all ϵ satisfying

$$\epsilon \geqslant x_{1-\beta/2}\frac{c}{1-c}\sqrt{\frac{n-n_0}{n}(1+c^3)}\frac{1}{\sqrt{nm}}. \tag{4.18}$$

The right-hand side of (4.18) constitutes the additional relative error in estimating π_j. For any $\beta>0$, this additional error can be exceeded with probability at most β. This completes the proof of the theorem. □

Theorem 4.1 has two important consequences. First, it states that the estimator $\bar{\pi}_j$ converges to π_j in probability when m goes to infinity. Thus, the estimator $\bar{\pi}_j$ is consistent. Second, Theorem 4.1 states that the error in the estimate of π_j originates mainly from estimating $w_{\cdot j}$. The additional relative error caused by estimating γ as $\left[\sum \bar{W}_{\cdot j}\right]^{-1}$, is of the order $1/\sqrt{mn}$ with arbitrarily high probability, and thus this error can essentially be neglected.

It follows from the above analysis that the quality of the estimator $\bar{\pi}_j$ as well as the complexity of the algorithm can be evaluated by the estimator $\bar{W}_{\cdot j}$. We proceed by analyzing the confidence intervals. Consider the confidence interval for $\bar{W}_{\cdot j}$ defined as

$$\mathbb{P}(|\bar{W}_{\cdot j}-w_{\cdot j}|<\varepsilon w_{\cdot j})\geqslant 1-\alpha. \tag{4.19}$$

From (4.9) and (4.10), we have

$$\mathbb{E}(\bar{W}_{\cdot j})=w_{\cdot j},\quad \mathrm{Var}(\bar{W}_{\cdot j})\leqslant\frac{1}{m}\frac{1+q_{jj}}{1-q_{jj}}w_{\cdot j}.$$

Since $\bar{W}_{\cdot j}$ is a sum of a large number of terms, the random variable $[\bar{W}_{\cdot j}-w_{\cdot j}]/\sqrt{\mathrm{Var}(\bar{W}_{\cdot j})}$ has approximally a standard normal distribution. Thus, from (4.19) we deduce

$$\varepsilon w_{\cdot j}/\sqrt{\mathrm{Var}(\bar{W}_{\cdot j})}\geqslant x_{1-\alpha/2},$$

which results in

$$m\geqslant\frac{1+q_{jj}}{1-q_{jj}}\frac{x^2_{1-\alpha/2}}{\varepsilon^2 w_{\cdot j}}.$$

Now applying $w_{\cdot j}=\gamma^{-1}\pi_j$, we get

$$m\approx\frac{1+q_{jj}}{1-q_{jj}}\frac{\gamma x^2_{1-\alpha/2}}{\varepsilon^2\pi_j}. \tag{4.20}$$

Note that $\pi_j \geqslant \gamma$ for all $j = \overline{1,n}$. Thus, with a high probability, a couple of hundreds iterations allows to evaluate the PageRank of all pages with relative error at most 0.1. In practice, however, it is essential to evaluate well the PageRank of important pages in a short time. We argue that a typical user of a search engine does not check more than a dozen of first answers to his/her query. Therefore, let us evaluate the relative error ε for a given value of π_j. Using (4.4), from (4.20) we derive

$$\varepsilon \approx x_{1-\alpha/2}\sqrt{\frac{1+q_{jj}}{1-q_{jj}}}\frac{\sqrt{1-c+c\sum_{i\in\mathcal{I}_0}\pi_i}}{\sqrt{\pi_j}\sqrt{mn}}. \tag{4.21}$$

Strikingly, it follows from (4.21) that the Monte Carlo method gives good results for important pages in one iteration only, that is, when $m = 1$. From the examples of PageRank values presented in [71], it follows that the PageRank of popular pages is at least 10^4 times greater than the PageRank of an average page. Since the PageRank value is bounded from below by $(1-c)/n$, the formula (4.21) implies that if the important pages have PageRank 10^4 times larger than the PageRank of the pages with the minimal PageRank value, the Monte Carlo method achieves an error of about 1% for the important pages already after the first iteration. In contrast, the Power method takes into account only the weighted sum of the number of incoming links after the first iteration.

Let us now compare the precision of the end-point version and the complete path version of the Monte Carlo method. According to Algorithm 4.1, the end-point version estimates π_j simply as a fraction of $N = mn$ random walks that ended at page j. Using standard techniques for such estimate, we construct a confidence interval

$$\mathbb{P}(|\widehat{\pi}_{j,N} - \pi_{j,N}| < \varepsilon\pi_{j,N}) = 1-\alpha.$$

Using again the standard normal distribution, we get

$$\varepsilon = x_{1-\alpha/2}\frac{\sqrt{1-\pi_j}}{\sqrt{\pi_j}\sqrt{mn}}. \tag{4.22}$$

Forgetting for a moment about slight corrections caused by the trade-off between random and cyclic start, we see that the choice between the end-point version and the complete-path version essentially depends on two factors: the total PageRank of dangling nodes and the probability of a cycle when a random walk started from j returns back to j. If the Web graph has many short cycles then the extra information from registering visits to every page is obtained at cost of a high extra variability which leads to a worse precision. If total rank of dangling nodes is high, the random walk will often reach dangling nodes and stop. This can have a negative impact on the complete path algorithm. The above mentioned two phenomena, if present, can make the difference between the end-point and the complete-path versions negligible. The experiments

of the next section on the real data however indicate that the real Web structure is such that the complete path version is more efficient than the end-point version.

We remark that if the results of the first iteration are not satisfactory, it is hard to improve them by increasing m. After m iterations, the relative error of the Monte Carlo method will reduce on average only by the factor $1/\sqrt{m}$ whereas the error of the Power method decreases exponentially with m. However, because of simplicity in implementation (in particular, simplicity in parallel implementation), the Monte Carlo algorithms can be still advantageous even if a high precision is required.

Let us also evaluate a magnitude of π_j's for which a desired relative error ε is achieved. Rewriting (4.21), we get

$$\pi_j \approx x^2_{1-\alpha/2} \frac{1+q_{jj}}{1-q_{jj}} \frac{(1-c+c\sum_{i\in\mathcal{I}_0}\pi_i)}{\varepsilon^2 mn}. \tag{4.23}$$

Finally, we would like to emphasize that the Monte Carlo algorithms have natural parallel implementation and they allow to perform a continuous update of the PageRank vector. Indeed, each available processor can run an independent Monte Carlo simulation. Since the PageRank vector changes significantly during one month, Google prefers to recompute the PageRank vector starting from the uniform distribution rather than to use the PageRank vector of the previous month as the initial approximation [58]. Then, it takes about a week to compute a new PageRank vector. It is possible to update the PageRank vector using linear algebra methods [59]. However, one needs first to separate new nodes and links from the old ones. This is not necessary if one uses Monte Carlo algorithms. Specifically, we suggest to run Monte Carlo algorithms continuously while the database is updated with new data and hence to have an up-to-date estimation of the PageRank for relatively important pages with high accuracy. Then, once in a while one can run the Power method to have a good PageRank estimation for all pages. In particular, the continuous update should eliminate the negative reaction of users to the so-called "Google dance" [75].

4.5 Numerical experiments

For our numerical experiments we have taken the Web site of INRIA Sophia Antipolis `http://www-sop.inria.fr/`. It is a typical Web site with about 50.000 Web pages and 200.000 hyperlinks. Accordingly, datasets of similar sizes have been extensively used in experimental studies of novel algorithms for PageRank computation [1, 58, 59]. To collect the Web graph data, we construct our own Web crawler which works with the Oracle database. The crawler consists of two parts: the first part is realized based on Java and is responsible for downloading pages from the Internet, parsing the pages and inserting their hyperlinks into the database; the second part is realized with the help of the stored procedures written in PL/SQL language and

is responsible for the data management. The program allows to run several crawlers in parallel to use efficiently the network and computer resources. Since the multi-user access is already realized in Oracle database management system, it is relatively easy to organize the information collection by several crawlers. Interested reader is referred to [15] for further details about the crawler and data analysis program. We have implemented the Power method (PI, for short) and the following three Monte Carlo algorithms in PL/SQL language:

- MC complete path stopping in dangling nodes,
  MC comp path dangl nodes, for short;
- MC end-point with cyclic start,
  MC end-point cycl start, for short;
- MC complete path with random start,
  MC comp path rand start, for short.

First, we performed the sufficient number of the power iterations to obtain the value of PageRank with 20-digit accuracy. We sorted the PageRank vector in the decreasing order and plotted it in the loglog scale (see Figure 4.1). It is interesting to observe that the PageRank vector follows very closely a power law. One can also see in Figure 4.2 how well the power low approximates the PageRank vector in linear scale starting from approximately the 100^{th} largest element. Then, we have chosen four elements from the sorted PageRank vector:

$$\begin{aligned} \pi_1 &= 0.004093834, \\ \pi_{10} &= 0.001035867, \\ \pi_{100} &= 0.000546446, \\ \pi_{1000} &= 0.000097785. \end{aligned} \tag{4.24}$$

We have performed 10 iterations of the PI method and 10 iterations of the three implemented MC algorithms. In Figures 4.3-4.6, we compare the results of 10 iterations of PI method and MC complete path stopping in dangling nodes method for the four chosen pages (4.24). Indeed, as predicted by formula (4.21), already the first iteration of MC complete path stopping in dangling nodes algorithm gives a small error for important Web pages. In fact, from Figures 4.3-4.6 one can see that MC complete path stopping in dangling nodes algorithm outperforms PI method even for the first 1000 most important pages. In Figures 4.3-4.6, we also plotted 95% confidence intervals for the MC method. As expected, there are some randomness in the convergence pattern of the Monte Carlo method and some points might fall outside of confidence intervals. However, as one can see from Figures 4.3-4.4, the PI method does not converge monotonously for the first few iterations as well.

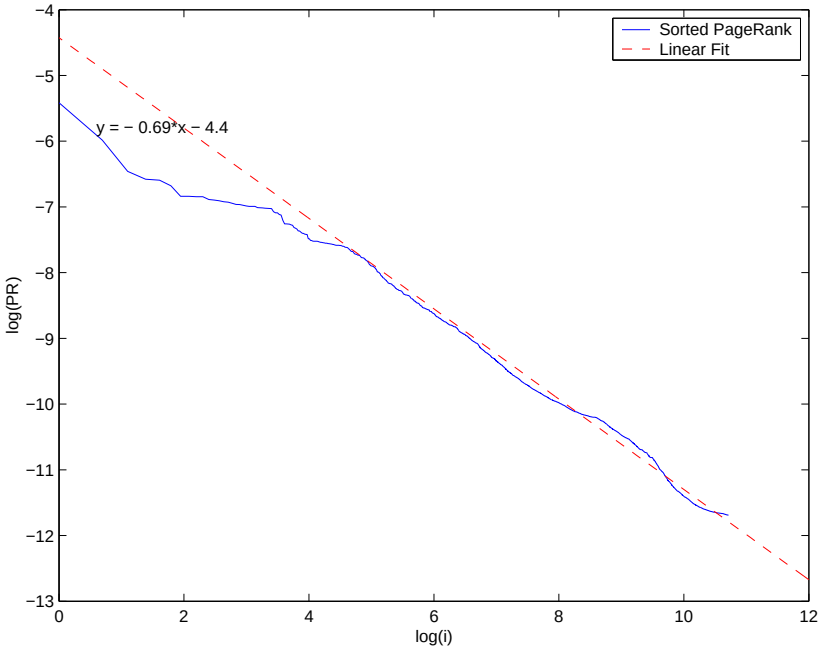


Figure 4.1: Sorted PageRank in loglog scale.

At first sight, it looks surprising that one iteration gives a relative error of only 7% with 95% confidence for pages with high PageRank. On the other hand, such result is to be expected. Roughly speaking, we use $5 * 10^4$ independent samples in order to estimate the probability $\pi = 0.004$. A binomial random variable B with parameters $n = 5 * 10^4$, $p = 0.004$ has mean 200 and standard deviation 14.1, and thus, with a high probability, a relative error of a standard estimator $\tilde{\pi} = B/n$ will be less than 11%. The additional gain that we get in (4.21) is due to regular visits to every page and the usage of the complete path information.

Next, in Figures 4.7-4.10 we compare three versions of the Monte Carlo method: MC complete path stopping in dangling nodes, MC end-point with cyclic start, and MC complete path with random start. We plotted actual relative error and the estimated 95% confidence intervals. It turns out that on our dataset MC complete path stopping in dangling nodes performs the best, followed by MC complete path with random start. MC end-point with cyclic start has the worst performance. The better performance of MC with cyclic start in respect to MC with random start was expected from the preliminary analysis of Section 2. MC is not trapped in cycles in our instance of the Web graph and the total PageRank of dangling nodes is relatively small

$$\sum_{i \in \mathcal{I}_0} \pi_i = 0.23,$$

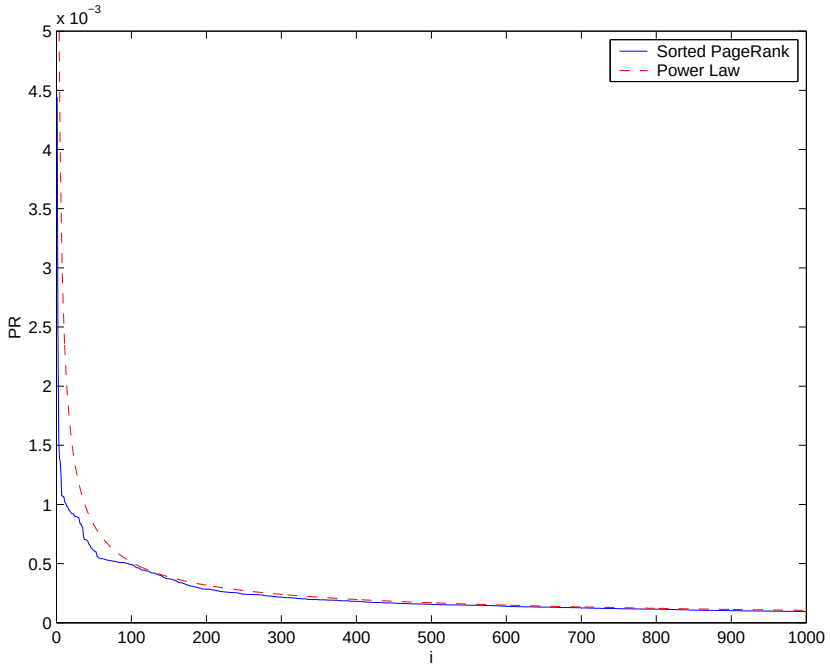


Figure 4.2: Sorted PageRank in linear scale.

hence, we have

$$\varepsilon_{\text{comp.path}} \approx \sqrt{1 - c + c \sum_{i \in \mathcal{I}_0} \pi_i}\, \varepsilon_{\text{end-point}} \approx 0.59 \varepsilon_{\text{end-point}}.$$

To check if the presence of cycles hinder the convergence of the Monte Carlo methods, we took into account the intra-page hyperlinks. On the modified graph the Monte Carlo methods have shown a very slow convergence. It is thus fortunate for MC methods that the original definition of the PageRank excludes the intra-page hyperlinks.

4.6 Conclusions

We have considered several Monte Carlo algorithms in this chapter. In particular, we have proposed a new Monte Carlo algorithm that takes into account not only the information about the last visited page, but about all visited pages during the simulation run. We have shown that MC algorithms with cyclic start outperform MC algorithms with random start. Our theoretical and experimental results have demonstrated that the Monte Carlo algorithms determine the PageRank of relatively important pages already after the first iteration. Here is a sharp contrast with the Power method that takes into account only the weighted sum of the number of incoming links after the first iteration. The other advantages of MC algorithms are natural parallel

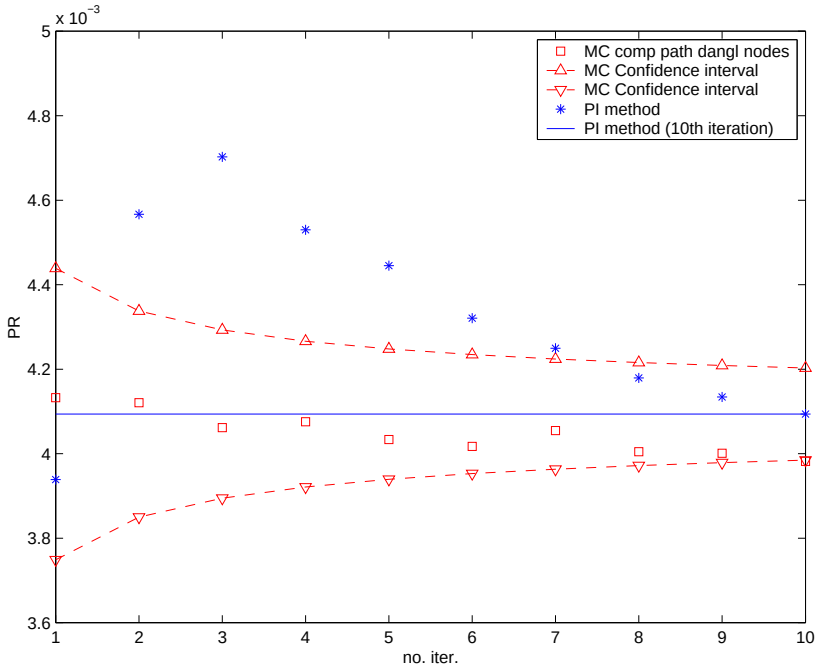


Figure 4.3: PI vs. MC comp path dangl nodes: π_1.

implementation and the possibility of the continuous PageRank update while the crawler brings new data from the Web.

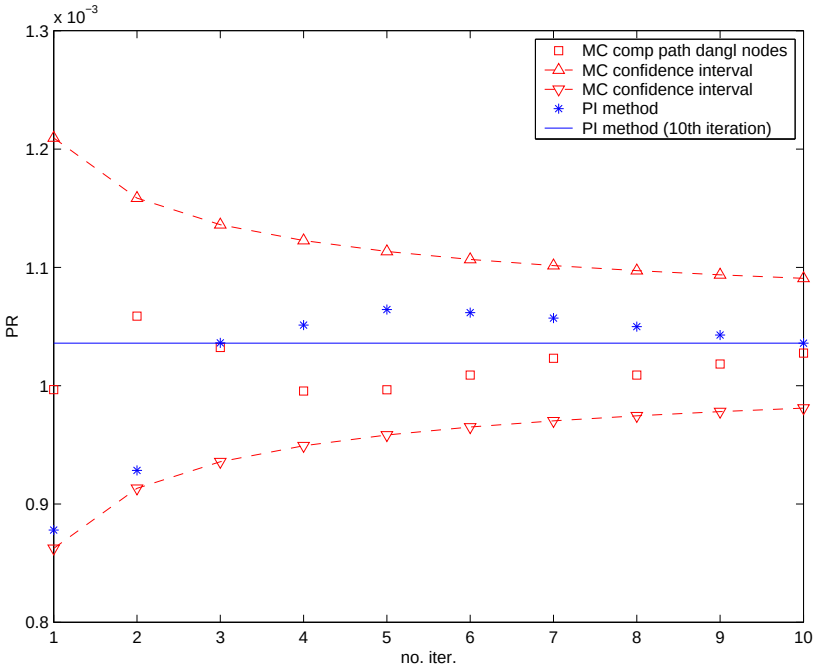


Figure 4.4: PI vs. MC comp path dangl nodes: π_{10}.

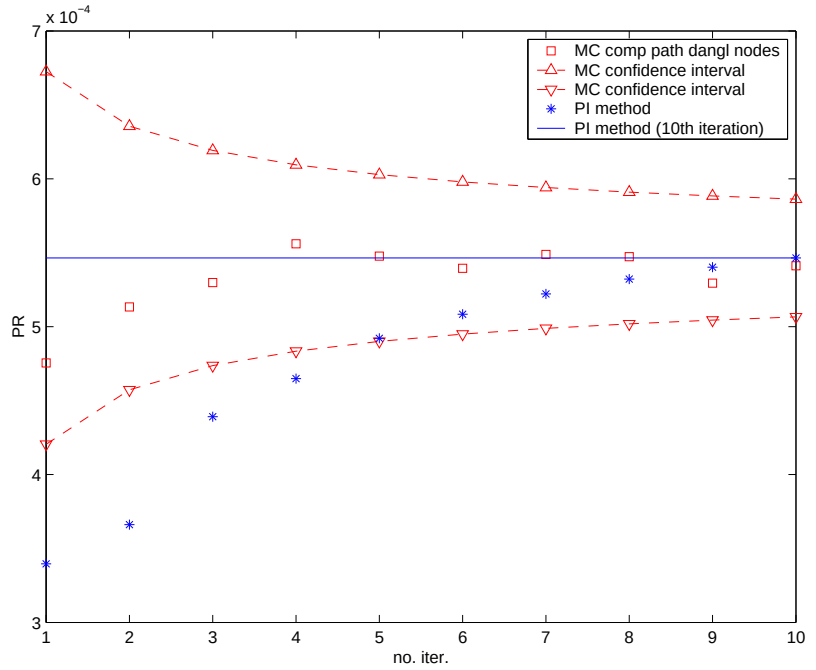


Figure 4.5: PI vs. MC comp path dangl nodes: π_{100}.

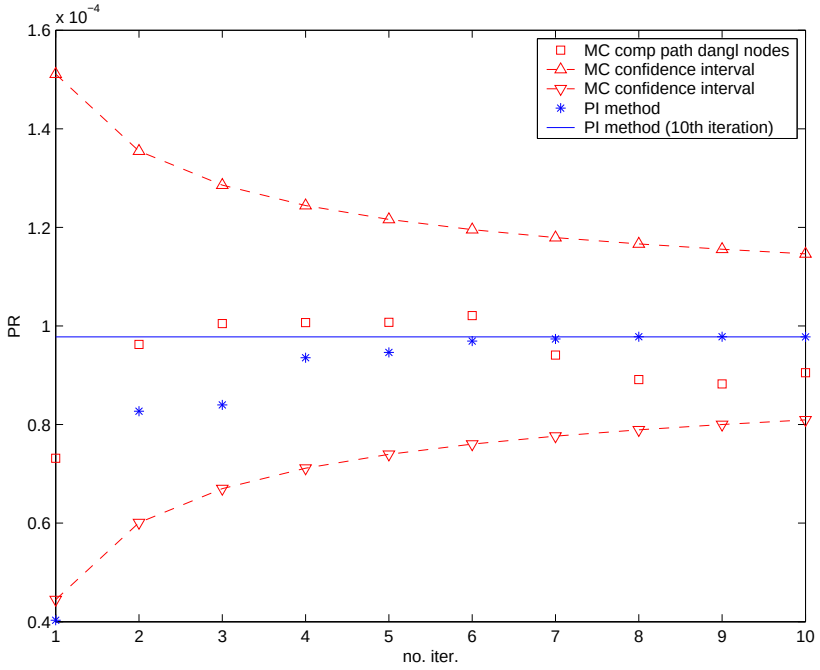


Figure 4.6: PI vs. MC comp path dangl nodes: π_{1000}.

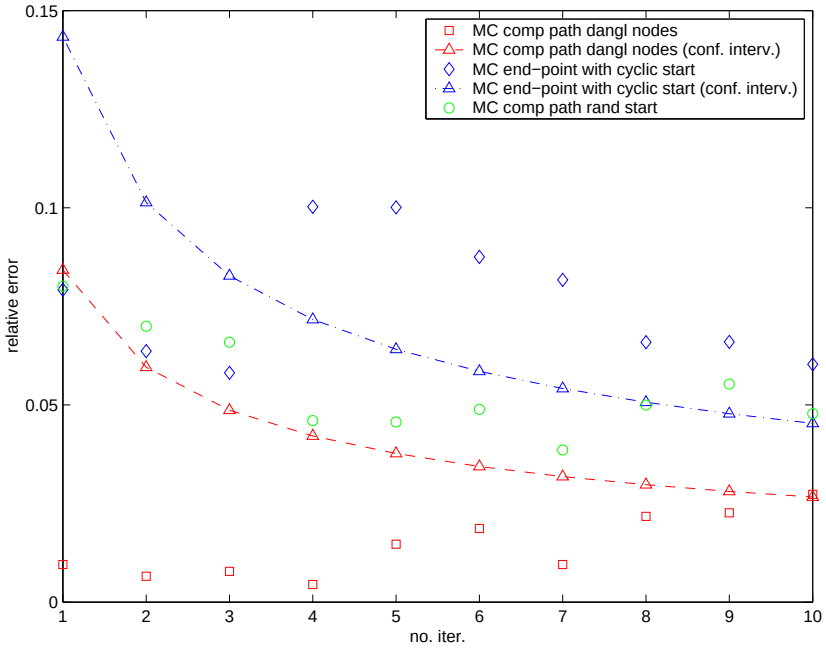


Figure 4.7: Comparison of MC algorithms: π_1.

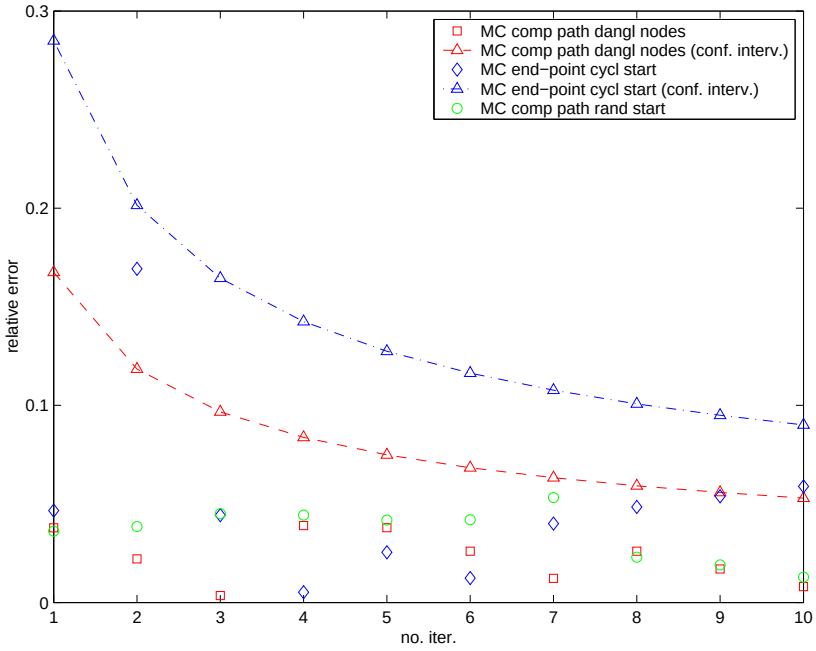


Figure 4.8: Comparison of MC algorithms: π_{10}.

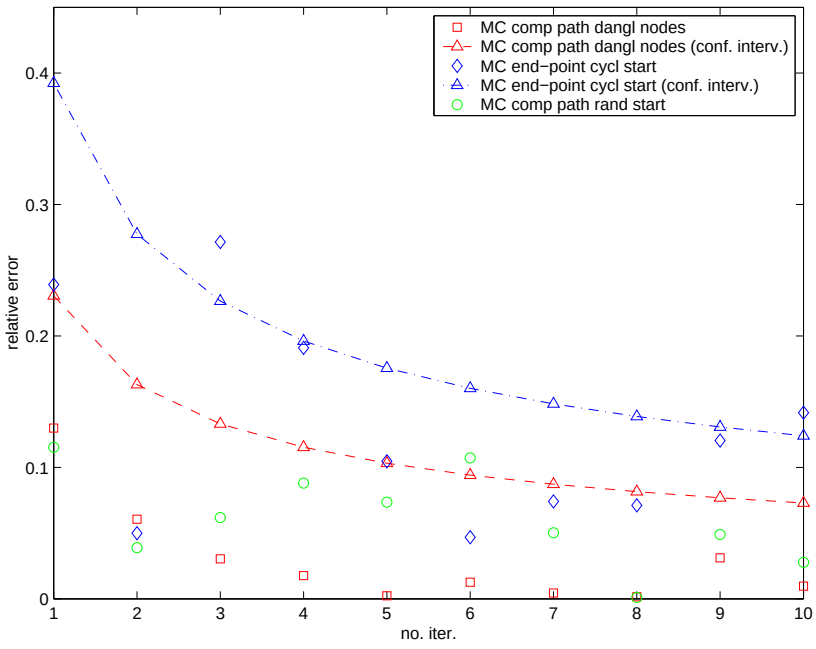


Figure 4.9: Comparison of MC algorithms: π_{100}.

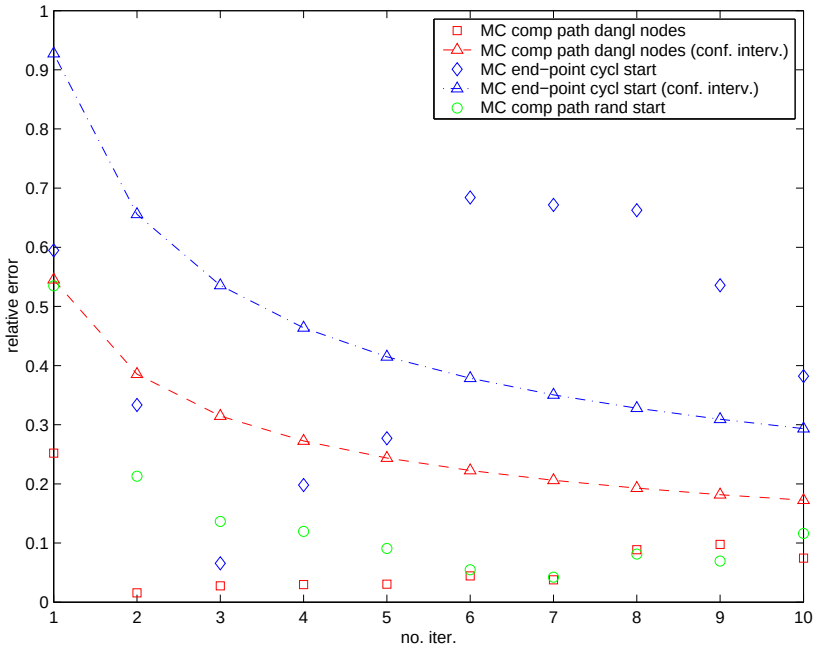


Figure 4.10: Comparison of MC algorithms: π_{1000}.

5

FINDING TOP-K LISTS WITH MONTE CARLO PERSONALIZED PAGERANK

5.1 Summary

There are a lot of applications where highly ranked pages play an important role. Monte Carlo methods applied to Personalized PageRank are considered with the aim to discover the ranking of the number of pages having high Personalized PageRank values. Three Monte Carlo methods are proposed and compared by the variance based performance of their estimators. The confidence intervals for the estimators are discovered by using the central limit theorem approximation. The probabilities to correctly reveal the top ordered or unordered list of highly ranked pages are analytically calculated and estimamet by Monte Carlo methods and by Bonferroni inequalities. The number of numerical experiments are carried out to illustrate theoretical results.

5.2 Introduction

Personalized PageRank or Topic-Sensitive PageRank [41] has a number of applications. In the original paper [41] Personalized PageRank was used to introduce the personalization in the Web search. In [29, 85] Personalized PageRank was suggested for finding related entities. In [70] Green measure, which is closely related to Personalized PageRank, was suggested for finding related pages in Wikipedia. In [4, 5] Personalized PageRank was used for finding local cuts in graphs and in [11] the Personalized PageRank was applied for clustering large hypertext document collections. In all the above mentioned applications one needs to find nodes with reasonably high values of Personalized PageRank. As was shown in [10] and presented in the previous chapter, the Monte Carlo methods are efficient for the estimation of PageRank for popular pages. Following up [10], in this chapter we propose to use Monte Carlo methods for finding top lists of pages with large values of Personalized PageRank.

Given a directed or undirected graph connecting some entities, the Personalized PageRank with a seed node i and a damping parameter c is defined as a solution of the following equations

$$\pi(i,c) = c\pi(i,c)P + (1-c)\mathbf{1}_i^T,$$

$$\sum_{j=1}^{n} \pi_j(i,c) = 1.$$

where $\mathbf{1}_i^T$ is a row unit vector with one in the i^{th} entry, P is the transition matrix associated with the entity graph and n is the number of entities. We shall also use the following Google matrix

$$G = cP + (1-c)\mathbf{1}\mathbf{1}_i^T, \tag{5.1}$$

where $\mathbf{1}$ is a column vector of ones. Equivalently, the Personalized PageRank can be given by the explicit formula

$$\pi(i,c) = (1-c)\mathbf{1}_i^T[I - cP]^{-1}. \tag{5.2}$$

Whenever the values of i and c are clear from the context we shall simply write π.

We would like to note that often the Personalized PageRank is defined with a general distribution ν in place of $\mathbf{1}_i^T$. However, typically distribution ν has a small support. Then, due to linearity, the problem of Personalized PageRank with distribution ν reduces to the problem of Personalized PageRank with distribution $\mathbf{1}_i^T$.

In this chapter we consider three Monte Carlo algorithms. The first algorithm is inspired by the following observation. Consider a random walk $\{X_t\}_{t\geqslant 0}$ that starts from node i, i.e, $X_0 = i$. Let at each step the random walk terminate with probability $1-c$ and make a transition according to the matrix P with probability c. Then, the end-points of such a random walk has the distribution $\pi(i,c)$.

Algorithm 5.1 (MC End Point) *Simulate* m *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at node* i. *Evaluate* π_j *as a fraction of* m *random walks which end at node* $j = \overline{1,n}$.

We note that a similar algorithm was employed in [10, 25] for the computation of the standard PageRank and can be found in the previous chapter.

The next observation leads to another Monte Carlo algorithm for Personalized PageRank. Denote $Z := [I - cP]^{-1}$. We have the following interpretation for the elements of matrix Z: $z_{ij} = E_i[N_j]$, where N_j is the number of visits to node j by a random walk before a restart, and $E_i[\cdot]$ is the expectation assuming that the random walk started at node i. Namely, z_{ij} is the expected number of visits to node j by the random walk initiated at state i with the run time geometrically distributed with parameter c. Thus, the formula (5.2) suggests the following estimator for Personalized PageRank

$$\hat{\pi}_j(i,c) = (1-c)\frac{1}{m}\sum_{r=1}^{m} N_j(i,r), \tag{5.3}$$

where $N_j(i,r)$ is the number of visits to state j during the run r of the random walk initiated at node i. Thus, we can suggest the second Monte Carlo algorithm.

Algorithm 5.2 (MC Complete Path) *Simulate* m *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at node* i. *Evaluate* π_j *as the total number of visits to node* j *multiplied by* $(1-c)/m$.

It was shown in [10] and in the previous chapter the MC Complete Path algorithm has a faster convergence than MC End Point for the standard PageRank.

We note that $m/(1-c)$ corresponds to the expected number of the random walk transitions after m restarts. Instead of the expected value we can use the actual count of the transition steps. This leads to the following modification of the MC Complete Path.

Algorithm 5.3 (MC Complete Path Transition Count) *Simulate* τ *steps of the random walk* $\{Y_t\}_{t\geqslant 0}$ *governed by the Google matrix* (5.1). *Evaluate* π_j *as the number of visits to node* j *divided by the total number of steps* τ.

We would like to note that the MC Complete Path and the MC Complete Path Transition Count produce the same ranking.

As outputs of the proposed algorithms we would like to obtain with high probability either a *top-k list* of nodes or a *top-k basket* of nodes.

Definition 7 *The* top-k list *of nodes is a list of* k *nodes with largest Personalized PageRank values arranged in a descending order of their Personalized PageRank values.*

Definition 8 *The* top-k basket *of nodes is a set of* k *nodes with largest Personalized PageRank values with no ordering required.*

We take the following technical assumption which is not restrictive and is satisfied in most practical applications.

Assumption 5.1 *We assume that* $\pi_1 > \pi_2 > ... > \pi_k > \pi_{k+1} \geqslant \pi_j$ *for* $j \geqslant k+2$.

It turns out that it is beneficial to relax our goal and to obtain a top-k basket with a small number of erroneous elements.

Definition 9 *We call* relaxation-l *top-*k *basket a realization when we allow at most* l *erroneous elements from top-*k *basket.*

In the present work we aim to estimate the numbers of random walk runs m sufficient for obtaining top-k list or top-k basket or relaxation-l top-k basket with high probability. In particular, we demonstrate that ranking converges considerably faster than the values of Personalized PageRank and that a relaxation-l with really small l helps significantly.

5.3 Variance based performance comparison

In the MC End Point algorithm the distribution of end points is multinomial. Namely, if we denote by L_j the number of paths that end at node j after m runs, then we have

$$P\{L_1 = l_1, L_2 = l_2, \ldots, L_n = l_n\} = \frac{m!}{l_1! l_2! \cdots l_n!} \pi_1^{l_1} \pi_2^{l_2} \cdots \pi_n^{l_n}. \tag{5.4}$$

Thus, the standard deviation of the MC End Point estimator for the k^{th} element is given by

$$\sigma(\hat{\pi}_k) = \sigma(L_k/m) = \frac{1}{\sqrt{m}} \sqrt{\pi_k(1-\pi_k)}. \tag{5.5}$$

An expression for the standard deviation of the MC Complete Path is more complicated. From (5.3), it follows that

$$\sigma(\hat{\pi}_k) = \frac{(1-c)}{\sqrt{m}} \sigma(N_k) = \frac{(1-c)}{\sqrt{m}} \sqrt{E_i\{N_k^2\} - E_i\{N_k\}^2}. \tag{5.6}$$

First, we recall that

$$E_i\{N_k\} = Z_{ik} = \pi_k(i)/(1-c). \tag{5.7}$$

Then, from [51], it is known that the second moment of N_k is given by

$$E_i\{N_k^2\} = [Z(2Z_{dg} - I)]_{ik},$$

where Z_{dg} is a diagonal matrix having as its diagonal the diagonal of matrix Z and $[A]_{ik}$ denotes the $(i,k)^{\text{th}}$ element of matrix A. Thus, we can write

$$E_i\{N_k^2\} = \mathbf{1}_i^T Z(2Z_{dg} - I)\mathbf{1}_k = \frac{1}{1-c}\pi(i)(2Z_{dg} - I)\mathbf{1}_k = \frac{1}{1-c}\left(\frac{1}{1-c}\pi_k(i)\pi_k(k) - \pi_k(i)\right). \tag{5.8}$$

Substituting (5.7) and (5.8) into (5.6), we obtain

$$\sigma(\hat{\pi}_k) = \frac{1}{\sqrt{m}}\sqrt{\pi_k(i)(2\pi_k(k) - (1-c) - \pi_k(i))}. \tag{5.9}$$

Since $\pi_k(k) \approx 1 - c$, we can approximate $\sigma(\hat{\pi}_k)$ with

$$\sigma(\hat{\pi}_k) \approx \frac{1}{\sqrt{m}}\sqrt{\pi_k(i)((1-c) - \pi_k(i))}.$$

Comparing the latter expression with (5.5), we can see that MC End Point requires approximately $1/(1-c)$ steps more than MC Complete Path. This was expected as MC End Point uses only information from end points of the random walks. We would like to emphasize that $1/(1-c)$ is a significant coefficient. For instance, if $c = 0.85$, then $1/(1-c) \approx 6.7$.

To calculate the variance of MC Complete Path Transition Count we use an expression for the variance of the Markov chain state frequency estimator from [2, 51]. If the simulation is run for τ steps, the standard deviation of MC Complete Path Transition Count is given by

$$\sigma(\hat{\pi}_k) = \frac{1}{\sqrt{\tau}}\sqrt{\pi_k(2H_{kk} - 1 + \pi_k)},$$

where H_{kk} is the $(k,k)^{\text{th}}$ element of the deviation matrix $H = \sum_{t=0}^{\infty}[G^t - \mathbf{1}\pi(i,c)]$. Next, we substitute into the above equation an expression for H_{kk} in terms of the second moment of the return time f_k to node k [51]

$$H_{kk} = \frac{1}{2}(\pi_k^2 E_k\{f_k^2\} - \pi_k)$$

to obtain

$$\sigma(\hat{\pi}_k) = \frac{1}{\sqrt{\tau}}\sqrt{\pi_k(\pi_k^2 E_k\{f_k^2\} - 1)}. \tag{5.10}$$

To compare the variance of MC Complete Path with the variance of MC Complete Path Transition Count we need to change the time scale in MC Complete Path from cycles to individual transitions. After changing the time scale, we need to compare

$$\sqrt{\pi_k(\pi_k^2 E_k\{f_k^2\} - 1)}$$

with

$$\frac{\sqrt{\pi_k(i)(2\pi_k(k) - (1-c) - \pi_k(i))}}{\sqrt{1-c}}.$$

It is clear that for the values of c very close to one, it is better to use MC Complete Path Transition Count. However, for the values of c not too close to one, it might happen that the return paths have very high second moment and then it is better to apply MC Complete Path rather than MC Complete Path Transition Count.

5.4 CLT Approximations

Let us provide central limit type theorems for our estimators.

Theorem 5.1 *For large* m*, a multivariate normal density approximation to the multinomial distribution* (5.4) *is given by*

$$f(l_1, l_2, \ldots, l_n) = \left(\frac{1}{2\pi m}\right)^{(n-1)/2} \left(\frac{1}{n\pi_1\pi_2\cdots\pi_n}\right)^{1/2} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\frac{(l_i - m\pi_i)^2}{m\pi_i}\right\}, \qquad (5.11)$$

subject to $\sum_{i=1}^{n} l_i = m$.

Proof See [53] and [77]. □

Now we consider MC Complete Path. First, we note that the vectors $N(i,r) = (N_1(i,r), \ldots, N_n(i,r))$ with $r = 1, 2, \ldots$ form a sequence of i.i.d. random vectors. Hence, we can apply the multivariate central limit theorem. Denote

$$\hat{N}(i,m) = \frac{1}{m}\sum_{r=1}^{m} N(i,r). \qquad (5.12)$$

Theorem 5.2 *Let* m *go to infinity. Then, we have the following convergence in distribution to a multivariate normal distribution*

$$\sqrt{m}\left(\hat{N}(i,m) - \bar{N}\right) \xrightarrow{D} \mathcal{N}(0, \Sigma(i)),$$

where $\bar{N}(i) = \mathbf{1}_i^T Z$ *and* $\Sigma(i) = E\{N^T(i,r)N(i,r)\} - \bar{N}^T(i)\bar{N}(i)$ *is a covariance matrix, which can be expressed as*

$$\Sigma(i) = \sum_{j=1}^{n} z_{ij}\left(D(j)Z + ZD(j) - D(j)\right) - Z^T\mathbf{1}_i\mathbf{1}_i^T Z, \qquad (5.13)$$

where $D(j)$ *is defined by*

$$d_{kl}(j) = \begin{cases} 1, & \text{if } k = l = j, \\ 0, & \text{otherwise}. \end{cases}$$

Proof The convergence follows from the multivariate central limit theorem. Expression (5.13) follows from expression for mixed expectation values proved in the Appendix. □

Next, let us provide a central limit type theorem for the estimator of MC Complete Path Transition Count.

Theorem 5.3 *Let $M_j(\tau)$ be the number of visits to state j after τ steps of a Markov chain governed by the Google matrix* (5.1)*. Then, we have the following convergence in distribution to a multivariate normal distribution*

$$\sqrt{\tau}\left(\frac{1}{\tau}M(\tau) - \pi(i,c)\right) \xrightarrow{D} \mathcal{N}(0, \Gamma(i)),$$

where $M(\tau) = (M_1(\tau), ..., M_n(\tau))$ and the elements of the covariance matrix $\Gamma(i)$ are given by

$$\Gamma_{jk}(i) = \pi_j(i,c)H_{jk}(i) + \pi_k(i,c)H_{kj}(i) + \pi_j(i,c)\pi_k(i,c) - \pi_j(i,c)\delta_{jk}, \tag{5.14}$$

for $j, k = \overline{1,n}$, and where $H(i) = \sum_{t=0}^{\infty}[G^t - \mathbf{1}\pi(i,c)]$ is a deviation matrix.

Proof The convergence of the random vector of the Markov chain state frequencies to a multivariate normal random variable is established, for instance, in [79]. The expression (5.14) for the elements of the covariance matrix follows from equation (5.11) in Section 3, Chapter 2 of [2]. (An expression for the covariance matrix provided in [79] is more cumbersome.) □

The next proposition provides a nice connection between the deviation matrix $H(i)$ and the matrix Z.

Proposition 5.1 *The deviation matrix of the Markov chain with transition matrix* (5.1) *is given by*

$$H(i) = [I - \mathbf{1}\pi(i,c)]Z, \tag{5.15}$$

with $Z = [I - cP]^{-1}$.

Proof It follows by the application of the Sherman-Morrison updating formula to the equation

$$H(i) = [I - G + \mathbf{1}\pi(i,c)]^{-1} - \mathbf{1}\pi(i,c) = [(I - cP) + \mathbf{1}(\pi(i,c) - (1-c)\mathbf{1}_i^T)]^{-1} - \mathbf{1}\pi(i,c).$$

□

Using (5.15) we can obtain more insights on the structure of the covariance matrix $\Gamma(i)$ and simplify its calculation.

In Theorems 5.2 and 5.3, we proved that we can approximate Personalized PageRank by two methods: MC Complete Path and MC Complete Path Transition Count. The methods have different covariance matrices, and the better method should have smaller entries in its covariance matrix. Because of relation (5.2) between random variables used in the theorems for the methods, we should compare matrix $(1-c)^2\Sigma(i)$ with matrix $\Gamma(i)$.

Let us define matrix $\Omega(i) = \{\omega_{jk}(i)\}$ as

$$\omega_{jk}(i) = \begin{cases} z_{ij}, & \text{if } j = k, \\ 0, & \text{otherwise.} \end{cases}$$

Proposition 5.2

$$\Gamma(i) = (1-c)\left(\Omega(i)Z + Z^T\Omega(i) - \Omega(i)\right) + (1-c)^2 Z^T\left(\mathbf{1}_i\mathbf{1}_i^T - \mathbf{1}_i\mathbf{1}_i^T Z - Z^T\mathbf{1}_i\mathbf{1}_i^T\right)Z.$$

Proof Let us define matrix $\Phi(i)$ as

$$\Phi(i) = diag\{\pi_j(i,c)\},$$

Since $\pi_j(i,c) = (1-c)z_{ij}$ by equation (5.2), we have that

$$\Phi(i) = (1-c)\,\Omega(i).$$

We can rewrite expression (5.14) in a matrix form as follows:

$$\Gamma(i) = \Phi(i)H(i) + H^T(i)\Phi(i) + \pi^T(i)\pi(i) - \Phi(i).$$

We recall that $\pi(i) = (1-c)\mathbf{1}_i^T Z$ by equation (5.2) and $H(i) = (I - \mathbf{1}\pi(i))\,Z$ by equation (5.15). Hence, we have $H(i) = \left(I - \mathbf{1}(1-c)\mathbf{1}_i^T Z\right)Z$.

Let us simplify matrix $\Gamma(i)$.

$$\begin{aligned}
\Gamma(i) &= (1-c)\,\Omega(i)\left(I - \mathbf{1}(1-c)\mathbf{1}_i^T Z\right)Z + Z^T\left(I - Z^T\mathbf{1}_i\mathbf{1}^T(1-c)\right)(1-c)\,\Omega(i) + \\
&+ (1-c)Z^T\mathbf{1}_i(1-c)\mathbf{1}_i^T Z - (1-c)\,\Omega(i), \\
\Gamma(i) &= (1-c)\,\Omega(i)Z - (1-c)^2\,\Omega(i)\mathbf{1}\mathbf{1}_i^T ZZ + (1-c)\,Z^T\Omega(i) - (1-c)^2\,Z^T Z^T\mathbf{1}_i\mathbf{1}^T\Omega(i) + \\
&+ (1-c)^2\,Z^T\mathbf{1}_i\mathbf{1}_i^T Z - (1-c)\,\Omega(i), \\
\Gamma(i) &= (1-c)\left(\Omega(i)Z + Z^T\Omega(i) - \Omega(i)\right) + (1-c)^2\left(Z^T\mathbf{1}_i\mathbf{1}_i^T Z - \Omega(i)\mathbf{1}\mathbf{1}_i^T ZZ - Z^T Z^T\mathbf{1}_i\mathbf{1}^T\Omega(i)\right).
\end{aligned}$$

We note that $\mathbf{1}^T\Omega(i) = \mathbf{1}_i^T Z$, and we complete the proof by

$$\Gamma(i) = (1-c)\left(\Omega(i)Z + Z^T\Omega(i) - \Omega(i)\right) + (1-c)^2 Z^T\left(\mathbf{1}_i\mathbf{1}_i^T - \mathbf{1}_i\mathbf{1}_i^T Z - Z^T\mathbf{1}_i\mathbf{1}_i^T\right)Z.$$

□

Now we consider matrix $\Sigma(i)$.

Proposition 5.3

$$\Sigma(i) = \Omega(i)\,Z + Z^T\Omega(i) - \Omega(i) - Z^T\mathbf{1}_i\mathbf{1}_i^T Z.$$

Proof

From (5.13) we have

$$\Sigma(i) = \sum_{j=1}^{n} z_{ij}\left(D(j)Z + ZD(j) - D(j)\right) - Z^T\mathbf{1}_i\mathbf{1}_i^T Z.$$

Let us consider $\sum_{j=1}^{n} z_{ij} D(j) Z$ in component form.

$$\sum_{j=1}^{n} z_{ij} \sum_{\varphi=1}^{n} d_{l\varphi}(j) z_{\varphi k} = \sum_{j=1}^{n} z_{ij} \delta_{lj} z_{jk} = z_{il} z_{lk} = \sum_{j=1}^{n} \omega_{lj}(i) z_{jk},$$

and it implies that $\sum_{j=1}^{n} z_{ij} D(j) Z = \Omega(i) Z$. Simetrically, $\sum_{j=1}^{n} z_{ij} Z D(j) = Z^T \Omega(i)$. Equality $\sum_{j=1}^{n} z_{ij} D(j) = \Omega(i)$ can be easily established. This completes the proof. □

Let us compare the covariance matrices $\Gamma(i)$ and $\Sigma(i)$ for the values of c close to 1. When $c \to 1$, we have that $\Phi(i) = diag\{\pi_j(i, c)\} \to diag\{\pi_j\}$, where π_j are the elements of the stationary distribution of the unperturbed matrix P and $H(i) \to H$, where $H = (I - P + \mathbf{1}\pi)^{-1} - \mathbf{1}\pi$ is the deviation matrix corresponding to P. Thus, we have that

$$\Gamma(i) \to diag\{\pi_j\} H + H^T diag\{\pi_j\} + \pi^T \pi - diag\{\pi_j\}.$$

Now let us consider the covariance matrix $\Sigma(i)$. Using the asymptotics

$$Z = \frac{1}{1 - c} \mathbf{1}\pi + O(1),$$

we obtain that

$$\Sigma(i) = \frac{1}{(1 - c)^2} \pi^T \pi + o((1 - c)^{-2}).$$

The latter implies that

$$\left(\hat{N}(i, m) - \pi\right) \xrightarrow{D} \pi \mathcal{N}(0, 1)$$

Of course, one can use the joint confidence intervals for the CLT approximations to estimate the quality of top-k list or basket. However, it appears that we can propose more efficient methods. Let us consider as an example mutual ranking of two elements k and l from a list. For illustration purpose, assume that the elements are independent. Suppose that we apply some version of CLT approximation. Then, we need to compare two normal random variables Y_k and Y_l with means π_k and π_l, and with the same variance σ^2. Without loss of generality we assume that $\pi_k > \pi_l$. Then, it can be shown that one needs twice as more experiments to guarantee that the random variable Y_k and Y_l inside their confidence intervals with the confidence level α than to guarantee that $P\{Y_k \geqslant Y_l\} = \alpha$.

5.5 Ranking probabilities

For the three introduced Monte Carlo algorithms we would like to calculate or to estimate a probability that after a given number of steps we correctly obtain top-k list or top-k basket. Namely, we need to calculate the probabilities $P\{Y_1 > \cdots > Y_k > Y_j, \forall j > k\}$ and $P\{Y_i > Y_j, \forall i, j : i \leqslant k < j\}$ respectively, where Y_k, $k = \overline{1, n}$, can be either the Monte Carlo estimates of

the ranked elements or their CLT approximations. We refer to these probabilities as the ranking probabilities and we refer to complement probabilities as misranking probabilities. It turns out that the ranking probabilities of top-k list and top-k basket can be estimated with the help of Bonferroni inequality for reasonably large values of m.

5.5.1 Estimation by Bonferroni inequality

Drawing correctly the top-k basket is defined by the event

$$\bigcap_{i \leqslant k < j} \{Y_i > Y_j\}.$$

Let us apply to this event the Bonferroni inequality

$$P\left\{\bigcap_s A_s\right\} \geqslant 1 - \sum_s P\left\{\bar{A}_s\right\}.$$

We obtain

$$P\left\{\bigcap_{i \leqslant k < j} \{Y_i > Y_j\}\right\} \geqslant 1 - \sum_{i \leqslant k < j} P\left\{\overline{\{Y_i > Y_j\}}\right\}.$$

Equivalently, we can write

$$1 - P\left\{\bigcap_{i \leqslant k < j} \{Y_i > Y_j\}\right\} \leqslant \sum_{i \leqslant k < j} P\{Y_i \leqslant Y_j\}.$$

We note that it is very good that we obtain an upper bound in the above expression for misranking probabilities, since the upper bound will provide a guarantee on the performance of our algorithms. For all MC algorithms discussed above, we can use the CLT approximation. First, we obtain an expression for misranking probability for two nodes

$$P\{Y_i \leqslant Y_j\} = 1 - \Phi(\sqrt{m}\rho_{ij}),$$

where $\Phi(\cdot)$ is the cumulative distribution function for the standard normal random variable and

$$\rho_{ij} = \frac{\pi_i - \pi_j}{\sqrt{\sigma_i^2 + \gamma_{ij} + \sigma_j^2}}.$$

For large m, the above expression can be bounded by

$$P\{Y_i \leqslant Y_j\} \leqslant \frac{1}{\sqrt{2\pi}} e^{-\frac{\rho_{ij}^2}{2} m}$$

Since the misranking probability for two nodes $P\{Y_i \leqslant Y_j\}$ decreases when j increases, we can write

$$1 - P\left\{\bigcap_{i \leqslant k < j} \{Y_i > Y_j\}\right\} \leqslant \sum_{i=1}^{k} \left(\sum_{j=k+1}^{j^*} P\{Y_i \leqslant Y_j\} + \sum_{j=j^*+1}^{n} P\{Y_i \leqslant Y_{j^*}\} \right),$$

for some j^*. This gives the following upper bound

$$1 - P\left\{\bigcap_{i \leqslant k < j} \{Y_i > Y_j\}\right\} \leqslant \sum_{i=1}^{k} \sum_{j=k+1}^{j^*} (1 - \Phi(\sqrt{m}\rho_{ij})) + \frac{n - j^*}{\sqrt{2\pi}} \sum_{i=1}^{k} e^{-\frac{\rho_{ij^*}^2}{2} m}. \tag{5.16}$$

Since we have a finite number of terms in the right hand side of expression (5.16), we conclude that

Theorem 5.4 *The misranking probability of the top-k basket tends to zero with geometric rate, that is,*

$$1 - P\left\{\bigcap_{i \leqslant k < j} \{Y_i > Y_j\}\right\} \leqslant C a^m,$$

for some $C > 0$ and $a \in (0, 1)$.

Particularly, if we use the Monte Carlo estimates from MC End Point algorithm, we can write

$$P\{Y_i \leqslant Y_j\} = \sum_{l_i + l_j \leqslant m,\ l_i \leqslant l_j} \frac{m!}{l_i! l_j! (m - l_i - l_j)!} \pi_i^{l_i} \pi_j^{l_j} (1 - \pi_i - \pi_j)^{m - l_i - l_j}. \tag{5.17}$$

We note that ρ_{ij} has a simple expression in the case of the multinomial distribution

$$\rho_{ij} = \frac{\pi_i - \pi_j}{\sqrt{(\pi_i + \pi_j)(1 - \pi_i - \pi_j)}}.$$

The Bonferroni inequality for the top-k list gives

$$P\{Y_1 > \cdots > Y_k > Y_j, \forall j > k\} \geqslant 1 - \sum_{1 \leqslant i \leqslant k-1} P\{Y_i \leqslant Y_{i+1}\} - \sum_{k+1 \leqslant j \leqslant n} P\{Y_k \leqslant Y_j\}.$$

Using misranking probability for two elements, one can obtain more informative bounds for the top-k list as was done above for the case of top-k basket. For the misranking probability of the top-k list we also have a geometric rate of convergence.

5.5.2 Exact ranking probabilities

Let us calculates exact ranking probabilities for MC End Point algorithm. The calculation of the ranking probabilities directly using (5.4) even for small values of k and moderately large values of n appears to be very cumbersome. In [32] it was suggested to use products of

substochastic matrices to calculate efficiently the distributions for the multinomial maximum and minimum. Here we also employ the technique based on products of substochastic matrices. Define a cumulative counter $s_i = s_{i-1} + l_i$ with $s_0 = 0$ and $s_n = m$, where l_i is the number of paths that end at node i after m runs. Then, the multinomial distribution (5.4) can be equivalently defined as follows [52, 47]:

$$P\{L_1 = l_1, L_2 = l_2, ..., L_n = l_n\} = \prod_{i=1}^{n-1} C_{m-s_{i-1}}^{s_i - s_{i-1}} (\tilde{\pi}_i)^{s_i - s_{i-1}} (1 - \tilde{\pi}_i)^{m - s_{i-1}}, \tag{5.18}$$

where

$$\tilde{\pi}_i = \pi_i / \sum_{j=i}^{n} \pi_j. \tag{5.19}$$

We can see that the sequence $s_i, i = \overline{0, n}$, can be regarded as a non-homogeneous Markov chain with the transitions defined by

$$P\{s_i | s_{i-1}\} = \begin{cases} C_{m-s_{i-1}}^{s_i - s_{i-1}} (\tilde{\pi}_i)^{s_i - s_{i-1}} (1 - \tilde{\pi}_i)^{m - s_{i-1}}, & \text{for } s_i \geqslant s_{i-1}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.20}$$

Let us define the transition matrix for the i^{th} step of this non-homogeneous Markov chain by Q_i. Since $s_0 = 0$, then matrix Q_1 is actually is a row vector,

$$Q_1 = \left(P(0|0\}, P\{1|0\}, \ldots, P\{m|0\}\right).$$

Similarly, since $s_n = m$, then Q_n is a column vector of all unities. We note that $\prod_{i=1}^{n} Q_i = 1$ is the probability that the non-homogeneous Markov chain has passed all its steps.

If we want to calculate a probability of some other event which can potentially happen with the Markov chain, we vary the entries of matrices Q_i forming substochastic matrices. For example, setting to zero selected transition probabilities yields probability distributions for multinomial maximum and minimum. Specifically, to calculate the distribution $P\{\max\{L_i\} \leqslant a\}$, set the probabilities $P\{s_i|s_{i-1}\} = 0$ for s_i such that $s_i > s_{i-1} + a$. We note that computing the probability $P\{\max\{L_i\} \leqslant a\}$ requires n product of $(m + 1) \times (m + 1)$ matrices, which is linear in n and hence much less computationally demanding than a brute force enumeration. Moreover, many elements in the transition matrices are zeros, which further reduces the cost of computation.

Next, using the distributions for multinomial maximum and minimum, we can calculate the probability of drawing correctly the top-k basket. The probability of drawing correctly the top-k basket is given by $P\left\{\min_{i \leqslant k}\{L_i\} > \max_{i > k}\{L_i\}\right\}$, which can be calculated as follows:

$$P\left\{\min_{i \leqslant k}\{L_i\} > \max_{i > k}\{L_i\}\right\} = \sum_{a=0}^{m} P\left\{\min_{i \leqslant k}\{L_i\} = a, \max_{i > k}\{L_i\} < a\right\} =$$

$$= \sum_{a=0}^{m}\sum_{\sigma=0}^{m} P\Big\{\min_{i\leqslant k}\{L_i\} = a \;\Big|\; \sum_{i\leqslant k} L_i = \sigma\Big\} P\Big\{\max_{i>k}\{L_i\} < a \;\Big|\; \sum_{i>k} L_i = m-\sigma\Big\} P\Big\{\sum_{i\leqslant k} L_i = \sigma\Big\} =$$

$$= \sum_{a=0}^{m}\sum_{\sigma=0}^{m}\left(P\Big\{\min_{i\leqslant k}\{L_i\} \geqslant a \;\Big|\; \sum_{i\leqslant k} L_i = \sigma\Big\} - P\Big\{\min_{i\leqslant k}\{L_i\} \geqslant a+1 \;\Big|\; \sum_{i\leqslant k} L_i = \sigma\Big\}\right) \times$$

$$\times \quad P\Big\{\max_{i>k}\{L_i\} < a \;\Big|\; \sum_{i>k} L_i = m-\sigma\Big\} P\Big\{\sum_{i\leqslant k} L_i = \sigma\Big\}, \tag{5.21}$$

where

$$P\Big\{\sum_{i\leqslant k} L_i = \sigma\Big\} = C_m^{\sigma}\left(\sum_{i\leqslant k}\pi_i\right)^{\sigma}\left(1-\sum_{i\leqslant k}\pi_i\right)^{m-\sigma}.$$

An approach similar to the above can also be applied for the calculation of the ranking probabilities for the MC Complete Path algorithms (with and without Transition Count). Here we use the fact that for a sufficiently large number of steps, the estimates of the ranked elements have approximately multivariate normal distribution. We can write

$$P\Big\{\min_{i\leqslant k}\{Y_i\} > \max_{i>k}\{Y_i\}\Big\} = \int P\Big\{\min_{i\leqslant k}\{Y_i\} > \max_{i>k}\{Y_i\} \;\Big|\; \sum_{i\leqslant k} Y_i = \sigma\Big\} f(Y_i = \sigma)\,d\sigma =$$

$$= \int\int f_{\min_{i\leqslant k}\{Y_i\}}\Big(y \;\Big|\; \sum_{i\leqslant k} Y_i = \sigma\Big) F_{\max_{i>k}\{Y_i\}}\Big(y \;\Big|\; \sum_{i>k} Y_i = S-\sigma\Big)\,dy\,d\sigma,$$

where $f_{\min_{i\leqslant k}\{Y_i\}}\Big(y \;\Big|\; \sum_{i\leqslant k} Y_i = \sigma\Big)$ is a probability density function of the minimal element of a multivariate normal random variable, $F_{\max_{i>k}\{Y_i\}}\Big(y \;\Big|\; \sum_{i>k} Y_i = S-\sigma\Big)$ is a distribution function of the maximal element of a multivariate normal random variable, and $S = \sum_{i=1}^{n} Y_i$. The functions $f_{\min_{i\leqslant k}\{Y_i\}}$ and $F_{\max_{i>k}\{Y_i\}}$ are calculated with the help of the following formula for the probability density function of the maximal element of an absolutely continuous r-dimensional random vector [7]

$$f_{\min_{1\leqslant i\leqslant r}\{Y_i\}}(y) = \sum_{i=1}^{r} f_{Y_i}(y) F_{Y_{-i}|Y_i=y}(y\mathbf{1}),$$

where f_{Y_i} is the marginal probability density function of Y_i, $F_{Y_{-i}|Y_i=y}$ is the conditional cumulative distribution function of $Y_{-i} = (Y_1,\ldots,Y_{i-1},Y_{i+1},\ldots,Y_r)$ given $Y_i = y$, and $\mathbf{1}$ is a vector of ones of the dimension $r-1$.

Drawing correctly all the elements of top-k basket with high probability can be very computationally consuming. Let us relax the conditions and calculate the probability to draw at least l, where $l < k$, correct nodes from top-k basket. We call it relaxation-$(k-l)$ top-k basket.

First of all, we calculate the probability to draw correctly exactly l nodes from top-k basket, where $l < k$. Let us denote by $\mathcal{I}$ the set of all the nodes, $|\mathcal{I}| = n$. Let us denote by $\mathcal{K}$ the set of

top-k nodes according to Personalized PageRank, $|\mathcal{K}| = k$. Let Γ_l be the set of all the subsets of set $\mathcal{K}$ of cardinality l. Let us denote by γ_a, where $a = \overline{1, \binom{k}{l}}$, an element of set Γ_l. Set γ_a is the set of l nodes which are drawn correctly in top-k. Let us denote by $\bar{\mathcal{K}}$ the set $\mathcal{I} \setminus \mathcal{K}$. Let Θ_l be the set of all the subsets of set $\bar{\mathcal{K}}$ of cardinality $k - l$. Let us denote by ϑ_b, where $b = \overline{1, \binom{n-k}{k-l}}$, an element of set Θ_l. Set ϑ_b is the set of $k - l$ nodes which do not belong to top-k basket, but they were placed there due to rough estimation. The probability to draw correctly a particular set of elements of cardinality l is given by

$$P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} > \max_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)}\{L_i\}\} = \sum_{\rho=0}^{m} P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} = \rho, \max_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)}\{L_i\} < \rho\}$$

$$= \sum_{\rho=0}^{m}\sum_{\sigma=0}^{m} P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} = \rho | \sum_{i\in\gamma_a\cup\vartheta_b} L_i = \sigma\}$$

$$P\{\max_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)}\{L_i\} < \rho | \sum_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)} L_i = m - \sigma\} P\{\sum_{i\in\gamma_a\cup\vartheta_b} L_i = \sigma\}$$

$$= \sum_{\rho=0}^{m}\sum_{\sigma=0}^{m}\left(P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} \geqslant \rho | \sum_{i\in\gamma_a\cup\vartheta_b} L_i = \sigma\} - P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} \geqslant \rho + 1 | \sum_{i\in\gamma_a\cup\vartheta_b} L_i = \sigma\}\right)$$
$$P\{\max_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)}\{L_i\} < \rho | \sum_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)} L_i = m - \sigma\} P\{\sum_{i\in\gamma_a\cup\vartheta_b} L_i = \sigma\}, \quad (5.22)$$

where

$$P\{\sum_{i\in\gamma_a\cup\vartheta_b} L_i = \sigma\} = C_m^{\sigma}\left(\sum_{i\in\gamma_a\cup\vartheta_b}\pi_i\right)^{\sigma}\left(1 - \sum_{i\in\gamma_a\cup\vartheta_b}\pi_i\right)^{m-\sigma}.$$

The probability to draw correctly any set of elements of cardinality l is given by

$$\sum_{a=1}^{\binom{k}{l}}\sum_{b=1}^{\binom{n-k}{k-l}} P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} > \max_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)}\{L_i\}\}.$$

And the probability of relaxation-$(k-l)$ top-k basket is give by

$$\sum_{c=l}^{k}\sum_{a=1}^{\binom{k}{c}}\sum_{b=1}^{\binom{n-k}{k-c}} P\{\min_{i\in\gamma_a\cup\vartheta_b}\{L_i\} > \max_{i\in\mathcal{I}\setminus(\gamma_a\cup\vartheta_b)}\{L_i\}\}.$$

Let us now calculate the ranking probability for top-k list. We recall that the ranking probability is

$$P\{Y_1 > \cdots > Y_k > Y_i, \forall i > k\}.$$

We use the cumulative counters and (5.20). We rewrite the ranking probability in the terms of the cumulative counter.

$$P\{Y_1 > \cdots > Y_k > Y_i, \forall i > k\} = P\{s_1 - s_0 > s_2 - s_1 > \cdots > s_k - s_{k-1} > s_i - s_{i-1}, \forall i > k\}.$$

We note that

$$\prod_{i=a}^{b} P\{s_i|s_{i-1}\} = P\{s_a, s_{a+1}, \ldots, s_b|s_{a-1}\}. \tag{5.23}$$

Using the above expression we can determine $P\{s_i, s_{i+1}|s_{i-1}, s_i - s_{i-1} > s_{i+1} - s_i\}$, where $i < k$, as

$$P\{s_i, s_{i+1}|s_{i-1}, s_i - s_{i-1} > s_{i+1} - s_i\} = \begin{cases} P\{s_i, s_{i+1}|s_{i-1}\}, & \text{for } s_i - s_{i-1} > s_{i+1} - s_i, \\ 0, & \text{otherwise,} \end{cases}$$

and $P\{s_k, s_{k+1}, \ldots, s_n|s_{k-1}, s_k - s_{k-1} > s_{i+1} - s_i, \forall i \geqslant k\}$ as

$$\begin{aligned} &P\{s_k, s_{k+1}, \ldots, s_n|s_{k-1}, s_k - s_{k-1} > s_{i+1} - s_i, \forall i \geqslant k\} = \\ = &\begin{cases} P\{s_k, s_{k+1}, \ldots, s_n|s_{k-1}\}, & \text{for } s_k - s_{k-1} > s_{i+1} - s_i, \forall i \geqslant k, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

Equipped with such probabilities we form substochastic matrices $R_i = \{r^{(i)}_{s_{i-1}s_i}\}$ related to the discussed above non-homogeneous Markov chain.

$$r^{(i)}_{s_{i-1}s_i} = \sum_{s_{i+1}} P\{s_i, s_{i+1}|s_{i-1}, s_i - s_{i-1} > s_{i+1} - s_i\},$$

where $i < k$. Similarly

$$r^{(k)}_{s_{k-1}s_k} = \sum_{s_{k+1}} \sum_{s_{k+2}} \cdots \sum_{s_n} P\{s_k, s_{k+1}, \ldots, s_n|s_{k-1}, s_k - s_{k-1} > s_{i+1} - s_i, \forall i \geqslant k\}. \tag{5.24}$$

We set $R_i = Q_i$, for all $i > k$. Therefore, we defined substochastic matrices R_i, $\forall i = \overline{1, n}$, which correspond to Q_i. Hence, the ranking probability for top-k list is determined as

$$P\{s_1 - s_0 > s_2 - s_1 > \cdots > s_k - s_{k-1} > s_t - s_{t-1}, \forall t > k\} = \prod_{i=1}^{n} R_i.$$

We note that R_i, where $i > k$, are stochastic matrices, and, moreover, R_n is a column vector of unities, then it is not required to calculate the above product for all the values of index i. It is enough to multiply only R_i, where $i \leqslant k$, and sum entries of the resulted row vector.

$$P\{s_1 - s_0 > s_2 - s_1 > \cdots > s_k - s_{k-1} > s_t - s_{t-1}, \forall t > k\} = \left(\prod_{i=1}^{k} R_i\right) \mathbf{1},$$

where **1** is a column vector of unities. Calculation of this product is less computationally demanded if one multiplies matrices starting from R_1 or **1** because they are vectors. Doing so we avoid the matrix multiplication of the intermediate matrices.

The calculation R_k according to equation (5.24) is very computationally consuming. Then we propose another, simpler, approach to calculate top-k list ranking probability. Let us denote by K the event $\{Y_1 > \cdots > Y_k\}$. We note that this event does not represent the event of the drawing correctly top-k list. We need to add a condition that the top k elements have the greatest values of the variable Y. Namely, we need to consider the event $\left\{K, \min_{i\leqslant k}\{L_i\} > \max_{i>k}\{L_i\}\right\}$, which probability can be calculated as follows:

$$P\left\{K, \min_{i\leqslant k}\{L_i\} > \max_{i>k}\{L_i\}\right\} = \sum_{a=0}^{m} P\left\{K, \min_{i\leqslant k}\{L_i\} = a, \max_{i>k}\{L_i\} < a\right\} =$$

$$= \sum_{a=0}^{m}\sum_{\sigma=0}^{m} P\left\{K, \min_{i\leqslant k}\{L_i\} = a \;\middle|\; \sum_{i\leqslant k} L_i = \sigma\right\} P\left\{\max_{i>k}\{L_i\} < a \;\middle|\; \sum_{i>k} L_i = m-\sigma\right\} P\left\{\sum_{i\leqslant k} L_i = \sigma\right\}.$$

Let us consider probability $P\{K, \min_{i\leqslant k}\{L_i\} = a | \sum_{i\leqslant k} L_i = \sigma\}$.

$$P\left\{K, \min_{i\leqslant k}\{L_i\} = a \;\middle|\; \sum_{i\leqslant k} L_i = \sigma\right\} = P\left\{K, \min_{i\leqslant k}\{L_i\} = 0 \;\middle|\; \sum_{i\leqslant k} L_i = \sigma - ka\right\}. \tag{5.25}$$

The last probability is the probability of ordering k elements if MC End Point does $\sigma - ka$ steps. To calculate the probability we repeat the analysis for top-k list, but in previous approach assuming that we do not have elements below k and MC End Point does $\sigma - ka$ steps.

5.6 Numerical results

We calculate ranking probabilities for top-k basket and its relaxation in numerical experiments.

We used artificially generated graphs for our experiments. We took a undirected graph, place its node in a circle and connect each node to its neighbours clockwise and counterclockwise. We rewired each end of an edge of the graph with some low probability, and we obtain the graph which follows the small-world model which describes the graph structure of the World Wide Web fairly nice. The reader is referred to [82] for details about small-world graphs and the way to generate them. We generated two graphs according to the procedure mentioned above. The characteristics of the graphs are placed in Table 5.1. The graphical representation of the adjacency matrix of graph G_1 is given at Figure 5.1.

We used MC end point and MC complete path to estimate ranking probabilities for top-k basket and its relaxations. One can see at Figures 5.2 and 5.3 that allowing even one misranked

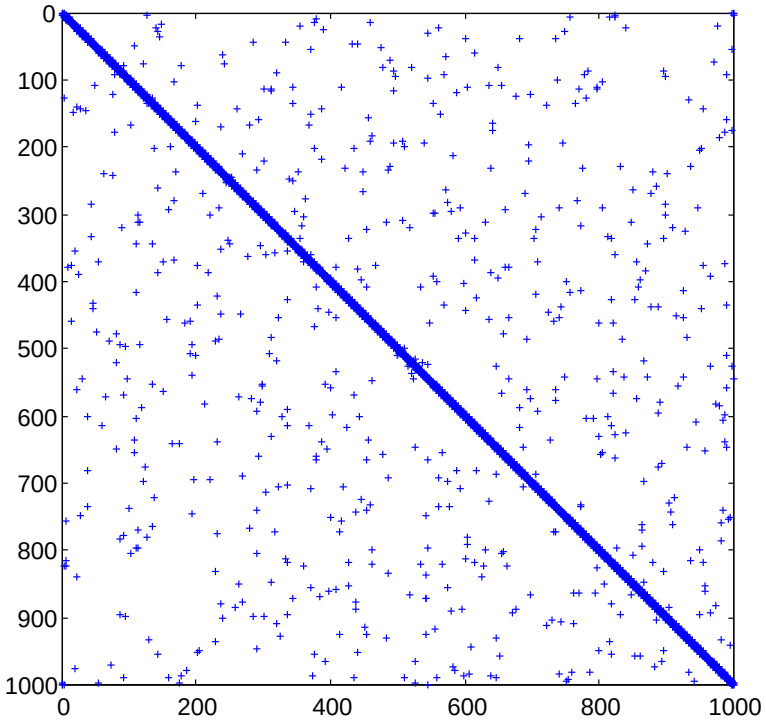


Figure 5.1: Adjacency matrix for graph G_1

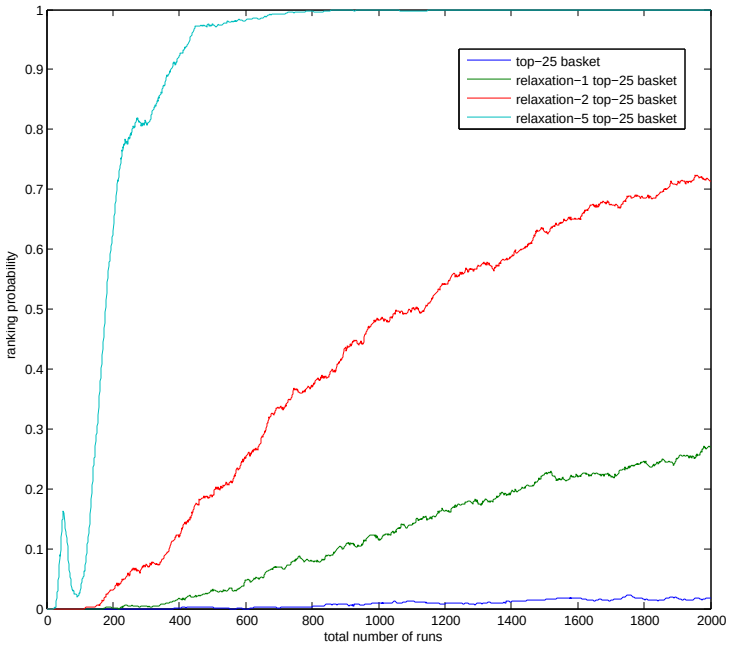


Figure 5.2: MC End Point estimation of ranking probability for graph G_1

	G_1	G_2
Total size	1000	50
Number of neighbours	6	4
Rewiring probability	0.1	0.1
Top-k	25	4

Table 5.1: Experiment graphs

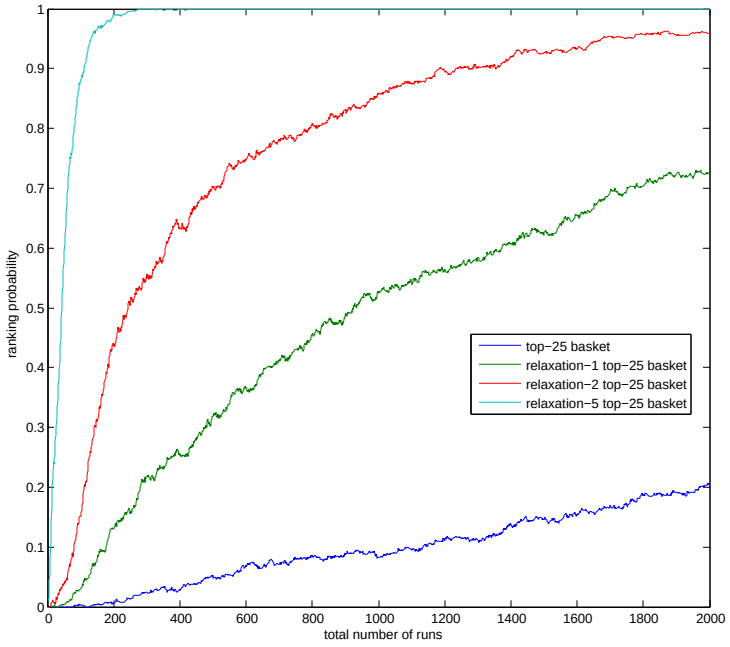


Figure 5.3: MC Complete Path estimation of ranking probability for graph G_1

element in top-k basket significantly increases the probability to detect the relaxation. If we allow several elements to be misranked, then the ranking probability increases further.

Comparing the Figures 5.2 and 5.3 one can see that MC complete path produces higher ranking probability for smaller number of runs.

We compared the estimation of ranking probability produced by MC End Point and by Bonferroni inequality at Figure 5.4. One can see that the lower bound for ranking probability produced by Bonferroni inequality follows the estimation by MC End Point starting from some number of runs.

We present the estimation of ranking probabilities for top-k basket and top-k list by MC End Point at Figure 5.5. One can see that top-k basket is significantly easier to detect.

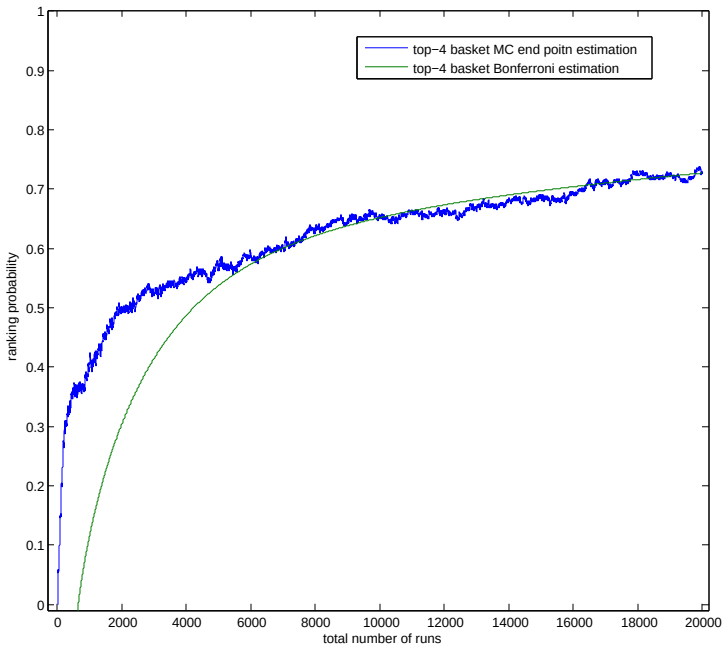


Figure 5.4: MC End Point estimation and Bonferroni estimation of ranking probability for graph G_2

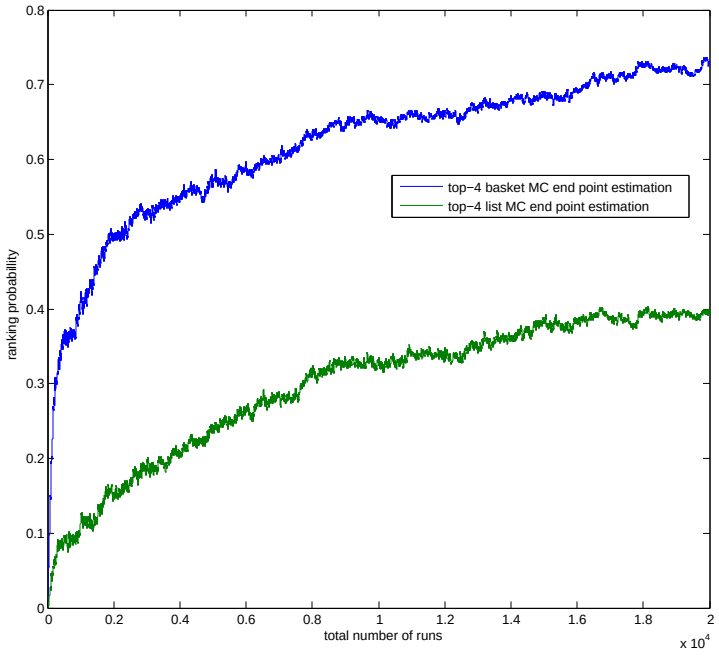


Figure 5.5: MC End Point estimation for top-4 basket and top-4 list for G_2

5.7 Conclusions

We analyzed Monte Carlo methods applied to Personalized PageRank with the aim to discover the ranking of the number of pages having high Personalized PageRank values. Three Monte Carlo methods were proposed and compared by the variance based performance of the estimators that they produce. The ranking probabilities calculated analytically give the level of certainty that the top-k list or the top-k basket are revealed correctly. We also provided estimation based on Bonferroni inequalities. In particular, we showed that the ranking probabilities converge exponentially. We considered a relaxation of top-k basket by allowing some number of erroneous elements. This relaxed top-k basket is significantly easier to detect. We carried out a number of numerical experiments to illustrate our theoretical results.

6

TENSOR APPROACH TO MIXED HIGH-ORDER MOMENTS OF ABSORBING MARKOV CHAINS

6.1 Summary

The moments of the number of the visits in an absorbing Markov chain are considered. The first moments and the non-mixed second moments of the number of the visits are calculated in classical textbooks such as the book of J. Kemeny and J. Snell “Finite Markov Chains”. The first moments and the non-mixed second moments can be easily expressed in a matrix form using the fundamental matrix of the absorbing Markov chain. Since the representation of the mixed moments of higher orders in a matrix form is not straightforward, if ever possible, they were not calculated. The gap is filled now. A tensor approach to the mixed high-order moments is proposed and compact closed-form expressions for the moments are discovered.

6.2 Introduction

Let us consider an absorbing Markov chain and let matrix P be its transition matrix. By renumbering the states, we can decompose matrix P in the following way

$$P = \begin{pmatrix} I & 0 \\ S & Q \end{pmatrix},$$

where submatrix Q is a substochastic matrix corresponding to transient states. Let T be the set of transient states and $\bar{T}$ be the set of absorbing states. We can define a fundamental matrix Z of the absorbing Markov chain

$$Z = (I - Q)^{-1} = I + Q + Q^2 + \dots.$$

Fundamental matrix $Z = \{z_{ij}\}_{i,j\in T}$ has the following probabilistic interpretation.

Definition 10 *Define* N_j *to be a function giving the total number of times before absorption that the absorbing Markov chain visits a transient state* j.

The values of the function N_j depends on the state where the Markov chain starts. Let us denote by $E_i[N_j]$ the first moment of function N_j assuming that the Markov chain starts at state i, where $i, j \in T$. Then

$$Z = \{E_i[N_j]\}_{i,j\in T}$$

as it is noted in [51, Theorem 3.2.4]. Non-mixed second moments $E_i[N_j^2]$ can also be found [51, Theorem 3.3.3] with the help of matrix Z as

$$\left\{E_i\left[N_j^2\right]\right\}_{i,j\in T} = Z\left(2Z_{dg} - I\right),$$

where Z_{dg} is the same matrix as Z, but all the off-diagonal elements are set to zero.

However, the mixed second moments $E_i[N_jN_k]$ and the mixed higher-order moments $E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right]$ are not so easy to calculate and to the best of our knowledge never has been done in a general context. Here we address this problem by tensor approach. Possible applications of our general result are Personalized PageRank [41] and polling system [3]. The mixed second moments are also used in the previous chapter related to Personalized PageRank.

6.3 Mixed second moments in matrix form

First we consider mixed second moments $E_i[N_jN_k]$ to show that calculation of them is not straightforward in matrix form.

Let us denote by u_j^k the indicator function $1_{\{X_k=j\}}$, where X_k is the value of the Markov chain at the k^{th} timestep. We note that $N_j = \sum_{\varphi=0}^{\infty} u_j^{\varphi}$.

Theorem 6.1 $E_i[N_jN_k]$ *is finite.*

Proof When we have proven the statement, it justifies our algebra with the series below.

$$E_i[N_jN_k] = E_i\left[\left(\sum_{\varphi=0}^{\infty} u_j^{\varphi}\right)\left(\sum_{\psi=0}^{\infty} u_k^{\psi}\right)\right] = E_i\left[\sum_{\varphi=0}^{\infty}\sum_{\psi=0}^{\infty} u_j^{\varphi}u_k^{\psi}\right] = \sum_{\varphi=0}^{\infty}\sum_{\psi=0}^{\infty} E_i\left[u_j^{\varphi}u_k^{\psi}\right].$$

$E_i\left[u_j^{\varphi}u_k^{\psi}\right]$ is the probability that the process is in state j at step φ and in state k at step ψ, starting in state i.

We need to consider three cases

- Let $\varphi = \psi$. If states j and k are equal, then we have that $E_i\left[u_j^{\varphi}u_k^{\varphi}\right] = p_{ij}^{(\varphi)}$; if states j and k are not equal, then we have that $E_i\left[u_j^{\varphi}u_k^{\varphi}\right] = 0$, since the process cannot be in two different states at the same step $\varphi = \psi$. Hence, we write $E_i\left[u_j^{\varphi}u_k^{\varphi}\right] = p_{ij}^{(\varphi)}\delta_{jk}$, where δ_{jk} here and below is the Kronecker symbol.
- Let $\varphi < \psi$, and let $d_1 = \psi - \varphi$. Then, $E_i\left[u_j^{\varphi}u_k^{\psi}\right]$ is the probability that the process is in state j at step φ, and in state k at step $\varphi + d_1$. Hence, $E_i\left[u_j^{\varphi}u_k^{\psi}\right] = p_{ij}^{(\varphi)}p_{jk}^{(d_1)}$.
- Let $\varphi > \psi$, and let $d_2 = \varphi - \psi$. Then, $E_i\left[u_j^{\varphi}u_k^{\psi}\right]$ is the probability that the process is in state k at step ψ, and in state j at step $\psi + d_2$. Hence, $E_i\left[u_j^{\varphi}u_k^{\psi}\right] = p_{ik}^{(\psi)}p_{kj}^{(d_2)}$.

We proceed as follows:

$$\begin{aligned} E_i[N_jN_k] &= \sum_{\varphi=0}^{\infty}\sum_{\psi=0}^{\infty} E_i\left[u_j^{\varphi}u_k^{\psi}\right] = \\ &= \sum_{\varphi=0}^{\infty}\left(\sum_{\psi=0}^{\varphi-1} E_i\left[u_j^{\varphi}u_k^{\psi}\right] + E_i\left[u_j^{\varphi}u_k^{\varphi}\right] + \sum_{\psi=\varphi+1}^{\infty} E_i\left[u_j^{\varphi}u_k^{\psi}\right]\right) = \\ &= \sum_{\varphi=0}^{\infty}\left(\sum_{\psi=0}^{\varphi-1} p_{ik}^{(\psi)}p_{kj}^{(\varphi-\psi)} + p_{ij}^{(\varphi)}\delta_{jk} + \sum_{\psi=\varphi+1}^{\infty} p_{ij}^{(\varphi)}p_{jk}^{(\psi-\varphi)}\right). \end{aligned}$$

According to [51, Corollary 3.1.2], there are numbers $b > 0$, $0 < d < 1$ such that $p_{ij}^{\varphi} \leqslant bd^{\varphi}$, and we can give the following estimate.

$$\begin{aligned} E_i[N_jN_k] &\leqslant \sum_{\varphi=0}^{\infty}\left(\sum_{\psi=0}^{\varphi-1}\left(bd^{\psi}\right)\left(bd^{\varphi-\psi}\right) + bd^{\varphi}\delta_{jk} + \sum_{\psi=\varphi+1}^{\infty}\left(bd^{\varphi}\right)\left(bd^{\psi-\varphi}\right)\right) = \\ &= \sum_{\varphi=0}^{\infty}\left(b^2\sum_{\psi=0}^{\varphi-1} d^{\varphi} + bd^{\varphi}\delta_{jk} + b^2\sum_{\psi=\varphi+1}^{\infty} d^{\psi}\right) = \end{aligned}$$

$$= \sum_{\varphi=0}^{\infty} \left(b^2 \varphi d^{\varphi} + b d^{\varphi} \delta_{jk} + b^2 d^{\varphi+1} \sum_{\psi=0}^{\infty} d^{\psi} \right) = \sum_{\varphi=0}^{\infty} \left(b^2 \varphi d^{\varphi} + b d^{\varphi} \delta_{jk} + b^2 \frac{d^{\varphi+1}}{1-d} \right) =$$

$$= \sum_{\varphi=0}^{\infty} b^2 \varphi d^{\varphi} + \sum_{\varphi=0}^{\infty} b d^{\varphi} \delta_{jk} + b^2 \frac{1}{1-d} \sum_{\varphi=0}^{\infty} d^{\varphi+1} = b^2 \sum_{\varphi=0}^{\infty} \varphi d^{\varphi} + b \delta_{jk} \frac{1}{1-d} + b^2 \frac{d}{(1-d)^2}.$$

Since $\frac{(\varphi+1)d^{\varphi+1}}{\varphi d^{\varphi}} = \frac{\varphi+1}{\varphi} d \to d < 1$, when $\varphi \to \infty$, the series $\sum_{\varphi=0}^{\infty} \varphi d^{\varphi}$ converges. This completes the proof.

□

Now that we have proven that $E_i[N_j N_k]$ is finite, let us calculate its value. We define matrix $\Lambda(i)$ as

$$\Lambda_{jk}(i) = E_i[N_j N_k].$$

Theorem 6.2 *The matrix of the mixed second order moment is given by*

$$\Lambda(i) = \sum_{\nu \in T} z_{i\nu} \left(D(\nu) Z + Z D(\nu) - D(\nu) \right),$$

where matrix $D(\nu)$ *is defined by*

$$D_{jk}(\nu) = \begin{cases} 1, & \text{if } \nu = j = k, \\ 0, & \text{otherwise.} \end{cases}$$

Proof Let us calculate $E_\nu [N_j N_k]$. We shall follow the principal idea of [51, Theorem 3.3.3], where the result is proven for the non-mixed second moments. We ask where the process can go in one step, from its starting position ν. It can go to state φ with probability $p_{\nu\varphi}$. If the new state is absorbing, then we can never reach states j or k again, and the only possible contribution is from the initial state, which is $\delta_{\nu j}\delta_{\nu k}$. If the new state is transient, then we will be in state j $\delta_{\nu j}$ times from the initial state, and N_j times from the later steps, and we will be in state k $\delta_{\nu k}$ times from the initial state, and N_k times from the later steps. Let us denote by T the set of transient states, and by $\bar{T}$ the set of absorbing states. We have

$$E_\nu [N_j N_k] = \sum_{\varphi \in \bar{T}} p_{\nu\varphi} \delta_{\nu j} \delta_{\nu k} + \sum_{\varphi \in T} p_{\nu\varphi} E_\varphi \left[(N_j + \delta_{\nu j}) (N_l + \delta_{\nu k}) \right] =$$

$$= \sum_{\varphi \in \bar{T}} p_{\nu\varphi} \delta_{\nu j} \delta_{\nu k} + \sum_{\varphi \in T} p_{\nu\varphi} \left(E_\varphi [N_j N_k] + \delta_{\nu j} E_\varphi [N_k] + E_\varphi [N_j] \delta_{\nu k} + \delta_{\nu j} \delta_{\nu k} \right) =$$

$$= \sum_{\varphi \in T} p_{\nu\varphi} \left(E_\varphi [N_j N_k] + \delta_{\nu j} E_\varphi [N_k] + E_\varphi [N_j] \delta_{\nu k} \right) + \delta_{\nu j} \delta_{\nu k} =$$

$$= \sum_{\varphi \in T} p_{\nu\varphi} E_\varphi [N_j N_k] + \sum_{\varphi \in T} p_{\nu\varphi} \left(\delta_{\nu j} E_\varphi [N_k] + E_\varphi [N_j] \delta_{\nu k} \right) + \delta_{\nu j} \delta_{\nu k}. \tag{6.1}$$

We recall that $z_{\varphi j} = E_\varphi[N_j]$. Let us denote $\varepsilon(\varphi, j, k) = E_\varphi[N_j N_k]$. Let us continue as follows

$$E_\nu[N_j N_k] = \sum_{\varphi\in T} p_{\nu\varphi} E_\varphi[N_j N_k] + \sum_{\varphi\in T} p_{\nu\varphi}(\delta_{\nu j} E_\varphi[N_k] + E_\varphi[N_j]\delta_{\nu k}) + \delta_{\nu j}\delta_{\nu k}.$$
$$\varepsilon(\nu, j, k) - \sum_{\varphi\in T} p_{\nu\varphi}\varepsilon(\varphi, j, k) = \sum_{\varphi\in T} p_{\nu\varphi}(\delta_{\nu j} z_{\varphi k} + z_{\varphi j}\delta_{\nu k}) + \delta_{\nu j}\delta_{\nu k}.$$
$$\sum_{\varphi\in T}(\delta_{\nu\varphi} - p_{\nu\varphi})\,\varepsilon(\varphi, j, k) = \sum_{\varphi\in T} p_{\nu\varphi}(\delta_{\nu j} z_{\varphi k} + z_{\varphi j}\delta_{\nu k}) + \delta_{\nu j}\delta_{\nu k}.$$

Let us multiply the last expression by $z_{i\nu}$ and sum over ν.

$$\sum_{\nu\in T} z_{i\nu}\sum_{\varphi\in T}(\delta_{\nu\varphi} - p_{\nu\varphi})\,\varepsilon(\varphi, j, k) = \sum_{\nu\in T} z_{i\nu}\sum_{\varphi\in T} p_{\nu\varphi}(\delta_{\nu j} z_{\varphi k} + z_{\varphi j}\delta_{\nu k}) + \delta_{\nu j}\delta_{\nu k}.$$

Next let us consider the left-hand side of the expression. Let us fix j, k for the moment. We can reformulate the left-hand side in matrix terms. We can consider $\varepsilon(\varphi, j, k)$ as a vector indexed by φ, let say $\varepsilon(\varphi, j, k) = \lambda_\varphi(j, k)$. Let us form matrices $Q = \{p_{\nu\varphi}\}_{\nu,\varphi\in T}$, $Z = \{z_{\varphi j}\}_{\varphi, j\in T}$, and $I = \{\delta_{\nu\varphi}\}_{\nu,\varphi\in T}$ is the identity matrix. One can see that the left-hand side can be formulated as

$$Z(I - Q)\lambda(j, k),$$

and, since $Z = (I - Q)^{-1}$, we have

$$Z(I - Q)\lambda(j, k) = \lambda(j, k),$$

or, written in a component form,

$$\sum_{\nu\in T} z_{i\nu}\sum_{\varphi\in T}(\delta_{\nu\varphi} - p_{\nu\varphi})\,\varepsilon(\varphi, j, k) = \varepsilon(i, j, k).$$

Now we consider the right-hand side.

$$\sum_{\varphi\in T} p_{\nu\varphi}(\delta_{ij} z_{\varphi k} + z_{\varphi j}\delta_{\nu k}) + \delta_{\nu j}\delta_{\nu k} =$$
$$\delta_{\nu j}\sum_{\varphi\in T} p_{\nu\varphi} z_{\varphi k} + \delta_{\nu k}\sum_{\varphi\in T} p_{\nu\varphi} z_{\varphi j} + \delta_{\nu j}\delta_{\nu k}. \tag{6.2}$$

One can see that $\delta_{\nu j}\delta_{\nu k} = D_{jk}(\nu)$ and $\delta_{\nu j}\sum_{\varphi\in T} p_{\nu\varphi} z_{\varphi k} = \{D(\nu)QZ\}_{jk}$, and $\delta_{ik}\sum_{\varphi\in T} p_{\nu\varphi} z_{\varphi j} = \{QZD(\nu)\}_{jk}$. Hence, we can write (6.2) in a matrix form.

$$D(\nu)QZ + QZD(\nu) + D(\nu).$$

Let us analyse the last expression.

$$D(\nu)QZ + QZD(\nu) + D(\nu) = D(\nu)(Z - I) + (Z - I)D(\nu) + D(\nu) =$$
$$= D(\nu)Z - D(\nu) + ZD(\nu) - D(\nu) + D(\nu) = D(\nu)Z + ZD(\nu) - D(\nu).$$

Thus, we can complete the proof by concluding that

$$\Lambda(i) = \sum_{\nu \in T} z_{i\nu} \left(D(\nu) Z + Z D(\nu) - D(\nu) \right).$$

□

One can see that we have to consider the mixed second moments either as a vector $\lambda(j,k)$ depending on two indices or as a matrix $\Lambda(i)$ depending on one indices in the above proof. We need this trick because of poverty of matrix operations. In the contrast to the matrix approach, calculation of the mixed second moments and the mixed high-order moments is natural in a tensor form, as we shall show below.

6.4 Introduction to tensors

We give a brief introduction to basic facts from the tensor theory which we shall use in the further sections. We do not present the tensor theory in its completeness, we just define what we need for our application to the mixed high-order moments. A interested reader is referred to [74, 63] for more details.

Tensors are a generalization of such notions as vector and linear operator. Firstly, let us remind the notion of vector.

We consider a vector as an objective quantity having a magnitude and a direction. The vector does not depend on the way we describe the world. We denote the vector under consideration by **a**. If we fix a coordinate system with its basis, $(e_1, e_2, \ldots, e_n)$, we can represent the vector as an array of real numbers, coordinates of the vector, $\left(a^1, a^2, \ldots, a^n\right)$,

$$\mathbf{a} = \sum_{i=1}^{n} a^i e_i.$$

When we change the basis or the coordinate system, we recalculate the coordinates by certain rules, but the vector itself does not change, see Figure 6.1. Let us assume now that we know only the vector and we do not know the coordinates of the vector. How can we determine them? It turns out that we can find a vector e^i, multiply it and vector **a** by inner product to determine coordinate a^i.

$$a^i = \mathbf{a} \cdot e^i, \qquad i = \overline{1, n}.$$

Vectors e^i are linear independent and, hence, form other basis which is called dual basis. Dual basis $\left(e^1, e^2, \ldots, e^n\right)$ relates to basis $(e_1, e_2, \ldots, e_n)$ as

$$e^i \cdot e_j = \delta^i_j,$$

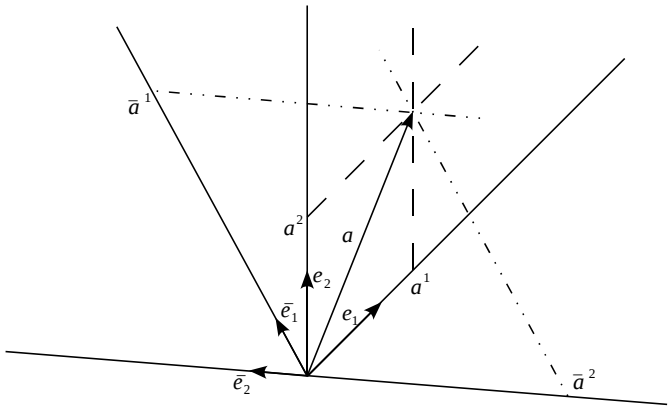


Figure 6.1: Change of a basis of a coordinate system. At the figure we have vector **a**, basis (e_1, e_2), coordinates of vector **a** in the basis (a^1, a^2), new basis $(\bar{e}_1, \bar{e}_2)$, coordinates of vector **a** in the new basis $(\bar{a}^1, \bar{a}^2)$.

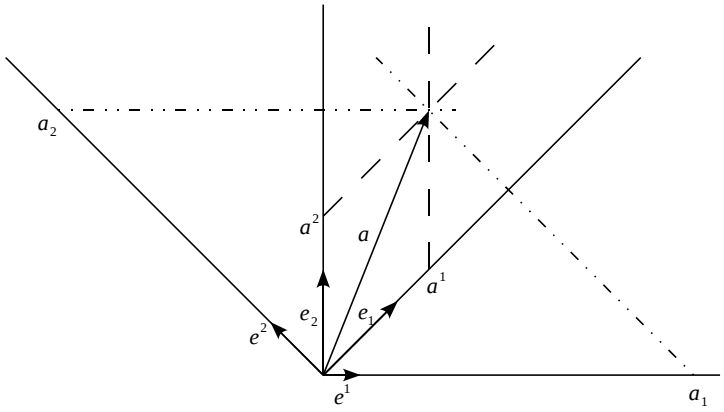


Figure 6.2: Dual basis. At the figure we have vector **a**, basis (e_1, e_2), coordinates of vector **a** in the basis (a^1, a^2), dual basis (e^1, e^2), coordinates of vector **a** in the dual basis (a_1, a_2).

where δ^i_j is the Kronecker symbol. As in any other basis, we can find coordinates of vector $\mathbf{a}$ in the dual basis, $(a_1, a_2, \ldots, a_n)$, see Figure 6.2.

$$\mathbf{a} = \sum_{i=1}^{n} a_i e^i.$$

The coordinates of the vector in the main basis are called contravariant components of the vector. The coordinates of the vector in the dual basis are called covariant components of the vector. The notions "contravariant" and "covariant" are justified by the fact that when we change basis we use different rules to recalculate the covariant and contravariant coordinates of the vector. Further, we shall always write covariant components with subscripts and contravariant components with superscripts. One can see that one-dimensional array of real numbers is enough to determine a vector.

Asides from vetors, there are other entities for which one-dimensional array of real numbers is not enough. They are linear operators, linear mappings of a vector space to another vector space. If we fix coordinate systems in the vector spaces we can express a linear operator $\mathcal{A}$ by a matrix, say matrix A. If we change the basis, we recalculate entries of matrix A obtaining another matrix $\bar{A}$, but the both matrices correspond to same linear operator $\mathcal{A}$.

$$A = \{a^i_j\} \quad \bar{A} = \{\bar{a}^i_j\}.$$

Matrices corresponding to linear operator $\mathcal{A}$ can be written in the main basis and/or the dual basis. Let matrices A, B, C correspond to linear operator $\mathcal{A}$, but they are expressed in different bases.

$$A = \{a^i_j\} \quad B = \{b^{ij}\} \quad C = \{c_{ij}\}.$$

Components of matrix A are one-time contravariant and one-time covariant, components of matrix B are twice contravariant, components of matrix C are twice covariant, but all the matrices corresponds to same linear operator $\mathcal{A}$. One can see that a linear operator can be expressed by a two-dimensional array of real numbers.

But there are entities which cannot be represented by a two-dimensional array of real numbers. They are multilinear operators which are also called tensors. Let us express a tensor $\mathcal{A}$ by components which are n-times contravariant and m-times covariant. The order of tensor is $n + m$ and its component form is given by

$$a^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_m}.$$

Let us introduce tensor operations which we need for further development. Tensor product $\otimes$ of a tensor $\mathcal{A}$ which is n-times contravariant and m-times covariant and a tensor $\mathcal{B}$ which is

s-times contravariant and t-times covariant is a tensor $\mathcal{C}$ which is $n+s$-times contravariant and $m+t$-times covariant (6.3).

$$\mathcal{A} \otimes \mathcal{B} = \mathcal{C}, \tag{6.3}$$

where components of tensor $\mathcal{C}$ in some basis can be found by formula

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{p_1 p_2 \ldots p_s}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n p_1 p_2 \ldots p_s}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t},$$

where indices $i_1, i_2, \ldots, i_n, k_1, k_2, \ldots, k_m, p_1, p_2, \ldots, p_s$ and $h_1, h_2, \ldots, h_t$ take all possible values. Further we shall write tensor product $\otimes$ as

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \otimes b^{p_1 p_2 \ldots p_s}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n p_1 p_2 \ldots p_s}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t},$$

assuming that indices $i_1, i_2, \ldots, i_n, k_1, k_2, \ldots, k_m, p_1, p_2, \ldots, p_s$ and $h_1, h_2, \ldots, h_t$ take all possible values.

In some cases, we need to consider only components of tensors having same indices in tensor product (6.4).

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \otimes b^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t} = a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t}. \tag{6.4}$$

Also let us define tensor contraction $\odot$ by formula in (6.5).

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_m} a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t}. \tag{6.5}$$

We note that tensor contraction is equivalent to matrix product if the matrices are written in one-time contravariant and one-time covariant components. Products with and without contraction follows the association algebra rule.

Proposition 6.1 *Association rule for products with and without contraction.*

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot \left(b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} \otimes c^{p_1 p_2 \ldots p_u}_{s_1 s_2 \ldots s_q} \right) = \left(a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} \right) \otimes c^{p_1 p_2 \ldots p_u}_{s_1 s_2 \ldots s_q}$$

Proof

$$\begin{aligned}
& a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot \left(b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} \otimes c^{p_1 p_2 \ldots p_u}_{s_1 s_2 \ldots s_q} \right) = \\
= & \sum_{k_1} \sum_{k_2} \cdots \sum_{k_m} a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \left(b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} c^{p_1 p_2 \ldots p_u}_{s_1 s_2 \ldots s_q} \right) = \\
= & \left(\sum_{k_1} \sum_{k_2} \cdots \sum_{k_m} a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} \right) c^{p_1 p_2 \ldots p_u}_{s_1 s_2 \ldots s_q} = \\
= & \left(a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} \right) \otimes c^{p_1 p_2 \ldots p_u}_{s_1 s_2 \ldots s_q}
\end{aligned}$$

□

We shall use tensors and the tensor operations in application to the mixed second moments and the mixed high-order moments.

6.5 Mixed second moments in tensor form

Having introduced tensor operation in the previous section, we shall show that the mixed second moments can be calculated from the tensor point of view without tricks which we used in the matrix form.

We denote $\varepsilon_j^i = E_i[N_j]$ and $\varepsilon_{jk}^i = E_i[N_jN_k]$, where ε_j^i and ε_{jk}^i we consider as tensors.

Theorem 6.3 *The mixed second moments are given by*

$$\varepsilon_{jk}^i = \varepsilon_\nu^i \odot \left(\varepsilon_j^\nu \otimes \delta_k^\nu + \varepsilon_k^\nu \otimes \delta_j^\nu - \delta_k^\nu \otimes \delta_j^\nu\right).$$

Proof We begin the proof as in Theorem 6.2 arriving to expression (6.1).

$$E_\nu[N_jN_k] = \sum_{\varphi\in T} p_{\nu\varphi}E_\varphi[N_jN_k] + \sum_{\varphi\in T} p_{\nu\varphi}\left(\delta_{\nu j}E_\varphi[N_k] + E_\varphi[N_j]\,\delta_{\nu k}\right) + \delta_{\nu j}\delta_{\nu k}.$$

Now we rewrite the above expression in the tensor form. Hence, $E_\varphi[N_j] = \varepsilon_j^\varphi$, $E_i[N_jN_k] = \varepsilon_{jk}^i$. We consider matrix $Q = \{p_{\nu\varphi}\}_{\nu,\varphi\in T}$ as tensor q_φ^ν. Kroneker symbol $\delta_{\nu k}$ we treat as δ_k^ν tensor.

$$\varepsilon_{jk}^\nu = q_\varphi^\nu \odot \varepsilon_{jk}^\varphi + q_\varphi^\nu \odot \left(\varepsilon_j^\varphi \otimes \delta_k^\nu + \varepsilon_k^\varphi \otimes \delta_j^\nu\right) + \delta_j^\nu \otimes \delta_k^\nu,$$
$$\varepsilon_{jk}^\nu - q_\varphi^\nu \odot \varepsilon_{jk}^\varphi = q_\varphi^\nu \odot \left(\varepsilon_j^\varphi \otimes \delta_k^\nu + \varepsilon_k^\varphi \otimes \delta_j^\nu\right) + \delta_j^\nu \otimes \delta_k^\nu.$$

Since $\varepsilon_{jk}^\nu = \delta_\varphi^\nu \odot \varepsilon_{jk}^\varphi$, we write

$$(\delta_\varphi^\nu - q_\varphi^\nu) \odot \varepsilon_{jk}^\varphi = q_\varphi^\nu \odot \left(\varepsilon_j^\varphi \otimes \delta_k^\nu + \varepsilon_k^\varphi \otimes \delta_j^\nu\right) + \delta_j^\nu \otimes \delta_k^\nu.$$

Let us multiply the above expression from left by ε_ν^i by tensor product with contraction.

$$\varepsilon_\nu^i \odot (\delta_\varphi^\nu - q_\varphi^\nu) \odot \varepsilon_{jk}^\varphi = \varepsilon_\nu^i \odot \left(q_\varphi^\nu \odot \left(\varepsilon_j^\varphi \otimes \delta_k^\nu + \varepsilon_k^\varphi \otimes \delta_j^\nu\right) + \delta_j^\nu \otimes \delta_k^\nu\right).$$

Since tensor ε_ν^i corresponds to matrix Z, and tensor $(\delta_\varphi^\nu - q_\varphi^\nu)$ corresponds to matrix $I - Q$, and $Z = (I - Q)^{-1}$, we obtain that

$$\varepsilon_\nu^i \odot (\delta_\varphi^\nu - q_\varphi^\nu) \odot \varepsilon_{jk}^\varphi = \delta_\varphi^i \odot \varepsilon_{jk}^\varphi = \varepsilon_{jk}^i.$$

Now we can continue using the following observation. Since tensor q_φ^ν corresponds to matrix Q, and tensor ε_j^φ corresponds to matrix Z, and $QZ = Z - I$, then $q_\varphi^\nu \odot \varepsilon_j^\varphi = \varepsilon_j^\nu - \delta_j^\nu$.

$$\begin{aligned}
\varepsilon_{jk}^i &= \varepsilon_\nu^i \odot \left(q_\varphi^\nu \odot \left(\varepsilon_j^\varphi \otimes \delta_k^\nu + \varepsilon_k^\varphi \otimes \delta_j^\nu\right) + \delta_j^\nu \otimes \delta_k^\nu\right) = \\
&= \varepsilon_\nu^i \odot \left(q_\varphi^\nu \odot \left(\varepsilon_j^\varphi \otimes \delta_k^\nu\right) + q_\varphi^\nu \odot \left(\varepsilon_k^\varphi \otimes \delta_j^\nu\right) + \delta_j^\nu \otimes \delta_k^\nu\right) = \\
&= \varepsilon_\nu^i \odot \left(\left(q_\varphi^\nu \odot \varepsilon_j^\varphi\right) \otimes \delta_k^\nu + (q_\varphi^\nu \odot \varepsilon_k^\varphi) \otimes \delta_j^\nu + \delta_j^\nu \otimes \delta_k^\nu\right) = \\
&= \varepsilon_\nu^i \odot \left(\left(\varepsilon_j^\nu - \delta_j^\nu\right) \otimes \delta_k^\nu + (\varepsilon_k^\nu - \delta_k^\nu) \otimes \delta_j^\nu + \delta_j^\nu \otimes \delta_k^\nu\right) = \\
&= \varepsilon_\nu^i \odot \left(\varepsilon_j^\nu \otimes \delta_k^\nu - \delta_j^\nu \otimes \delta_k^\nu + \varepsilon_k^\nu \otimes \delta_j^\nu - \delta_k^\nu \otimes \delta_j^\nu + \delta_j^\nu \otimes \delta_k^\nu\right) = \\
&= \varepsilon_\nu^i \odot \left(\varepsilon_j^\nu \otimes \delta_k^\nu + \varepsilon_k^\nu \otimes \delta_j^\nu - \delta_k^\nu \otimes \delta_j^\nu\right).
\end{aligned}$$

Concluding that

$$\varepsilon^{i}_{jk} = \varepsilon^{i}_{\nu} \odot \left(\varepsilon^{\nu}_{j} \otimes \delta^{\nu}_{k} + \varepsilon^{\nu}_{k} \otimes \delta^{\nu}_{j} - \delta^{\nu}_{k} \otimes \delta^{\nu}_{j} \right),$$

we complete the proof. □

One can see that we used the natural tensor operations to calculate the mixed second moments in the above proof and we need not the trick with representation of the moments as in the matrix form.

6.6 Auxiliary combinatorial result

Before we deal with the mixed high-order moments, we need an auxiliary combinatorial result.

Let M be a finite set of elements of any nature with cardinality m. Let $M = \{k_0, k_1, \ldots, k_{m-1}\}$.

Let us enumerate all the combinations of the elements of set M having length j and let us index them by ψ, where $j = \overline{0, m}$ and $\psi = \overline{0, \binom{m}{j} - 1}$. Let us define a function $f(M, j, \psi)$. Value $f(M, j, \psi)$ is the combination of the elements of set M having length j and index ψ. Let us denote $\bar{f}(M, j, \psi) = M \setminus f(M, j, \psi)$.

Let us consider $f(M, j, \psi)$, where $\psi = \overline{0, \binom{m}{j} - 1}$. Since the order of the elements in combination $f(M, j, \psi)$ does not matter, we can assume any order. Let $f(M, j, \psi) = \left\{k_{\omega_0}, k_{\omega_1}, \ldots, k_{\omega_{j-1}}\right\}$, where $\omega_x = \overline{0, m-1}$, $x = \overline{0, j-1}$. We shall assume that $\omega_0 \leqslant \omega_1 \leqslant \cdots \leqslant \omega_{j-1}$. According to [54] we can calculate ψ as

$$\psi = \sum_{x=0}^{j-1} \binom{\omega_x}{x+1},$$

where $\binom{a}{b} = 0$, if $a < b$. Such indexing provide lexicographic ordering to combinations $f(M, j, \psi)$. It means that, for example, when $m = 3$ and $j = 2$, combinations will be ordered like this: k_0k_1, k_0k_2, k_1k_3. We need the lexicographic ordering only to prove Proposition 6.2 below, although the proposition holds for any ordering. In any other discussion, we assume any, but fixed, ordering.

Let us denote by A the set of all combinations of elements of set M with length $\varkappa$.

$$A = \left\{ f(M, \varkappa, \rho) \,\middle|\, \rho = \overline{0, \binom{m}{\varkappa} - 1} \right\}.$$

Let us denote by B the following multiset.

$$B = \left\{ f\left(f(M, j, \psi), \varkappa, \chi\right) \,\middle|\, \psi = \overline{0, \binom{m}{j} - 1}, \chi = \overline{0, \binom{j}{\varkappa} - 1} \right\}.$$

One can see that multiset B consists of the same elements as set A. Let us establish a precise relation between set A and multiset B.

Proposition 6.2 *B is a multiset of the elements of set A and each element of set A is taken $\binom{m-\varkappa}{m-j}$ times.*

Proof Let us consider the following set

$$D(M,j,\varkappa)=\left\{(f(f(M,j,\psi),\varkappa,\chi),(\psi,\chi))\,\middle|\,\psi=\overline{0,\binom{m}{j}-1},\chi=\overline{0,\binom{j}{\varkappa}-1}\right\}.$$

It is the set of elements of multiset B equipped by its indices, therefore, we can distinguish equal elements of multiset B and compose the set. We shall write $D(M,j,\varkappa)=D$ if it does not produce any ambiguity.

Let us consider the following set

$$G=\left\{(f(M,\varkappa,\rho),(\rho,\iota))\,\middle|\,\rho=\overline{0,\binom{m}{\varkappa}-1},\iota=\overline{0,\binom{m-\varkappa}{m-j}-1}\right\}.$$

It is the set of the elements of set A equipped with its index ρ and auxiliary index ι. Due to index ι, each element of set A is repeated $\binom{m-\varkappa}{m-j}$ times in set G.

To prove the proposition we need to show that there is an one-to-one mapping from (ρ,ι) to (ψ,χ), which we denote by $(\psi,\chi)(\rho,\iota)$, such that

$$D=F,$$

where

$$F(M,j,\varkappa)=\left\{(f(M,\varkappa,\rho),(\psi,\chi)(\rho,\iota))\,\middle|\,\rho=\overline{0,\binom{m}{\varkappa}-1},\iota=\overline{0,\binom{m-\varkappa}{m-j}-1}\right\}.$$

We shall write $F(M,j,\varkappa)=F$ if it does not produce any ambiguity.

First of all, we prove that the cardinalities of sets D and F are equal. The cardinality of set D is equal to

$$|D|=\binom{m}{j}\binom{j}{\varkappa}=\frac{m!}{j!\,(m-j)\,!}\frac{j!}{\varkappa!\,(j-\varkappa)\,!}=\frac{m!}{(m-j)\,!\varkappa!\,(j-\varkappa)\,!}.$$

And the cardinality of set F is equal to

$$|F|=\binom{m}{\varkappa}\binom{m-\varkappa}{m-j}=\frac{m!}{\varkappa!\,(m-\varkappa)\,!}\frac{(m-\varkappa)\,!}{(m-j)\,!\,(m-j-m+\varkappa)\,!}=\frac{m!}{\varkappa!\,(m-j)\,!\,(\varkappa-j)\,!}.$$

Hence, one can see that $|D|=|F|$ and the one-to-one mapping can be potentially established. Therefore, the definition of set F is valid. We should prove that $D=F$.

We shall assume lexicographic ordering of combinations discussed above in the further development of the proof.

We shall continue the proof using the mathematical induction. We lead the induction by the cardinality of set M. Hence, let us prove the base of induction.

- Let $m = 1$ and $M = \{k_0\}$. We have following options for $(j, \varkappa)$: $(0,0)$, $(1,0)$, $(1,1)$. Let us consider each option.

  - $(j, \varkappa) = (0,0)$

$$\begin{aligned} & D(M,0,0) = \\ = & \left\{ (f(f(M,0,\psi),0,\chi),(\psi,\chi)) \middle| \psi = \overline{0,\binom{1}{0}-1}, \chi = \overline{0,\binom{0}{0}-1} \right\} = \\ = & \{(\emptyset,(0,0))\}. \end{aligned}$$

$$\begin{aligned} & F(M,0,0) = \\ = & \left\{ (f(M,0,\rho),(\psi,\chi)(\rho,\iota)) \middle| \rho = \overline{0,\binom{1}{0}-1}, \iota = \overline{0,\binom{1-0}{1-0}-1} \right\} = \\ = & \{(\emptyset,(\psi,\chi)(0,0))\} = \\ = & \{(\emptyset,(0,0))\}. \end{aligned}$$

  - $(j, \varkappa) = (1,0)$

$$\begin{aligned} & D(M,1,0) = \\ = & \left\{ (f(f(M,1,\psi),0,\chi),(\psi,\chi)) \middle| \psi = \overline{0,\binom{1}{1}-1}, \chi = \overline{0,\binom{1}{0}-1} \right\} = \\ = & \{(\emptyset,(0,0))\}. \end{aligned}$$

$$\begin{aligned} & F(M,1,0) = \\ = & \left\{ (f(M,0,\rho),(\psi,\chi)(\rho,\iota)) \middle| \rho = \overline{0,\binom{1}{0}-1}, \iota = \overline{0,\binom{1-0}{1-1}-1} \right\} = \\ = & \{(\emptyset,(\psi,\chi)(0,0))\} = \\ = & \{(\emptyset,(0,0))\}. \end{aligned}$$

  - $(j, \varkappa) = (1,1)$

$$\begin{aligned} & D(M,1,1) = \\ = & \left\{ (f(f(M,1,\psi),1,\chi),(\psi,\chi)) \middle| \psi = \overline{0,\binom{1}{1}-1}, \chi = \overline{0,\binom{1}{1}-1} \right\} = \\ = & \{(\{k_0\},(0,0))\}. \end{aligned}$$

$$
\begin{aligned}
&F(M,1,1) = \\
&= \left\{ (f(M,1,\rho),(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{0,\binom{1}{1}-1}, \iota = \overline{0,\binom{1-1}{1-1}-1} \right\} = \\
&= \{(\{k_0\},(\psi,\chi)(0,0))\} = \\
&= \{(\{k_0\},(0,0))\}.
\end{aligned}
$$

■ Let $m = 2$ and $M = \{k_0, k_1\}$. We have following options for $(j,\varkappa)$: $(0,0)$, $(1,0)$, $(1,1)$, $(2,0)$, $(2,1)$, $(2,2)$. Let us consider each option.

– $(j,\varkappa) = (0,0)$

$$
\begin{aligned}
&D(M,0,0) = \\
&= \left\{ (f(f(M,0,\psi),0,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,\binom{2}{0}-1}, \chi = \overline{0,\binom{0}{0}-1} \right\} = \\
&= \{(\emptyset,(0,0))\}.
\end{aligned}
$$

$$
\begin{aligned}
&F(M,0,0) = \\
&= \left\{ (f(M,0,\rho),(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{0,\binom{2}{0}-1}, \iota = \overline{0,\binom{2-0}{2-0}-1} \right\} = \\
&= \{(\emptyset,(\psi,\chi)(0,0))\} = \\
&= \{(\emptyset,(0,0))\}.
\end{aligned}
$$

– $(j,\varkappa) = (1,0)$

$$
\begin{aligned}
&D(M,1,0) = \\
&= \left\{ (f(f(M,1,\psi),0,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,\binom{2}{1}-1}, \chi = \overline{0,\binom{1}{0}-1} \right\} = \\
&= \left\{ (\emptyset,(\psi,\chi)) \,\middle|\, \psi = \overline{0,1}, \chi = 0 \right\} = \\
&= \{(\emptyset,(0,0)),(\emptyset,(1,0))\}.
\end{aligned}
$$

$$
\begin{aligned}
&F(M,1,0) = \\
&= \left\{ (f(M,0,\rho),(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{0,\binom{2}{0}-1}, \iota = \overline{0,\binom{2-0}{2-1}-1} \right\} = \\
&= \left\{ (\emptyset,(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = 0, \iota = \overline{0,1} \right\} = \\
&= \{(\emptyset,(\psi,\chi)(0,0)),(\emptyset,(\psi,\chi)(0,1))\} = \\
&= \{(\emptyset,(0,0)),(\emptyset,(1,0))\}.
\end{aligned}
$$

– $(j, \varkappa) = (1, 1)$

$$\begin{aligned}
&D(M,1,1) = \\
&= \left\{ (f(f(M,1,\psi),1,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0, \binom{2}{1} - 1}, \chi = \overline{0, \binom{1}{1} - 1} \right\} = \\
&= \left\{ (f(f(M,1,\psi),1,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,1}, \chi = 0 \right\} = \\
&= \{(\{k_0\},(0,0)),(\{k_1\},(1,0))\}.
\end{aligned}$$

$$\begin{aligned}
&F(M,1,1) = \\
&= \left\{ (f(M,1,\rho),(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{0, \binom{2}{1} - 1}, \iota = \overline{0, \binom{2-1}{2-1} - 1} \right\} = \\
&= \left\{ (f(M,1,\rho),(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{0,1}, \iota = 0 \right\} = \\
&= \{(\{k_0\},(\psi,\chi)(0,0)),(\{k_1\},(\psi,\chi)(1,0))\} = \\
&= \{(\{k_0\},(0,0)),(\{k_1\},(1,0))\}.
\end{aligned}$$

– $(j, \varkappa) = (2, 0)$

$$\begin{aligned}
&D(M,2,0) = \\
&= \left\{ (f(f(M,2,\psi),0,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0, \binom{2}{2} - 1}, \chi = \overline{0, \binom{2}{0} - 1} \right\} = \\
&= \{(f(f(M,2,\psi),0,\chi),(\psi,\chi)) \,|\, \psi = 0, \chi = 0\} = \\
&= \{(\emptyset,(0,0))\}.
\end{aligned}$$

$$\begin{aligned}
&F(M,2,0) = \\
&= \left\{ (f(M,0,\rho),(\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{0, \binom{2}{0} - 1}, \iota = \overline{0, \binom{2-0}{2-2} - 1} \right\} = \\
&= \{(f(M,0,\rho),(\psi,\chi)(\rho,\iota)) \,|\, \rho = 0, \iota = 0\} = \\
&= \{(\emptyset,(\psi,\chi)(0,0))\} = \\
&= \{(\emptyset,(0,0))\}.
\end{aligned}$$

– $(j, \varkappa) = (2, 1)$

$$\begin{aligned}
&D(M,2,1) = \\
&= \left\{ (f(f(M,2,\psi),1,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0, \binom{2}{2} - 1}, \chi = \overline{0, \binom{2}{1} - 1} \right\} = \\
&= \left\{ (f(f(M,2,\psi),1,\chi),(\psi,\chi)) \,\middle|\, \psi = 0, \chi = \overline{0,1} \right\} = \\
&= \{(\{k_0\},(0,0)),(\{k_1\},(0,1))\}.
\end{aligned}$$

$$
\begin{aligned}
& F(M,2,1) = \\
= & \left\{ (f(M,1,\rho),(\psi,\chi)(\rho,\iota)) \left| \rho = \overline{0,\binom{2}{1}-1}, \iota = \overline{0,\binom{2-1}{2-2}-1} \right. \right\} = \\
= & \left\{ (f(M,1,\rho),(\psi,\chi)(\rho,\iota)) \left| \rho = \overline{0,1}, \iota = 0 \right. \right\} = \\
= & \{(\{k_0\},(\psi,\chi)(0,0)),(\{k_1\},(\psi,\chi)(1,0))\} = \\
= & \{(\{k_0\},(0,0)),(\{k_1\},(0,1))\}.
\end{aligned}
$$

– $(j,\varkappa) = (2,2)$

$$
\begin{aligned}
& D(M,2,2) = \\
= & \left\{ (f(f(M,2,\psi),2,\chi),(\psi,\chi)) \left| \psi = \overline{0,\binom{2}{2}-1}, \chi = \overline{0,\binom{2}{2}-1} \right. \right\} = \\
= & \{(f(f(M,2,\psi),2,\chi),(\psi,\chi)) | \psi = 0, \chi = 0\} = \\
= & \{(\{k_0,k_1\},(0,0))\}.
\end{aligned}
$$

$$
\begin{aligned}
& F(M,2,2) = \\
= & \left\{ (f(M,2,\rho),(\psi,\chi)(\rho,\iota)) \left| \rho = \overline{0,\binom{2}{2}-1}, \iota = \overline{0,\binom{2-2}{2-2}-1} \right. \right\} = \\
= & \{(f(M,2,\rho),(\psi,\chi)(\rho,\iota)) | \rho = 0, \iota = 0\} = \\
= & \{(\{k_0,k_1\},(\psi,\chi)(0,0))\} = \\
= & \{(\{k_0,k_1\},(0,0))\}.
\end{aligned}
$$

Having proven the induction base, we continue with the induction step.

Let us consider set F. Since combinations $f(M,\varkappa,\rho)$ are ordered in lexicographic order, we know that combinations $f(M,\varkappa,\rho)$ containing element k_0 have indices $\rho = \overline{0,\binom{m-1}{\varkappa-1}-1}$, and we can write

$$
\begin{aligned}
F = & \left\{ (f(M,\varkappa,\rho),(\psi,\chi)(\rho,\iota)) \left| \rho = \overline{0,\binom{m-1}{\varkappa-1}-1}, \iota = \overline{0,\binom{m-\varkappa}{m-j}-1} \right. \right\} \cup \\
\cup & \left\{ (f(M,\varkappa,\rho),(\psi,\chi)(\rho,\iota)) \left| \rho = \overline{\binom{m-1}{\varkappa-1},\binom{m}{\varkappa}-1}, \iota = \overline{0,\binom{m-\varkappa}{m-j}-1} \right. \right\} = \\
= & F_1 \cup F_2.
\end{aligned}
$$

Combinations $f(M,\varkappa,\rho)$ as elements of set F_1 contain element k_0 and the combinations of set F_2 do not contain element k_0.

Let us consider set D. We again exploit that the combinations are ordered in lexicographic order, then combinations $f(M,j,\psi)$ containing element k_0 have indices $\psi = \overline{0,\binom{m-1}{j-1}-1}$. Thus,

we can write

$$\begin{aligned} D &= \left\{ (f(f(M,j,\psi),\varkappa,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,\binom{m-1}{j-1}-1}, \chi = \overline{0,\binom{j}{\varkappa}-1} \right\} \cup \\ &\cup \left\{ (f(f(M,j,\psi),\varkappa,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{\binom{m-1}{j-1},\binom{m}{j}-1}, \chi = \overline{0,\binom{j}{\varkappa}-1} \right\} = \\ &= D_1 \cup D_2. \end{aligned}$$

We do the same with set D_1. Combinations $f(f(M,j,\psi),\varkappa,\chi)$ containing element k_0 have indices $\chi = \overline{0,\binom{j-1}{\varkappa-1}-1}$, and we express D_1 as follows

$$\begin{aligned} D_1 &= \left\{ (f(f(M,j,\psi),\varkappa,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,\binom{m-1}{j-1}-1}, \chi = \overline{0,\binom{j-1}{\varkappa-1}-1} \right\} \cup \\ &\cup \left\{ (f(f(M,j,\psi),\varkappa,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,\binom{m-1}{j-1}-1}, \chi = \overline{\binom{j-1}{\varkappa-1},\binom{j}{\varkappa}-1} \right\} = \\ &= D_a \cup D_b. \end{aligned}$$

Hence, we partition set D as $D = D_a \cup D_b \cup D_2$.

Let us prove that $D_a = F_1$. We can rewrite F_1 as follows

$$\begin{aligned} F_1 = \Bigg\{ &\{k_0, (f(M \setminus \{k_0\}, \varkappa - 1, \rho)\}, (\psi,\chi)(\rho,\iota)) \,\Bigg| \\ &\rho = \overline{0,\binom{m-1}{\varkappa-1}-1}, \iota = \overline{0,\binom{m-\varkappa}{m-j}-1} \Bigg\}. \end{aligned}$$

Considering set D_a, one can see that each element of the set contains k_0 and all the combinations of the elements of set M containing element k_0 are counted.

$$\begin{aligned} D_a &= \left\{ (f(f(M,j,\psi),\varkappa,\chi),(\psi,\chi)) \,\middle|\, \psi = \overline{0,\binom{m-1}{j-1}-1}, \chi = \overline{0,\binom{j-1}{\varkappa-1}-1} \right\} = \\ &= \Bigg\{ (\{k_0, f(f(M \setminus \{k_0\}, j-1, \psi), \varkappa - 1, \chi)\}, (\psi,\chi)) \,\Bigg| \\ &\quad \psi = \overline{0,\binom{m-1}{j-1}-1}, \chi = \overline{0,\binom{j-1}{\varkappa-1}-1} \Bigg\}. \end{aligned}$$

One can see that $D_a = D(M \setminus \{k_0\}, j-1, \varkappa-1)$, therefore, by induction, we can conclude that

$$D_a = F_1.$$

Next we shall prove that $F_2 = D_b \cup D_2$. One can easily see that

$$F_2 = \left\{ (f(M \setminus \{k_0\}, \varkappa, \rho), (\psi,\chi)(\rho,\iota)) \,\middle|\, \rho = \overline{\binom{m-1}{\varkappa-1},\binom{m}{\varkappa}-1}, \iota = \overline{0,\binom{m-\varkappa}{m-j}-1} \right\},$$

and, renumbering elements,

$$F_2 = \left\{ (f(M \setminus \{k_0\}, \varkappa, \rho), (\psi, \chi)(\rho, \iota)) \middle| \rho = \overline{0, \binom{m-1}{\varkappa} - 1}, \iota = \overline{0, \binom{m-\varkappa}{m-j} - 1} \right\}.$$

Let us consider D_2. Since combinations $f(M, j, \psi)$ of set D_2 do not contain k_0, we write, renumbering elements,

$$D_2 = \left\{ (f(f(M \setminus \{k_0\}, j, \psi), \varkappa, \chi), (\psi, \chi)) \middle| \psi = \overline{0, \binom{m-1}{j} - 1}, \chi = \overline{0, \binom{j}{\varkappa} - 1} \right\}.$$

One can see that $D_2 = D(M \setminus \{k_0\}, j, \varkappa)$, therefore, we conclude by induction that

$$D_2 = \left\{ (f(M \setminus \{k_0\}, \varkappa, \rho), (\psi, \chi)(\rho, \iota)) \middle| \rho = \overline{0, \binom{m-1}{\varkappa} - 1}, \iota = \overline{0, \binom{m-1-\varkappa}{m-1-j} - 1} \right\}.$$

Let us consider D_b. Renumbering elements of set D_b we have

$$D_b = \left\{ (f(f(M, j, \psi), \varkappa, \chi), (\psi, \chi)) \middle| \psi = \overline{0, \binom{m-1}{j-1} - 1}, \chi = \overline{0, \binom{j-1}{\varkappa} - 1} \right\}.$$

Since combinations $f(f(M, j, \psi), \varkappa, \chi)$ do not contains element k_0, we do not need it in combinations $f(M, j, \psi)$, hence, we write

$$D_b = \left\{ (f(f(M, j, \psi) \setminus \{k_0\}, \varkappa, \chi), (\psi, \chi)) \middle| \psi = \overline{0, \binom{m-1}{j-1} - 1}, \chi = \overline{0, \binom{j-1}{\varkappa} - 1} \right\},$$

or it is the same as

$$D_b = \left\{ (f(f(M \setminus \{k_0\}, j-1, \psi), \varkappa, \chi), (\psi, \chi)) \middle| \psi = \overline{0, \binom{m-1}{j-1} - 1}, \chi = \overline{0, \binom{j-1}{\varkappa} - 1} \right\}.$$

Now one can see that $D_2 = D(M \setminus \{k_0\}, j-1, \varkappa)$, therefore, we conclude by induction that

$$D_b = \left\{ (f(M \setminus \{k_0\}, \varkappa, \rho), (\psi, \chi)(\rho, \iota)) \middle| \rho = \overline{0, \binom{m-1}{\varkappa} - 1}, \iota = \overline{0, \binom{m-1-\varkappa}{m-j} - 1} \right\}.$$

We renumber elements of D_b as follows

$$D_b = \left\{ (f(M \setminus \{k_0\}, \varkappa, \rho), (\psi, \chi)(\rho, \iota)) \middle| \rho = \overline{0, \binom{m-1}{\varkappa} - 1}, \iota = \overline{\binom{m-1-\varkappa}{m-1-j}, \binom{m-\varkappa}{m-j} - 1} \right\},$$

and we obtain that

$$D_b \cup D_2 = F_2.$$

Thus, we have

$$D_a \cup D_b \cup D_2 = F_1 \cup F_2,$$

and, consequently,

$$D = F,$$

which completes the proof. □

The auxiliary combinatorial result plays an important role in our treatment of the mixed high-order moment which we consider in the next section.

6.7 Mixed high-order moments

Let us now consider the mixed moments of higher order. Before we formulate the mixed high-order moments in tensor formalism, let us prove that the moments are finite.

Let us consider the conditional moment generating function of the absorbing Markov chain, $M_i(y) = E_i\left[e^{\sum y_j N_j}\right]$, where summation is performed over all states of the Markov chain and the process starts at a transient state i. We need to prove that the moment generating function is analytical in the origin.

Let us define vector ζ and matrix $\Theta = \{\vartheta_{ik}\}_{i,k\in T}$

$$\begin{aligned} \zeta_i &= e^{y_i}\left(1 - \sum_{k\in T} p_{ik}\right), \\ \vartheta_{ik} &= \delta_{ik} - e^{y_i} p_{ik}. \end{aligned}$$

Proposition 6.3 *If all y_i are small enough, moment generating function $M(y)$ is give by*

$$M(y) = \Theta^{-1}\zeta.$$

Proof We ask where the process can go in one step, from its starting position i.

$$\begin{aligned} E_i\left[e^{\sum y_j N_j}\right] &= \sum_{k\in\bar{T}} p_{ik} e^{y_i} + \sum_{k\in T} p_{ik} E_k\left[e^{\sum_{j\neq i} y_j N_j + y_i(N_i+1)}\right] = \\ &= \sum_{k\in\bar{T}} p_{ik} e^{y_i} + \sum_{k\in T} p_{ik} E_k\left[e^{\sum y_j N_j + y_i}\right] = \\ &= \sum_{k\in\bar{T}} p_{ik} e^{y_i} + \sum_{k\in T} p_{ik} E_k\left[e^{\sum y_j N_j} e^{y_i}\right] = \\ &= e^{y_i}\left(1 - \sum_{k\in T} p_{ik}\right) + e^{y_i} \sum_{k\in T} p_{ik} E_k\left[e^{\sum y_j N_j}\right]. \end{aligned}$$

And we solve the above equation in the following way.

$$\begin{aligned} \mathrm{E}_i\left[e^{\sum y_j N_j}\right] &= e^{y_i}\left(1-\sum_{k\in T} p_{ik}\right)+e^{y_i}\sum_{k\in T} p_{ik}\mathrm{E}_k\left[e^{\sum y_j N_j}\right], \\ \mathrm{E}_i\left[e^{\sum y_j N_j}\right]-e^{y_i}\sum_{k\in T} p_{ik}\mathrm{E}_k\left[e^{\sum y_j N_j}\right] &= e^{y_i}\left(1-\sum_{k\in T} p_{ik}\right), \\ \sum_{k\in T}\left(\delta_{ik}-e^{y_i}p_{ik}\right)\mathrm{E}_k\left[e^{\sum y_j N_j}\right] &= e^{y_i}\left(1-\sum_{k\in T} p_{ik}\right). \end{aligned} \tag{6.6}$$

Then, we can rewrite (6.6) in matrix form

$$\Theta M(y)=\zeta.$$

Let us show that matrix Θ is invertible. Let us denote by $t=|T|$ and by Ξ the matrix

$$\Xi=\begin{pmatrix} e^{y_1} & 0 & \dots & 0 \\ 0 & e^{y_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & e^{y_t} \end{pmatrix}.$$

We express matrix Θ as

$$\Theta=I-\Xi Q,$$

where Q corresponds to the transient states of the absorbing Markov chain. Since matrix Ξ is diagonal, then matrix ΞQ is matrix Q whose rows are multiplied by diagonal elements of matrix Ξ. Matrix Q is substochastic, hence,

$$Q\mathbf{1}=q,$$

where $q=(q_1,q_2,\dots,q_t)^T$, and $0\leqslant q_i\leqslant 1$, $\forall i=\overline{1,t}$, and $\exists i: q_i<1$. Then,

$$\Xi Q\mathbf{1}=q=(e^{y_1}q_1,e^{y_2}q_2,\dots,e^{y_t}q_t)^T.$$

Matrix ΞQ is substochastic, if $0\leqslant e^{y_i}q_i\leqslant 1$, $\forall i=\overline{1,t}$, and $\exists i: e^{y_i}q_i<1$. Therefore, if $y_i\leqslant -\ln q_i$, $\forall i=\overline{1,t}$, and $\exists i: y_i<-\ln q_i$, matrix Θ is invertible.

And we can determine the conditional moment generating function by

$$M(y)=\Theta^{-1}\zeta.$$

□

One can see that the conditional moment generating function $M(y)$ is analytical at the origin, and, hence, there exist all the mixed high-order moments and they are finite.

We denote

$$\begin{aligned}\varepsilon^{i}_{j} &= \mathrm{E}_i\left[N_j\right],\\ \varepsilon^{i}_{jk} &= \mathrm{E}_i\left[N_j N_k\right],\ldots,\\ \varepsilon^{i}_{k_0 k_1 \ldots k_{m-1}} &= \mathrm{E}_i\left[\prod_{j=0}^{m-1} N_{k_j}\right],\end{aligned}$$

where m is a natural number. Let us denote $M = \{k_0, k_1, \ldots, k_{m-1}\}$. The cardinality of set M is m. We call set M the basis set.

We have got the mixed moments of higher order in tensor representation. Since the product is a commutative operation, the order of indices $k_0 k_1 \ldots k_{m-1}$ in $\varepsilon^{i}_{k_0 k_1 \ldots k_{m-1}}$ does not matter, and we can write $\varepsilon^{i}_{k_0 k_1 \ldots k_{m-1}} = \varepsilon^{i}_{M}$.

Let us denote ${}_0a^{\nu}_{k_0 k_1 \ldots k_{m-1}} = \bigotimes_{\iota=0}^{m-1} \delta^{\nu}_{k_\iota}$. Let us define tensor ${}_ja^{\nu}_{k_0 k_1 \ldots k_{m-1}}$ as follows:

$${}_ja^{\nu}_{k_0 k_1 \ldots k_{m-1}} = \sum_{\psi=0}^{\binom{m}{j}-1} \varepsilon^{\nu}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)}.$$

Since index ψ passes all possible values, the order of indices $k_0 k_1 \ldots k_{m-1}$ in ${}_ja^{\nu}_{k_0 k_1 \ldots k_{m-1}}$ does not matter, and we can write ${}_ja^{\nu}_{k_0 k_1 \ldots k_{m-1}} = {}_ja^{\nu}_{M}$. We note that ${}_ma^{\nu}_{M} = \varepsilon^{\nu}_{M}$.

Let us define tensor ${}_jb^{\varphi\nu}_{k_0 k_1 \ldots k_{m-1}}$ as follows:

$${}_jb^{\varphi\nu}_{k_0 k_1 \ldots k_{m-1}} = \sum_{\psi=0}^{\binom{m}{j}-1} \varepsilon^{\varphi}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)}.$$

Since index ψ passes all possible values, the order of indices $k_0 k_1 \ldots k_{m-1}$ in ${}_jb^{\varphi\nu}_{k_0 k_1 \ldots k_{m-1}}$ does not matter, and we can write ${}_jb^{\varphi\nu}_{k_0 k_1 \ldots k_{m-1}} = {}_jb^{\varphi\nu}_{M}$. We note that ${}_mb^{\varphi\nu}_{M} = \varepsilon^{\varphi}_{M}$, ${}_0b^{\varphi\nu}_{M} = {}_0a^{\nu}_{M}$, and, in general, ${}_jb^{\nu\nu}_{M} = {}_ja^{\nu}_{M}$.

Let us define tensor ${}_{\varkappa j}c^{\nu}_{k_0 k_1 \ldots k_{m-1}}$, where $\varkappa \leqslant j$, as follows:

$${}_{\varkappa j}c^{\nu}_{k_0 k_1 \ldots k_{m-1}} = \sum_{\psi=0}^{\binom{m}{j}-1} {}_\varkappa a^{\nu}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)}.$$

Since index ψ passes all possible values, the order of indices $k_0 k_1 \ldots k_{m-1}$ in ${}_{\varkappa j}c^{\nu}_{k_0 k_1 \ldots k_{m-1}}$ does not matter, and we can write ${}_{\varkappa j}c^{\nu}_{k_0 k_1 \ldots k_{m-1}} = {}_{\varkappa j}c^{\nu}_{M}$.

Proposition 6.4 *The following formula takes place:*

$${}_{0j}c^{\nu}_{M} = \binom{m}{j} {}_0a^{\nu}_{M}.$$

Proof

$$ {}_0c^{\nu}_{M} = \sum_{\psi=0}^{\binom{m}{j}-1} {}_0a^{\nu}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)} = \sum_{\psi=0}^{\binom{m}{j}-1} {}_0a^{\nu}_{M} = \binom{m}{j} {}_0a^{\nu}_{M}. $$

□

Proposition 6.5 *The following formula takes place:*

$$ {}_jc^{\nu}_{M} = {}_ja^{\nu}_{M}. $$

Proof

$$ {}_jc^{\nu}_{M} = \sum_{\psi=0}^{\binom{m}{j}-1} {}_ja^{\nu}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)} = \sum_{\psi=0}^{\binom{m}{j}-1} \varepsilon^{\nu}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)} = {}_ja^{\nu}_{M}. $$

□

Propositions 6.4 and 6.5 are the particular cases of the following proposition.

Proposition 6.6 *The following formula takes place:*

$$ {}_{\varkappa}c^{\nu}_{M} = \binom{m-\varkappa}{m-j} {}_{\varkappa}a^{\nu}_{M}. $$

Proof We can write ${}_{\varkappa}c^{\nu}_{M}$ as follows

$$ {}_{\varkappa}c^{\nu}_{M} = \sum_{\psi=0}^{\binom{m}{j}-1} {}_{\varkappa}a^{\nu}_{f(M,j,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)} = $$

$$ = \sum_{\psi=0}^{\binom{m}{j}-1} \sum_{\chi=0}^{\binom{j}{\varkappa}-1} \varepsilon^{\nu}_{f(f(M,j,\psi),\varkappa,\chi)} \otimes {}_0a^{\nu}_{\bar{f}(f(M,j,\psi),\varkappa,\chi)} \otimes {}_0a^{\nu}_{\bar{f}(M,j,\psi)}. \tag{6.7} $$

We can write ${}_{\varkappa}a^{\nu}_{M}$ as follows:

$$ {}_{\varkappa}a^{\nu}_{M} = \sum_{\psi=0}^{\binom{m}{\varkappa}-1} \varepsilon^{\nu}_{f(M,\varkappa,\psi)} \otimes {}_0a^{\nu}_{\bar{f}(M,\varkappa,\psi)}. \tag{6.8} $$

Since operations "$+$" and "$\otimes$" are commutative, to prove the statement of the proposition it is enough to show that every term of summation (6.8) can be found in summation (6.7) $\binom{m-\varkappa}{m-j}$ times.

Indices $f(f(M,j,\psi),\varkappa,\chi)$ in (6.7) make up multiset B from Section 6.6 and indices $f(M,\varkappa,\psi)$ in (6.8) make up set A from Section 6.6. Therefore, we apply Proposition 6.2 and complete the proof. □

Theorem 6.4 *The mixed high-order moments of the absorbing Markov chain is given by*

$$\varepsilon^{i}_{M} = \varepsilon^{i}_{\nu} \odot \sum_{\varkappa=0}^{m-1} (-1)^{m-\varkappa+1}{}_{\varkappa}a^{\nu}_{M}.$$

Proof Let us assume that the theorem is proven for smaller values of m, particularly for $m = 2$ Theorem 6.3.

We start with the non-tensor representation of the high-order mixed moments, $E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right]$.

Let us calculate $E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right]$. Following the approach of [51, Theorem 3.3.3] we ask where the process can go in one step, from its starting position i. It can go to state φ with probability $p_{i\varphi}$. If the new state is absorbing, then we can never reach states k_j, where $j = \overline{0, m-1}$, again, and the only possible contribution is from the initial state, which is $\prod_{j=0}^{m-1} \delta_{ik_j}$. If the new state is transient then we will be in state k_j δ_{ik_j} times from the initial state, and N_{k_j}, where $j = \overline{0, m-1}$, times from the later steps. We have

$$E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right] = \sum_{\varphi\in\bar{T}} p_{i\varphi} \prod_{j=0}^{m-1} \delta_{ik_j} + \sum_{\varphi\in T} p_{i\varphi} E_\varphi\left[\prod_{j=0}^{m-1} \left(N_{k_j} + \delta_{ik_j}\right)\right] =$$

$$= \sum_{\varphi\in\bar{T}} p_{i\varphi} \prod_{j=0}^{m-1} \delta_{ik_j} +$$

$$+ \sum_{\varphi\in T} p_{i\varphi}\left(E_\varphi\left[\prod_{\kappa\in M} N_\kappa\right] + \sum_{\psi=0}^{\binom{m}{m-1}-1} E_\varphi\left[\prod_{\kappa\in f(M,m-1,\psi)} N_\kappa\right] \prod_{\kappa\in\bar{f}(M,m-1,\psi)} \delta_{i\kappa} +\right.$$

$$+ \cdots + \sum_{\psi=0}^{\binom{m}{j}-1} E_\varphi\left[\prod_{\kappa\in f(M,j,\psi)} N_\kappa\right] \prod_{\kappa\in\bar{f}(M,j,\psi)} \delta_{i\kappa} + \cdots +$$

$$\left. + \sum_{\psi=0}^{\binom{m}{1}-1} E_\varphi\left[\prod_{\kappa\in f(M,1,\psi)} N_\kappa\right] \prod_{\kappa\in\bar{f}(M,1,\psi)} \delta_{i\kappa} + \prod_{\kappa\in M} \delta_{i\kappa}\right).$$

We note that

$$\sum_{\varphi\in\bar{T}} p_{i\varphi} \prod_{j=0}^{m-1} \delta_{ik_j} + \sum_{\varphi\in T} p_{i\varphi} \prod_{\kappa\in M} \delta_{i\kappa} = \prod_{\kappa\in M} \delta_{i\kappa}\left(\sum_{\varphi\in\bar{T}} p_{i\varphi} + \sum_{\varphi\in T} p_{i\varphi}\right) = \prod_{\kappa\in M} \delta_{i\kappa}.$$

And we continue

$$E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right] = \prod_{\kappa\in M} \delta_{i\kappa} + \sum_{\varphi\in T} p_{i\varphi}\left(E_\varphi\left[\prod_{\kappa\in M} N_\kappa\right] + \sum_{j=1}^{m-1} \sum_{\psi=0}^{\binom{m}{j}-1} E_\varphi\left[\prod_{\kappa\in f(M,j,\psi)} N_\kappa\right] \prod_{\kappa\in\bar{f}(M,j,\psi)} \delta_{i\kappa}\right).$$

Let us rewrite the last expression in the tensor form. We will use index ν in place of i for further development. We note that $\prod_{\kappa\in\bar{f}(M,j,\psi)}\delta_{\nu\kappa}$ is represented in the tensor form as $\bigotimes_{\kappa\in\bar{f}(M,j,\psi)}\delta^{\nu}_{\kappa}={}_0a^{\nu}_{\bar{f}(M,j,\psi)}$. Hence, we write

$$\begin{aligned}\varepsilon^{\nu}_{M} &= {}_0a^{\nu}_{M}+q^{\nu}_{\varphi}\odot\left(\varepsilon^{\varphi}_{M}+\sum_{j=1}^{m-1}\sum_{\psi=0}^{\binom{m}{j}-1}\varepsilon^{\varphi}_{f(M,j,\psi)}\otimes{}_0a^{\nu}_{\bar{f}(M,j,\psi)}\right)=\\ &= q^{\nu}_{\varphi}\odot\varepsilon^{\varphi}_{M}+q^{\nu}_{\varphi}\odot\sum_{j=1}^{m-1}{}_jb^{\varphi\nu}_{M}+{}_0a^{\nu}_{M}.\end{aligned}$$

Let us consider $q^{\nu}_{\varphi}\odot\sum_{j=1}^{m-1}{}_jb^{\varphi\nu}_{M}+{}_0a^{\nu}_{M}$ and, in particular, $q^{\nu}_{\varphi}\odot{}_jb^{\varphi\nu}_{M}$ as one term of the summation.

$$q^{\nu}_{\varphi}\odot{}_jb^{\varphi\nu}_{M}=q^{\nu}_{\varphi}\odot\sum_{\psi=0}^{\binom{m}{j}-1}\varepsilon^{\varphi}_{f(M,j,\psi)}\otimes{}_0a^{\nu}_{\bar{f}(M,j,\psi)}.$$

Next we consider $q^{\nu}_{\varphi}\odot\varepsilon^{\varphi}_{f(M,j,\psi)}$ and proceed further by induction.

$$q^{\nu}_{\varphi}\odot\varepsilon^{\varphi}_{f(M,j,\psi)}=q^{\nu}_{\varphi}\odot\varepsilon^{\varphi}_{\mu}\odot\sum_{\varkappa=0}^{j-1}(-1)^{j-\varkappa+1}{}_\varkappa a^{\mu}_{f(M,j,\psi)}.$$

Since $q^{\nu}_{\varphi}\odot\varepsilon^{\varphi}_{\mu}=\varepsilon^{\nu}_{\mu}-\delta^{\nu}_{\mu}$.

$$\begin{aligned}q^{\nu}_{\varphi}\odot\varepsilon^{\varphi}_{f(M,j,\psi)} &= \left(\varepsilon^{\nu}_{\mu}-\delta^{\nu}_{\mu}\right)\odot\sum_{\varkappa=0}^{j-1}(-1)^{j-\varkappa+1}{}_\varkappa a^{\mu}_{f(M,j,\psi)}=\\ &= \varepsilon^{\nu}_{f(M,j,\psi)}-\delta^{\nu}_{\mu}\odot\sum_{\varkappa=0}^{j-1}(-1)^{j-\varkappa+1}{}_\varkappa a^{\mu}_{f(M,j,\psi)}=\\ &= \varepsilon^{\nu}_{f(M,j,\psi)}+\sum_{\varkappa=0}^{j-1}(-1)^{j-\varkappa}{}_\varkappa a^{\nu}_{f(M,j,\psi)}.\end{aligned}$$

Now we come back to $q^{\nu}_{\varphi} \odot_j b^{\varphi\nu}_{M}$.

$$\begin{aligned} q^{\nu}_{\varphi} \odot_j b^{\varphi\nu}_{M} &= \sum_{\psi=0}^{\binom{m}{j}-1} \left(\varepsilon^{\nu}_{f(M,j,\psi)} + \sum_{\varkappa=0}^{j-1} (-1)^{j-\varkappa}{}_{\varkappa}a^{\nu}_{f(M,j,\psi)} \right) \otimes_0 a^{\nu}_{\tilde{f}(M,j,\psi)} = \\ &= {}_j a^{\nu}_{M} + \sum_{\varkappa=0}^{j-1} (-1)^{j-\varkappa} \sum_{\psi=0}^{\binom{m}{j}-1} {}_{\varkappa}a^{\nu}_{f(M,j,\psi)} \otimes_0 a^{\nu}_{\tilde{f}(M,j,\psi)} = \\ &= {}_j a^{\nu}_{M} + \sum_{\varkappa=0}^{j-1} (-1)^{j-\varkappa}{}_{\varkappa j}c^{\nu}_{M} = \\ &= \sum_{\varkappa=0}^{j} (-1)^{j-\varkappa}{}_{\varkappa j}c^{\nu}_{M}. \end{aligned}$$

Now we come back to $q^{\nu}_{\varphi} \odot \sum_{j=1}^{m-1} {}_j b^{\varphi\nu}_{M} + {}_0 a^{\nu}_{M}$.

$$\begin{aligned} q^{\nu}_{\varphi} \odot \sum_{j=1}^{m-1} {}_j b^{\varphi\nu}_{M} + {}_0 a^{\nu}_{M} &= \sum_{j=1}^{m-1} \sum_{\varkappa=0}^{j} (-1)^{j-\varkappa}{}_{\varkappa j}c^{\nu}_{M} + {}_0 a^{\nu}_{M} = \\ &= \sum_{j=1}^{m-1} \sum_{\varkappa=1}^{j} (-1)^{j-\varkappa}{}_{\varkappa j}c^{\nu}_{M} + \sum_{j=1}^{m-1} (-1)^{j}{}_{0j}c^{\nu}_{M} + {}_0 a^{\nu}_{M} = \\ &= \sum_{\varkappa=1}^{m-1} \sum_{j=\varkappa}^{m-1} (-1)^{j-\varkappa}{}_{\varkappa j}c^{\nu}_{M} + \sum_{j=0}^{m-1} (-1)^{j} \binom{m}{j} {}_0 a^{\nu}_{M}. \end{aligned}$$

We note that $\sum_{j=0}^{m} (-1)^j \binom{m}{j} = 0$, and, therefore,

$$\sum_{j=0}^{m-1} (-1)^{j-\varkappa} \binom{m}{j} = (-1)^{m+1}.$$

Let us consider $\sum_{j=\varkappa}^{m-1} (-1)^{j-\varkappa}{}_{\varkappa j}c^{\nu}_{M}$. We recall that ${}_{\varkappa j}c^{\nu}_{M} = \binom{m-\varkappa}{m-j}{}_{\varkappa}a^{\nu}_{M}$ according to Proposition 6.6, and we consider $\sum_{j=\varkappa}^{m-1} (-1)^{j-\varkappa} \binom{m-\varkappa}{m-j}$.

$$\begin{aligned} \sum_{j=\varkappa}^{m-1} (-1)^{j-\varkappa} \binom{m-\varkappa}{m-j} &= \sum_{j=0}^{m-\varkappa-1} (-1)^{j} \binom{m-\varkappa}{m-j-\varkappa} = \sum_{j=0}^{m-\varkappa-1} (-1)^{j} \binom{m-\varkappa}{j} = \\ &= \sum_{j=0}^{m-\varkappa} (-1)^{j} \binom{m-\varkappa}{j} - (-1)^{m-\varkappa} \binom{m-\varkappa}{m-\varkappa} = (-1)^{m-\varkappa+1}. \end{aligned}$$

Hence, for $q^{\nu}_{\varphi} \odot \sum_{j=1}^{m-1} {}_j b^{\varphi\nu}_{M} + {}_0 a^{\nu}_{M}$ we have

$$q_{\varphi}^{\nu} \odot \sum_{j=1}^{m-1} {}_{j}b_{M}^{\varphi\nu} + {}_{0}a_{M}^{\nu} = \sum_{\varkappa=1}^{m-1} (-1)^{m-\varkappa+1} {}_{\varkappa}a_{M}^{\nu} + (-1)^{m+1} {}_{0}a_{M}^{\nu} = \sum_{\varkappa=0}^{m-1} (-1)^{m-\varkappa+1} {}_{\varkappa}a_{M}^{\nu}.$$

And, finally, we obtain

$$\begin{aligned} \varepsilon_{M}^{\nu} &= q_{\varphi}^{\nu} \odot \varepsilon_{M}^{\varphi} + q_{\varphi}^{\nu} \odot \sum_{j=1}^{m-1} {}_{j}b_{M}^{\varphi\nu} + {}_{0}a_{M}^{\nu} = \\ &= q_{\varphi}^{\nu} \odot \varepsilon_{M}^{\varphi} + \sum_{\varkappa=0}^{m-1} (-1)^{m-\varkappa+1} {}_{\varkappa}a_{M}^{\nu}. \end{aligned}$$

Next we consider $\varepsilon_{M}^{\nu} - q_{\varphi}^{\nu} \odot \varepsilon_{M}^{\varphi}$.

$$\varepsilon_{M}^{\nu} - q_{\varphi}^{\nu} \odot \varepsilon_{M}^{\varphi} = (\delta_{\varphi}^{\nu} - q_{\varphi}^{\nu}) \odot \varepsilon_{M}^{\varphi},$$

and multiplying by ε_{ν}^{i} from left, and recalling that $\varepsilon_{\nu}^{i} \odot (\delta_{\varphi}^{\nu} - q_{\varphi}^{\nu}) = \delta_{\varphi}^{i}$ we have

$$\varepsilon_{\nu}^{i} \odot (\delta_{\varphi}^{\nu} - q_{\varphi}^{\nu}) \odot \varepsilon_{M}^{\varphi} = \delta_{\varphi}^{i} \odot \varepsilon_{M}^{\varphi} = \varepsilon_{M}^{i}.$$

We complete the proof with

$$\varepsilon_{M}^{i} = \varepsilon_{\nu}^{i} \odot \sum_{\varkappa=0}^{m-1} (-1)^{m-\varkappa+1} {}_{\varkappa}a_{M}^{\nu}.$$

□

One can see that tensor formalism allows us to calculate the mixed high-order moments by compact formula. The mixed high-order moments are determined by the moments of lower orders.

6.8 Conclusion

We considered the mixed high-order moments of an absorbing Markov chain. While the first moments and the non-mixed seconds moment can be expressed in a matrix form, it can hardly be done for the mixed high-order moments. Using tensor formalism, we developed a compact close-form expression for the mixed high-order moments.

7

CONCLUSIONS

In this thesis, we have addressed problems related to ranking of Web pages. There are a lot of approaches to ranking but we have drawn our attention to hyper-link based ranking technics, particularly, to PageRank invented by the founders of Google corporation.

One of the most important problem in the topic of PageRank is computation. Since the size of the World Wide Web is huge, the calculation of PageRank of all the pages on the Web is very computationally consuming. According to information publicly available, Google uses the Power algorithm to calculate PageRank. Although, the convergence rate of the Power algorithm is controlled by the damping factor, the Power algorithm is still too computationally consuming. In the second chapter we discussed methods to accelerate the calculation of the PageRank vector. We mentioned the adaptive algorithm in which once a PageRank value converged, it is fixed and is not recomputed in further iterations. This simple method speeds up the PageRank computation by 30%. Another algorithm uses an extrapolation to accelerate computations and achieves 38% and 23% of speed up correspondingly by the Aitken and Quadratic extrapolations. The other group of methods uses aggregation-disaggregation approach to make the PageRank computations faster. The main idea of the methods is to divide the set of all the pages on the Web into subsets of smaller size and to calculate vectors related to the subsets. Having the vectors one can reconstruct the vector of PageRank for the whole Web. We sketchy reviewed several aggregation-disaggregation algorithms and considered two of them in details. The first algorithm, BlockRank algorithm, consists of three stages. At the first stage, PageRank is determined separately for each site. It can be done locally by a site. After that, at the second stage, BlockRank is calculated by composing an aggregated matrix with local PageRank vectors of the sites as aggregation vectors. And, at the last stage, global PageRank is composed using local PageRank vectors taken with corresponding Block-

Rank values as weights. It was empirically shown that the BlockRank algorithm is faster than the Power method by a factor of two. Another algorithm, Fast Two-Stage Algorithm, exploits the presence of the dangling pages, lumping them at the first stage and calculating the PageRank vector of the aggregated matrix with lumped dangling pages, and, at the second stage, aggregating other pages. The first stage requiring less computation work than the Power method does and converges at least as fast as the Power method. The second stage usually converges after about three iterations. We have considered in details two methods: the full aggregation-disaggregation method and the partial aggregation-disaggregation method. The methods can be applied both to the PageRank computation and to the computation of the stationary distribution of a general Markov chain. The full aggregation-disaggregation method and partial aggregation method are similar. Having the set of pages decomposed into non-intersecting subsets (two subsets in the case of the partial aggregation-disaggregation method), one constructs an aggregated matrix using the vector resulted from the previous iteration (at the first iteration, any probability distribution). The way, how the aggregated matrix is constructed, ensures that the matrix is a stochastic matrix. After that, one calculates stationary distribution of the aggregated matrix and uses its values as weights to reconstruct the full vector. Then, to obtain the vector resulted from the iteration, one performs several iterations of the Power algorithm. In case of the partial aggregation-disaggregation method, only one subset of the Web pages is aggregated into one state of the aggregated Markov chain, the other subset stays unmodified. The full aggregation-disaggregation converges locally in two cases. Firstly, if the original transition matrix is positive and greater then a matrix whose row are the stationary distribution of the transition matrix multiplied by a positive factor. Secondly, if the original transition matrix has at least one positive column. Both the cases take place for the Google matrix and we can estimate the rate of convergence for the Google matrix. In case of huge graphs, which the Web certainly is, the full aggregation-disaggregation methods converges with the rate estimated as square root of the damping factor. The estimate is worse than for the Power method, but there are examples of graphs fro which the full aggregation-disaggregation method converges faster than the Power method. The partial aggregation-disaggregation method behaves better than the full aggregation-disaggregation method in term of convergence. If the Power method converges, it implies that the partial aggregation-disaggregation methods converges, too. In the case of the Google matrix, there is such a decomposition of the set of the Web pages that the partial aggregation-disaggregation method converges faster than the Power algorithm. The convergence properties of the partial aggregation-disaggregation method make it preferable to the full aggregation-disaggregation method. In the same time, the full aggregation-disaggregation method is less computationally consuming, since it deals with fully aggregated matrix at each iteration which has only two rows and two columns if the set of the Web pages in decomposed into two subsets while the partial aggregation-disaggregation

method deals with partially aggregated matrix which dimensions depends on the number of the pages in the subsets. We can say that the full aggregation-disaggregation method reduces the computation cost at each iteration at the expense of convergence properties. In the second chapter, of the thesis we discovered the conditions when the full aggregation-disaggregation method and the partial aggregation-disaggregation methods are equivalent in the sense that they produce the same sequence of vectors obtained at each iteration. The conditions are the following. The off-diagonal block of the aggregated part of the transition matrix should have rank one. In that case, starting with a specially chosen initial distribution, both the methods give the same sequence of intermediate vectors, which means that they are equivalent. Another observation which was made is that this specially chosen initial distribution can be obtained by the first iteration of the partial aggregation-disaggregation method initiated by any probability distribution. This enable us to formulate a novel algorithm which we called the mixed aggregation-disaggregation algorithm. The mixed aggregation-disaggregation algorithm does its first iteration as the partial aggregation-disaggregation method and the others as the full aggregation-disaggregation method. On the one hand, in the case when the off-diagonal block of the aggregated part of the transition matrix has rank one, the mixed aggregation-disaggregation algorithm produce the same sequence of the intermediate results as the partial aggregation-disaggregation method and, hence, it has the same properties of a convergence as the partial aggregation-disaggregation method. On the other hand, all the iteration excepting the first one are performed by the mixed aggregation-disaggregation algorithm as the full aggregation-disaggregation method, therefore, the mixed aggregation-disaggregation algorithm is almost as computationally consuming as the full aggregation-disaggregation method. Thus, the mixed aggregation-disaggregation algorithms possess the advantages of both the aggregation-disaggregation methods and avoids their drawbacks.

The choice of the damping factor, one of the most important parameter of the PageRank algorithm, is not evident. Although significant attention was attracted to the discussion of the problem, scientist have not arrived to the common conclusion. Google's choice 0.85 as the value of the damping factor is supported by observation that surfing through the Web a user usually does 6 click on the hyper-links and later jumps to an arbitrary page, which is the case for the random surfer model with such a value of the damping factor. Some authors proposed to avoid a particular choice of the damping factor introducing the damping functions, integrating over all the range of the values of the damping factor or considering the damping factor as a random variable. Other authors suggested particular values of the damping factor arguing it in different ways. In the citation network of scientific papers, scientists unlikely follows more than two levels of the references from a paper. This observation leads to 0.5 as the value of the damping factor. The value 0.5 is supported by other arguments, too. The distribution of PageRank among the principal components of the Web graph is more fair if the

damping factor equals to 0.5. In the thesis we explored the centrality measures which can be used instead of PageRank and which are free from the damping factor. We have decomposed the Web graph equipped with artificial links into its principal components: the Extended Giant Strongly Connected Component (ESCC) and the Pure Out Component (POUT). POUT is small in size but if the damping factor is chosen equal to one, the random walk absorbs with probability one into POUT. As we showed in the numerical experiments, a large majority of pages and nearly all important pages are in the ESCC. We also noted that even if the damping factor is chosen close to one, the random walk can spend a significant amount of time in the ESCC before the absorption. Therefore, we drew our attention to the ranking of the Web pages inside the ESCC and suggested to use quasi-stationary distributions for it since they represent the dynamics of the random walk before it leaves the ESCC. We have considered four versions of the quasi-stationary distributions and discovered that they are closed to each other by Kendall's tau metric and angular measure. It allows us to calculate only one of them depending on the computation efficiency. The four quasi-stationary distributions are close to each other, but they are rather different from PageRank vector with the damping factor equal to 0.85.

Although the iterative methods of the PageRank calculation are highly developed, asides from them, there are other probabilistic methods aiming for this purpose. In Chapter 4, we propose and analyze Monte Carlo type methods for the PageRank computation. To the best of our knowledge only in two works the Monte Carlo methods were applied to the PageRank computation. We have considered five versions of the Monte Carlo type methods two of which were proposed in the two works mentioned above. PageRank is a stationary distribution of a particularly specified Markov chain which models behaviour of the random surfer. In general, one run of a Monte Carlo method simulates a random walk on the Web graph sampling the path of the random surfer. The five Monte Carlo methods that we considered differ in the way how they start runs, what data they collect at each run, and how they deal with the dangling pages during a run. Runs can be started at an arbitrary page or be iterated over all the pages, which we call a cyclic start. We can keep either the information about the last visited page or about all the pages visited by the random surfer during a run. During a run, the random surfer can either jump from a dangling page to an arbitrary page on the Web graph or to stop at the page, thus, finishing the run. We have analysed the Monte Carlo methods analytically and concluded that the Monte Carlo methods with cyclic start outperforms the analogous Monte Carlo methods with random start. Further analysis showed that keeping the information about all the visited page during a simulation run leads to better convergence than keeping the information about the last visited page only. The analytical analysis is supported by the experiments made on a real graph. Experiments showed that already after one iteration a good approximation of PageRank can be obtained for popular pages, while the Power methods gives just a weighted sum of in-coming links.

In many cases just the relative ranking of the Web pages playes an essential role and actual values of PageRank are not important. Monte Carlo method applied to Personalized PageRank was analized with the aim to discover the ranking of the number of pages having high Personalized PageRank values either in the order of their Personalized PageRank values or disregarding to the order. We compared three Monte Carlo type methods by their performance using the variance of the estimators of Personalized PageRank that they produce. We provided analitical expressions and estimation of the probabilities with which the Monte Carlo methods correctly reveal a certain number of Web pages with highest values of Personalized PageRank. We carried out a number of numerical experiments to illustrate our theoretical results.

The last chapter was devoted to analysis of the mixed high-order moments of an absorbing Markov chain, which were used in the previous chapter in application to Personalized PageRank and Monte Carlo methods. While the first and the second non-mixed moments of the number of visits of an absorbing Markov chain can be easily expressed in a matrix form, it can be hardly done for the mixed high-order moments. We have applied tensor theory as a generalization of the matrices to the mixed high-order moments and obtained compact closed-form expressions.

A

SUMMARY IN ENGLISH

A.1 Introduction

With the rapid development of the Internet and the World Wide Web, the problem of information retrieval becomes extremely important. Due to the huge size of the Web, the retrieved results of a search are so enormous that the problem of their sorting becomes actual. Among other criteria, the results can be sorted according to their authoritativeness. We shall consider one way to estimate the authoritativeness of the Web pages based on the hyper-link structure of the Web, namely, the *PageRank algorithm* [71, 27]. The major idea of the method is that authoritativeness of a Web page depends on the number and the quality of hyper-links to the page placed on other Web pages. A hyper-link to a Web page is called an in-coming link. Intuitively, the larger the number of in-coming links to a Web page is, the higher the authoritativeness of the page is. But the authoritativeness of the Web page where the in-coming link is placed also plays an important role. In contrast to the scientific citation index, for example, the PageRank algorithm takes it into account.

Let us introduce the PageRank algorithm formally. We consider the Web as a directed graph. A Web page is a node and a hyper-link is an arc where the tail of the arc is the Web page where the hyper-link is placed and the head of the arc is the Web page which the hyper-link refers to. This directed graph is called the *Web Graph*. We shall use term "page" and "node" interchangeably in the further discussion. Let us assume that there are n Web pages on the Web and let us enumerate the Web pages by integer numbers from 1 to n and define the $n \times n$

hyperlink matrix H such that

$$h_{ij} = \begin{cases} 1/d_i, & \text{if page } i \text{ links to } j, \\ 0, & \text{otherwise,} \end{cases} \tag{A.1}$$

for $i, j = \overline{1, n}$, where d_i is the number of out-going links from page i. If a page has no out-going links then the page is called a *dangling page* and the row of matrix H corresponding to it is the zero row. We fix it by assuming that a dangling page refers to all the pages on the Web. These imaginary links are called *artificial links*. Let us introduce a column vector a which element $a_i = 1$ if the i^{th} row of matrix H corresponds to a dangling page and 0 otherwise. Let us define a stochastic matrix P by

$$P = H + av, \tag{A.2}$$

where v is a uniform probability distribution. We define a stochastic matrix which is called the *Google matrix* and expresses as

$$G = cP + (1 - c)\mathbf{1}v, \tag{A.3}$$

where $\mathbf{1}$ is a column vector of appropriate dimension whose all the entries are equal to one and $0 < c < 1$. The PageRank vector is defined as eigenvector of matrix G corresponding to its principal eigenvalue.

$$\pi = \pi G, \tag{A.4a}$$

$$\pi\mathbf{1} = 1, \tag{A.4b}$$

The *PageRank vector* can be found from (A.4) as [21, 58, 68]

$$\pi = v(1 - c)(I - cP)^{-1}, \tag{A.5}$$

where I is an identity matrix.

One can see that if we consider the Google matrix as a transition matrix of a Markov chain, then PageRank is a stationary probability distribution of the Markov chain. Let us imagine a surfer on the Web staying at one of the Web pages. With probability $1 - c$ she jumps to an arbitrary Web page and, with probability c, she chooses to follow one of the out-going links of the page where she is at the moment. The particular out-going link is chosen uniformly from the set of out-going links the current Web page. If we imagine that a lot of surfers spread uniformly on the Web follow the behaviour described above, then the number of the surfers on a Web page will be proportional to its PageRank value after a while. It implies that the probability to find a surfer on a Web page is the PageRank value of the Web page. It is clear that the higher the PageRank value is, the more visited the page is. Thus, PageRank can be considered as a result of a particular random walk on the Web Graph and the result is a centrality measure defined on the Web Graph that determines the relative importance of a node within the graph.

Above we assumed that the surfer has not any preference in choosing a Web page when she jumps onto an arbitrary page, but this model is not adequate. One user can prefer sport, another - news, and a third - arts. We can take into account such preference by vector v assuming not a uniform, but general distribution. Vector v is called a *personalization vector*. PageRank is called, in this case, *Personalized PageRank*.

Calculation of PageRank is very computationally consuming since the dimension of the PageRank vector is huge. The dimension of PageRank is the number of indexed Web pages on the Web. The estimation from November 1997 until March 2009 moves from 200 million documents to 25 billion. The dimension size of PageRank makes impractical the using of direct methods, such as Gauss elimination, to determine PageRank. Some approximating methods have to be used.

According to information which is available publicly, Google is using the Power method to calculate PageRank [71]. Starting from the initial approximation as the uniform distribution vector $\pi^{(0)} = (1/n)\mathbf{1}^T$, the k^{th} approximation vector is calculated by

$$\pi^{(k)} = \pi^{(k-1)}G, \quad k \geqslant 1. \tag{A.6}$$

The method stops when the required precision ε is achieved. The number of flops needed for the method to converge is of the order $\frac{\log \varepsilon}{\log c}\, nnz(P)$, where $nnz(P)$ is the number of non-zero elements of the matrix P [58].

A.2 Aggregation-disaggregation methods for PageRank calculation

PageRank calculation is a very computationally expensive operation. Because of direct methods are very time consuming [76, §2], Google uses the Power method to compute the PageRank vector [71], but the convergence rate can be low [57]. Some accelerating methods were proposed in [49, 50, 44, 59, 48, 64]. The authors of [49, 50] accelerated the computation of PageRank vector by modifications of the Power method, while the authors of [44, 59, 48, 64] used aggregation-disaggregation approach.

One of the contributions of the thesis is the equivalence condition of two aggregation-disaggregation methods that allows one to compose a novel method possessing advantages of the mentioned methods and avoiding their drawbacks.

We consider aggregation-disaggregation methods below applied to a general Markov chain and its transition matrix with its stationary distribution. For the consideration of particular case of PageRank, an interested reader is referred to the thesis. Aggregation-disaggregation methods (A/D methods) for the computation of the stationary distribution use the decomposition of the state space which we denote by $\mathcal{I}$. Let us assume that the set $\mathcal{I}$ is decomposed into two non-intersecting sets $\mathcal{I}^{(i)}$, $i = 1, 2$ (the general case of the decompostion into finite number of the

sets is considered in the thesis), such that

$$\begin{aligned} \mathcal{I}^{(1)} &= \{1, \ldots, n_1\}, \\ \mathcal{I}^{(2)} &= \{n_1 + 1, \ldots, n_1 + n_2\}, \end{aligned} \tag{A.7}$$

where $n_1 + n_2 = n$.

According to the decomposition of the states space, the transition matrix can also be partitioned as follows:

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}, \tag{A.8}$$

where P_{ij} is a block with dimension $n_i \times n_j$. Following the partitioning of the transition matrix, its stationary distribution is partitioned into components:

$$\pi = (\pi_1, \pi_2), \tag{A.9}$$

where π_i is a row vector with $\dim(\pi_i) = n_i$.

All aggregation methods use an *aggregated matrix* A. The matrix A is a matrix whose each element corresponds to a block of matrix P, i.e., $a_{ij} \leftrightarrow P_{ij}$. Typically, the elements of the matrix A are formed as $a_{ij} = \zeta_i P_{ij} \mathbf{1}$, where ζ_i is a probability distribution vector. We call the vector ζ_i the *aggregation vector*. Each aggregation method forms the aggregation matrix in its own way using different probability distributions as aggregation vectors and different partitioning. One can consider the aggregated matrix as a transition matrix of a Markov chain with its state space formed by subsets of the state space $\mathcal{I}$.

Aggregation-disaggregation algorithms

We shall review only two aggregation-disaggregation methods. An interested reader is referred to the thesis for a detailed review of other methods.

Full aggregation-disaggregation method (FAM). Determine an approximation $\pi^{(k)}$ to stationary distribution π of stochastic matrix P in k iterations.

1. Select a vector $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}\right)$ with $\pi^{(0)}\mathbf{1} = 1$.
2. Do $k = 0, 1, 2 \ldots$
   (a) Normalize $\sigma_i^{(k)} = \pi_i^{(k)} / \|\pi_i^{(k)}\|_1$, $i = 1, 2$.
   (b) Form aggregated matrix $A^{(k)}$

$$A^{(k)} = \begin{pmatrix} \sigma_1^{(k)} P_{11}\mathbf{1} & \sigma_1^{(k)} P_{12}\mathbf{1} \\ \sigma_2^{(k)} P_{21}\mathbf{1} & \sigma_2^{(k)} P_{22}\mathbf{1} \end{pmatrix}.$$

(c) Determine stationary distribution $\nu^{(k)}$ of $A^{(k)}$

$$\nu^{(k)} = \nu^{(k)} A^{(k)}.$$

(d) Determine disaggregated vector $\widetilde{\pi}^{(k)}$

$$\widetilde{\pi}^{(k)} = \left(\nu_1^{(k)}\sigma_1^{(k)}, \nu_2^{(k)}\sigma_2^{(k)}\right).$$

(e) Do l steps of the Power method

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^l.$$

Partial aggregation-disaggregation method (PAM). Determine an approximation $\pi^{(k)}$ to stationary distribution π of stochastic matrix P in k iterations.

1. Select a vector $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}\right)$ with $\pi^{(0)}\mathbf{1} = 1$.
2. Do $k = 0, 1, 2 \ldots$
   (a) Normalize $\sigma_2^{(k)} = \pi_2^{(k)} / \|\pi_2^{(k)}\|_1$.
   (b) Form aggregated matrix $A_1^{(k)}$

$$A_1^{(k)} = \begin{pmatrix} P_{11} & P_{12}\mathbf{1} \\ \sigma_2^{(k)} P_{21} & \sigma_2^{(k)} P_{22}\mathbf{1} \end{pmatrix}.$$

   (c) Determine stationary distribution $\alpha^{(k)}$ of $A_1^{(k)}$

$$\alpha^{(k)} = \alpha^{(k)} A_1^{(k)}.$$

   (d) Partition $\alpha^{(k)}$

$$\alpha^{(k)} = (\omega_1^{(k)}, \rho^{(k)}).$$

   (e) Determine disaggregated vector $\widetilde{\pi}^{(k)}$

$$\widetilde{\pi}^{(k)} = \left(\omega_1^{(k)}, \rho^{(k)}\sigma_2^{(k)}\right).$$

   (f) Do l steps of the Power method

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^l.$$

When $\mathrm{rank} P_{21} = 1$, two theorems are proven in the thesis stating that the two above algorithm are equivalent in the sense that they produce the same sequence of intermediate results. These theorems allow us to formulate new method of finding stationary distribution

π, which perform its first iteration as the partially aggregation-disaggregation method and all the other iterations as the full aggregation-disaggregation method, which we call the mixed aggregation-disaggregation algorithm. On the one hand, in the case when the off-diagonal block of the aggregated part of the transition matrix has rank one, the mixed aggregation-disaggregation algorithm produce the same sequence of the intermediate results as the partial aggregation-disaggregation method and, hence, it has the same properties of a convergence as the partial aggregation-disaggregation method. On the other hand, all the iteration excepting the first one are performed by the mixed aggregation-disaggregation algorithm as the full aggregation-disaggregation method, therefore, the mixed aggregation-disaggregation algorithm is almost as computationally consuming as the full aggregation-disaggregation method. Thus, the mixed aggregation-disaggregation algorithms possess the advantages of both the aggregation-disaggregation methods and avoids their drawbacks.

A.3 Quasi-Stationary Distributions as Centrality Measures for the Giant Strongly Connected Component of a Reducible Graph

The choice the value of the damping factor which is an essential input parameter in the PageRank algorithm is an important problem having no evident solution so far. If someone fixes the damping factor equal to 0, she gets the uniform distribution as ranking of the Web pages. Obviously, it does not make any sense. In the same time, as the other extreme case, choosing the damping factor equal to unity is at the same level of rationality, since, in that case, the rank tends to concentrate at few pages called *rank sinks*.

We explore parameter-free centrality measures. Here we suggest centrality measures which take as an input only the adjacency list of a graph.

The Web Graph can be divided into principal components. If the artificial links from dangling nodes are taken into account, it is shown in [14] that the Web Graph can be divided into two components: the Extended Strongly Connected Component (ESCC) and Pure OUT component (POUT). the ESCC is the biggest strongly connected component of the Web Graph. All the other strongly connected components are of several magnitudes smaller than the ESCC. POUT is small in size but if the damping factor c is chosen equal to one, the random walk absorbs with probability one into POUT. As we show in the numerical experiments section a large majority of pages and nearly all important pages are in the ESCC. We also note that even if the damping factor is chosen close to one, the random walk can spend a significant amount of time in the ESCC before the absorption. Therefore, for ranking Web pages from the ESCC we suggest the use of quasi-stationary distributions [33, 73], since they represent the dynamics of the random walk before it leaves the ESCC.

As noted in [14], by renumbering the nodes, the transition matrix P can be transformed to

the following form

$$P = \begin{pmatrix} Q & 0 \\ R & T \end{pmatrix},$$

where the block T corresponds to the ESCC, the block Q corresponds to POUT, and the block R corresponds to the transitions from the ESCC to the nodes in POUT. Since matrix T corresponds to the ESCC, it is irreducible. Denote by π_T the part of the PageRank vector corresponding to the ESCC. Using formula (A.5) we conclude that

$$\pi_T(c) = \frac{1-c}{n}\mathbf{1}^T(I - cT)^{-1},$$

where $\mathbf{1}$ is a vector of ones of an appropriate dimension. Let us define the renormalized part of the PageRank vector corresponding to the ESCC:

$$\hat{\pi}_T(c) = \frac{\pi_T(c)}{\|\pi_T(c)\|_1}. \tag{A.10}$$

We note that this renormalization does not alter the rank among the nodes inside the ESCC.

We define four quasi-stationary distributions and provide intuitive explanations clarifying their meaning.

Definition 11 *The pseudo-stationary distribution $\hat{\pi}_T$ is given by*

$$\hat{\pi}_T = \frac{\mathbf{1}^T[I - T]^{-1}}{\mathbf{1}^T[I - T]^{-1}\mathbf{1}}.$$

The i^{th} component of $\hat{\pi}_T$ can be interpreted as a fraction of time the random walk (with $c = 1$) spends in node i prior to absorption.

Proposition A.1 *The following limit exists*

$$\hat{\pi}_T = \lim_{c \to 1} \hat{\pi}_T(c),$$

and the ranking of pages in the ESCC provided by the PageRank vector converges to the ranking provided by $\hat{\pi}_T$ as the damping factor goes to one. Moreover, these two rankings coincide for all values of c above some value c^*.

Definition 12 *The quasi-stationary distribution $\tilde{\pi}_T$ is defined by equation*

$$\tilde{\pi}_T T = \lambda_1 \tilde{\pi}_T, \qquad \tilde{\pi}_T \mathbf{1} = 1,$$

where λ_1 is the Perron-Frobenius eigenvalue of matrix T.

The quasi-stationary distribution $\bar{\pi}_T$ can be interpreted as a proper initial distribution on the non-absorbing states (states in the ESCC) which is such that the distribution of the random walk, conditioned on the non-absorption prior time t, is independent of t [35].

Denote by $\bar{T}$ the hyper-link matrix associated with the ESCC when the links leading outside of the ESCC are neglected. Clearly, we have

$$\bar{T}_{ij} = \frac{T_{ij}}{[T\mathbf{1}]_i},$$

where $[T\mathbf{1}]_i$ denotes the i^{th} component of the vector $T\mathbf{1}$.

Definition 13 *The quasi-stationary distribution $\bar{\pi}_T$ is defined by equation*

$$\bar{\pi}_T\bar{T} = \bar{\pi}_T, \qquad \bar{\pi}_T\mathbf{1} = 1.$$

The $\bar{T}_{ij}$ entry of the matrix $\bar{T}$ can be viewed as a conditional probability to jump from node i to node j under the condition that the random walk does not leave the ESCC at the jump. Then, $\bar{\pi}_T$ can be interpreted as the stationary distribution of the random walk under the above condition.

We can generalize the notion of $\bar{\pi}_T$. Namely, we consider the situation when the random walk stays inside the ESCC after some finite number of jumps N. An interested reader is referred to the thesis for the details. Let us now consider the limiting case, when N goes to infinity. We shall refer to the following limit as the twisted kernel

$$\check{T}_{ij} = \lim_{N\to\infty} \frac{T_{ij}T_j^{(N-1)}\mathbf{1}}{T_i^{(N)}\mathbf{1}}. \tag{A.11}$$

The existence is proven in the thesis.

Definition 14 *The quasi-stationary distribution $\check{\pi}_T$ is defined as the stationary distribution of the twisted kernel. Namely, it is the solution of the following eigenvector equation and normalization condition:*

$$\check{\pi}_T = \check{\pi}_T\check{T}, \qquad \check{\pi}_T\mathbf{1} = 1.$$

If we assume aperiodicity in addition, $\check{T}_{ij}$ can be given the interpretation of the probability of transition from i to j in the ESCC for the chain, conditioned on the fact that it never leaves the ESCC.

Analytic analysis made in the thesis allows us to conclude that the considered quasi-stationary distributions are close to each other which supported by the numerical experiments.

Numerical experiments

For our numerical experiments we have used the Web site of INRIA (http://www.inria.fr, the dataset is available from the author upon request). It is a typical Web site with about 300.000 Web pages and 2.200.000 hyper-links. In our experiments we compute $\bar{\pi}_T$, $\tilde{\pi}_T$, $\hat{\pi}_T$, and $\check{\pi}_T$ with 5-digit precision. Also we compute $\hat{\pi}_T(0.85)$ which is the normalized PageRank vector of the ESCC with the damping factor equal to 0.85. For each pair of these vectors we calculated Kendall's τ metric. Kendall's τ metric shows how two rankings are different in terms of the number of swaps which are needed to transform one ranking to the other. Kendall's τ metric has the value of one if two rankings are identical and minus one if one ranking is the inverse of the other.

In our experiments, Kendall's τ metric for all the pairs is very close to one. Thus, we conclude that all the four quasi-stationarity based centrality measures produce very similar rankings.

We have also analyzed Kendall's τ metric between $\tilde{\pi}_T$ and PageRank of the ESCC as a function of the damping factor (see Figure A.1). As c goes to one, Kendall's τ approaches one. This is in agreement with Proposition A.1.

We have also compared the ranking produced by the quasi-stationary distributions and PageRank of the ESCC using the θ rank correlation measure. The measure is defined as follows

$$\theta_i = \arctan(r_i^1/r_i^2),$$

where r_i^1 is ranking of node i in a vector and r_i^2 is ranking of the same node i in an other vector. By the term ranking, we mean here the place of node i in a vector if we sort the entries of the vector in decreasing order. If ranking of node i is the same in both the vectors, then θ_i is equal to $\pi/4$. As one can see from Figure A.1, cumulative distribution over θ_i corresponding to any quasi-stationary distributions is close to vertical line at $\pi/4$ which means that rankings produced by the vectors are close to each other.

A.4 Monte Carlo methods in PageRank computation: When one iteration is sufficient

Although the iterative methods of the PageRank calculation are highly developed, asides from them, there are other probabilistic methods aiming for this purpose. Here we study Monte Carlo (MC) type methods for the PageRank computation. To the best of our knowledge, only in two works [25, 37] the Monte Carlo methods are applied to the PageRank computation. The principal advantages of the probabilistic Monte Carlo type methods over the deterministic methods are: the PageRank of important pages is determined with high accuracy already after

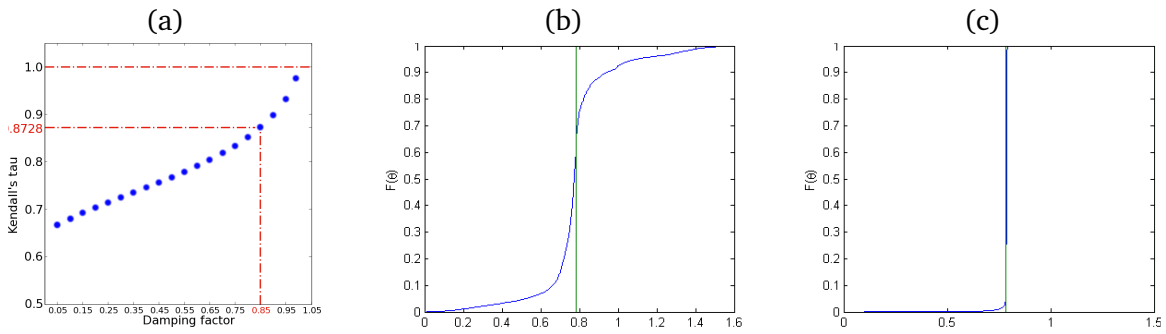


Figure A.1: (a) Kendall's τ metric between $\tilde{\pi}_T$ and PageRank of the ESCC $\hat{\pi}_T(c)$ as a function of the damping factor. The cumulative distribution of θ rank correlation measure: (b) $\hat{\pi}_T(0.85)$ and $\bar{\pi}_T$, (c) $\bar{\pi}_T$ and $\hat{\pi}_T$.

the first iteration; MC methods have natural parallel implementation; and MC methods allow continuous update of the PageRank as the structure of the Web changes.

One can suggest the following algorithm employed in [25].

Algorithm A.1 (MC end-point with random start) *Simulate* N *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at a randomly chosen page. Evaluate* π_j *as a fraction of* N *random walks which end at page* $j = \overline{1,n}$.

The above algorithm produces unnecessary randomness by random choice of a starting page. The starting page can be chosen iteratively. This results in the following algorithm whose version was used in [37] for computing Personalized PageRank.

Algorithm A.2 (MC end-point with cyclic start) *Simulate* $N = mn$ *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at each page exactly* m *times. Evaluate* π_j *as a fraction of* N *random walks which end at page* $j = \overline{1,n}$.

The two above algorithms keep only the information about the last page which is visited by the simulated random walks. We can construct an algorithm which takes into account all the visited pages during a random walk.

Algorithm A.3 (MC complete path) *Simulate the random walk* $\{X_t\}_{t\geqslant 0}$ *exactly* m *times from each page. For any page* i, *evaluate* π_j *as the total number of visits to page* j *multiplied by* $(1-c)/(n*m)$.

The simulated by the above algorithm random walk jumps onto an arbitrary page arriving to a dangling page. Obviously, it decouples the two parts of a random walk before the visit of a dangling page and after it. In this case, it is can be better to stop at a dangling page, which leads to another algorithm.

Algorithm A.4 (MC complete path stopping at dangling nodes) *Simulate the random walk* $\{Y_t\}_{t\geqslant 0}$ *starting exactly* m *times from each page. For any page* j*, evaluate* π_j *as the total number of visits to page* j *divided by the total number of visited pages.*

We note that the complete path versions of the Monte Carlo methods also admit a random start. The corresponding algorithm is as follows.

Algorithm A.5 (MC complete path with random start) *Simulate* N *samples of the random walk* $\{Y_t\}_{t\geqslant 0}$ *started at a random page. For any page* j*, evaluate* π_j *as the total number of visits to page* i *divided by the total number of visited pages.*

We concluded that the MC algorithms with cyclic start are preferable to the analogous the MC algorithms with random start. We thoroughly analyzed and compare the MC complete path stopping at dangling nodes with the MC end-point. We showed that under natural conditions the MC complete path stopping at dangling nodes outperforms the MC end-point. For the supporting analytical arguments, a reader is referred to the thesis.

Numerical experiments

For our numerical experiments, we have taken the Web site of INRIA Sophia Antipolis `http://www-sop.inria.fr/`. It is a typical Web site with about 50.000 Web pages and 200.000 hyperlinks. First, we performed the sufficient number of the power iterations to obtain the value of PageRank with 20-digit accuracy. We sorted the PageRank vector in the decreasing order and plotted it in the loglog scale (see Figure A.2).

We have performed 10 iterations of the PI method and 10 iterations of the three implemented MC algorithms. In Figure A.2, we compare the results of 10 iterations of the PI method and the MC complete path stopping in dangling nodes method for the 1000^{th} by the PageRank value page. Indeed, already the first iteration of the MC complete path stopping in dangling nodes algorithm gives a small error for the Web page.

Next, in Figure A.2, we compare three versions of the Monte Carlo method: the MC complete path stopping in dangling nodes, the MC end-point with cyclic start, and the MC complete path with random start. We plotted actual relative error and the estimated 95% confidence intervals. It turns out that on our dataset the MC complete path stopping in dangling nodes performs the best, followed by the MC complete path with random start. The MC end-point with cyclic start has the worst performance.

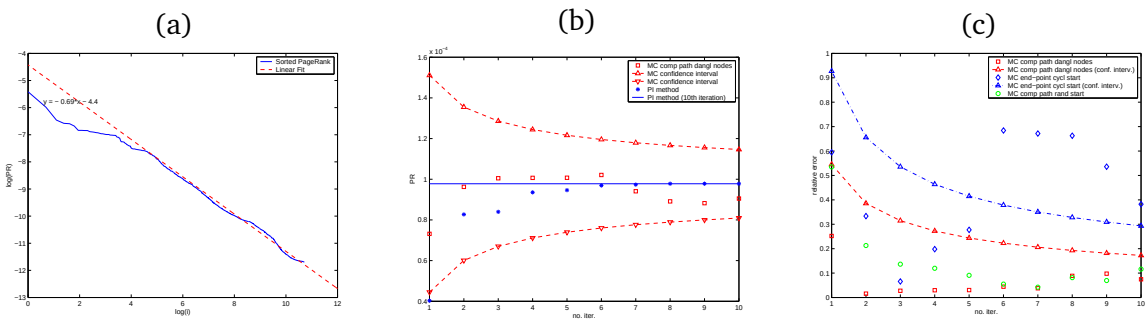


Figure A.2: (a) Sorted PageRank in loglog scale. (b) PI vs. MC comp path dangl nodes: π_{1000}. (c) Comparison of MC algorithms: π_{1000}.

A.5 Finding top-k lists with Monte Carlo Personalized PageRank

Personalized PageRank or Topic-Sensitive PageRank [41] has a number of applications. In the original paper [41], Personalized PageRank was used to introduce the personalization in the Web search. In [29, 85], Personalized PageRank was suggested for finding related entities. In [70], Green measure which is closely related to Personalized PageRank, was suggested for finding related pages in Wikipedia. In [4, 5], Personalized PageRank was used for finding local cuts in graphs and, in [11], the Personalized PageRank was applied for the clustering large hyper-text document collections. In all the above mentioned applications, one needs to find nodes with reasonably high values of Personalized PageRank. As was shown in [10], and presented in the previous chapter, the Monte Carlo methods are efficient for the estimation of PageRank for popular pages. Following up [10], we propose to use Monte Carlo methods for finding top lists of pages with large values of Personalized PageRank.

We consider Personalized PageRank with the particular personalization vector v in equation (A.2) and equation (A.3) equal to $\mathbf{1}_i^T$ without lose of generality due to the linearity of PageRank.

We consider three Monte Carlo methods simulating random walks over the Web graph. All the Monte Carlo methods produce estimators of Personalized PageRank. The first Monte Carlo method takes into account the nodes where the random walks stop.

Algorithm A.6 (MC End Point) *Simulate* m *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at node* i. *Evaluate* π_j *as a fraction of* m *random walks which end at node* $j = \overline{1,n}$.

The second Monte Carlo method takes into account all the nodes visited by a random walk and finds estimation of Personalized PageRank as the ratio of the number of visits to the expected number of transitions made during the random walk.

Algorithm A.7 (MC Complete Path) *Simulate* m *runs of the random walk* $\{X_t\}_{t\geqslant 0}$ *initiated at node* i*. Evaluate* π_j *as the total number of visits to node* j *multiplied by* $(1-c)/m$*.*

The last Monte Carlo method takes into account all the nodes visited by a random walk but divides the number of visits by the actual number of transitions performed during the random walk.

Algorithm A.8 (MC Complete Path Transition Count) *Simulate* τ *steps of the random walk* $\{Y_t\}_{t\geqslant 0}$ *governed by the Google matrix. Evaluate* π_j *as the number of visits to node* j *divided by the total number of steps* τ*.*

As outputs of the proposed algorithms we would like to obtain with high probability either a *top*-k *list* of nodes or a *top*-k *basket* of nodes.

Definition 15 *The* top-k list *of nodes is a list of* k *nodes with largest Personalized PageRank values arranged in a descending order of their Personalized PageRank values.*

Definition 16 *The* top-k basket *of nodes is a set of* k *nodes with largest Personalized PageRank values with no ordering required.*

We take the following technical assumption which is not restrictive and is satisfied in most practical applications.

Assumption A.1 *We assume that* $\pi_1 > \pi_2 > \ldots > \pi_k > \pi_{k+1} \geqslant \pi_j$ *for* $j \geqslant k+2$*.*

It turns out that it is beneficial to relax our goal and to obtain a top-k basket with a small number of erroneous elements.

Definition 17 *We call* relaxation-l *top*-k *basket a realization when we allow at most* l *erroneous elements from top*-k *basket.*

We compared three Monte Carlo type methods by their performance using the variance of the estimators of Personalized PageRank that they produce. We calculate analytically ranking probabilities which give us the level of certainty that the top-k list or the top-k basket are revealed correctly. We also provide estimation based on Bonferroni inequalities and Monte Carlo methods. In particular, we showed that the ranking probabilities converge exponentially. We considered a relaxation of top-k basket by allowing some number of erroneous elements. This relaxed top-k basket is significantly easier to detect. We carried out a number of numerical experiments to illustrate our theoretical results.

A.6 Tensor approach to mixed high-order moments of absorbing Markov chains

In the mathematical analysis used in the consideration of Monte Carlo methods for Personalized PageRank, mixed high-order moments of the number of visits are exploited. Since the mixed high-order moments are hard to express in a matrix form, we used a tensor approach to calculate them. Compact close-form formulae are obtained in the thesis.

Introduction

Let us consider an absorbing Markov chain and let matrix P be its transition matrix. By renumbering the states we can decompose matrix P in the following way:

$$P = \begin{pmatrix} I & 0 \\ S & Q \end{pmatrix},$$

where submatrix Q is a substochastic matrix corresponding to transient states. Let T be the set of the transient states and $\bar{T}$ be the set of the absorbing states. We can define a fundamental matrix Z of the absorbing Markov chain as

$$Z = (I - Q)^{-1} = I + Q + Q^2 + \dots.$$

Fundamental matrix $Z = \{z_{ij}\}_{i,j\in T}$ has the following probabilistic interpretation.

Definition 18 *Define* N_j *to be a function giving the total number of times before absorption that the absorbing Markov chain visits a transient state* j.

Let us denote by $E_i[N_j]$ the first moment of function N_j assuming that the Markov chain starts at state i, where $i, j \in T$. Then

$$Z = \{E_i[N_j]\}_{i,j\in T}$$

as it is noted in [51, Theorem 3.2.4]. The non-mixed second moments $E_i[N_j^2]$ can also be found [51, Theorem 3.3.3] with the help of matrix Z as

$$\left\{E_i\left[N_j^2\right]\right\}_{i,j\in T} = Z\left(2Z_{dg} - I\right),$$

where Z_{dg} is the same matrix as Z, but all the off-diagonal elements are set to zero.

However, the mixed second moments $E_i[N_j N_k]$ and the mixed higher-order moments $E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right]$ are not so easy to calculate. Here we address this problem by tensor approach.

Introduction in tensors

We give a brief introduction to basic facts from the tensor theory which we shall use in the further sections. We do not present the tensor theory in its completeness, we just define what we need for our application to the mixed high-order moments. A interested reader is referred to [74, 63] for more details.

Let us introduce tensor operations which we need for further development. Tensor product $\otimes$ of a tensor $\mathcal{A}$ which is n-times contravariant and m-times covariant and a tensor $\mathcal{B}$ which is s-times contravariant and t-times covariant is a tensor $\mathcal{C}$ which is $n+s$-times contravariant and $m+t$-times covariant (A.12).

$$\mathcal{A} \otimes \mathcal{B} = \mathcal{C}, \tag{A.12}$$

where components of tensor $\mathcal{C}$ in some basis can be found by formula

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{p_1 p_2 \ldots p_s}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n p_1 p_2 \ldots p_s}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t},$$

where indices $i_1, i_2, \ldots, i_n$,$k_1, k_2, \ldots, k_m$,$p_1, p_2, \ldots, p_s$ and $h_1, h_2, \ldots, h_t$ take all the possible values. Further we shall write tensor product $\otimes$ as

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \otimes b^{p_1 p_2 \ldots p_s}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n p_1 p_2 \ldots p_s}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t},$$

assuming that indices $i_1, i_2, \ldots, i_n$,$k_1, k_2, \ldots, k_m$,$p_1, p_2, \ldots, p_s$ and $h_1, h_2, \ldots, h_t$ take all the possible values.

In some cases, we need to consider only the components of tensors having the same indices in the tensor product (A.13).

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \otimes b^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t} = a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t}. \tag{A.13}$$

Also let us define tensor contraction $\odot$ by formula in (A.14).

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_m} a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t}. \tag{A.14}$$

We note that tensor contraction is equivalent to matrix product if the matrices are written in one-time contravariant and one-time covariant components.

We shall use tensors and the tensor operations in application to the mixed second moments and the mixed high-order moments.

Mixed high-order moments

Let us now consider the mixed moments of higher order. Before we formulate the mixed high-order moments in tensor formalism, let us prove that the moments are finite.

Let us consider the conditional moment generating function of the absorbing Markov chain, $M_i(y) = E_i\left[e^{\sum y_j N_j}\right]$, where summation is performed over all states of the Markov chain and the process starts at a transient state i. We need to prove that the moment generating function is analytical at the origin.

Let us define vector ζ and matrix $\Theta = \{\vartheta_{ik}\}_{i,k\in T}$

$$\begin{aligned} \zeta_i &= e^{y_i}\left(1 - \sum_{k\in T} p_{ik}\right), \\ \vartheta_{ik} &= \delta_{ik} - e^{y_i} p_{ik}. \end{aligned}$$

Proposition A.2 *If all y_i are small enough, moment generating function $M(y)$ is give by*

$$M(y) = \Theta^{-1}\zeta.$$

One can see that the conditional moment generating function $M(y)$ is analytical at the origin, and, hence, there exist all the mixed high-order moments and they are finite.

We denote

$$\begin{aligned} \varepsilon^i_j &= E_i\left[N_j\right], \\ \varepsilon^i_{jk} &= E_i\left[N_j N_k\right], \ldots, \\ \varepsilon^i_{k_0 k_1 \ldots k_{m-1}} &= E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right], \end{aligned}$$

where m is a natural number. Let us denote $M = \{k_0, k_1, \ldots, k_{m-1}\}$. The cardinality of set M is m.

Let us give the result for the mixed second moments, firstly.

Theorem A.1 *The mixed second moments are given by*

$$\varepsilon^i_{jk} = \varepsilon^i_\nu \odot \left(\varepsilon^\nu_j \otimes \delta^\nu_k + \varepsilon^\nu_k \otimes \delta^\nu_j - \delta^\nu_k \otimes \delta^\nu_j\right).$$

We have got the mixed moments of higher order in tensor representation. Since the product is a commutative operation, the order of indices $k_0 k_1 \ldots k_{m-1}$ in $\varepsilon^i_{k_0 k_1 \ldots k_{m-1}}$ does not matter, and we can write $\varepsilon^i_{k_0 k_1 \ldots k_{m-1}} = \varepsilon^i_M$.

Let us denote ${}_0a^\nu_{k_0 k_1 \ldots k_{m-1}} = \bigotimes_{\iota=0}^{m-1} \delta^\nu_{k_\iota}$. Let us define tensor ${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}}$ as follows:

$${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}} = \sum_{\psi=0}^{\binom{m}{j}-1} \varepsilon^\nu_{f(M,j,\psi)} \otimes {}_0a^\nu_{\bar{f}(M,j,\psi)}.$$

Since index ψ passes all possible values, the order of indices $k_0 k_1 \ldots k_{m-1}$ in ${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}}$ does not matter, and we can write ${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}} = {}_ja^\nu_M$. We note that ${}_ma^\nu_M = \varepsilon^\nu_M$.

Theorem A.2 *The mixed high-order moments of the absorbing Markov chain is given by*

$$\varepsilon_M^i = \varepsilon_\nu^i \odot \sum_{\varkappa=0}^{m-1} (-1)^{m-\varkappa+1}{}_\varkappa a_M^\nu.$$

One can see that tensor formalism allows us to calculate the mixed high-order moments by a compact formula. The mixed high-order moments are determined by the moments of lower orders.

B

PRÉSENTATION DES TRAVAUX DE THÈSE EN FRANCAIS

B.1 Introduction

Avec le développement rapide de l'Internet et le World Wide Web, le problème de recherche d'information devient extrêmement important. En raison de la taille du Web, les résultats trouvés d'une recherche sont tellement énormes que le problème de leur tri se pose. Parmi d'autres critères, les résultats peuvent être triés selon à leur autorité. Nous allons examiner une façon d'estimer l'autorité des pages Web en fonction de la structure hyper-lien du Web, à savoir, *l'algorithme PageRank* [71, 27]. L'idée majeure de la méthode est que l'autorité d'une page Web dépend du nombre et de la qualité des hyper-liens vers la page placés sur d'autres pages Web. Un hyper-lien vers une page Web est appelé un lien entrant. Intuitivement, plus le nombre de liens entrants à une page Web est élevé, le plus l'autorité de la page l'est. Mais l'autorité de la page où le lien entrant est placé joue également un rôle important. A la différence de l'index des citations scientifiques, par exemple, l'algorithme PageRank le prend en considération.

Présentons maintenant l'algorithme PageRank formellement. Nous considérons le Web

comme un graphe orienté. Une page Web est un nœud et un hyper-lien est un arc dont la queue de l'arc est la page Web où l'hyper-lien est placé et la tête de l'arc est la page Web sur laquelle l'hyper-lien fait référence. Ce graphe orienté est appelé *le Graphe du Web*. Nous utiliserons les termes de "page" et "nœud" de façon interchangeable dans la suite. Supposons qu'il existe n pages Web sur le Web que nous numérotons de 1 à n et définissons la matrice d'hyper-liens H de taille $n\ times n$ par

$$h_{ij} = \begin{cases} 1/d_i, & \text{if page } i \text{ links to } j, \\ 0, & \text{otherwise,} \end{cases} \tag{B.1}$$

avec $i, j = \overline{1, n}$, où d_i est le nombre de liens sortant de la page i. Si une page n'a pas de liens sortant, elle est appelé *page ballante* et la ligne de la matrice H qui lui correspond est la ligne zéro. Nous le remettons en order en supposant que les pages ballantes se réfère à toutes les pages sur le Web. Ces liens imaginaires sont appelés *liens artificiels*. Nous allons présenter une colonne vecteur de a quel élément $a_i = 1$ si le i^e ligne de la matrice H correspond à une page ballants et 0 autrement. Nous définissons une stochastique matrice P par

$$P = H + av, \tag{B.2}$$

où v est une distribution de probabilité uniforme. Nous définissons une matrice stochastique qui est appelé *matrice Google* par

$$G = cP + (1 - c)\mathbf{1}v, \tag{B.3}$$

où $\mathbf{1}$ est un vecteur colonne de dimension appropriée, dont toutes les entrées sont égales à un, et $0 < c < 1$. Le vecteur du PageRank est défini comme le vecteur propre de la matrice G correspondant à sa valeur propre principale.

$$\pi = \pi G, \tag{B.4a}$$

$$\pi\mathbf{1} = 1. \tag{B.4b}$$

Le *vecteur PageRank* peut être consultée à (B.4) dans [21, 58, 68]

$$\pi = v(1 - c)(I - cP)^{-1}, \tag{B.5}$$

où I est la matrice d'identité.

On peut voir que si l'on considère la matrice Google comme une matrice de transition d'une chaîne de Markov, alors PageRank est une distribution de probabilité stationnaire de la chaîne de Markov. Imaginons un internaute sur le Web sur une des pages Web. Avec une probabilité de $1-c$, il passe à une page Web arbitraire et, avec une probabilité c, il choisit de suivre un des liens sortant de la page où il est à pour le moment. Le lien particulier sortant est choisi de manière uniforme dans l'ensemble des sortants des liens de la page Web en cours. Si l'on imagine que beaucoup de surfeurs répartis uniformément sur le Web suivent le comportement décrit ci-dessus, le numéro d'internautes sur une page Web sera proportionnel à sa valeur PageRank après un certain temps. Cela implique que la probabilité de trouver un internaute sur une page Web est la valeur PageRank de la page Web. Il est clair que plus la valeur PageRank est grande, plus la page est visitée. Ainsi, le PageRank peut être considéré comme la suite d'une marche aléatoire sur le Graphe du Web et le résultat est une mesure de centralité définies sur le Graphe du Web qui détermine l'importance relative d'un nœud à l'intérieur du graphe.

Ci-dessus, nous avons supposé que l'internaute n'a pas de préférence dans le choix d'une page Web quand il saute sur une page arbitraire, mais ce modèle n'est pas adéquat. Un utilisateur peut préférer le sport, l'autre l'actualité, et un troisième l'art. Nous pouvons prendre en compte cette préférence par le vecteur v en ne supposant pas une loi uniforme, mais une distribution générale. Vector v est appelé un vecteur de la *personnalisation*. Le PageRank est appelé, dans ce cas, *PageRank Personnalisée*.

Le calcul du PageRank est très consommateur puisque la dimension du vecteur PageRank est énorme. La dimension du PageRank est le nombre de pages indexées sur le Web. L'estimation de Novembre 1997 jusqu'à Mars 2009 passe de 200 millions à 25 milliards de documents. La taille de la dimension PageRank rend impossible utilization de méthodes directes, telles que le pivot de Gauss, à déterminer le PageRank. Certaines méthodes d'approximation doivent être utilisées.

Selon des informations qui sont disponibles publiquement, Google utilise la méthode d'itération de puissance pour le calcul du PageRank [71]. Partant du vecteur de distribu-

tion uniforme $\pi^{(0)} = (1/n)\mathbf{1}^T$ en approximation initial, le k^e vecteur rapprochement est calculé par

$$\pi^{(k)} = \pi^{(k-1)}G, \quad k \geqslant 1. \tag{B.6}$$

La méthode s'arrête lorsque la précision requise ε est atteint. Le nombre de flops nécessaires pour la méthode de convergence est de l'ordre $\frac{\log \varepsilon}{\log c}\, nnz(P)$, où $nnz(P)$ est le nombre de éléments non nuls de la matrice P [58].

B.2 Méthodes d'agrégation-désagrégation pour le calcul de PageRank

Le calcul du PageRank est une opération très coûteuse en calcul. Comme les méthodes directes sont très consommatrices en temps [76, §2], Google utilise la méthode d'itération de puissance pour calculer le vecteur du PageRank [71], mais le taux de convergence peut être faible [57]. Certaines méthodes ont été proposées pour l'accélération [49, 50, 44, 59, 48, 64]. Les auteurs de [49, 50] ont accéléré le calcul de vecteur du PageRank par des modifications de la méthode d'itération de puissance, tandis que les auteurs de [44, 59, 48, 64] ont utilisé une approche d'agrégation-désagrégation.

L'une des contributions de la thèse est la condition d'équivalence de deux méthodes d'agrégation-désagrégation qui permet de composer une nouvelle méthode possédant des avantages des méthodes mentionnées et éviter leurs inconvénients.

Nous considérons les méthodes d'agrégation-désagrégation ci-dessous s'appliquées à une chaîne de Markov générale et sa matrice de transition avec sa distribution stationnaire. Pour l'examen du cas particulier de PageRank, un lecteur intéressé se reportera à la thèse. Les méthodes d'agrégation-désagrégation pour le calcul de la distribution stationnaire utilisent de la décomposition d'espace d'états que nous noterons $\mathcal{I}$. Supposons que l'ensemble $\mathcal{I}$ est décomposé en deux ensembles non-croisés $\mathcal{I}^{(i)}$, $i = 1, 2$ (le cas général de decompostion en

nombre fini d'ensembles est pris en considération dans la thèse), de telle sorte que

$$\begin{aligned} \mathcal{I}^{(1)} &= \{1, \ldots, n_1\}, \\ \mathcal{I}^{(2)} &= \{n_1 + 1, \ldots, n_1 + n_2\}, \end{aligned} \tag{B.7}$$

où $n_1 + n_2 = n$.

Selon la décomposition d'espace d'états, la matrice de transition est également partitionnée comme suit:

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}, \tag{B.8}$$

où P_{ij} est un bloc de dimension $n_i \times n_j$. Suite à la partitionnement de la matrice de transition, la distribution stationnaire est partitionnée en composants:

$$\pi = (\pi_1, \pi_2), \tag{B.9}$$

où π_i est un vecteur ligne avec $dim(\pi_i) = n_i$.

Toutes les méthodes d'agrégation utilisent une *matrice agrégée* A. La matrice A est une matrice dont chaque élément correspond à un bloc de la matrice P, soit $a_{ij} \leftrightarrow P_{ij}$. En règle générale, les éléments de la matrice A sont formés que $a_{ij} = \zeta_i P_{ij}\mathbf{1}$, où ζ_i est un vecteur d'une distribution de probabilité. Nous appelons le vecteur ζ_i *vecteur d'agrégation*. Chaque forme méthode d'agrégation crée une matrice d'agrégation à sa manière avec à partir de differentes distributions de probabilités en tant que vecteurs d'agrégation et un partitionnement différent. On peut considérer la matrice agrégée comme une matrice de transition d'une chaîne de Markov avec son espace d'état formé par les sous-ensembles de l'espace d'état $\mathcal{I}$.

Algorithmes d'agrégation-désagrégation

Nous passons en revue deux méthodes d'agrégation-désagrégation. Un lecteur intéressé est renvoyé à la thèse pour d'examen détaillé des autres méthodes.

Méthode d'agrégation-désagrégation complète (FAM). Déterminer une arroximation $\pi^{(k)}$ de la distribution stationnaire π de la matrice stochastique P en k itérations.

1. Sélectionnez un vecteur $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)}\right)$ avec $\pi^{(0)}\mathbf{1} = 1$.

2. Faites $k = 0, 1, 2 \ldots$

   (a) Normalisez $\sigma_i^{(k)} = \pi_i^{(k)} / \|\pi_i^{(k)}\|_1$, $i = 1, 2$.

   (b) Formez la matrice agrégée $A^{(k)}$

$$A^{(k)} = \begin{pmatrix} \sigma_1^{(k)} P_{11} \mathbf{1} & \sigma_1^{(k)} P_{12} \mathbf{1} \\ \sigma_2^{(k)} P_{21} \mathbf{1} & \sigma_2^{(k)} P_{22} \mathbf{1} \end{pmatrix}.$$

   (c) Déterminez la distribution stationnaire $\nu^{(k)}$ de $A^{(k)}$

$$\nu^{(k)} = \nu^{(k)} A^{(k)}.$$

   (d) Déterminez le vecteur de désagrégation $\widetilde{\pi}^{(k)}$

$$\widetilde{\pi}^{(k)} = \left(\nu_1^{(k)} \sigma_1^{(k)}, \nu_2^{(k)} \sigma_2^{(k)} \right).$$

   (e) Faites ℓ étapes de la méthode d'itération de puissance

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)} P^{l}.$$

Méthode d'agrégation-désagrégation partielle (PAM). Déterminer une approximation $\pi^{(k)}$ de la distribution stationnaire π de la matrice stochastique P en k itérations.

1. Sélectionnez un vecteur $\pi^{(0)} = \left(\pi_1^{(0)}, \pi_2^{(0)} \right)$ avec $\pi^{(0)} \mathbf{1} = 1$.

2. Faites $k = 0, 1, 2 \ldots$

   (a) Normalisez $\sigma_2^{(k)} = \pi_2^{(k)} / \|\pi_2^{(k)}\|_1$.

   (b) Formez la matrice agrégée $A_1^{(k)}$

$$A_1^{(k)} = \begin{pmatrix} P_{11} & P_{12} \mathbf{1} \\ \sigma_2^{(k)} P_{21} & \sigma_2^{(k)} P_{22} \mathbf{1} \end{pmatrix}.$$

   (c) Déterminez la distribution stationnaire $\alpha^{(k)}$ de $A_1^{(k)}$

$$\alpha^{(k)} = \alpha^{(k)} A_1^{(k)}.$$

(d) Divisez $\alpha^{(k)}$

$$\alpha^{(k)} = (\omega_1^{(k)}, \rho^{(k)}).$$

(e) Déterminez le vecteur de désagrégation $\widetilde{\pi}^{(k)}$

$$\widetilde{\pi}^{(k)} = \left(\omega_1^{(k)}, \rho^{(k)}\sigma_2^{(k)}\right).$$

(f) Faites ℓ étapes de la méthode d'itération de puissance

$$\pi^{(k+1)} = \widetilde{\pi}^{(k)}P^{l}.$$

Lorsque $rankP_{21} = 1$, deux théorèmes qui sont prouvées dans la thèse affirment que les deux algorithmes mentionnés ci-dessus sont équivalentes au sens qu'ils produisent la même séquence des résultats intermédiaires. Ces théorèmes nous permettent de formuler nouvelle méthode de trouver distribution stationnaire π, qui effectuent la première itération de la méthodes d'agrégation-désagrégation partielle et tous l'autres itérations de la méthode d'agrégation-désagrégation complète, que nous appelons l'algorithme d'agrégation-désagrégation mixte. D'une part, dans le cas où la bloc hors-diagonale de la part agrégée de la matrice de transition a rang un, l'algorithme d'agrégation-désagrégation mixte produire la même séquence de l'résultats intermédiaires que la méthode d'agrégation-désagrégation partielle et, par conséquent, il a les mêmes propriétés d'une convergence de la méthode d'agrégation-désagrégation partielle. D'autre part, toutes les itérations à l'exception le premier est effectué par l'algorithme d'agrégation-désagrégation mixtes comme à la méthode d'agrégation-désagrégation complète, par conséquent, l'algorithme d'agrégation-désagrégation mixte est presque aussi consommer de calcul que la méthode d'agrégation-désagrégation complète. Ainsi, l'algorithme d'agrégation-désagrégation mixte possède les avantages des deux méthodes d'agrégation-désagrégation et évite les inconvénients.

B.3 Distributions quasi-stationnaire que les mesures de centralité pour le Géant Composante Fortement Connexe d'un graphe réductible

Le choix de la valeur du facteur d'amortissement qui est un paramètre d'entrée essentielle dans l'algorithme PageRank est un problème important qui n'ayant pas de solution évidente si loin. Si quelqu'un corrige le facteur d'amortissement égal à 0, elle reçoit l'uniforme distribution comme le classement des pages Web. Évidemment, il ne fait pas de sens. Dans le même temps, comme l'autre cas extrêmes, le choix du facteur d'amortissement égal à l'unité est au même niveau de rationalité, puisque, dans ce cas, le classement tend de se concentrer à quelques pages appelé *éviers du rang*.

Nous explorons des mesures centralité sans paramètres. Ici, nous vous suggérons de centralité des mesures qui tiennent à titre de contribution que la liste d'adjacence d'un graphe.

Le Graphe du Web peut être divisée en composantes principales. Si les liens artificiels à partir de nœuds pendants sont pris en compte, il est indiqué dans [14] que le Graphe du Web peut être divisé en deux composantes: la Composante Fortement Connexe Étendue (ESCC) et la Pur OUT composante (POUT). L'ESCC est la plus grande composante fortement connexe du Graphe du Web. Tous les autres composantes sont étroitement liées de plusieurs magnitudes plus petites que l'ESCC. POUT est de petite taille, mais si le facteur d'amortissement c est choisi égal à un, la marche aléatoire avec probabilité une absorbe en POUT. Comme nous le montrons dans le numériques section expériences, une grande majorité des pages et près de tous les pages importants sont en l'ESCC. Nous notons également que, même si le facteur d'amortissement est choisi à proximité à l'un, la marche aléatoire peut passer beaucoup de temps dans l'ESCC avant l'absorption. Par conséquent, classement des pages Web à partir de l'ESCC nous suggérons l'utilisation de distributions quasi-stationnaire [33, 73], car elles représentent la dynamique de la marche aléatoire avant qu'il quitte l'ESCC.

Comme indiqué dans [14], par la renumérotation de nœuds, le matrice de transition P peut

être transformée en la forme suivante

$$P = \begin{pmatrix} Q & 0 \\ R & T \end{pmatrix},$$

où le bloc T correspond à l'ESCC, le bloc de Q correspond à POUT, et le bloc de R correspond à la transition de l'ESCC pour les nœuds dans POUT. Parce que la matrice T correspond à l'ESCC, elle est irréductible. Notons π_T la part du vecteur PageRank correspondant à l'ESCC. En utilisant formule (B.5), nous concluons que

$$\pi_T(c) = \frac{1-c}{n}\mathbf{1}^T(I-cT)^{-1},$$

où $\mathbf{1}$ est un vecteur unitaire de dimension appropriée. Nous allons définir la partie renormalisée du vecteur PageRank correspondant à l'ESCC:

$$\hat{\pi}_T(c) = \frac{\pi_T(c)}{||\pi_T(c)||_1}. \tag{B.10}$$

Nous notons que cette renormalisation ne modifie pas le classement entre les nœuds à l'intérieur de l'ESCC.

Nous définissons quatre distributions quasi-stationnaires et fournissons de explications intuitives pour clarifier leur sens.

Définition 1 *La distribution pseudo-stationnaire $\hat{\pi}_T$ est donnée par*

$$\hat{\pi}_T = \frac{\mathbf{1}^T[I-T]^{-1}}{\mathbf{1}^T[I-T]^{-1}\mathbf{1}}.$$

La i^e composante de $\hat{\pi}_T$ peut-être interprétée comme la fraction du temps que la marche aléatoire (avec $c = 1$) passe dans le nœud i avant son absorption.

Proposition B.1 *La limite suivante existe*

$$\hat{\pi}_T = \lim_{c\to 1} \hat{\pi}_T(c),$$

et le classement des pages dans l'ESCC fournies par le vecteur PageRank converge vers le classement fourni par $\hat{\pi}_T$ quand le facteur d'amortissement c tend vers un. En outre, ces deux classements coïncident pour tous les valeurs de c de plus une certaine valeur c^*.

Définition 2 *La distribution quasi-stationnaire $\tilde{\pi}_T$ est définie par l'équation*

$$\tilde{\pi}_T T = \lambda_1 \tilde{\pi}_T, \qquad \tilde{\pi}_T \mathbf{1} = 1,$$

où λ_1 est la valeur propre de Perron-Frobenius de la matrice T.

La distribution quasi-stationnaire $\tilde{\pi}_T$ peut-être interprétée comme une bonne distribution initiale sur les états non-absorbants (états dans l'ESCC) qui sont de nature que la distribution de la marche aléatoire, conditionnée à la non-absorption préalable au temps t, est indépendant de t [35].

Notons $\bar{T}$ la matrice d'hyperliens associée à l'ESCC lorsque les liens donnant sur l'extérieur de l'ESCC sont négligés. De toute évidence, nous avons

$$\bar{T}_{ij} = \frac{T_{ij}}{[T\mathbf{1}]_i},$$

où $[T\mathbf{1}]_i$ est la i^e composante du vecteur $T\mathbf{1}$.

Définition 3 *La distribution quasi-stationnaire $\bar{\pi}_T$ est définie par l'équation*

$$\bar{\pi}_T \bar{T} = \bar{\pi}_T, \qquad \bar{\pi}_T \mathbf{1} = 1.$$

L'entrée $\bar{T}_{ij}$ de la matrice $\bar{T}$ peut-être considérée comme une condition probabilité de passer du nœud i au nœud j à la condition que la marche aléatoire ne laisse pas l'ESCC la sauter. Alors, $\bar{\pi}_T$ peut-être interprétée comme la distribution stationnaire de la marche aléatoire sous la condition ci-dessus.

On peut généraliser la notion de $\bar{\pi}_T$. Notamment, nous examinons la situation lorsque la marche aléatoire reste à l'intérieur de l'ESCC, après un nombre fini de sauts N. Un lecteur intéressé se reportera à la thèse pour les détails. Examenons maintenant les cas limite, lorsque N tend vers l'infini. Nous nous référons à ce qui suivons la limit du noyau torsadé

$$\check{T}_{ij} = \lim_{N \to \infty} \frac{T_{ij} T_j^{(N-1)} \mathbf{1}}{T_i^{(N)} \mathbf{1}}. \tag{B.11}$$

L'existence est prouvée dans la thèse.

Définition 4 *La distribution quasi-stationnaire $\breve{\pi}_T$ est définie comme la distribution stationnaire du noyau torsadé. A savoir, il est la solution de l'équation de vecteur propre et condition de normalisation suisants:*

$$\breve{\pi}_T = \breve{\pi}_T \breve{\Upsilon}, \qquad \breve{\pi}_T \mathbf{1} = 1.$$

Si nous supposons en plus l'apériodicité, $\breve{\Upsilon}_{ij}$ tiens de l'interprétation de la probabilité de transition à partir de i pour j dans l'ESCC, conditionné sur le fait qu'il ne quitte jamais l'ESCC.

L'analyse analytique, faite dans la thèse, nous permet de conclure que les distributions quasi-stationnaires considérées sont proches les unes des autres ce qui est soutenu par les expériences numériques.

Expériences numériques

Pour nos expériences numériques, nous avons utilisé le site Web de l'INRIA (http://www.inria.fr, l'ensemble de données est disponible auprès de l'auteur sur demande). C'est un site Web typique avec environ 300.000 pages Web et 2.200.000 hyper-liens. Dans nos expériences, nous calculons $\bar{\pi}_T$, $\tilde{\pi}_T$, $\hat{\pi}_T$, et $\breve{\pi}_T$ avec 5 chiffres de précision. Aussi, nous calculons $\hat{\pi}_T(0.85)$ qui est le vecteur normalisé PageRank de l'ESCC avec le facteur d'amortissement égal à 0.85. Pour chaque paire de ces vecteurs, nous avons calculé métrique τ de Kendall. La métrique τ de Kendall montre comment deux métriques de classements sont différentes en termes de nombre de swaps qui sont nécessaires pour transformer un rang à l'autre. La métrique τ de Kendall a la valeur de un si les deux classements sont identiques et moins un si le premier rang est l'inverse de l'autre.

Dans nos expériences, la métrique τ de Kendall pour toutes les paires est très proche de un. Ainsi, nous concluons que la centralité produrent par les quatre mesures de classements basées sur la quasi-stationnarité est très similaires.

Nous avons également analysé la métrique τ de Kendall entre $\tilde{\pi}_T$ et le PageRank de l'ESCC, en fonction du facteur d'amortissement (voir Figure B.1). Quand c tend vers l'un, τ de Kendall s'approche de un. Ceci est en accord avec la proposition B.1.

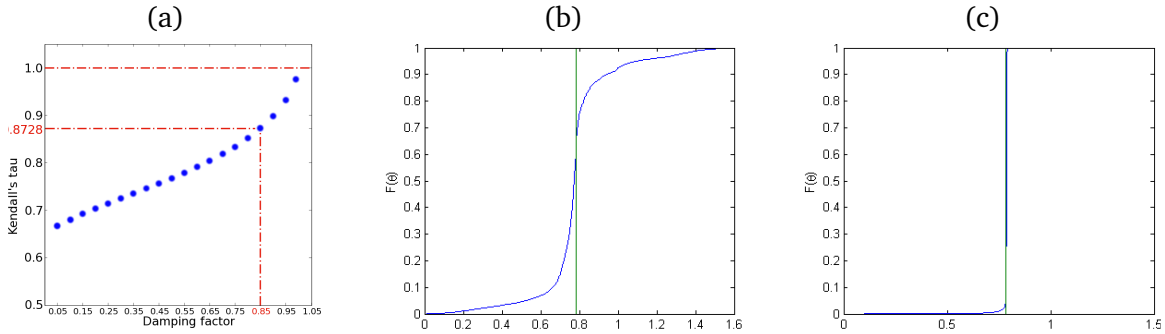


Figure B.1: (a) métriques τ de Kendall entre $\tilde{\pi}_T$ et le PageRank de l'ESCC $\hat{\pi}_T(c)$ en fonction du facteur d'amortissement. Le cumulatif la distribution de θ mesure la corrélation de rang: entre $\hat{\pi}_T(0.85)$ et $\tilde{\pi}_T$ (b), entre $\tilde{\pi}_T$ et $\hat{\pi}_T$ (c).

Nous avons également comparé le classement produit par les distributions quasi-stationnaires ce PageRank et de l'ESCC par l'aide de la θ mesure de corrélation de rang. La mesure est définie comme suit

$$\theta_i = \arctan(r_i^1/r_i^2),$$

où r_i^1 est le classement du nœud i dans un vecteur et r_i^2 est le classement du même nœud i dans un autre vecteur. Par le classement terme, nous entendons ici le lieu du nœud i dans un vecteur si nous trions les entrées du vecteur dans l'ordre décroissant. Si classement de nœud i est le même dans les deux vecteurs, alors θ_i est égal à $\pi/4$. Comme on peut le voir sur la Figure B.1, la distribution cumulative sur θ_i correspondant à toutes les distributions quasi-stationnaires est proche de la ligne verticale à $\pi/4$, ce qui signifie que le classement des produits par les vecteurs sont proches les uns des autres.

B.4 Méthodes de Monte Carlo pour le calcul PageRank: Quand une itération est suffisante

Bien que les méthodes itératives de calcul PageRank soient très développées, à part eux, il existe d'autres méthodes probabilistes visant à ce but. Ici, nous étudions les méthodes de type Monte Carlo (MC) pour le calcul du classement PageRank. A notre connaissance, dans seule-

ment deux ouvrages [25, 37] les méthodes Monte Carlo sont appliquées au calcul du PageRank. Les principaux avantages méthodes probabiliste de type Monte Carlo sur les méthodes déterministes sont: le PageRank des pages importantes est déterminé avec une grande précision déjà après la première itération, les méthodes MC ont dans mises en œuvre parallèles naturelles et les méthodes de MC permettent la mise à jour continue de la PageRank quand la structure de l'Internet évolue.

On peut suggérer l'algorithme suivant employé dans [25].

Algorithme B.1 (MC point final avec démarrage aléatoire) *Simuler* N *marches aléatoires* $\{X_t\}_{t\geqslant 0}$ *et chaque marche est initiée à une page choisie au hasard. Évaluer* π_j *comme une fraction de* N *marches aléatoires qui finissent à la page* $j = \overline{1,n}$.

L'algorithme ci-dessus produit l'hasard inutiles par le choix aléatoire d'une page de démarrage. La page de départ peut être choisi de manière itérative. Il en résulte l'algorithme suivant utilisé dans [37] pour le calcul PageRank Personnalisé.

Algorithme B.2 (MC point final avec démarrage cycliques) *Simuler* $N = mn$ *marches aléatoires* $\{X_t\}_{t\geqslant 0}$ *et les marches initiée à chaque page exactement* m *fois. Évaluer* π_j *comme une fraction de* N *marches aléatoires qui se terminent à la page* $j = \overline{1,n}$.

Les deux algorithmes ci-dessus ne conservent que les informations sur la dernière page qui est visitée par les marchees aléatoires simulées. Nous pouvons construire un algorithme qui prend compte toutes les pages visitées lors d'une marche aléatoire.

Algorithme B.3 (MC chemin d'accès complet avec démarrage cyclique) *Simuler la marche aléatoire* $\{X_t\}_{t\geqslant 0}$ *exactement* m *fois à partir de chaque page. Pour n'importe quelle page* i*, évaluer* π_j *le nombre total de visites à la page* j *multiplié par* $(1-c)/(n*m)$.

La simulation par l'algorithme ci-dessus pass sur une page arbitraire si elle arrive à une page ballants. Évidemment, il dissocie les deux parties d'une marche aléatoire avant la visite d'une page ballante et après. Dans ce cas, il est peut être mieux de s'arrêter à une page ballante, ce qui conduit à un autre algorithme.

Algorithme B.4 (MC chemin d'accès complet au niveau des nœuds d'arrêt ballants) *Simuler la marche aléatoire* $\{Y_t\}_{t\geqslant 0}$ *à partir de chaque page exactement* m *fois. Pour toute page* j*, évaluer* π_j *le nombre total de visites à la page* j *divisé par le nombre total de pages visitées.*

Nous notons que les versions de chemin d'accès complet des méthodes de Monte Carlo aussi admet une choix de la page au hasard. L'algorithme correspondant est le suivant.

Algorithme B.5 (MC chemin complet avec démarrage aléatoire) *Simuler* N *des échantillons de la marche aléatoire* $\{Y_t\}_{t\geqslant 0}$ *à partire d'une page au hasard. Pour touts page* j*, évaluer* π_j *le nombre total de visites à la page* i *divisé par le nombre total de pages visitées.*

Nous avons conclu que les algorithmes MC avec démarrage cycliques sont préférables aux analogues algorithmes de MC avec départ aléatoire. Nous avons analysé de manière approfondie et comparé les MC chemin d'accès complet au niveau des nœuds d'arrêt ballants à la MC point final. Nous avons montré que dans les conditions naturelles de la MC chemin d'accès complet au niveau des nœuds d'arrêt ballants surpasse la MC point final. L'analyse rigoureuse est présentée dans la thèse.

Expériences numériques

Pour nos expériences numériques, nous avons pris le site Web de l'INRIA Sophia Antipolis `http://www-sop.inria.fr/`. Il s'agit d'un site Web typique, avec environ 50.000 pages Web et 200.000 hyper-liens. Tout d'abord, nous avons effectué un nombre d'itérations suffisant de la méthode d'itérations de la puissance pour obtenir la valeur du PageRank dans 20 chiffres de précision. Nous avons réglé le vecteur PageRank dans l'ordre décroissant et tracé avec dans l'échelle loglog (voir la Figure B.2).

Nous avons effectué 10 pas de la méthode d'itérations de la puissance (PI) et 10 itérations de trois algorithmes mis en œuvre MC. Dans la Figure B.2, nous comparons les résultats de 10 itérations de la méthode PI et le MC chemin d'accès complet d'arrêt dans la méthode des nœuds pendants pour les 1000^e par la page de valeur PageRank. En effet, déjà la première itération du

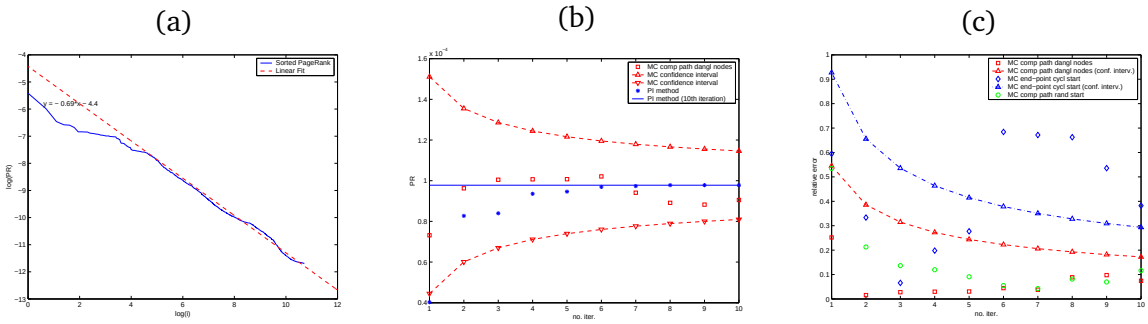


Figure B.2: (a) Tri PageRank dans loglog échelle. (b) PI vs MC chemin d'accès complet au niveau des nœuds d'arrêt ballants: π_{1000}. (c) Comparaison des algorithmes MC: π_{1000}.

MC chemin d'accès complet d'arrêt en balançant algorithme de nœuds donne une petite erreur de la page Web.

Ensuite, dans la Figure B.2, nous comparons les trois versions de la méthode de Monte Carlo: le MC chemin d'accès complet d'arrêt dans les nœuds ballants, le MC point final avec démarrage cyclique, et le MC chemin d'accès complet avec départ aléatoire. Nous avons tracé l'erreur relative réelle et le montant estimaté avec 95% d'intervalles de confiance. Il s'avère que sur notre jeu de données de la MC chemin d'accès complet d'arrêt dans les nœuds ballants obtient les meilleurs résultats, suivie par le MC chemin d'accès complet avec départ aléatoire. Le MC point final avec démarrage cyclique a la plus mauvaise performance.

B.5 Trouver des listes du haut-k avec Monte Carlo PageRank Personnalisé

Les PageRank Personnalisé ou PageRank thématiques sensibles [41] ont nombre d'applications. Dans le papier original [41], le PageRank Personnalisé a été utilisé pour introduire la personnalisation dans le Web de recherche. Dans [29,85], le PageRank Personnalisé a été suggéré pour trouver des entités liées. Dans [70], la mesure Green, qui est étroitement liée à la composante PageRank Personnalisé, a été proposé pour trouver des pages liées à Wikipédia. Dans [4, 5], le PageRank Personnalisé a été utilisé pour trouver des coupes locales graphiques

et, dans [11], le PageRank Personnalisé a été appliqué pour le regroupement de grandes collections de documents hyper-texte. Dans tous les applications mentionnées ci-dessus, il faut trouver des nœuds avec des valeurs relativement élevées des PageRank Personnalisé. Comme a été montré dans [10], et présenté dans le chapitre précédent, les méthodes de Monte Carlo sont efficaces pour l'estimation du PageRank pour les populaires pages. À la suite [10], nous proposons d'utiliser des méthodes de Monte Carlo pour trouver des listes en haut des pages avec de grandes valeurs de PageRank Personnalisé.

Nous considérons le PageRank Personnalisé avec le vecteur de personnalisation v dans l'équation (B.2) et l'équation (B.3) égal à $\mathbf{1}_i^T$ sans perdre de généralité en raison de la linéarité du PageRank.

Nous considérons trois méthodes de Monte Carlo simulant une marche aléatoire sur le Graphe du Web. Toutes les méthodes de Monte Carlo produisent des estimateurs le PageRank Personnalisé. La première méthode de Monte Carlo prend en compte les zones où les marches aléatoires s'arrêtent.

Algorithme B.6 (MC point final) *Simuler* m *va de la marche aléatoire* $\{X_t\}_{t\geqslant 0}$ *initiée au nœud* i. *Évaluer* π_j *comme une fraction de* m *marches aléatoires qui se terminent au nœud* $j = \overline{1,n}$.

La deuxième méthode de Monte Carlo prend en compte tous les nœuds visités par un marche aléatoire et trouve estimation du PageRank Personnalisé comme le rapport de la nombre de visites sur le nombre attendu de transitions a la marche aléatoire.

Algorithme B.7 (MC chemin d'accès complet) *Simuler* m *marches aléatoires* $\{X_t\}_{t\geqslant 0}$ *lancées au noeud* i. *Évaluer* π_j *par le nombre total de visites au nœud* j *multiplié par* $(1-c)/m$.

La dernière méthode de Monte Carlo prend en compte tous les nœuds visités par un marche aléatoire, mais divise le nombre de visites effectuées par le nombre réel de transitions effectuées au cours de la marche aléatoire.

Algorithme B.8 (MC chemin d'accès complet compte de transitions) *Simuler* τ *étapes de la marche aléatoire* $\{Y_t\}_{t\geqslant 0}$ *régie par la matrice Google. Évaluer* π_j *par le nombre de visites au noeud*

j *divisé par le nombre total de mesures* τ.

Puis les résultats des algorithmes proposés, nous aimerions obtenir soit une *liste du haut-*k de nœuds ou un *panier supérieur-*k de nœuds avec une forte probabilité.

Définition 5 *La* liste du haut-k *de nœuds est une liste de* k *nœuds avec les plus grandes valeurs de PageRank Personnalisée disposées dans un ordre décroissant de leurs valeurs PageRank Personnalisé.*

Définition 6 *Le* panier supérieur-k *de nœuds est un ensemble de* k *nœuds avec les plus grandes valeurs de PageRank Personnalisée sans ordre requis.*

Nous prenons l'hypothèse suivante technique qui n'est pas restricte et est satisfaite dans la plupart des applications pratiques.

Supposition B.1 *Nous supposons que* $\pi_1 > \pi_2 > ... > \pi_k > \pi_{k+1} \geqslant \pi_j$ *pour* $j \geqslant k+2$.

Par fois c'est bénéfique de détendre notre objectif d'obtenir un panier supérieur-k avec un petit nombre d'éléments erronés.

Définition 7 *Nous appelons* relaxation-l *d'un panier supérieur-*k *une réalisation lorsque nous permettons des* l *éléments erronés dans le panier supérieur-*k.

Nous avons comparé les performance de trois méthodes de type Monte Carlo en utilisant la variance des estimateurs des PageRank Personnalisé qu'ils produisent. Nous calculons analytiquement les probabilités de classement qui nous donnent le niveau de sécurité que la liste du haut-k ou le panier supérieur-k soit révélé correctement. Nous avons également fourni une estimation basée sur les inégalités de Bonferroni et des méthodes de Monte Carlo. En particulier, nous avons montré que les probabilités de classement convergent de façon exponentielle. Nous avons considéré un assouplissement le panier supérieur-k en permettant à certains nombre de fau éléments. Cette relaxation du panier supérieur-k est beaucoup plus facile à détecter. Nous effectué un certain nombre d'expériences numériques pour illustrer nos résultats théoriques.

B.6 Une approche tensorielle pour le calcul des moments mixtes d'ordre supérieur des chaîne de Markov avec absorption

Dans l'analyse mathématique utilisée dans l'examen des méthodes de Monte Carlo pour les applications de PageRank Personnalisée, les moments mixtes d'ordre supérieur du nombre de visites sont exploitées. Étant donné que les moments mixtes d'ordre supérieur sont difficiles à exprimer dans une forme matrice, nous avons utilisé une approche tensorielle. Des formules approchées compactes sont obtenues dans la thèse.

Introduction

Prenons une chaîne de Markov absorbante avec sa matrice transaction P. Par la renumérotation des états on peut décomposer la matrice P de la façon suivante:

$$P = \begin{pmatrix} I & 0 \\ S & Q \end{pmatrix},$$

où la sous-matrice Q est une matrice substochastic correspondant à états transitoires. Soit T l'ensemble des états transitoires et $\bar{T}$ l'ensemble des états absorbants. Nous pouvons définir une matrice fondamentale Z de la chaîne de Markov absorbante par

$$Z = (I - Q)^{-1} = I + Q + Q^2 + \dots.$$

La matrice fondamentale $Z = \{z_{ij}\}_{i,j \in T}$ a la interprétation probabiliste suivante.

Définition 8 *Soit* N_j *la fonction donnant le nombre total de fois que la chaîne de Markov absorbante visites un état transitoire* j *avant de d'absorption.*

Notons $E_i[N_j]$ le premier moment de la fonction N_j en supposant que la chaîne de Markov commence à l'état i, où $i, j \in T$. Donc,

$$Z = \{E_i[N_j]\}_{i,j \in T}$$

comme il est noté dans [51, Theorem 3.2.4]. Le moments deuxieme non-mixtes $E_i[N_j^2]$ peut également être trouvé [51, Theorem 3.3.3] avec l'aide de la matrice Z comme

$$\left\{E_i\left[N_j^2\right]\right\}_{i,j\in T} = Z\left(2Z_{dg} - I\right),$$

où Z_{dg} est la même matrice Z, mais tous les éléments hors diagonale sont mis à zéro.

Cependant, les moments mixtes deuxieme $E_i[N_jN_k]$ et les moments mixtes d'ordre supérieur $E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right]$ ne sont pas si faciles à calculer. Ici, nous abordons ce problème par l'approche de tenseur.

Introduction des tenseurs

Nous donnons une brève introduction des faits de base de la théorie tenseur que nous utilisons dans les autres sections. Nous ne présentons pas la théorie tenseur dans son invégalité, nous venons de définir ce dont nous avons besoin pour notre application aux moments mixtes d'ordre supérieur. Un lecteur intéressé se reportera à [74, 63] pour plus de détails.

Nous vous présentons les opérations de tenseur dont nous avons besoin pour le développement ultérieur. Le produit tenseur $\otimes$ d'un tenseur $\mathcal{A}$ qui est n-fois contravariant et m-fois covariant-fois et un tenseur $\mathcal{B}$ qui est s-fois contravariant et t-fois covariant est un tenseur $\mathcal{C}$ qui est $n+s$-fois contravariant et $m+t$-fois covariant (B.12).

$$\mathcal{A}\otimes\mathcal{B} = \mathcal{C}, \tag{B.12}$$

où les composantes du tenseur de $\mathcal{C}$ dans une base peuvent être trouvés par la formule

$$a^{i_1\,i_2\ldots i_n}_{k_1\,k_2\ldots k_m}\, b^{p_1\,p_2\ldots p_s}_{h_1\,h_2\ldots h_t} = c^{i_1\,i_2\ldots i_n\,p_1\,p_2\ldots p_s}_{k_1\,k_2\ldots k_m\,h_1\,h_2\ldots h_t},$$

où les indices $i_1, i_2, \ldots, i_n$,$k_1, k_2, \ldots, k_m$, $p_1, p_2, \ldots, p_s$ et $h_1, h_2, \ldots, h_t$ prennent toutes les valeurs possibles. De plus, nous écrirons produit tensoriel $\otimes$

$$a^{i_1\,i_2\ldots i_n}_{k_1\,k_2\ldots k_m} \otimes b^{p_1\,p_2\ldots p_s}_{h_1\,h_2\ldots h_t} = c^{i_1\,i_2\ldots i_n\,p_1\,p_2\ldots p_s}_{k_1\,k_2\ldots k_m\,h_1\,h_2\ldots h_t},$$

en supposant que les indices $i_1, i_2, \ldots, i_n$, $k_1, k_2, \ldots, k_m$, $p_1, p_2, \ldots, p_s$ et $h_1, h_2, \ldots, h_t$ prennent toutes les valeurs possibles.

Dans certains cas, nous devons considérer que les composantes des tenseurs ayant les mêmes indices dans le produit du tenseur (B.13):

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \otimes b^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t} = a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m h_1 h_2 \ldots h_t}. \tag{B.13}$$

Aussi laissez-nous définir la contraction tenseur $\odot$ par la formule (B.14):

$$a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} \odot b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} = \sum_{k_1} \sum_{k_2} \cdots \sum_{k_m} a^{i_1 i_2 \ldots i_n}_{k_1 k_2 \ldots k_m} b^{k_1 k_2 \ldots k_m}_{h_1 h_2 \ldots h_t} = c^{i_1 i_2 \ldots i_n}_{h_1 h_2 \ldots h_t}. \tag{B.14}$$

Nous notons que la contraction tensorielle est équivalente a produit des matrices si les matrices sont écrites en une seule composantes contravariant et une seule covariantes.

Nous allons utiliser les tenseurs et les opérations tenseur en les application aux moments mixtes deuxieme et aux moments mixtes d'ordre supérieur.

Les moments mixtes d'ordre supérieur

Considérons maintenant les moments mixtes d'ordre supérieur. Avant de formuler les moments mixtes d'ordre supérieur dans le formalisme tensoriel, prouvons que les moments sont limitées

Prenons la fonction génératrice des moments conditionnelles de la chaîne de Markov absorbante, $M_i(y) = E_i\left[e^{\sum y_j N_j}\right]$, où la sommation est effectuée sur tous les états de la chaîne de Markov et le processus commence à un état transitoire i. Nous avons besoin de prouver que la fonction génératrice des moments est analytique à l'origine.

Nous allons définir vecteur ζ et la matrice $\Theta = \{\vartheta_{ik}\}_{i,k\in T}$

$$\begin{aligned} \zeta_i &= e^{y_i}\left(1 - \sum_{k\in T} p_{ik}\right), \\ \vartheta_{ik} &= \delta_{ik} - e^{y_i} p_{ik}. \end{aligned}$$

Proposition B.2 *Si tous les y_i sont assez petits, la fonction génératrice des moments $M(y)$ est indiquée par*

$$M(y) = \Theta^{-1}\zeta.$$

On peut voir que la fonction génératrice des moments conditionnelles $M(y)$ est analytique à l'origine, et, par conséquent, il existe tous les moments mixtes d'ordre élevé et ils sont finis.

Nous notons

$$\begin{aligned}
\varepsilon^i_j &= E_i[N_j], \\
\varepsilon^i_{jk} &= E_i[N_j N_k], \ldots, \\
\varepsilon^i_{k_0 k_1 \ldots k_{m-1}} &= E_i\left[\prod_{j=0}^{m-1} N_{k_j}\right],
\end{aligned}$$

où m est un entier naturel. Notons $M = \{k_0, k_1, \ldots, k_{m-1}\}$. Le cardinal de mettre M est m.

Donnons le résultat pour les moments mixtes deuxiemes, tout d'abord.

Théorème B.1 *Les moments mixtes deuxiemes sont donnés par*

$$\varepsilon^i_{jk} = \varepsilon^i_\nu \odot \left(\varepsilon^\nu_j \otimes \delta^\nu_k + \varepsilon^\nu_k \otimes \delta^\nu_j - \delta^\nu_k \otimes \delta^\nu_j\right).$$

Nous avons eu des moments mixtes d'ordre supérieur dans la représentation tenseur. Comme le produit est une opération commutative, l'ordre des indices $k_0 k_1 \ldots k_{m-1}$ dans $\varepsilon^i_{k_0 k_1 \ldots k_{m-1}}$ n'a pas d'importance, et nous pouvons écrire $\varepsilon^i_{k_0 k_1 \ldots k_{m-1}} = \varepsilon^i_M$.

Notons ${}_0a^\nu_{k_0 k_1 \ldots k_{m-1}} = \bigotimes_{\iota=0}^{m-1} \delta^\nu_{k_\iota}$. Permettez-nous de définir le tenseur ${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}}$ comme suit:

$${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}} = \sum_{\psi=0}^{\binom{m}{j}-1} \varepsilon^\nu_{f(M,j,\psi)} \otimes {}_0a^\nu_{\bar{f}(M,j,\psi)}.$$

Comme l'index ψ passe toutes les valeurs possibles, l'ordre des indices $k_0 k_1 \ldots k_{m-1}$ dans ${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}}$ n'a pas d'importance, et nous pouvons écrire ${}_ja^\nu_{k_0 k_1 \ldots k_{m-1}} = {}_ja^\nu_M$. Nous notons que ${}_ma^\nu_M = \varepsilon^\nu_M$.

Théorème B.2 *Les moments mixtes d'ordre supérieur de la chaîne de Markov absorbante sont données par*

$$\varepsilon^i_M = \varepsilon^i_\nu \odot \sum_{\varkappa=0}^{m-1} (-1)^{m-\varkappa+1} {}_\varkappa a^\nu_M.$$

On peut voir que le formalisme tensoriel nous permet de calculer la moments mixtes d'ordre supérieur par une formule compacte. Les moments mixtes d'ordre supérieur sont déterminés par les moments d'ordre inférieur.

BIBLIOGRAPHY

[1] Serge Abiteboul, Mihai Preda, and Gregory Cobena. Adaptive on-line page importance computation. In *Proceedings of the twelfth international conference on World Wide Web*, pages 280–290, Budapest, Hungary, 2003. ACM Press.

[2] D.J. Aldous and J.A. Fill. Reversible Markov chains and random walks on graphs. Book in preparation, 2001.

[3] Eitan Altman and Dieter Fiems. Expected waiting time in symmetric polling systems with correlated walking times. *Queueing Syst.*, 56(3-4):241–253, 2007.

[4] Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Local graph partitioning using pagerank vectors. In *FOCS*, pages 475–486. IEEE Computer Society, 2006.

[5] Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Local partitioning for directed graphs using pagerank. In Anthony Bonato and Fan R. K. Chung, editors, *WAW*, volume 4863 of *Lecture Notes in Computer Science*, pages 166–178. Springer, 2007.

[6] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan. Searching the web. *ACM Transactions on Internet Technology (TOIT)*, 1(1):2–43, August 2001.

[7] Reinaldo B. Arellano-Valle and Marc G. Genton. On the exact distribution of the maximum of absolutely continuous dependent random variables. *Statistics & Probability Letters*, 78(1):27–35, January 2008.

[8] Aristotle. *Metaphysics*.

[9] Aristotle. *Physics*.

[10] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova. Monte carlo methods in pagerank computation: When one iteration is sufficient. *SIAM J. Numer. Anal.*, 45(2):890–904, 2007.

[11] Konstantin Avrachenkov, Vladimir Dobrynin, Danil Nemirovsky, Son Kim Pham, and Elena Smirnova. Pagerank based clustering of hypertext document collections. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 873–874, New York, NY, USA, 2008. ACM.

[12] Konstantin Avrachenkov and Nelly Litvak. Decomposition of the google pagerank and optimal linking strategy. Technical report, INRIA Research Report no. 5101, 2004.

[13] Konstantin Avrachenkov and Nelly Litvak. The effect of new links on google pagerank. *Stoch. Models*, 22:2006, 2006.

[14] Konstantin Avrachenkov, Nelly Litvak, and Kim Son Pham. Distribution of pagerank mass among principle components of the web. *CoRR*, abs/0709.2016, 2007. informal publication.

[15] Konstantin Avrachenkov, Danil Nemirovsky, and Natalia Osipova. Web graph analyzer tool. In Luciano Lenzini and Rene L. Cruz, editors, *VALUETOOLS*, volume 180 of *ACM International Conference Proceeding Series*, page 54. ACM, 2006.

[16] Konstantin E. Avrachenkov. *Analytic Perturbation Theory and its Applications*. PhD thesis, University of South Australia, 2005.

[17] Konstantin E. Avrachenkov, Moshe Haviv, and Phil G. Howlett. Inversion of analytic matrix functions that are singular at the origin. *SIAM J. Matr. Anal. Appl*, 1998.

[18] Ricardo Baeza-Yates, Paolo Boldi, and Carlos Castillo. Generalizing pagerank: Damping functions for link-based ranking algorithms. In *Proceedings of ACM SIGIR*, pages 308–315, Seattle, Washington, USA, August 2006. ACM Press.

[19] A. Berman and R.J. Plemmons. Nonnegative matrices in the mathematical sciences. *SIAM Classics In Applied Mathematics, SIAM*, 1994.

[20] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public web search engines. In *Proceedings of the 7th International Conference on World Wide Web*, 1998.

[21] Monica Bianchini, Marco Gori, and Franco Scarselli. Inside pagerank. *ACM Trans. Inter. Tech.*, 5(1):92–128, February 2005.

[22] Paolo Boldi. Totalrank: ranking without damping. In *Poster proceedings of the 14th international conference on World Wide Web*, pages 898–899, Chiba, Japan, 2005. ACM Press.

[23] Paolo Boldi, Massimo Santini, and Sebastiano Vigna. Pagerank as a function of the damping factor. In *Proceedings of the 14th international conference on World Wide Web*, pages 557–566, Chiba, Japan, 2005. ACM Press.

[24] L. A. Breyer and G. O. Roberts. Catalytic perfect simulation. Technical report, Methodol. Comput. Appl. Probab, 2000.

[25] L.A. Breyer. Markovian page ranking distributions: some theory and simulations. Available at `http://www.lbreyer.com/preprints.html`, 2002. Technical report.

[26] Claude Brezinski, Michela Redivo-Zaglia, and Stefano Serra-Capizzano. Extrapolation methods for PageRank computations. *C. R. Math. Acad. Sci. Paris*, 340(5):393–397, 2005.

[27] Sergey Brin and Larry Page. The anatomy of a Large-Scale hypertextual web search engine. http://ilpubs.stanford.edu:8090/361/, 1998.

[28] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer Networks*, 33:309–320, 2000.

[29] Soumen Chakrabarti. Dynamic personalized pagerank in entity-relation graphs. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 571–580, New York, NY, USA, 2007. ACM Press.

[30] P. Chen, H. Xie, Sergei Maslov, and Sidney Redner. Finding scientific gems with google's pagerank algorithm. *J. Informetrics*, 1(1):8–15, 2007.

[31] P. G. Constantine and D. F. Gleich. Using polynomial chaos to compute the infuence of multiple random surfers in the pagerank model. In *Proceedings of the 5th Workshop on Algorithms and Models for the Web Graph*. Springer, 2007.

[32] Charles J. Corrado. The Exact Joint Distribution for the Multinomial Maximum and Minimum and the Exact Distribution for the Multinomial Range. *SSRN eLibrary*, 2007.

[33] J. N. Darroch and E. Seneta. On quasi-stationary distributions in absorbing discrete-time finite markov chains. *Journal of Applied Probability*, 2(1):88–100, 1965.

[34] Maurice de Kunder. World wide web size. `http://www.worldwidewebsize.com/`.

[35] Erik A. Van Doorn. Quasi-stationary distributions and convergence to quasi-stationarity of birth-death processes. *Advances in Applied Probability*, 23(4):683–700, 1991.

[36] W. J. Ewens. The diffusion equation and a pseudo-distribution in genetics. *Journal of the Royal Statistical Society. Series B (Methodological)*, 25(2):405–412, 1963.

[37] Dániel Fogaras, Balázs Rácz, Károly Csalogány, and Tamás Sarlós. Towards scaling fully personalized pageRank: algorithms, lower bounds, and experiments. *Internet Math.*, 2(3):333–358, 2005.

[38] Silvia Gabrielli and Stefano Mizzaro. Negotiating a multidimensional framework for relevance space. In Stephen W. Draper, Mark D. Dunlop, Ian Ruthven, and C. J. van Rijsbergen, editors, *MIRA*, Workshops in Computing. BCS, 1999.

[39] Chris Gosden. *Prehistory: A very short introduction*. Oxford University Press, Oxford, 2003.

[40] Antonio Gulli and Alessio Signorini. The indexable Web is more than 11.5 billion pages. In *Poster proceedings of the 14th international conference on World Wide Web*, pages 902–903, Chiba, Japan, 2005. ACM Press.

[41] Taher H. Haveliwala. Topic-sensitive pagerank. In *Proceedings of the Eleventh World Wide Web Conference*, pages 517–526, Honolulu, Hawaii, USA, May 2002. ACM Press.

[42] Monika Rauch Henzinger. Algorithmic challenges in web search engines. *Internet Mathematics*, 1(1), 2003.

[43] Roger A. Horn and Stefano Serra-Capizzano. A general setting for the parametric Google matrix. *Internet Math.*, 3(4):385–411, 2006.

[44] Ilse C. F. Ipsen and Steve Kirkland. Convergence analysis of a PageRank updating algorithm by Langville and Meyer. *SIAM J. Matrix Anal. Appl.*, 27(4):952–967, 2006.

[45] Ilse C. F. Ipsen and Teresa M. Selee. Pagerank computation, with special attention to dangling nodes. *SIAM J. Matrix Anal. Appl.*, 29(4):1281–1296, 2007.

[46] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 271–279, New York, NY, USA, 2003. ACM Press.

[47] Norman L. Johnson, Samuel Kotz, and N. Balakrishnan. *Discrete multivariate distributions*. A Wiley-Interscience publication. Wiley, New York, NY [u.a.], 1997.

[48] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Exploiting the block structure of the web for computing pagerank, 2003.

[49] Sepandar D. Kamvar, Taher H. Haveliwala, and Gene H. Golub. Adaptive methods for the computation of pagerank. *Technical Report*, 2003.

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

[51] John George Kemeny and James Laurie Snell. *Finite Markov chains*. University series in undergraduate mathematics. VanNostrand, New York, repr edition, 1969.

[52] C. D. Kemp and Adrienne W. Kemp. Rapid generation of frequency tables. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 36(3):277–282, 1987.

[53] C.G. Khatri and S.K. Mitra. Some identities and approximations concerning positive and negative multinomial distributions. In *Multivariate Analysis, Proc. 2nd Int. Symp.*, pages 241–260. Academic Press, 1969.

[54] Donald E. Knuth. The art of computer programming: Pre-fascicle 3a, a draft of section 7.2.1.3: Generating all combinations.

[55] I. Kontoyiannis and S.P. Meyn. Spectral theory and limit theorems for geometrically ergodic markov processes. In *the 2001 INFORMS Applied Probability Conference*, pages 304–362, 2001.

[56] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tompkins, and E. Upfal. The web as a graph. In *PODS'00: Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 1–10, 2000.

[57] Amy Nicole Langville and Carl Dean Meyer. Updating pagerank using the group inverse and stochastic complementation. Technical report, North Carolina State University, November 2002. Available at `http://www.ncsu.edu/crsc/reports/reports02.html`.

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

[59] Amy Nicole Langville and Carl Dean Meyer. Updating pagerank with iterative aggregation. In Stuart I. Feldman, Mike Uretsky, Marc Najork, and Craig E. Wills, editors, *WWW (Alternate Track Papers and Posters)*, pages 392–393. ACM, 2004.

[60] Amy Nicole Langville and Carl Dean Meyer. Updating the stationary vector of an irreducible markov chain with an eye on google's pagerank. Technical report, North Carolina State University, September 2004. Available at `http://www.ncsu.edu/crsc/reports/reports02.html`.

[61] Amy Nicole Langville and Carl Dean Meyer. *Google's PageRank and beyond: the science of search engine rankings*. Princeton University Press, 2006.

[62] Amy Nicole Langville and Carl Dean Meyer. A reordering for the pagerank problem. *SIAM J. Scientific Computing*, 27(6):2112–2120, 2006.

[63] L. P. Lebedev and Michael J. Cloud. *Tensor analysis*. World Scientific, 2003.

[64] Chris P. Lee, Gene H. Golub, and Stefanos A. Zenios. A fast two-stage algorithm for computing pagerank and its extensions. Technical report, Stanford University, 2004.

[65] Ivo Marek and Ivana Pultarová. A note on local and global convergence analysis of iterative aggregation-disaggregation methods. *Linear Algebra and its Applications*, 413(2-3):327 – 341, 2006. Special Issue on the 11th Conference of the International Linear Algebra Society, Coimbra, 2004.

[66] John H. Mathews and Kurtis D. Fink. *Numerical methods using MATLAB*. Pearson Prentice Hall, 4, illustrated edition, 2004.

[67] C. D. Meyer. Stochastic complementation, uncoupling markov chains, and the theory of nearly reducible systems. *SIAM Review*, 31:240–272, 1989.

[68] Cleve Moler. *Numerical Computing with MATLAB*. SIAM, 2004.

[69] Danil A. Nemirovsky. Analysis of iterative methods for pagerank computation based on decomposition of the web graph. *Master thesis, St.Petersburg State University*, 2005.

[70] Yann Ollivier and Pierre Senellart. Finding related pages using green measures: An illustration with wikipedia. In *Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI 2007)*, 2007.

[71] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia, 1998.

[72] Edward Alexander Parsons. *The Alexandrian Library, Glory of the Hellenic World; Its Rise, Antiquities, and Destructions*. Elsevier, Amsterdam - New York, 1952.

[73] Eugene Seneta. *Non-Negative Matrices and Markov Chains*. Springer, 2006.

[74] James G. Simmonds. *A Brief on Tensor Analysis*. Springer, 1997.

[75] Markus Sobek. Google dance. `http://dance.efactory.de`, 2002. eFactory GmbH & Co. KG Internet-Agentur.

[76] William J. Stewart. *Introduction to the numerical solution of Markov chains*. Princeton Univ. Press, Princeton, NJ, 1994.

[77] Kunio Tanabe and Masahiko Sagae. An exact cholesky decomposition and the generalized inverse of the variance-covariance matrix of the multinomial distribution, with applications. *Journal of the Royal Statistical Society. Series B (Methodological)*, 54(1):211–219, 1992.

[78] Robert S. Taylor. Process of asking questions. *American Documentation*, 13:391–396, 1962.

[79] Samis Trevezas and Nikolaos Limnios. Variance estimation in the central limit theorem for markov chains. *Journal of Statistical Planning and Inference*, 139(7):2242–2253, 07/2009 2009.

[80] Y. V. Volkovich, N. Litvak, and B. Zwart. A framework for evaluating statistical dependencies and rank correlations in power law graphs. Memorandum 1868, University of Twente, Enschede, June 2008.

[81] Xuanhui Wang, Tao Tao, Jian-Tao Sun, Azadeh Shakery, and Chengxiang Zhai. Dirichlet-trank: Solving the zero-one gap problem of pagerank. *ACM Transactions on Information Systems*, 26(2):1–29, 2008.

[82] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

[83] Rebecca S. Wills and Ilse C. F. Ipsen. Ordinal ranking for Google's PageRank. *SIAM J. Matrix Anal. Appl.*, 30(4):1677–1696, 2008.

[84] T.D. Wilson. On user studies and information needs. *Journal of Librarianship*, 37(1):3–15, 1981.

[85] Alexander D. Wissner-Gross. Preparation of topical reading lists from the link structure of wikipedia. In *ICALT*, pages 825–829. IEEE Computer Society, 2006.

RÉSUMÉ

Une des façons de trouver l'autorité des pages sur le World Wide Web est l'algorithme PageRank, qui est basé sur l'hyper-liens entre les pages. D'algorithmes accélérer le calcul du PageRank sont considérés, et, en particulier, les méthodes d'agrégation-désagrégation sont discutées. Conditions d'équivalence de la méthode de la pleine agrégation-désagrégation et de la méthode de la partielle agrégation-désagrégation sont découvertes. Un choix du facteur d'amortissement, l'un des plus importants paramètres de l'algorithme PageRank, n'est pas évidente. Quasi-stationnaire distributions en tant que solutions de rechange sans paramètres de PageRank sont discutées. Une certain nombre de méthodes de Monte Carlo ont été examinées et il a été révélé que cette méthodes peuvent donner une bonne approximation des valeurs du PageRank des pages populaires déjà après une itération. Aussi les méthodes de Monte Carlo ont été examinées dans l'application au PageRank Personnalisé dans le but de découvrir le classement des nombre de pages ayant des taux élevés valeurs du PageRank Personnalisé. Le moments mixtes d'ordre supérieur du nombre de visites de la chaîne de Markov avec absorption sont considérés et la expression compacte et explicite basée sur la théorie tenseur est découverte.

Mots-clés: PageRank, l'agrégation, Monte Carlo, le classement, tenseur, moments, absorbant, chaîne de Markov.

ABSTRACT

One of the way to find authoritativeness of the pages on the World Wide Web is the PageRank algorithm which is based on hyper-links between pages. Algorithms accelerating the PageRank computation are considered, and, in particularly, aggregation-disaggregation methods are discussed. Equivalence conditions of the full aggregation-disaggregation method and the partial aggregation-disaggregation methods are discovered. A choice of the damping factor, one of the most important parameter of the PageRank algorithm, is not evident. Quasi-stationary distributions as a parameter-free alternatives to PageRank are discussed. A number of Monte Carlo methods were considered and it was revealed that such methods can give a good approximation of the PageRank values of popular pages already after one iteration. Also Monte Carlo methods were discussed in the application to Personalized PageRank with the aim to discover the ranking of the number of pages having high Personalized PageRank values. The mixed high-order moments of the number of visits of an absorbing Markov chain are considered and compact closed-form expression based on the tensor theory are discovered.

Keywords: PageRank, aggregation, Monte Carlo, ranking, tensor, moments, absorbing, Markov chain.