Académie de Montpellier

# Université Montpellier II

— Sciences et Technique du Languedoc —

## D I S S E R T A T I O N

presented at the University of Sciences and Technology of Languedoc
in fulfillment of the requirements for the degree of Doctor of Philosophy

Specialty : **Informatique**
*Doctoral Program* : **Informatique**
*Doctoral School* : **Information, Structures, Systèmes**

# Web Graphs
# PageRank-like Importance Measures

by Fabien Mathieu

Defended on December 8, 2004 before a committee of:

M. Serge Abiteboul, Directeur de recherche, INRIA.....reviewer
M. Vincent Blondel, Professor, UCL (Belgium).....guest
M. Pierre Fraigniaud, Directeur de recherche, CNRS.....reviewer
M. Michel Habib, Professor, LIRMM, Montpellier.....thesis advisor
M. Alain Jean-Marie, Directeur de recherche, INRIA.....chair
M. Laurent Viennot, Research Scientist, INRIA.....co-advisor

# *Disclaimer 2026*

I defended my PhD thesis in 2004, using the state-of-the-art LaTeX workflow back in the day: LaTeX to DVI, DVI to PS, PS to PDF. A full compilation took more than five minutes. A few years later, I realized that I could no longer compile the original LaTeX source files to PDF, as too many packages were outdated. Also, my manuscript was only available in French, and I wanted to have an English version as well. So I decided to convert the original LaTeX source files to Typst, a modern typesetting system that is easier to maintain and supports multilingual documents out of the box.

The document you are reading is the result of this conversion, performed in 2026. The original LaTeX source may have contained specific formatting, macros, or packages that do not have direct equivalents in Typst. As a result, some elements of the original document may not be perfectly preserved.

I decided to keep the content of the original document as much as possible, while adapting the formatting to fit the capabilities of Typst. For example, obvious typos have been fixed, but URLs that no longer work have been left as is. This is not a new version of the thesis, but rather a faithful reproduction of the original document in a new typesetting system. The goal is to make the content of the thesis more accessible and maintainable, while preserving the original ideas and arguments as much as possible.

# Table of Contents

# *Acknowledgments*

As I prepare to upload this thesis to the University of Montpellier II server, thereby setting it in stone for all eternity, I cannot help but acknowledge that it would not have been possible without the contributions of many people, to whom I wish to pay a well-deserved tribute.

First of all, even if it is hardly conventional, thank you to the one without whom Fabien Mathieu would be little more than the sum of two first names. Thank you Papa — it is thanks to you that this thesis exists, and it is only natural that I dedicate it to you.

Thank you also to the one I met when I was more or less finishing my DEA, and the thesis was still something strange and otherworldly to me. Without Laurent Viennot to bring me back down to earth, I might still be searching for the Fourier transform of the Web graph. I still do not know how he managed to supervise a PhD student like me, stubborn and living on New York time, but it is thanks to his patience that I am able to write these lines today, and I will be eternally grateful to him. Thank you Laurent — the researcher —, and thank you Laurent — the friend.

I also wish to thank Michel Habib, who agreed to supervise me with full knowledge of what he was getting into and placed his complete trust in me. His unwavering support allowed me to practice Mediterranean-Parisian ubiquity during those few years. May he in turn enjoy the same freedom that I was able to have thanks to him.

Thank you to the reviewers of this thesis for giving me their time and attention, and for managing to endure the delays I imposed on them: thank you to Serge Abiteboul, who revealed the weaknesses of my thesis and helped me correct some of them; thank you to Pierre Fraigniaud for risking even his morality in the service of science.

I also wish to thank the other members of the jury, Alain Jean-Marie, whose corrections were very helpful to me, and Vincent Blondel, whose interest in my thesis particularly touched me. Thank you as well to François Baccelli, who unfortunately had to send his apologies at the last moment.

Thank you to all the researchers, webmasters, and ordinary internet users for all the data they were kind enough to send me by email when I needed it. The list is long, but I have by no means forgotten their contributions — quite the contrary.

Obviously, none of this would have been possible without the support of the countless unsung workers whom I had the opportunity to press into service, particularly during the writing phase. A loyal inner circle, they tirelessly proofread the successive drafts leading up to the present version. If there are fewer than twenty mistaeks per page, it is thanks to them.

The Montpellier team first. Fabien (the other one), whose thesis source code allowed me to go from 0 to 100 pages in less time than it takes to type "copy/paste" with a jackhammer. I cannot help but acknowledge that these acknowledgments would not have been possible without his contributions, and I wish to pay him a well-deserved tribute. Mohamed, for his part, was there for me every time I was not there myself, and I also do not forget that without his programming skills, I would never have crossed the 100 million page mark within the allotted time.

The Parisian team next. Julien, who is unmatched when it comes to making a chocolate mousse with rinse-aid salt for 20 people, and who was kind enough, along with Anamaria, to lend me a base camp for the summer of 2004. Speaking of base camps, a big thank you to Fanfan. Her bed is too hard, but the dreams are beautiful. And I of course do not forget Mathilde, who overcame her handicap (being blonde and a literature student) to give me precious stylistic advice.

I also have a special thought for Danielle Croisy, from INRIA, although I cannot explain how she managed to put up with all those overdue travel orders.

Thank you to all the others whose support I have not forgotten. *Thank you very many* to Mickey and Thibaut for stopping by the LIRMM in December 2004. Thank you to the entire Halle Gorithme, with its rabbits, its crawlers, its pizzas, and its oysters.

Finally, a big thank you to my sister Nathalie, my aunt Marie-Paule, my niece Marion, and my nephew Quentin. After roaming the four corners of France and beyond, one comes to truly appreciate knowing that there is a place where one feels at home.

THANK YOU ALL
AND HAPPY READING!

# Introduction

*The scientific method, which selects, explains, and orders, acknowledges the limits imposed upon it by the fact that the use of the method transforms its object, and that consequently the method can no longer be separated from its object.*

*Heisenberg*

## Search engines today

Since the last few years, the Internet, and the Web in particular, have undergone profound transformations driven by multiple changes in orders of magnitude. Ever more data on ever more machines are accessible to ever more Internet users. The electronic economy has also expanded, and what was yesterday merely an experimental showcase for commercial enterprises has become, often at the cost of unsuccessful attempts, a fully-fledged economic sector. These changes have given rise to new behaviours both among Internet users and site administrators.

On the users' side, the problem that has emerged is finding one's way among the billions of available pages. The average user wants to have the means to access all the resources offered by the Web, and the usual methods (hyperlink navigation from a portal, knowing the right address through extra-Web means) are no longer sufficient.

On the sites' side, the symmetric problem of visibility arises. A site, however well designed, has value only if it is visited, like the tree that falls in the forest. This problem of visibility has many facets, which are the same as for visibility in the real world. The visibility of Mr. Durand's personal site is for him the assurance of being known or contacted by those who wish to do so. The visibility of a commercial site represents money. Being more visible than one's competitors makes it possible to acquire new customers at their expense, which, in a still limited but rapidly expanding market, is a near-necessary condition for survival.

While the two problems of accessibility and visibility are symmetric, they do not necessarily go hand in hand. If an Internet user wants to eat apples, their main concern will be to find Web pages that discuss apples, or even sell them. On the other side, a pear enthusiast's site will want to make itself known to the apple-eating

Internet user, if possible before they have found an apple site, in order to try to change their mind and convert them to pears.

The primary purpose of search engines is to ensure the accessibility of as many Web pages as possible to Internet users. In practice, from the largest possible selection of pages, the engine must return the pages that meet the user's needs, those needs being expressed through a *query*. But they have in fact also become the main medium of visibility for sites.

This strategic position, at the crossroads of the Internet user and the site, makes the search engine the centrepiece of today's Web, and it is no surprise to see that the company *Google*, which holds — to this day — a near monopoly in the search engine sector, is now publicly traded on the stock market.

## Knowing the Web

One of the foremost qualities of a search engine is having a substantial database, both from a quantitative and a qualitative standpoint. In Part I we propose to highlight the issues raised by these databases built by search engines.

We begin by attempting to define the Web, which, as we shall see in Chapter 1, is not as easy as one might initially think. Once this is done, we will be able, in Chapter 2, to study those subsets of the Web known as *crawls*. Crawls naturally carry a graph structure that we will detail in Chapter 3. In particular, we will try to put into perspective the bow-tie model, whose conclusions are too often misinterpreted, and we will demonstrate the existence of a very strong site-level organisation of Web pages linked to the URL decomposition tree. We will show in particular that the canonical structure of sites makes it possible to compute in linear time a good approximation of a certain site decomposition of a Web graph.

## Finding the right pages

More than the size of its database, it is the way a search engine uses it that determines its effectiveness. While for a given query the database will often contain several tens of thousands of pages likely to provide a suitable answer, Internet users will rarely look beyond the first 10 or 20 results returned by the engine. It is therefore necessary to sort through the possible answers in order to extract, automatically, the few pages best suited to a given query.

First-generation search engines used methods of lexical and semantic relevance to perform this sorting: the first pages returned were those that, from the standpoint of semantic analysis, were closest to the query. But fairly quickly, relevance-based ranking was undermined by the sites' need for visibility. Many sites began to pad the semantic content of their pages using techniques of varying ingenuity and honesty, for example by filling the page with white text on a white background. Very often, instead of being directed to the most relevant pages, the Internet user would end up

on pages that had nothing to do with their query, but whose desire for visibility was very strong.

To counter these techniques, new ranking methods were introduced, with the aim of inferring the importance of Web pages from the graph structure formed by the known pages. It is one of these families of ranking methods, the PageRanks, that will be studied in Part II.

Chapter 4 will recall some theoretical foundations on Markov chains, and will develop in particular certain connections between graph theory and the theory of stochastic processes that are necessary for a proper understanding of PageRank. Chapter 5 will provide an opportunity to present a near-exhaustive bestiary of classical PageRank algorithms. In particular, we will present a unified view of PageRank in terms of stochastic interpretation, and highlight the commonalities, differences, advantages, and drawbacks of the various methods.

Finally, in Chapter 6 and Chapter 7, we will present several original PageRank algorithms that we believe can bring significant improvements to existing algorithms.

The *BackRank* algorithm, originally designed to model the use of the *Back* button, has some interesting side effects, such as faster convergence than classical algorithms and the absence of problems related to the existence of dangling pages.

*FlowRank* is an algorithm that attempts to take into account in a fine-grained manner the role of internal pages, external pages, and the *zap* flow in an exact computation of PageRank. *BlowRank* is a practical adaptation of *FlowRank* inspired by another decomposed PageRank computation algorithm.

All these algorithms are potential tools aimed at facilitating the trade-off between accessibility and visibility, but they also make it possible to gain a better understanding of the mechanisms at play when one seeks to model stochastic behaviour on the Web.

## Publications

Here is the list to date of the publications of my work. [Mat01, MV02, MV03a] correspond to the work on Web graphs that forms the basis of Part I of this thesis. [BM03, MB04] are the starting point of Chapter 6, just as [MV03b, MV04] are for Chapter 7. The work on the classification of PageRanks in Chapter 5 as well as the detailed analyses of the *BackRank* and *BlowRank* algorithms are too recent and have not yet been published. Finally, [MR04] is a research report on a model for download problems in peer-to-peer networks, carried out jointly with Julien Reynier. It is a promising subject, but one that still needs to mature, which is why I have simply placed it in the appendix (Appendix C, page 137).

All these documents are available for download at

`http://www.lirmm.fr/~mathieu`

[Mat01] F. Mathieu. Structure supposée du graphe du Web. Première journée Graphes Dynamiques et Graphes du Web, décembre 2001.

[MV02] F. Mathieu et L. Viennot. Structure intrinsèque du Web. Rapport Tech. RR-4663, INRIA, 2002.

[MV03b] F. Mathieu et L. Viennot. Aspects locaux de l'importance globale des pages Web. In *Actes de Algotel03 5ème Rencontres Francophones sur les aspects Algorithmiques des Télécommunications*, 2003.

[BM03] M. Bouklit et F. Mathieu. Effet de la touche Back dans un modèle de surfeur aléatoire : application à PageRank. In *Actes des 1ères Journées Francophones de la Toile*, 2003.

[MV03a] F. Mathieu et L. Viennot. Local Structure in the Web. In *12th international conference on the World Wide Web*, 2003.

[MB04] M. Bouklit et F. Mathieu. The effect of the back button in a random walk: application for pagerank. In *13th international conference on World Wide Web*, 2004.

[MV04] F. Mathieu et L. Viennot. Local aspects of the Global Ranking of Web Pages. Rapport Tech. RR-5192, INRIA, 2004.

[MR04] F. Mathieu et J. Reynier. File Sharing in P2P: Missing Block Paradigm and Upload Strategies. Rapport Tech. RR-5193, INRIA, 2004.

Part I

# Structures of the Web

# Chapter 1
# What is the Web?

*I shall speak, for instance, of the Mind and of the "demiurge," while carefully refraining from defining too precisely what I mean by these terms: because I do not clearly know… In this dreadful state of our ignorance, must we really define so precisely?*

*Rémy Chauvin, la Biologie de l'Esprit*

## 1.1 Genesis of the Web

Sources: *A History of Networks* [Gen04].

Around the early 1960s, while a melody drifted through the sky (a bird called Sputnik), the United States decided to develop a form of decentralised network capable of withstanding a nuclear attack that would destroy one or more of its nerve centres. It was the year sixty-two.

A few years later, this project became *ARPANET*. It was 1969 and four American universities were connected by this network of a new kind. From that moment on, the network never stopped growing and evolving, until it became what is now called the Internet[1], that is to say a formidable network of networks.

In 1972, the *Acceptable Use Policy (AUP)* charter prohibited any commercial entity from connecting to the network.

In 1984, the milestone of 1,000 machines was reached and *Centre Européen pour la Recherche Nucléaire (CERN)* joined the Internet. Six years later, in 1990, while the number of connected computers reached 300,000, the largest Internet site in the world

[1]The term Internet was apparently introduced in 1974 by Vincent Cerf and Bob Kahn. Note in passing that internet and Internet do not mean the same thing! One is a common noun designating a meta-network structure, the other is a proper noun for THE meta-network using the TCP/IP protocol.

was that of CERN, the future birthplace of the Web, a vast worldwide collection of so-called hypertext and hypermedia documents distributed over the Internet.

That same year, 1990, the AUP ceased to exist, paving the way for what would become, a few years later, the *Dot-com Bubble*.

In 1991, Tim Berners-Lee of CERN introduced the concept of the *World Wide Web (WWW)*, sometimes referred to simply as the *Web*. The World Wide Web is the part of the Internet where the navigation method is HyperText and the protocol is *HyperText Transfer Protocol (HTTP)*.

The philosophy of HTTP lies in so-called *hypertext* links that connect pages to one another and allow navigation when selected. We speak of the "Web" — with a capital letter — even though it is in reality the "World Wide Web" or "W3".

## 1.2 A Definition of the Web

Originally, as we have just seen, the *Web* was characterised by both a protocol, *HTTP*, and a language, *HyperText Markup Language (HTML)*; the former serving to deliver "pages" (files) written in the latter, interpreted on the client side by Web browsers (*Browsers*).

Nowadays, one may question the validity of this dual characterisation:

- *HTML* is an easy-to-learn language for writing structured documents linked to one another by hyperlinks. In practice, it is no longer used exclusively through *HTTP* and can be found on many other media (CD-ROM, DVD-ROM…) for purposes as numerous as they are varied (documentation, education, encyclopaedia, help…).
- In order to deliver multimedia content, *HTTP* is designed to serve any type of file. Images, sounds, videos, texts in various formats, archives, executables… are all accessible through the *HTTP* protocol, in a spirit more or less removed from the original hypertext navigation concept. This tendency toward "all-HTTP" leads to sometimes paradoxical situations. For instance, to transfer files (even large ones), there is a certain tendency to abandon the appropriate protocol, *File Transfer Protocol (FTP)*, in favour of *HTTP*. As shown in Table 1.1[2], it appears that files traditionally transferred via *FTP* are now transferred via *peer-to-peer (P2P)*, but also via *HTTP* (for example, most download servers on *sourceforge.net* use *HTTP*).
- Certain documents that are not *HTML* documents allow hyperlink navigation: proprietary documents (Acrobat PDF, Flash…) or new languages (*WML*, *XML*…).

One can glimpse the difficulty of finding a "good" definition of the Web. For the sake of simplicity more than anything else, throughout this thesis we shall continue

[2]Many thanks to Philippe Olivier and Nabil Benameur for providing me with these data.

| **Year** | ***HTTP*** | ***FTP*** | ***P2P*** |
|---|---|---|---|
| 2001 | 13 % | 10 % | 35 % |
| 2002 | 14 % | 2 % | 50 % |
| 2004 | 20 % | *negligible* | 65 % |

Table 1.1: Evolution of *HTTP*, *FTP*, and *P2P* traffic (as a percentage of volume)

to define the Web according to its initial dual characterisation. Thus, we shall call *Web* the set of documents written in *HTML* and available on the Internet via the *HTTP* protocol. We are aware of the extremely restrictive nature of this definition, but prefer to work on a well-defined and widely studied set rather than to seek an exhaustiveness that may not even be attainable.

## 1.3 Accessibility of the Web

Within the Web we have just defined, the problem of visibility and accessibility of pages now arises. What can we see of the *Web* and how can we access it? Several structurings of the Web based on these questions of visibility, accessibility, and indexability have been proposed.

### 1.3.1 Depths of the Web

Michael K. Bergman, in 2000, proposed the metaphor of depth to distinguish the different *Webs* [Ber00]. One thus distinguishes:

**The Surface Web** the surface of the *Web*, according to Bergman, consists of all static and publicly available pages.

**The Deep Web** conversely, the deep *Web* consists of dynamic websites and databases accessible through a Web interface.

This vision of the *Web* remains rather Manichaean. Danny Sullivan [Sul00] proposes a third kind of *Web*, *The Shallow Web*[3], made up for instance of publicly available dynamic pages, such as those of the *Internet Movie Database (IMDB)* (`http://www.imdb.com`), or those of the site `http://citeseer.ist.psu.edu/`.

### 1.3.2 Visibility of the Web

Chris Sherman and Gary Price propose in their book *The Invisible Web* [SP01] an approach based on visibility by the major search engines. The equivalent of the

[3]To remain faithful to Bergman's analogy, one could translate *Shallow Web* as *near space*. I prefer the term *swamp*, which conveys rather well the effect of this zone of the Web on crawlers.

*Surface Web* is, for Sherman and Price, the set of pages indexed by search engines. According to them, the rest of the Web then breaks down into 4 categories:

**The Opaque Web** pages that could be indexed by search engines but are not (limitation on the number of pages indexed per site, indexing frequency, missing links to pages thus preventing crawling).

**The Private Web** web pages that are available but voluntarily excluded by webmasters (password, metatags, or files in the page to prevent the search engine robot from indexing it).

**The Proprietary web** pages accessible only to authorised persons (intranets, authentication systems…). The robot therefore cannot access them.

**The Truly Invisible Web** content that cannot be indexed for technical reasons. For example, format unknown to the search engine, dynamically generated pages (including characters such as ? and &)…

### 1.3.3 Accessible Web

Each of the two approaches we have just seen has its advantages and disadvantages. While Bergman's definition is fairly appealing (on the one hand, a static Web accessible by clicks; on the other, a dynamic Web reachable only through queries), it does not realistically describe the current Web. The approach of Sherman and Price, namely discrimination based on indexability by search engines, is more flexible, but does not, in my opinion, always associate the right causes with the right effects[4].

A third approach, used by [BB98, Dah00, Hen+99], provides a kind of synthesis of the models just mentioned. It is the model of the *accessible* Web:

**Definition 1.1**: The accessible Web is the set of Web pages that may be pointed to by a hyperlink.

More precisely:

- This is equivalent to considering as part of the accessible Web any page that can be accessed simply by typing the correct address — also referred to by the term Uniform Resource Locator (URL) [BMM94] — into a browser.
- Dynamic pages that do not have hidden variables, that is, whose possible variables are passed in the URL, are part of the accessible Web.

By convention, we shall exclude certain pages from our definition of the accessible Web:

- Error pages returned by a server (*4xx* errors, for example);
- Pages whose access by robots is forbidden by a `robots.txt` file;

[4]For example, all dynamic pages accessible from `http://www.liafa.jussieu.fr/~fmathieu/arbre.php` [GLM02] should logically belong to the *opaque* Web, whereas the categorisation of Sherman and Price seems to consign them to being completely invisible…

- Pages protected by login and/or password (even if the login and password can be given in the URL).

This definition obviously has its flaws as well. For example, it does not take into account at all the problem of duplicates (how should one consider two pages with strictly identical content — for instance two URLs corresponding to the same physical file?), nor that of the temporal dynamicity of pages and their content (what does the accessibility of the front page of a newspaper mean? The accessibility of a page that returns the time and date?). Strictly speaking, we should thus speak of the Web accessible at a given instant $t$, and accept identifying a page with its URL despite the inevitable redundancies. Even so, many grey areas persist. For instance, some ill-intentioned administrators do not return the same content depending on whether the requester of the page is human or not, in order to deceive search engines; others return pages adapted to the browser used by the visitor; still others check that the visitor has indeed passed through the home page before navigating within the site, and redirect them to the home page if this is not the case; finally, because of routing problems, it is entirely possible that at a given instant $t$, a server is perfectly visible from one IP address and inaccessible from another. In addition to the instant $t$, the address from which the request is made and all the information transmitted in the *HTTP* request must therefore also be defined.

## 1.4 Intermezzo: the page that linked to all pages

During a discussion with Jean-Loup Guillaume and Matthieu Latapy, in that hallowed place of scientific research known as the coffee machine room, while we were arguing at length about the *bow-tie* model proposed by Broder *et al.* [Bro+00][5], a whimsical idea fell from the cup: what if we set up a Web page capable of linking to all other pages?

After being first written by Matthieu Latapy, whose code I then took over, a somewhat improved version of the original page is now available at `http://www.liafa.jussieu.fr/~fmathieu/arbre.php` [GLM02].

The principle of this page is that of a typewriter, or rather a click-writer. To reach a given page, one simply clicks its letters one by one, the resulting address being kept in memory through a variable passed as a parameter in the URL. The page dynamically checks whether this address makes sense (whether it belongs to the indexable Web), and if so, a hyperlink is inserted to that address[6]. Figure 1.1 is a screenshot of the *page that links to all pages* in action.

[5] See Section 3.2, page 32.

[6] The *page that links to all pages* also displays the Google PageRank (a score from 0 to 10) when available. It does not yet make coffee, alas.

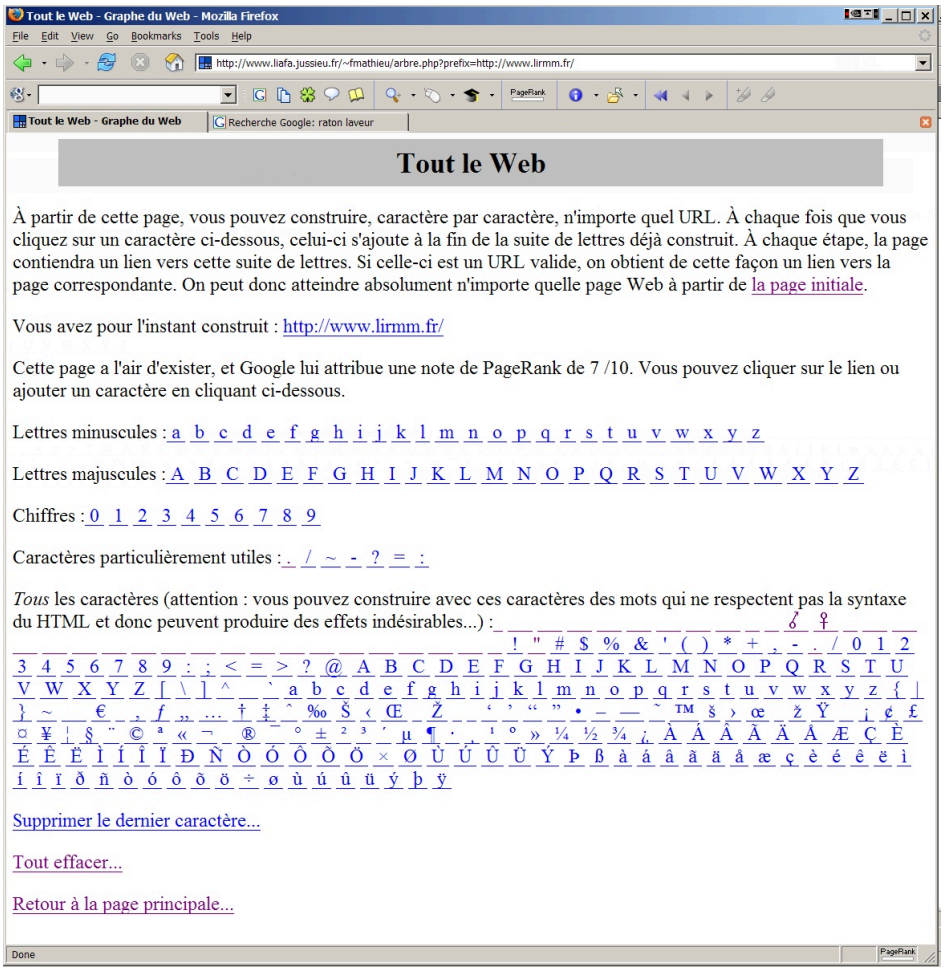



Figure 1.1: Screenshot of *the page that links to all pages* in action

Up to URL length limitations (and character escaping bugs that have not yet been fixed), the page's purpose, namely to be connected by hyperlinks to virtually every page of the indexable Web, is achieved.

This page is above all a playful exercise, but it allows one to take a step back from a good number of received ideas:

- The primary goal of this page was to deliver a small jab at the interpretation generally given to the bow-tie model, and I believe this goal was achieved.
- It allows one to take with a minimum of hindsight all claims about the structure of the Web. After all, I can claim to know a portion of the Web of approximately $256^{n-55}$ pages, give or take a factor of $10^{10}$, where $n$ is the maximum number of characters that a URL can contain[7]. This result, far exceeding all estimates of the Web's size put forward so far (see Section 2.2), also allows one to assert some astonishing statistics:

[7]The HTTP 1.1 protocol does not specify a maximum URL length (cf [Fie+99]). $n$ is therefore in principle as large as one wishes. In practice, each server has a fixed limit, on the order of a few kilobytes. Moreover, there was a time when sending a URL longer than 8 KB was an effective way to crash an *IIS* server...

- The average degree of the *Web I know* is $257 - \varepsilon$, where $\varepsilon$ is a perturbation due to *real* pages. Moreover, contrary to everything previously believed, the degree distribution does not follow a power law but closely resembles a Dirac delta.
- There exists a strongly connected component in the *Web I know*, and any page in the *Web I know* almost certainly belongs to this component.

Of course, these results should not be taken at face value! I would be the first to find suspicious a paper claiming that the *Web* has more than a googol of pages[8], or that heavy-tailed distributions do not exist. The *page that links to all pages* is merely a *Saturday night idea*[9] that was put into practice, and I am perfectly aware of this. But it has the advantage of showing us in full light the immensity of the accessible Web (and in particular the impossibility of indexing it[10]) and of urging us to understand clearly that all meaningful results one can obtain about the *Web* in fact concern crawls that represent an infinitesimal fraction of the *indexable Web* in terms of the number of pages.

To conclude this *intermezzo*, let us note that in terms of information theory, we doubt that the *page that links to all pages* and its dynamic pages are worth more than the few lines of code that lie behind them.

[8] A googol is a number equal to $10^{100}$. It does not honour the Russian writer Nikolai Gogol (1809–1852), but was coined in 1938 by the nephew of the American mathematician Edward Kasner (1878–1955), Milton, who was then 9 years old. Note that in French, googol is written… gogol! The name of the famous search engine is directly derived from the name invented by Kasner; there is in fact a legal dispute between Google and Kasner's heirs. Finally, let us mention that the number represented by a 1 followed by a googol of 0s is called googolplex.

[9] I borrow this concept of a *Saturday night idea* from Michel de Pracontal. It is "the kind of idea that researchers sometimes discuss during their leisure hours, without taking them too seriously" (L'IMPOSTURE SCIENTIFIQUE EN DIX LEÇONS, lesson 7, page 183).

[10] To give a sense of scale, performing a breadth-first traversal of the *page that links to all pages* at a rate of 10 pages per second, it would take approximately 100 million years to manage to type `http://free.fr`. As for the address `http://www.lirmm.fr`, the age of the universe is far too short for a robot to find it solely by crawling the *page that links to all pages*.

# Chapter 2
# Crawlers and Sizes of the Web

*The world is medium-sized.*

*Michel Houellebecq,* Lanzarote

*Size isn't everything. The whale is endangered while the ant is doing just fine.*

*Bill Vaughan*

*One must hide depth. Where? On the surface.*

*Hugo von Hofmannsthal*

Now that we have defined a framework on which to work (the accessible Web), the question of which objects we will study arises. From a mathematical point of view, the accessible Web can be countably infinite — according to RFC 2616 [Fie+99] — or finite — since there exists only a finite number of servers and each has a maximum URL length. In any case, it satisfies a certain physicist's definition of infinity, which is the one we shall use here: *large compared to the number of atoms in the universe*. The accessible Web indeed contains many nearly infinite nests of pages. The page that points to all pages, of course, but also misconfigured servers, or even commercial sites that, by creating an *infinity* of pages, try to boost their PageRank (see Part II, Section 7.5). To give a concrete example, during the presentation of their paper *Extrapolation methods for accelerating PageRank computations* [Kam+03a], the authors recounted how one of their experiments was disrupted by a German website that accounted for 90 % of the pages they had indexed. Upon investigation, the site in question turned out to be a pornographic website that hoped, through an infinity of tightly interlinked pages, to achieve a high ranking in search engines (the principle of *Free For All* pages, also known as *link farms* or *nurseries*).

In conclusion, it is impossible to index *all* of the accessible Web, just as talking about

its size no longer makes sense[1]. In the absence of physical access to servers, the only tangible data at our disposal are therefore *crawls*, that is, the regions of the Web actually discovered, and possibly indexed, by commercial enterprises (search engines) or public institutions. The purpose of this chapter is to define *crawls* precisely, to give orders of magnitude, and to see to what extent it is possible to compare *crawls* or to measure a certain quality.

## 2.1 Web Crawlers

*Crawlers*, or *spiders*, or *agents*, are the trawlers that traverse the Web in order to generate these pieces of the Web called *crawls*. There are two main types of crawlers: static crawlers and dynamic crawlers. The principle behind all static crawlers is the same: starting from an initial set of pages, they analyze the hyperlinks contained in these pages, attempt to retrieve the pages pointed to by these hyperlinks, and so on, thereby recovering an ever-growing portion of the pages of the accessible Web. Each page is *a priori* retrieved only once, and the process is stopped when deemed appropriate (enough pages retrieved, growth having become negligible…). The *static* crawl thus obtained consists of:

- a set of known pages, which is the union of the initial set and the set of discovered pages;
- a set of indexed pages, a subset of the known pages consisting of pages actually visited by the crawl;
- a set of hyperlinks originating from indexed pages and pointing to known pages.

A dynamic crawler, on the other hand, is not designed to stop, but to perpetually traverse the Web and periodically revisit the pages it has visited [APC03].

What we call a crawler is in fact the collection of software and hardware resources used to carry out a crawl. What determines the particular crawl that a crawler produces?

- The date on which the crawl begins.
- The set of seed pages.
- The exploration strategy for new pages (in what order should which pages be retrieved?).
- Technical limitations: bandwidth, storage and processing capacity, available time…

This applies to all types of crawlers, even if asymptotically, only the exploration strategies and technical limitations play an important role for dynamic crawlers.

[1]One may note that while, a few years ago, estimating the size of the Web was a *fashionable* topic (see [Ber00, BB98, LG98, LG99, Mur00]), no one has ventured into it for two years, and the viewpoint expressed here now seems widely shared (see in particular [Dah00, EMT04]).

### 2.1.1 Creation of the initial page set

It goes without saying that choosing a good seed set is fundamental to a crawler's efficiency. For commercial reasons, it is difficult to obtain the seed sets of major search engines. Many people agree, however, that major search engines use:

- a *well-chosen* subset of the pages obtained from previous crawls [CGP98]. The home pages of *directories*, in particular, seem to be prime candidates for inclusion in the initial set [Cra04];
- new pages submitted through one of the many existing *referencing* methods (sometimes paid);
- it is possible that some more or less clever techniques are also used to discover new pages, such as analyzing Web server logs, or attempting to traverse up the URLs tree of "isolated" pages (rumors read on various pages of the site `http://www.webrankinfo.com`).

To conclude regarding the choice of the initial set, let us note that it is this set that largely determines the bow-tie structure of a *Web graph* (see Section 3.2).

### 2.1.2 Exploration strategies

An ideal crawler that could retrieve and process an infinite number of pages per unit of time would have no difficulty obtaining the entire Web theoretically reachable by hyperlinks from the initial set. All exhaustive traversal methods (breadth-first or depth-first, for example) would yield the same optimal result. But real crawlers are limited by their bandwidth, their processing capacity, and the number of requests per server per minute, which must account for both politeness rules and the risk of being banned.

All these constraints mean that, at any given moment, there exists a finite limit to the number of pages that a specific search engine can index, and it seems that this number has always been, regardless of the search engine considered, considerably smaller than the finite estimates of the Indexable Web [Ber00, BB98, Hen+99, LG98, LG99].

Each engine therefore has a physical barrier to the number of pages it can index. For example, according to Google, this barrier is approximately 4 billion documents. The exploration strategy (coupled with the choice of the initial set) will determine which 4 billion pages these will be, and thus to a large extent the quality[2] of a crawl. A search engine whose agents got lost in nurseries and other *pages that point to all pages* would surely have little success. This is why some engines, such as *AltaVista* and *Teoma*, avoid the practice of *Deep Crawling*[3], while others, such as Google, maintain a blacklist of nurseries.

[2] In a vague sense…

[3] *Deep Crawling* consists of trying to retrieve the maximum number of web pages from a given site.

The strategy can also negatively affect an engine's barrier if it does not seek to optimize available resources. For example, due to the limit on the number of requests per site, the bandwidth consumed by exploring a single server is often negligible compared to the available bandwidth, hence the near-necessity of exploring multiple servers in parallel. For the same reasons, it is always better to have a fast server among those currently being queried.

With all these "physical" constraints taken into account, two main philosophies can be distinguished among known exploration strategies.

- Traditionally, search engines seem to have employed a *breadth-first* exploration, with pages discovered first being explored with priority (subject to the constraints described above) [Bro+00].
- Exploration methods inspired by random walks are also being developed (see [Hen+99] and, to some extent, the *greedy* strategy of [APC03]). These methods have the advantage of more easily retrieving pages that are important in the PageRank sense [Hen+99], or even of estimating the PageRank dynamically [APC03]. The connections between PageRank and random walks will be developed in Part II.

## 2.2 Sizes and Evolution of Crawls

Now that the concept of a crawl has been roughly explained, we will provide some statistics on the orders of magnitude involved when discussing crawls of the major search engines.

### 2.2.1 According to the operators

These data were mostly obtained from the site http://searchenginewatch.com/.

Figure 2.1 presents the claimed sizes of the five billion-page search engines as of September 2, 2003. If we study the respective evolution of the different search engines, several major periods of upheaval can be distinguished, corresponding to periods of intense competition among engines. These *Crawl Wars* are shown in detail Figure 2.2. Each of these wars resulted, for the winner or winners, in the crossing of a psychological threshold.

Between December 1997 and June 1999, the first *Crawl War* took place, at the end of which AltaVista surpassed the 150 million mark, closely followed by NorthernLight (Figure 2.2b).

The second *Crawl War* (September 1999 – June 2000) was marked by a tight match between AltaVista and AllTheWeb, which concluded with the crushing victory

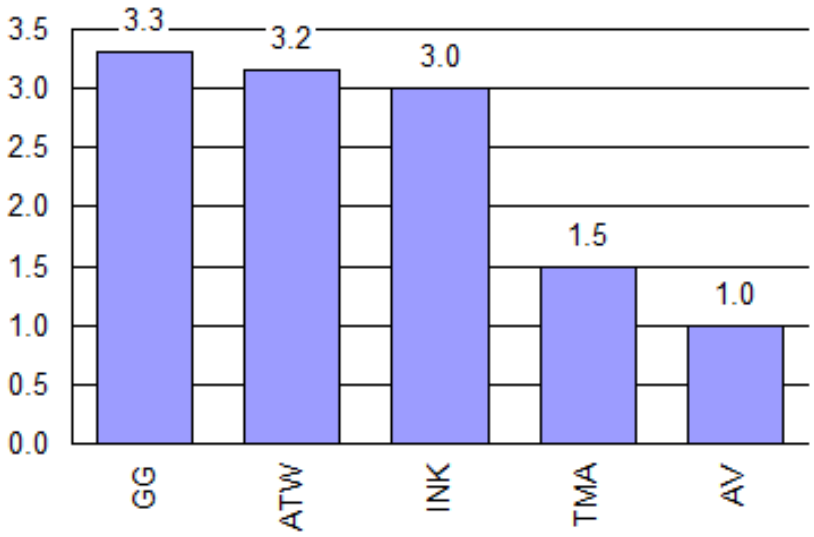



Figure 2.1: Number of pages claimed by various search engines as of September 2, 2003

of… Google, which went on to surpass one and a half billion claimed pages, setting a new standard (Figure 2.2c).

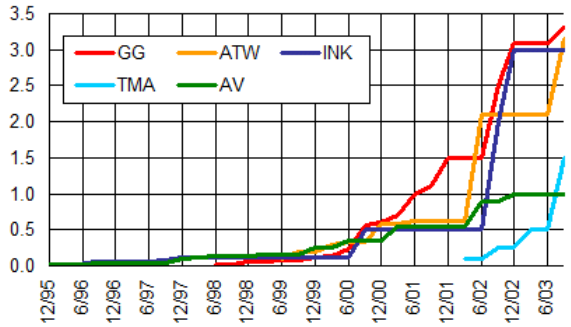
The third war (June 2002 – December 2002) began with a brief period of AllTheWeb in the lead. Google and Inktomi responded almost immediately, and within six months crossed the 3 billion web page mark. A few months later, AllTheWeb caught up (Figure 2.2d).

Currently (summer 2004), Google claims to index more than 4 billion pages, but the battlefield has shifted. The battle between Google and Yahoo (which outsourced to Google until February 2004, and has since used Inktomi's database) is attempting to become more subtle, the goal being to provide "the most understandable and most relevant" results. Yahoo is reportedly preparing its own version of Google's PageRank, called WebRank.

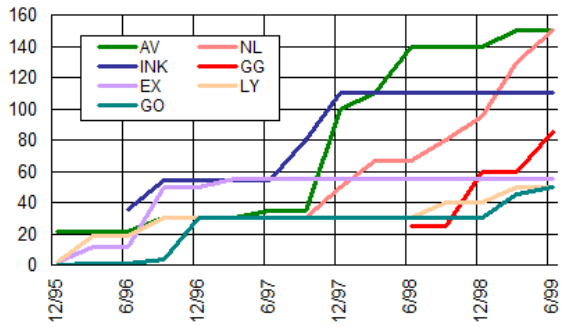
### 2.2.2 According to the auditors

All the figures we have just seen are those announced by the search engines themselves. Since we are in a context of commercial competition, it is legitimate to question the reliability of these figures, with the challenge of defining an adequate methodology.

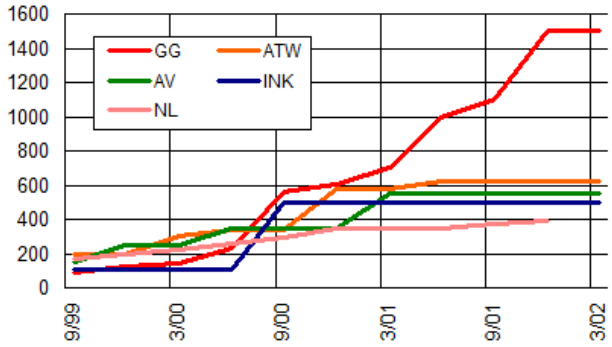
The site *SearchEngineShowdown* [Sea], administered by Greg Notess, proposes a rough method for estimating the effective size of search engines. The principle is to submit several queries in parallel to the different search engines one wishes to compare, and to postulate that the number of results returned by each engine is, to a first approximation, proportional to the number of URLs known by that search engine. If the actual size of one of the engines is known, then a simple rule of three can be used to estimate the actual size of the others. Now, in December 2002, it was



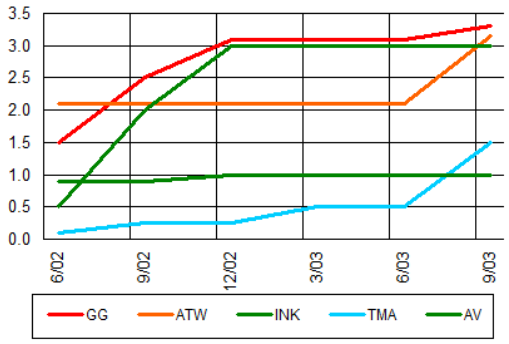

(a) Evolution of search engines (12–95 - 09–03)




(b) Crawl War I (12–97 - 06–99)




(c) Crawl War II (09–99 - 06–00)




(d) Crawl War III (06–02 - 12–02)

Figure 2.2: Crawl Wars. Legend: AV=AltaVista, ATW=AllTheWeb, EX=Excite, GG=Google, GO=GO/Infoseek, INK=Inktomi, LY=Lycos, NL=Northern Light, TMA=Teoma. Unit: 1 million pages (Figure 2.2b and Figure 2.2c) or 1 billion pages (Figure 2.2a and Figure 2.2d).

possible to know the exact size of the AllTheWeb engine simply by submitting the query `url.all:http`[4].

Figure 2.3 compares the claimed and estimated sizes (using Greg Notess's method) of several search engines. Three categories of search engines can be easily distinguished, which reveal both the possible exaggerations of the engines and the bias introduced by the queries:

**The realists** *Google*, *AllTheWeb*, and *WiseNut* have estimated and claimed values that are extremely close, and tend to prove that the estimation method has a certain reliability.

**The sites that wanted three billion** The estimated sizes of HotBot and MSN Search are in sharp contradiction with the claimed values. These are engines based on Inktomi technology. According to Inktomi, not all the computers forming their database are accessible at the same time, so one can never access the entire database. Moreover, it is possible that Inktomi's partners (HotBot and MSN Search in this case) use only a portion of the database, for practical or commercial reasons. For Greg Notess, these estimates therefore reflect the

[4]Thanks to Greg Notess, webmaster of the SearchEngineShowdown site, for sharing this method with me, even though it no longer works today. The query *url.all:http* only works with engines using Fast technology, which has no longer been used by major engines since April 1st, 2004 (and this is not a hoax…).

portion of the database actually available at the time the queries were made. It is clear that the bias of Notess's method prevents us from making categorical assertions, but one cannot help suspecting that the figure of three billion is a publicity response to Google's three billion (recall: we are at the end of the third *Crawl War*).

**The modest ones** AltaVista, Teoma, NLResearch, and Gigablast seem to claim much less than their actual size, which does not seem very logical. According to Greg Notess, the explanation for this paradox lies in how known but non-indexed pages are counted. Indeed, the sites we called *realists* seem to claim their known URLs, whether indexed or not, while the *modest ones* restrict themselves to pages actually indexed. When one knows that non-indexed pages represent at least 25 % of known pages, but less than 1 % of query results[5] (source: *SearchEngineShowdown*), we have a beginning of an explanation for the phenomenon.

### 2.2.3 Personal remarks

Before going further, I would like to highlight a personal observation regarding the extreme volatility of the figures put forward by search engines. For example, while the number of URLs claimed by Google is four billion, the query "the"[6] promises

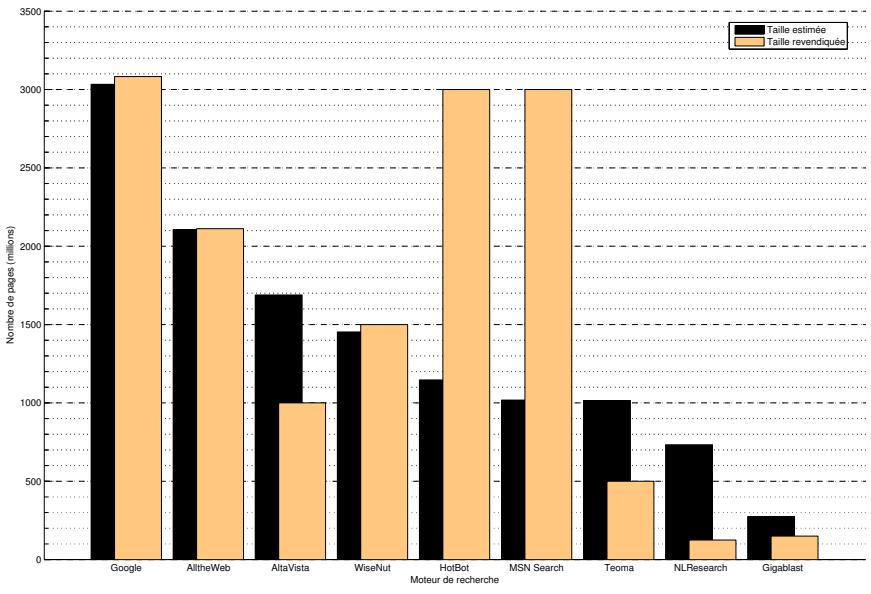



Figure 2.3: Estimated and claimed sizes of various search engines as of December 31, 2002 (source: [Sea])

[5]There do exist, however, queries returning only non-indexed pages. I recommend, on *Google*, the query `site:.xxx`, which returned, in August 2004, 765 non-indexed pages (since they were non-existent, the TLD .xxx not existing at the time of the experiment), of which only 495 were relevant…

[6]http://www.google.com/search?hl=fr&ie=UTF-8&safe=off&q=the&btnG=Rechercher&lr=

5 840 000 000 pages. On the other hand, it is in theory possible to know the number of known pages for a specific domain (for example `.fr`) by using a query of the type `site:tld`. By performing this operation on all TLDs, one should logically arrive at close to six billion pages, or more. However, the experiment yields only 749 485 183 (to within 0.26%, *Google* giving only 3 significant figures). One must conclude from these two tests either that, at the time of the experiments, Google indexed at least 5 840 000 000 pages and at most 749 485 183, or that more than 5 billion indexed pages do not belong to existing TLDs, or that the figures announced by Google should be taken with a degree of caution.

The new *Yahoo* search engine also gives sometimes peculiar results. For instance, I observed that the number of results could depend on the rank from which one wished to view the results. Thus, while the query `"pâte à crêpes"` returns 7480 results when displaying the first 20 pages, this number drops to 7310 when requesting pages 981 to 1000, with a regular decrease. Experimentally, the difference can reach 5% of the announced figure.

Finally, one must not forget to mention the problem of duplicates. Indeed, most major engines treat, for their indexes and algorithms, `www.domain.com/`, `domain.com/`, and `www.domain.com/index.html` as different pages. The grouping is done *a posteriori*, using hashing methods to identify identical pages (which *Google* euphemistically calls *pages with similar content*). In summary, duplicates are counted in the number of pages found, but are not returned in the results (unless explicitly requested by the user). To fix ideas, Table 2.1 presents the results of a few queries for which it is possible to know the number of pages excluding duplicates[7]. By convention, we have chosen to define the number of pages claimed by *Yahoo* as the one reported when requesting the last pages.

| | Claimed pages | | Unique pages | |
|---|---|---|---|---|
| **Query** | **Google** | **Yahoo** | **Google** | **Yahoo** |
| Anticonstitutionnellement | 752 | 795 | 442 | 485 |
| Raton laveur | 18200 | 23500 | 801 | >1000 |
| “Pâte à crêpe”* | 2690 | 1630 | 591 | 489 |
| “Pâte à crêpes”* | 7900 | 5950 | 745 | 971 |
| Webgraph | 9040 | 8180 | 603 | 759 |

Table 2.1: Claimed pages or pages with similar content? A few examples as of August 28, 2004. *Note that according to the *Petit Robert*, one should write *pâte à crêpe* and not *pâte à crêpes*.

[7]Since *Yahoo* and *Google* only return the first 1000 results of a query, it is only possible to count duplicates when the number of unique pages is less than 1000. One then simply needs to request the last 10 results to obtain the desired information.

## 2.3 Measurements on Crawls

The study and comparison of different crawls is a delicate subject to address. On one hand, there are commercial crawls produced by search engines, which can generally only be accessed through public queries made on the engines themselves, queries that are very often throttled. On the other hand, research laboratories often offer smaller sizes and extremely varied crawling methods. Because of these disparities, it can be difficult to compare different results, or even to know whether they are comparable. The purpose of this section is to catalogue different methods for evaluating crawls, in the case of both private and public data.

The method for estimating actual sizes proposed by Greg Notess (cf Section 2.2.2) can also serve as a measure of crawl quality. By counting the responses to 25 representative queries, one measures in a sense the ability of an engine to respond to these queries, effectively turning the query bias into a quality measure.

### 2.3.1 Overlap techniques

This system of comparison by queries is used in a more advanced way in the overlap comparison method, introduced by Bharat and Broder in 1998 [BB98] and revisited by Lawrence and Giles [LG98, LG99].

The principle of the overlap method is as follows: to measure the overlap between the respective indexes of two engines. For example, if one can determine that a fraction $p$ of an index $A$ is in $B$, and that a fraction $q$ of $B$ is in $A$, then one can deduce that $|A \cap B| = p|A| = q|B|$, and thus obtain the size ratio between the indexes:

$$\frac{|A|}{|B|} = \frac{q}{p} \tag{2.1}$$

The whole challenge is to estimate $p$ and $q$ in the absence of a freely accessible public index. This is where query-based sampling comes in: queries created by a random generator as uniform as possible are submitted to an engine. Each time, one of the results among the first 100 is chosen at random, and it is verified, using more or less strict methods, whether this result is in the other engine's index.

Compared to Notess's method, two additional sources of bias (in addition to the bias due to query selection and standard sampling biases) are introduced:

**Ranking bias** By choosing samples only from the first 100 results (a constraint imposed by the search engines themselves), the rankings (importances) of the pages considered are statistically higher than average.

**Verification bias** There is no perfect method for checking whether a given page belongs to an index. The various methods proposed by Bharat and Broder are based on analyzing a query supposed to define the page, and the validation

criteria range from *the engine returns the correct page* to *the engine returns a non-empty result.*

Overlap measurements had their heyday during the Crawl Wars[8] (see Section 2.2), but while they have the advantage of introducing a rigorous formalism, the cost in terms of bias and the relative complexity of the method mean that Notess's method remains very competitive.

### 2.3.2 Sampling by random walk

An interesting variant of Bharat and Broder's sampling method was proposed by Henzinger *et al.* [Hen+99]: rather than measuring one engine using another, the principle is to use as a yardstick a personal crawl derived from a random walk. This random walk does not provide a uniform sample of Web pages[9], but a sample that, up to statistical biases, obeys the probability distribution associated with the random walk. This distribution over Web pages, better known as the PageRank (see Part II), will allow, by measuring the fraction of samples known to the engine using overlap techniques, to estimate the amount of PageRank contained in the pages known to the engine.

The specific bias of this method is that behind the random walk sampling lies an implicit crawl, the crawl that would be obtained by letting the random walk run long enough. One therefore measures the PageRank of the engine under evaluation relative to the PageRank on this virtual crawl. In particular, any divergence between the crawl strategies of the random walk and those of the engine is detrimental to the evaluation of the latter: if the engine has indexed parts of the Web that the random walk, for various technical reasons, has never touched, those parts will not count in the evaluation. Conversely, if certain sites are deliberately avoided by the engines (nurseries, for example, see Section 2.1.2, page 23), this omission will be penalized in the evaluation.

[8]To go so far as to say that certain papers primarily served to show that AltaVista had the largest database is only a small step that I shall take Section 3.2…

[9]Choosing a Web page *at random* is impossible, the indexable Web being infinite.

# Chapter 3
# Graphs and structures of the web

*Life is an ephemeral butterfly bearing the wings of paradox.*

*Benoît Gagnon*

*Places do not change their appearance as men change their faces.*

*Tacitus*

*A good website is always under construction.*

*Anonymous*

Now that the concepts of the Web and of crawling have been clarified, we can study what can be said about them. Since this chapter is devoted to structures, it is appropriate to place a canonical structure on the crawls we wish to study. One of the simplest and most natural structures in the case of Web crawls is the graph structure. Web graphs, a definition of which is given in Section 3.1, are a frequent subject of study. After critically analyzing one of the best-known models, the butterfly model (Section 3.2), we will highlight in the remainder of this chapter the importance of the site structure in Web graphs.

## 3.1 Web graphs: definition

Consider a crawl $C$. This crawl consists of a set of URLs, some of which have been visited, and others merely detected through the hyperlinks of visited pages. It also contains the set of hyperlinks from visited pages. We will call the graph of crawl $C$ the directed graph $G = (V, E)$, where $V$ is the set of pages in the crawl, whether visited or not, and such that an arc $e \in E$ connects a page $i$ to a page $j$ if, and only

if, $i$ contains a hyperlink pointing to $j$. By convention, anchors[1] are ignored, and multiple links add nothing to the graph (if a page $i$ has two hyperlinks pointing to $j$, the graph will still have only one arc from $i$ to $j$). Likewise, links from a page to itself will not be taken into account.

By abuse of language, we will often use the term *Web graph* to designate such a graph $G$. One should always remember that these are in fact always crawl graphs.

## 3.2 The butterfly model revisited

The butterfly structure of the Web graph is today accepted by many people, and is very often mentioned in scientific articles [APC03, Ara+01, CF02, Kam+03b, LM04, RG03] or in popular science publications [GL02], sometimes in a completely irrelevant manner[2]. I consider for my part that the butterfly model applies poorly to dynamic graphs in general, and to Web graphs in particular. This section, whose ideas come from [Mat01], will try to justify this position by attempting to distinguish myth from reality in order to identify the true contribution of the article *Graph Structure in the Web*.

### 3.2.1 The butterfly model

*Graph Structure in the Web* [Bro+00] is an article that was presented at the 9th International Conference on the World Wide Web. The article is based on the analysis of two crawls provided by the AltaVista search engine, a crawl of 203 million pages dated May 1999 and one of 271 million pages from October 1999. The analysis yields several results on the structure of the Web graph, the most novel being the discovery of a butterfly structure:

- Approximately one quarter of the pages considered belong to a single strongly connected component, called the core or SCC.
- Another quarter consists of pages from which one can reach the core, while the reverse is not true. This is the IN component.
- Conversely, nearly one quarter of the pages are reachable from the core, while the reciprocal is false. These pages form the OUT component.

[1] Anchors allow pointing to a specific part of a Web page.

[2] For example, it is claimed in [LM04] and [Kam+03b] that Arasu *et al.* use the butterfly model to speed up their PageRank computation [Ara+01]. Upon verification, Arasu *et al.* do cite the butterfly model, but propose a PageRank computation method based on… writing $A$ in block triangular form derived from the strongly connected components decomposition (cf. Theorem 4.6). In particular, the only contribution of the butterfly model is the existence of one diagonal block larger than the others, the *Strongly Connected Component (SCC)*.

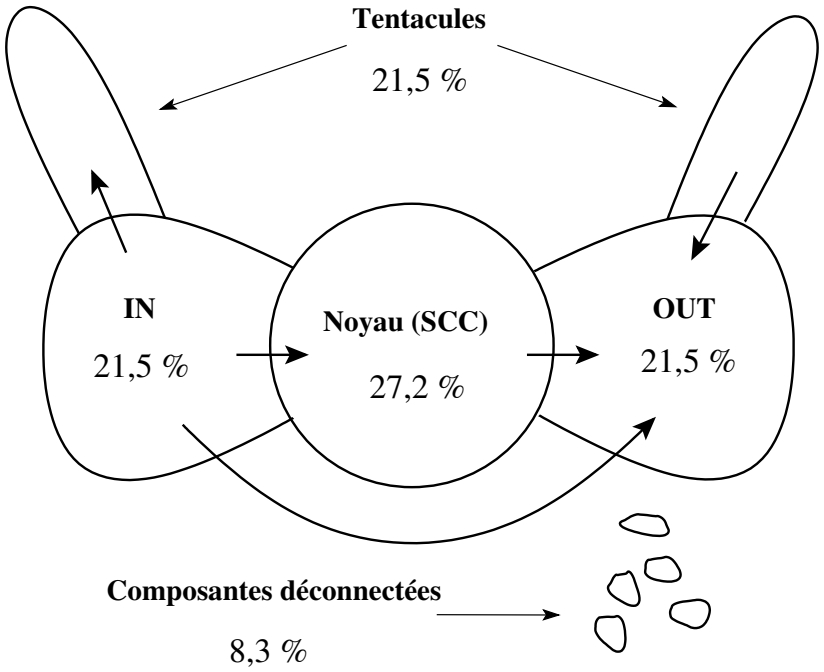



Figure 3.1: The butterfly structure

- Pages that are reachable from IN, or that allow access to OUT, and that belong to neither IN, nor OUT, nor SCC, form the tendrils (TENDRILS) and also represent nearly one quarter of the pages.
- The four quarters we have just mentioned actually form 90% of the pages in the crawl, constituting the largest connected component. The remaining 10% are components disconnected from the rest and of smaller size.

This structure is commonly represented by a diagram similar to that of Figure 3.1. The butterfly shape obtained from the IN, SCC, and OUT components gives its name to the model (*bow tie*).

From this article, many people have retained the idea that this roughly equal division into IN component, SCC, OUT, and tendrils is characteristic of the Web: this is the butterfly model.

### 3.2.2 Weaknesses of the butterfly model

**Does it still exist?**

The main drawback of the butterfly model is that in all likelihood, it no longer represents reality. The Web graph, that is, the graph of the accessible Web, cannot have a balanced butterfly structure, since it contains among other things the *page that points to all pages* (cf. Section 1.4). One must therefore consider that this model applies to graphs derived from large Web crawls (this is moreover the opinion of the authors[3]). However, one observes that:

[3]"This suggests that our results are relatively insensitive to the particular crawl we use, provided it is large enough.", *op. cit.*

- Today, most large crawls are the work of search engines.
- These engines list the sites they discover (through crawling or indexing) in directories.
- Directories obviously belong to the core of the Web.

What can then happen for a page that, for a given crawl, belongs to the IN component, or to the tendrils, or even to the disconnected components:

- If the page or one of its ancestors is indexed by the directories, it ends up *de facto* either in the core or in OUT.
- A page is part of the crawl because one of its ancestors was part of the initial crawl set. If no ancestor of the page has been added to the directory, one may doubt that an ancestor would be kept in the initial crawl set. Consequently, at the next crawl, the page will disappear.

In conclusion, yes, there can be an IN component, or disconnected components, but these components are necessarily transient, precisely because of the relationships between crawlers and directories. One must also take into account the fact that current large crawls contain a non-negligible proportion of non-indexed pages (at least 25% according to a 2002 study [Sea], 75% according to a 2004 article [EMT04]), which automatically inflate the OUT component. For all these reasons, I am inclined to strongly doubt that more than 2 (3?) billion pages indexed by Google belong to neither the core nor OUT.

**A model lacking robustness**

A question one may ask is whether the parameters considered are relevant for the study of Web graphs. On the one hand, we have crawls, which try as best they can to traverse a Web that is dynamic both spatially and temporally, and which often have a very low *overlap* between them [BB98]. On the other hand, definitions such as *IN: the set of pages that can reach the core but are not reachable from it.* Is it reasonable to combine highly variable crawls with fragile definitions? A concrete example: imagine that an unscrupulous individual seizes the set of initial pages from AltaVista's crawls and places links to these pages on their Web page. If this page is in the core, this means the collapse of all components onto SCC and OUT[4].

Contrary to what the authors claim[5], a single page, not even that large[6], suffices to undermine the butterfly model. We are in fact dealing with a methodological flaw: when faced with data subject to great uncertainty and dynamics (Web graphs), it makes no sense to treat variables that are extremely sensitive to initial conditions as

[4]This collapse phenomenon is in fact what tends to happen today with directories.

[5]"The structure that is now unfolding tells us that it is relatively insensitive to the particular large crawl we use. For instance, if AltaVista's crawler fails to include some links whose inclusion would add one of the tendrils to the SCC, we know that the resulting change in the sizes of SCC and TENDRIL will be small (since any individual tendril is small). Likewise, our experiments in which we found that large components survived the deletion of nodes of large in-degree show that the connectivity of the web is resilient to the removal of significant portions", *op. cit.*

[6]It appears that the starting set for AltaVista's crawls at the time consisted of a few hundred pages. A bookmarks page can easily contain a few hundred hyperlinks.

universal constants. For the same reasons, considering the diameter of the connected component is hardly relevant; indeed, for the 2 AltaVista crawls studied, it remains roughly constant (around 500), but upon closer inspection, evidence seems to suggest that the diameter of 500 is due to a single chain, possibly a misconfigured page or a robot trap that escaped the filters[7].

**An unreplicated experiment**

To my knowledge, the experiment of [Bro+00] has not been replicated on large crawls since. The only real experiments conducted on the butterfly model therefore concern two proprietary AltaVista crawls, close in time and thus likely based on the same technology (which may explain certain artifacts such as the constancy of the diameter). Moreover, if one compares the size and dates of these two crawls (May 1999, 203 million pages, October 1999, 271 million pages) with Figure 2.2b and Figure 2.2c (Figure 2.2b, page 26), one reaches the conclusion that the crawls used were certainly experimental.

In [Dil+01], it is nonetheless found that the butterfly structure is a fractal figure of the Web structure, meaning that one finds within sites and communities the same butterfly model as on the entire graph. Upon closer inspection, one notices that the global butterfly structure is taken for granted, and that the proportions of IN, OUT, and SCC observed in the subgraphs are extremely variable, which supports the preceding remarks.

### 3.2.3 The butterfly model in a sociological context

To properly understand the success of the butterfly model, it is important to place it in context. We are at the beginning of the year 2000. The *Internet Bubble* is still in full growth, and articles are appearing indicating that search engines can no longer keep up with the growth of the Web [Ber00, Bra97, LG99]. The AltaVista company had won the first *Crawl War* (see Figure 2.2b, page 26 as well as the conclusions of [BB98]) by surpassing 150 million indexed pages, but it lacked the assurance that this increase in size would allow it to keep pace with the evolution of the Web. In this context, *Graph Structure in the Web* was perfectly timed. Indeed, what is the takeaway of the article?

- There exists a global structure of the Web (with a capital W).
- This structure is invisible if one considers too small a portion of the Web, but there exists a threshold beyond which a crawl's structure is the structure of the Web.
- AltaVista's crawls are large enough to possess the structure of the Web.

In short, *Graph Structure in the Web* responds to an expectation that is both existential (Can we understand the Web?) and commercial (Our crawl is the Web.). One could almost see it as a mechanism arising from the laws of supply and demand.

[7] "Beyond a certain depth, only a few paths are being explored, and the last path is much longer than any of the others.", *op. cit.*

Ultimately, the question of whether its scientific foundations are sound or not becomes secondary, which is perhaps a beginning of an explanation for the success of this article.

### 3.2.4 Conclusion

The butterfly model most certainly had meaning at the time of its discovery, but that meaning is not *the Web has a butterfly structure*. It rather expresses a particular situation where, because of the *Internet Bubble*, new pages were arriving too quickly for the Web to assimilate them. But today, two phenomena mean that we are no longer in this situation: on the one hand, the *Bubble* has burst. The creation of new sites has become rare, and we are witnessing more the development of existing sites than the appearance of a swarm of unregistered sites (cf. Section 3.3.2), which is a beginning of an explanation for the fact that the butterfly structure is found within site graphs [Dil+01]. On the other hand, search engines no longer merely index the Web; they continuously modify its structure through directories. These two effects, combined with the model's lack of robustness, make it unlikely that large commercial crawls still have the same proportion of IN, tendrils, and other disconnected components as those found in *Graph Structure in the Web*.

One should nevertheless retain that Broder *et al.* highlighted the existence of a large core of pages all connected to one another, as well as the protectionist behavior of certain commercial sites that place themselves entirely within the OUT component. One should also retain that the proportion of the IN component in a crawl graph is a certain indication of the difficulty a large structured graph has in adapting to rapid expansion.

## 3.3 Role of servers and sites in the structure of the Web graph

The notion of site is fundamental in the organization of the Web. For example, many search engines have a policy of returning only a limited number of pages per site (with a *more pages from the same site* option available). A good decomposition into sites has numerous applications, ranging from PageRank computation methods [Kam+03b, MV04] to efficient compression methods for crawls [GLV02, Ran+01].

### 3.3.1 Web servers, sites, and communities

The concepts of Web server, site, and community are often encountered in the literature, with sometimes contradictory definitions. We will make explicit here what we mean by server, site, and community.

**Physical server** A physical Web server is defined as a specific physical machine connected to the Internet and identified by an IP address, which returns a 200 response code and a Web page in response to an *HTTP* request asking for the root (/) on port 80[8]. The physical website associated with the server consists of all pages hosted at the given IP address.

**Virtual server** Since IP addresses are not unlimited, solutions had to be devised to conserve addresses. One such solution is the use of virtual servers. Virtual servers, introduced with the *HTTP 1.1* standard, allow a single IP address to behave as several servers differentiated by their Domain Name System (DNS) address (*Hostname*). This method is used, for example, by *free* for hosting its clients' personal pages, or in Japan, where IP address resources are very limited. From the user's perspective, a virtual server is indistinguishable from a physical server, except that one cannot replace the DNS address with the IP address.

**Logical site** A logical website consists of a set of pages strongly connected by hyperlinks. Intuitively, a site is characterized by a certain homogeneity of style (imposed by the *Webmaster*) and easy navigation among the site's pages. The study of sites will be the subject of sections Section 3.3.3 and Section 3.3.4.

**Community** The notion of community is in some sense orthogonal to the notion of site. A community is a set of pages connected by a common interest. The search for communities is at the heart of Kleinberg's *Hyperlink-Induced Topic Search (HITS)* algorithm [Kle98] as well as the *CosmoWeb* project [BBM03].

### 3.3.2 Evolution of the number of servers

It is easier to estimate the number of Web servers (physical and virtual) than the number of Web pages.

To count physical servers, it "suffices" to make an *HTTP* request to all possible IP addresses. Now, if we ignore IPv6 addresses, which remain relatively negligible, there are only $2^{32}$ possible addresses, or approximately four billion[9]. This work was carried out by the *Web Characterization Project* [O'N+02], over the period 1998–2002, and the results concerning the number of servers are shown in Figure 3.2. The number of unique physical servers is the number of physical servers minus the duplicates, that is, servers returning exactly the same content as a previously tested server.

The main remark one can make regarding these figures is the observation of a stagnation in the number of physical servers since 2001. This stagnation can be

[8]Port 80 is the standard port for the HTTP protocol. Other ports are used more or less commonly, such as port 443 for secure HTTP or port 8080 for a secondary server, but we will not consider them here.

[9]There are in fact fewer if one removes certain *reserved* addresses – military addresses for example – which are not supposed to host an accessible Web server. The *Internet Assigned Numbers Authority (IANA)* is the organization responsible for managing the available and unavailable address ranges. One can thus eliminate approximately 48% of the available addresses.

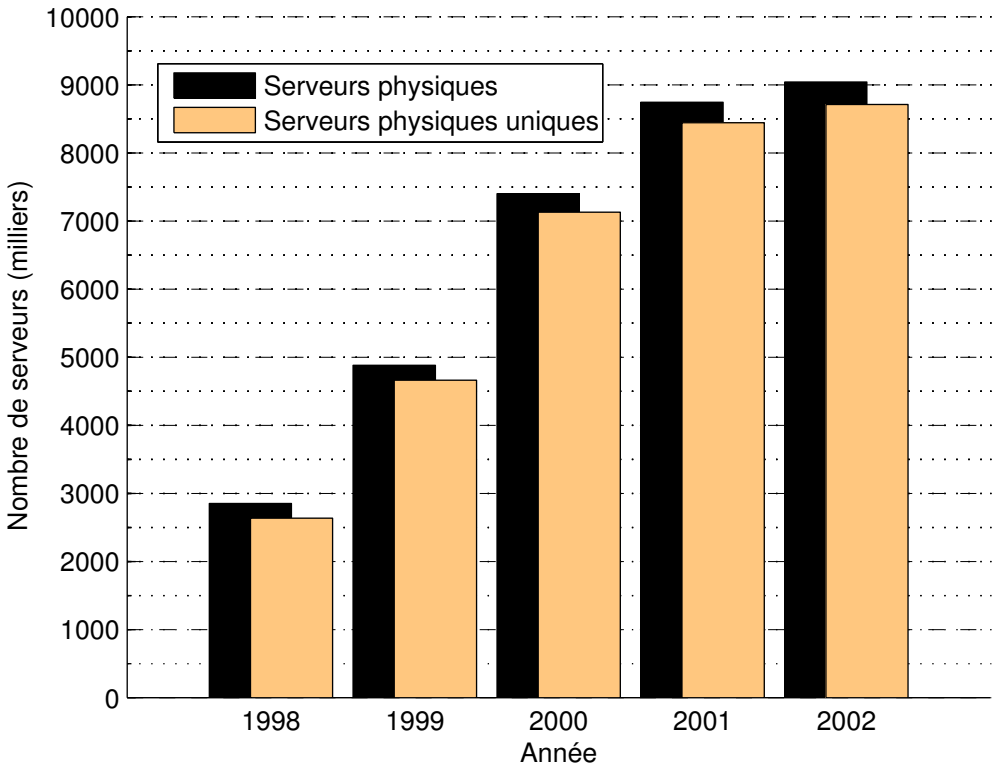



Figure 3.2: Physical Web servers (IP addresses)

explained by the end of the *Internet Bubble*. Indeed, the Web has ceased to be a new technology, and most academic, commercial, and social actors likely to integrate into the Web have already done so. According to the *Web Characterization Project*, almost no new Web servers are being created; growth occurs individually at the server level.

The study of virtual servers was carried out by the *Netcraft* website [Net04]. It consists of listing all declared DNS addresses (like the *WhoIs* site [Who04]), and testing by means of an *HTTP* request the existence of a server at each of the addresses obtained.

The evolution of virtual Web servers, according to *Netcraft* [Net04], shows a roughly linear constant progression in the number of active virtual Web servers over the last 4 years (cf. Figure 3.3). This is in any case not, a priori, a geometric progression.

### 3.3.3 Intuitive and visual approach to the notion of site

The concept of site is, in our view, fundamental to the structure of the Web. Its applications are numerous:

- More precise evaluation of the degree of completeness of a crawl (for each site, what proportion of pages has been visited?)

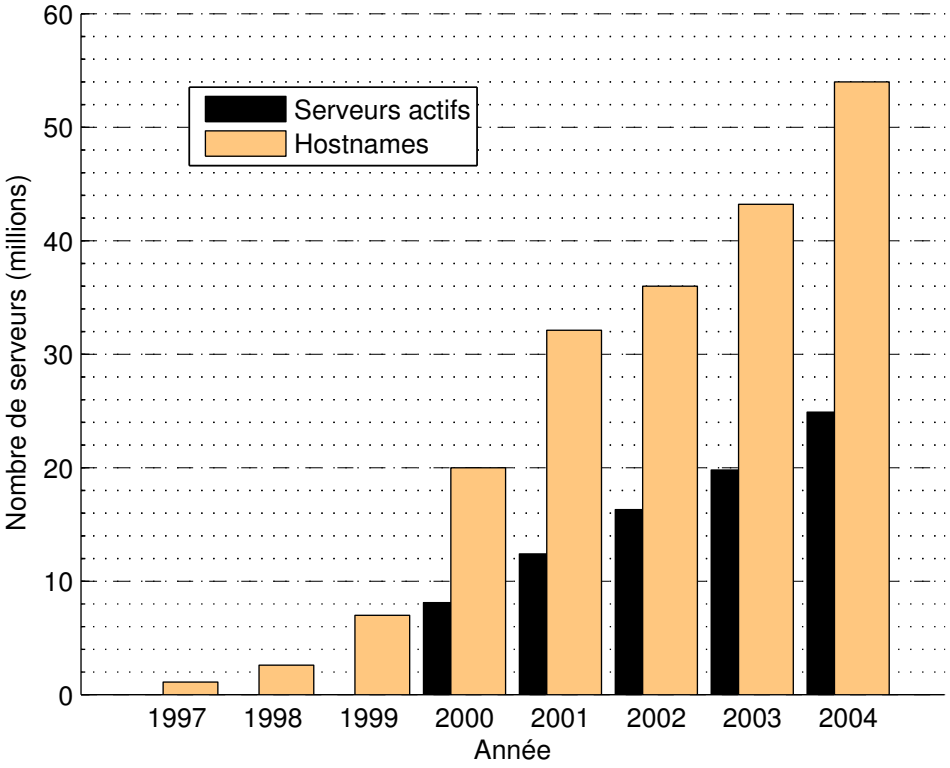



Figure 3.3: Virtual Web servers (domain names)

- Distinguishing purely navigational links from more relevant links (trust links to other sites).
- Allowing certain search engines to return only one or two pages per site, in order to avoid monopolization of query results.
- Refinement of the *zap* factor in PageRank computations (see Part II, Section 5.3, as well as [Bri+98, Hen+99])
- PageRank algorithms based on site structure (the FlowRank algorithm, presented in Part II, Chapter 7, as well as its "competitor" BlockRank [Kam+03b])

Most of the time, sites are approximated by servers ([Ada99]). But reality can sometimes be more complex. Large sites may rely on several servers, while conversely a server may host several sites. For example, one would want to group www.microsoft.com and download.microsoft.com and consider them as part of the same site. Conversely, not all personal site hosting providers use virtual domains; a large proportion places sites in the directories of a single server.

All these considerations lead us to raise the question of the definition of a logical site. Human users generally have no difficulty knowing what a site is, which constitutes a kind of empirical definition, but is hardly rigorous and does not lend itself to automation. Note that sites manually listed by *directories* fall under this definition.

The Web Characterization Project [O'N+02] proposes the following semantic definition:

> *[A Web Site (Information Definition) is] a set of related Web pages that, in the aggregate, form a composite object of informational relevance. Informational relevance implies that the object in question addresses a non-trivial information need.*
>
> [A logical website is] a set of related pages that, in the aggregate, form a composite object delivering relevant information. This relevant information implies that the object in question addresses a non-trivial information need.

To this semantic characterization, Li, Kolak, Vu, and Takano add in [Li+00] a structural approach to define the notion of *logical domain*:

> *A logical domain is a group of pages that has a specific semantic relation and a syntactic structure that relates them.*
>
> A logical domain is a group of pages that possess a specific semantic relation and a syntactic structure that connects them.

Based on this definition, they define a set of rules (semantic and structural) to identify sites.

More modestly, our approach is to see what can be achieved by considering only the structural aspect of pages. We therefore limit ourselves to, on the one hand, the graph structure induced by hyperlinks, and on the other hand, the tree structure of URLs that reveals natural clusters in crawl graphs. Indeed, very often, a logical site obeys hierarchical rules at the level of URLs (Uniform Resource Locators [BMM94]), the most common being the existence of a characteristic common prefix.

This decomposition tree[10] structure on graphs (*clustered graphs*) was introduced by Feng *et al.* in [FCE95] as a tool for representing large graphs. Its main use is therefore to allow drawing graphs in a way that reveals any existing hierarchy, and decomposition trees are mainly used in domains where implicit or explicit diagram structures exist[11].

The whole problem is then to find, for a given graph, the decomposition tree that offers the best structural representation [Bri03]. In the case of Web graphs, the URLs tree provides an intrinsic decomposition tree.

**Definition**
Let $G(V, E)$ be a graph. A decomposition tree of $G$ is a tree $T$ whose leaves correspond to the vertices $V$. Each internal node $n$ of $T$ defines a cluster (a set of vertices) $V_{T(n)}$. This cluster consists of the set of vertices of $V$ corresponding to the leaves of the subtree of $T$ rooted at $n$. For example, the cluster associated with the root of $T$ is the entire set $V$.

[10] Thanks to Fabien de Montgolfier for reminding me of the term *decomposition tree*.
[11] One may think, for example, of modular decomposition trees.

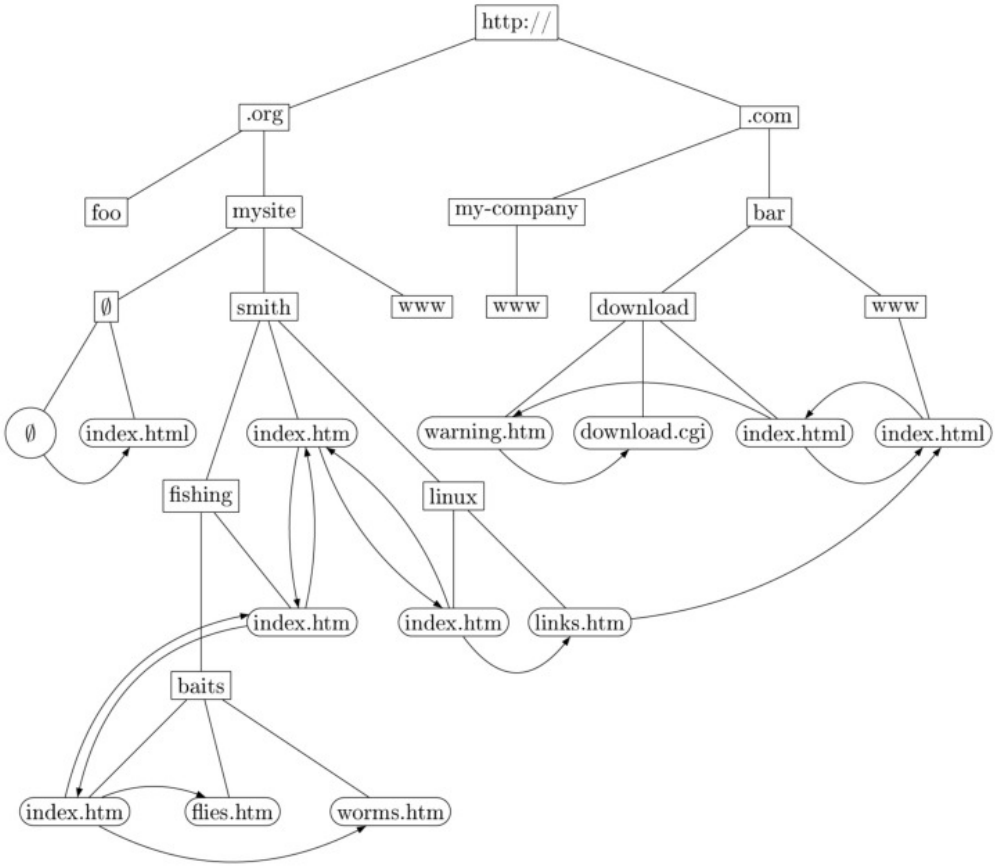



Figure 3.4: URLs: a natural decomposition tree for Web graphs

A good example of the use of decomposition trees is the modeling of human relationships. There exists indeed a fairly natural graph of human relationships, where vertices are people and where a person $A$ points to a person $B$ if $A$ knows $B$). A decomposition tree that can meaningfully complement this graph is that of geographic location: world, country, (state), city, neighborhood, street, …

**Web graphs and the URL decomposition tree**

URLs provide a natural decomposition tree structure on Web graphs. The set of servers[12] can be represented as a tree whose root is `http`, the depth-1 nodes are the TLDs[13], followed by the domain names proper and possibly subdomains. Each server, which is a leaf of the domain name tree, hosts a page hierarchy identical to the structure of the corresponding physical file system. The union of the file trees within the server tree yields the URLs decomposition tree.

For example, the URL `http://smith.mysite.org/linux/index.html` decomposes into *HTTP*, *org*, *mysite*, *smith*, *linux*, and finally *index.html*, thus forming a path from the root to the corresponding leaf in the decomposition tree (see Figure 3.4).

[12]For simplicity, we will restrict ourselves to servers identified by their DNS domain name and using port 80.

[13] *Top Level Domain (TLD)*, or top-level domain. TLDs are divided into two categories: generic top-level domains (gTLD) and country code top-level domains (ccTLD). TLDs are managed by IANA (`http://www.iana.org`).

One may also question the relevance of beginning the decomposition with the top-level domain (TLD). Indeed, there exist macro-sites that, in order to achieve better visibility, deploy across several TLDs (`.com` and the ccTLDs of the countries where they operate, quite classically). We nevertheless prefer to retain sorting by TLD, on the one hand because this corresponds to the original philosophy of domain names, and on the other hand because there also exist domain names for which the TLD makes a considerable difference. The discerning adult reader may, for example, note a definite semantic difference between the content of the site `http://www.france2.fr` and that of `http://www.france2.com`…

**Seeing the structure of sites**

Given that, in general, webmasters try to organize their sites, one can expect the concept of website to be intimately connected to the URL decomposition tree. We have confirmed this by observing the graphical representation of the adjacency matrix $M$ of the graph of a crawl of approximately 8 million URLs *sorted in lexicographic order* from `.fr`, carried out in June 2001 as part of the cooperative research action *Soleil Levant* [Sol01].

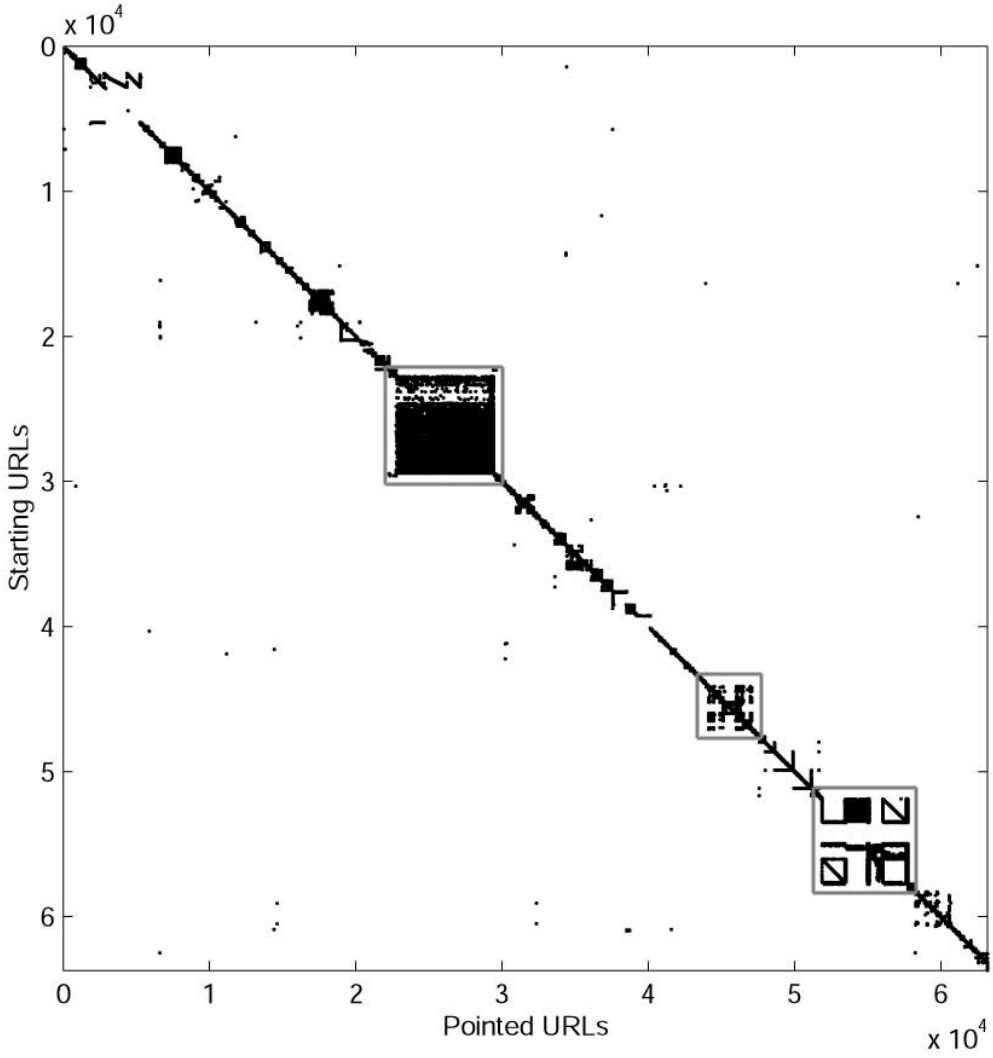



Figure 3.5: Adjacency matrix of $6 \cdot 10^4$ pages from a crawl of 8 million pages of `.fr`

We have represented a small portion of this crawl (60,000 pages) in Figure 3.5, and some zooms on interesting sub-portions in Figure 3.6. The first observation is that the adjacency matrix can obviously be decomposed into two terms, $M = D + S$, where $D$ is a block diagonal matrix, and where $S$ is a (very) sparse matrix.

Sites (and sub-sites) indeed appear as squares that coincide with the nodes of the URL tree. With a little practice, one can even guess the deep structure of sites from the appearance of the corresponding *square*. For example, pages with high out-degree (typically, the site map) result in horizontal lines, while those with high in-degree (the home pages) are characterized by vertical lines. "Noisy" squares, that is, with points exhibiting a pseudo-random structure (cf. Figure 3.6c), are often the sign of interactive documentation pages (dictionaries, for example). Let us finally note that the $D + S$ structure can exhibit a recursive character, as shown by the block in Figure 3.6d.

### 3.3.4 Proposed algorithm for partitioning into sites

We will attempt, using only the knowledge of the graph and URLs of a crawl, to generate as simply as possible a partition of the graph that reflects the notion of site. Our approach thus differs from that employed by [Li+00], who also uses a set of semantic rules to define sites.

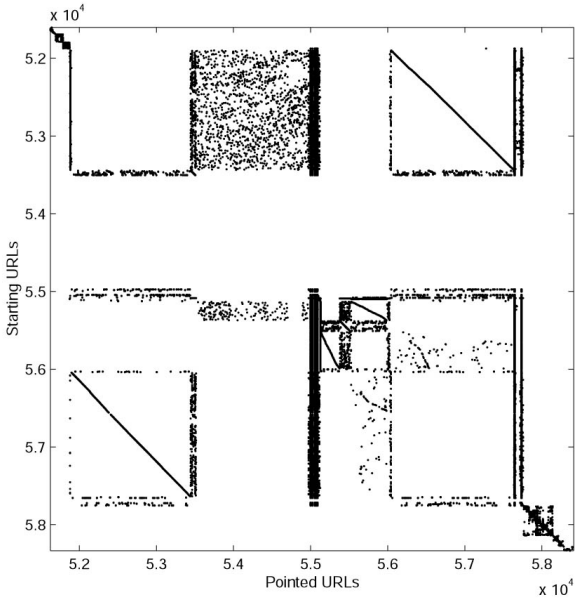
**Formal model**

Structurally, we believe that a site is defined as a set of pages tightly connected to one another by navigational links, as the adjacency matrix representations seem to confirm (cf. Figure 3.5).

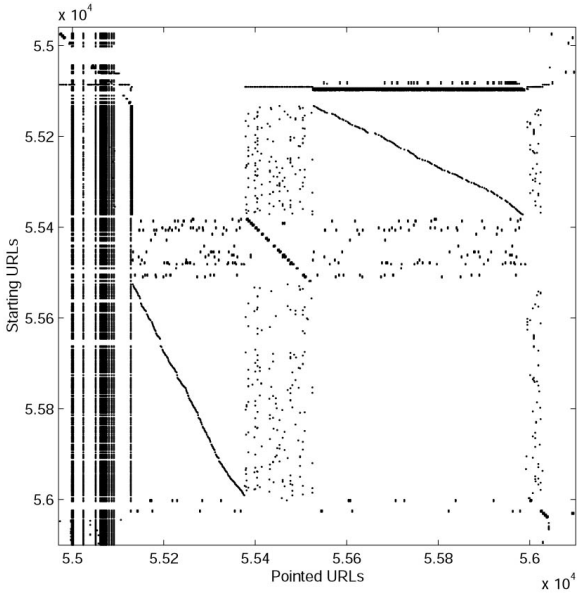
This leads us to try to define a function that measures the quality of a site partition relative to the rate of navigational links. This function, $f_G : \mathcal{S}$ partition of $G \to f(\mathcal{S}) \in \mathbb{R}$ must reach an extremum when $\mathcal{S}$ approaches (in a sense to be specified) what we would like to call a site partition. Once such a function is found, the standard method for partitioning $G$ consists of constructing an initial partition of the URLs (adapted to the situation considered), then performing local adjustments to try to optimize $f(\mathcal{S})$.

**Choice of the function $f_G$**

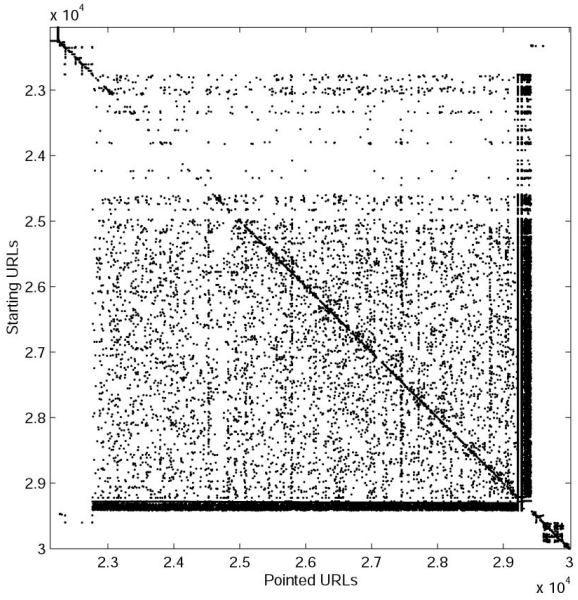
Since the most important factor in our view for estimating a site partition is the number of navigational links, it seems logical to seek a function $f_G$ that depends explicitly on $i$, the number of internal links. One must also take into account the number of sites: indeed, this number is not fixed in advance (otherwise, we would be in the classical framework of *mincut* or *maxcut* type algorithms). To avoid having the trivial partition $G$ as the extremum, and to try to obtain a fine decomposition, it also seems important to us to try to maximize the number of sites $p$. We thus have two global parameters, the number of sites $p$ and the number of internal links $i$, and



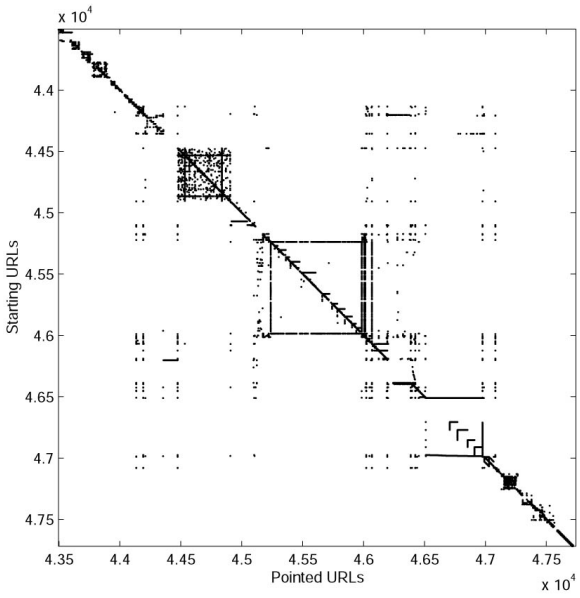

(a) allmacintosh.easynet.fr, a software site




(b) allmacintosh.(…).fr/osx, a sub-site for a specific OS




(c) aemiaif.lip6.fr/jargon, an interactive dictionary




(d) algo.inria.fr, the site of an INRIA research team

Figure 3.6: Visual approach to site structure: zooming in on clusters from Figure 3.5

the function $f$ we seek must satisfy both $\frac{\partial f}{\partial p} > 0$ and $\frac{\partial f}{\partial i} > 0$. Several simple formulas are a priori possible:

- $f : (p, i) \to \alpha p + i$, $\alpha \in \mathbb{R}$.

  This solution will not be retained here, mainly because the choice of $\alpha$, to be judicious, needs to equal the average number of links per page, which leaves little room for maneuver. Indeed, if $\alpha$ is larger than this number, one risks reducing the problem to maximizing $p$ (it becomes advantageous to isolate pages), while if it is smaller, for the same reasons, only $i$ will matter.

- $f : (p, i) \to p \cdot i^{\alpha}$, $\alpha \in \mathbb{R}$.

  This solution has the advantage of being more flexible regarding the ratio between $p$ and $i$.

- $f : (p, i) \to p^{\frac{i}{\ln}|E|}$, where $|E|$ is the total number of links.

  This last solution is deliberately parameter-free because we thus have a quantity with an intuitive interpretation: for a partition with no external links, we would have $f(p, i) = p$. More generally, $f$ remains of the same order of magnitude as $p$, approaching it all the more closely as internal cohesion is greater. We will therefore call the quantity $p^{\frac{i}{\ln}|E|}$ the site index, which can be interpreted as the equivalent number of isolated sites.

**Problem of isolated pages**

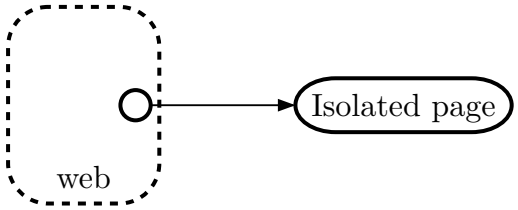



Figure 3.7: Isolated page

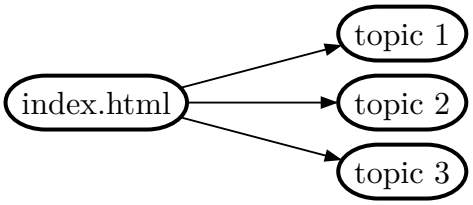



Figure 3.8: Structured site

The case of pages with in-degree 1 and out-degree zero poses a problem. While they may sometimes represent an isolated site reduced to a single page (Figure 3.7), one observes experimentally that in most cases these are pages that have been crawled but not visited (see Section 2.1) or terminal pages of structured sites (Figure 3.8). From a structural point of view, and in the absence of any other consideration, it seems legitimate to us to ensure that pages with in-degree 1 and out-degree zero are attached to the site of the parent page. This translates, at the level of the function $f(p, i)$ introduced previously, into the following inequality:

$$(\forall p, i) \quad f(p-1, i+1) > f(p, i) \tag{3.1}$$

This inequality, if one tries to impose it on our function $f$ (which is necessary if one wants $f(p, i)$ to suffice for performing the partition), raises the following problem: any partition with parameter $(p, i)$ such that

$$p \leq |E| - i + 1 \tag{3.2}$$

is less effective than the trivial partition:

$$f(p, i) < f(1, |E|) \tag{3.3}$$

Alas, Equation (3.2) is satisfied as soon as the graph is connected. More generally, for a graph composed of $k$ connected components and $|E|$ edges, for any partition with parameter $(p, i)$ we will necessarily have the inequality

$$f(p, i) \leq f(k, |E|) \tag{3.4}$$

**Summary:** There exists no universal function $f(p, i)$ that is maximal for a non-trivial site partition respecting terminal pages.

Two approaches are then possible to circumvent this problem, as well as others of the same kind that one might encounter when trying to refine our model:

- Continue to seek to maximize $f(p, i)$ by imposing external rules, which would amount, for example, to working on a subset of the set of partitions of the graph. This is the approach used for the structural part of the logical domain definition employed by [Li+00].
- Replace $p$ or $i$ by new variables that handle the problem implicitly. This is the approach we will study.

**Alteration of the variables $p$ and $i$**

A first idea would consist of weighting the edges so that links to isolated pages are harder to break. Thus, a fairly classical and natural idea is to take a weight inversely proportional to the in-degree (cf. [Kle98]). Alas, the new variable $i'$ thus introduced has numerous drawbacks, especially if it is the only alteration introduced:

First, one may consider the case of two sites (in the intuitive sense) connected by a single edge (Figure 3.9). If the algorithm is designed not to separate terminal pages, that is, if it respects the relation Equation (3.1), it will be forced to merge the two sites, which is not necessarily the desired effect.

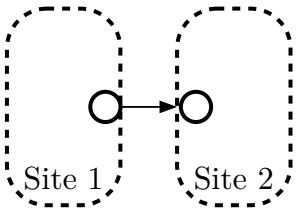



Figure 3.9: 1-connected sites

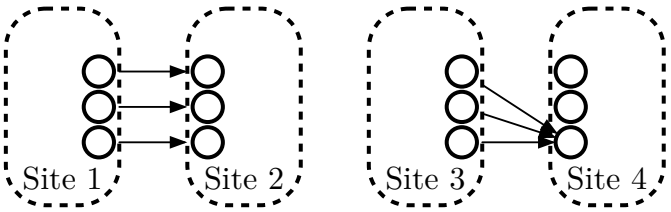



Figure 3.10: Examples of 3-connected sites

Furthermore, if one considers the two examples of 3-connected sites in Figure 3.10, one would like the decision to merge sites, all other things being equal, to be the same in both cases. With edge weights depending on the in-degree, this will be difficult, since the connection strength varies by a factor of three between the two cases.

An alteration by weighting partitions according to their size is another solution. More generally, if one considers a partition $V = \bigcup_{i=1}^{p} P_i$ and a weighting function $h : \mathbb{N}^* \to \mathbb{R}$, one can define the new variable $p' = \sum_{i=1}^{p} h(|P_i|)$. The simplest solution, which we will adopt here, consists of assigning zero weight to singleton partitions (which amounts in practice to working on $(p, i)$ while imposing the rule that singleton partitions are forbidden). The choice of other functions $h$ will not be treated here, but one may hope, by choosing the right function $h$, for some control over the partition distribution, for example avoiding the formation of overly large sites by weakening $h$ for large values.

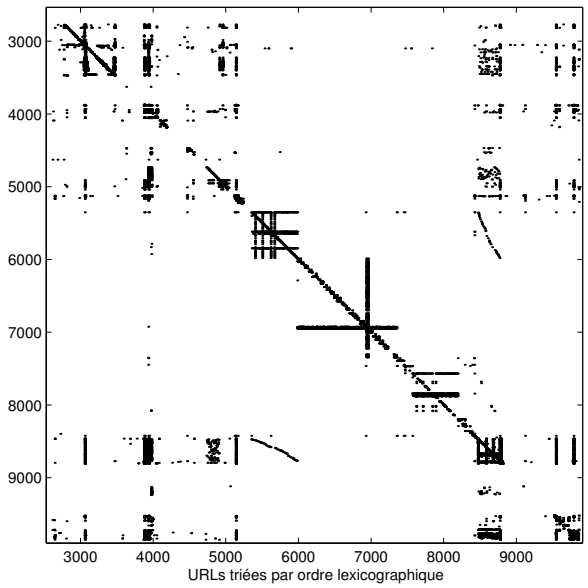



Figure 3.11: The five-on-a-die pattern: a block is interleaved within another

**Chosen function $f_G$**

Taking into account all the remarks we have just made, we choose as our evaluation function the function $f$ which, to a partition $\mathcal{S} = (S_1, ..., S_p)$ of $G$, associates $p'^{\frac{i}{\uparrow}E|}$, where $i$ is the number of internal links and $p' = \sum_{i=1}^{p} h(|S_i|)$, with $h = \chi_{([2,+\infty[)}$.

**Estimating a site partition using the URL decomposition tree**

Now that we possess an evaluation function $f$ for a partition, the question arises of producing an efficient partition. An intuitive method is based on the following hypothesis, supported by the observation of Figure 3.5 and Figure 3.6: The URL architecture of pages reflects the architecture of sites, and therefore blocks are *a priori* subtrees or unions of subtrees of the Tree-Graph tree. Instead of having to choose a partition from among all possible ones, we can restrict the set of candidates to those coinciding with the tree structure.

This solution is not perfect, and there will always be borderline cases. Thus, in Russian-doll configurations, the choice of partition level will be closely tied to the choice of functions $f$ and $h$. On the other hand, five-on-a-die configurations (a site embedded within another at the adjacency matrix level, see Figure 3.11), which are difficult to disentangle automatically using only the adjacency matrix (which nevertheless carries more information than the graph alone), will not cause problems with the decomposition tree provided that the sites coincide with the tree (which is the case for all examples tested).

We will give an example of a simple algorithm for performing an intelligent and fast partition of URLs using the tree-graph: the Filtered Breadth-First Search (Algorithm 3.1).

**Data**
A set of URLs together with the associated graph $G$.
**Result**
A site partition of $G$, each site having an entry page.
**begin**
    No vertex is marked.
    **while** there remains an unmarked vertex
        Choose an unmarked vertex among those of lowest height
        in the decomposition tree
        Perform a breadth-first search on the graph starting from this vertex
        ignoring arcs leaving the lexicographic cone
        Label the set of vertices obtained according to the starting vertex,
        which will be called the *entry page* of the site
        **if** a vertex already belonging to a site is encountered[14]
            Merge the two sites

Algorithm 3.1: Filtered Breadth-First Search (FBFS) Algorithm

**Definition 3.1**: The lexicographic cone of a URL is the cluster associated, in the URLs decomposition tree, with the internal node that is the parent of the leaf corresponding to the URL in question.

The advantages of this initial decomposition are as follows:

- One avoids amalgamating into a single site two sets that are very close in the tree but have no direct link in the graph. Thus, so-called "personal" sites are not merged under the sole label of the hosting provider (free, voila, multimania…); similarly, a researcher's page within a laboratory will only be part of the laboratory's site if it is referenced by the latter, which seems satisfactory to us.
- Conversely, sites hosted on several closely related servers (www.inria.fr and www-rocq.inria.fr, for example) are treated as a single entity.

There remain, unfortunately, cases that the algorithm we propose does not handle. For instance, certain sites straddle several servers that have no kinship in the lexicographic tree as we have defined it (personal sites shared between several hosting providers, commercial sites available in `.com`, `.fr`, `.de`…).

**Results**
We will compare here the partition obtained by Filtered Breadth-First Search with partitions obtained by quotienting by server, by directory at depth 1, and by directory

[14]This situation only occurs if the two vertices are siblings.

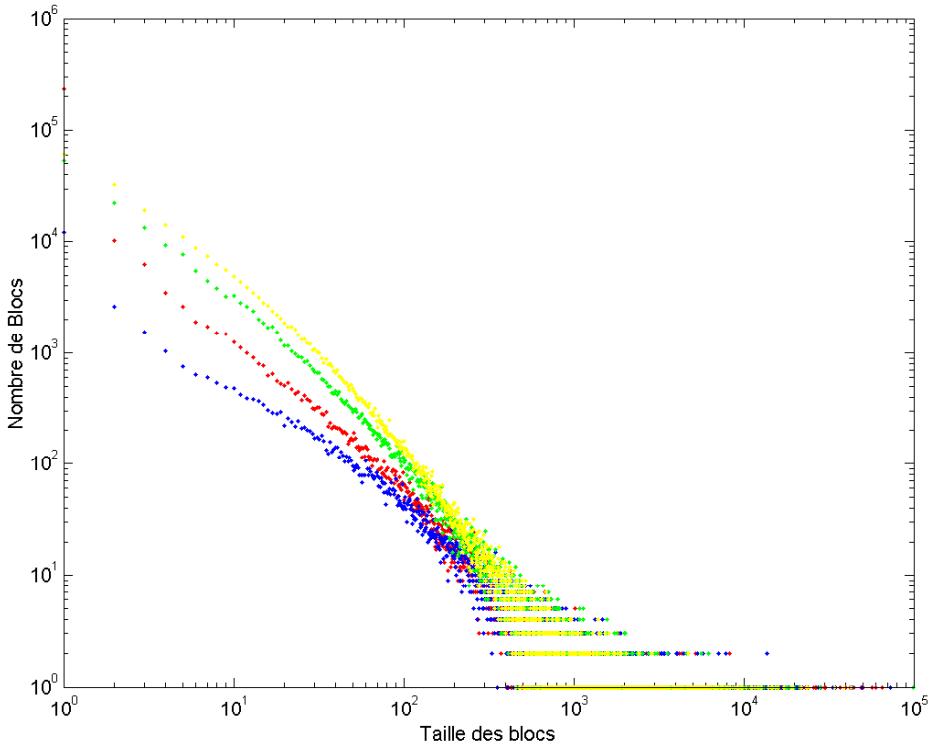



Figure 3.12: Distribution of partitions as a function of their size, on a log-log scale. Legend: FBFS in red, server in blue, 1-dir in green, 2-dir in yellow

at depth 2 (which typically corresponds to cuts at heights 3, 4, and 5 in the tree-graph). The study was conducted on the crawl of 8,212,791 URLs from `.fr` dated June 15, 2001. For brevity, we will call these partitions FBFS, server, 1-dir, and 2-dir.

Let us first observe how the different partitions are distributed. Figure 3.12 shows the size distributions of the different partitions. One observes that regardless of the method used (FBFS, server, 1-dir, or 2-dir), the partitions obtained are distributed according to a power law[15], that is, the number of blocks of size $x$ is of order $\frac{C}{x^\alpha}$. The values of the exponents (estimated by linear regression) are indicated in table Table 3.1.

The power law is a law considered characteristic of social networks and human activities. The fact of encountering such a law in all cases studied, while it proves nothing by itself, shows that from the sole point of view of page distribution

| **Partition** | server | 1-dir | 2-dir | FBFS |
|---|---|---|---|---|
| $\alpha$ | 1.32 | 1.75 | 1.89 | 1.59 |

Table 3.1: Power law coefficients of the different partitions

[15]By abuse of language, we will often call a power law any phenomenon that, when represented on a *log-log* scale, yields a curve forming approximately a straight line. The fact that the number of samples considered is finite, and the sensitivity of estimates to extreme values, means that it would be more accurate to merely speak of a *heavy tail* phenomenon.

| | $p$ | $p'$ | $\frac{i}{\uparrow}E\vert$ | $p'^{\frac{i}{\uparrow}E\vert}$ |
|---|---|---|---|---|
| server | 39305 | 27241 | 0.9517 | 16629 |
| FBFS | 289539 | 58094 | 0.9218 | 24624 |
| 1-dir | 182942 | 137809 | 0.7851 | 10831 |
| 2-dir | 250869 | 189698 | 0.7012 | 5025 |

Table 3.2: Characteristics of the different partitions before optimization

(without taking hyperlinks into account), the four distributions studied appear to be compatible with the human structure of the Web.

Let us also note the importance of "sites" of size 1 (the data in Figure 3.12 come from the partitions before any optimization), especially for the FBFS partition. The explanation is quite logical: size-1 partitions contain, among other things, all the errors that were not removed from the crawl: 4xx errors, errors in the URL format… The FBFS partition isolates more errors than the others by construction, since a URL that is isolated in the lexicographic cone of a site will be isolated by FBFS, unlike the other partitions.

Finally, let us compare the quality of the site decomposition of the different partitions, using the site index defined by the function $f$. The optimization is initially minimal: size-1 sites are merged with the site to which they are most strongly connected (most often, the connection is unique).

The results are reported in table Table 3.2. One will note that FBFS, whose results for $p'$ as well as for $i$ are midway between server and 1-dir, achieves by far the best site index.

**Optimization of partitions**

The preceding results come from partitions that do not *a priori* seek to maximize the site index. One can settle for these results, if one views the site index merely as a quality indicator, but one can also seek to maximize it. We will therefore examine what a local optimization by mixing yields, without FBFS and with FBFS. The principle of this optimization is simple: for each of the sites defined by the partition to be optimized, one attempts to improve the site index by replacing it with the local decomposition (server, 1-dir,…) that yields the best site index. It is therefore a greedy algorithm, and the result obtained, which is a local maximum, may depend on the order in which sites are processed.

The results are presented in table Table 3.3. Two different traversal orders of the partitions were performed, in order to assess the fluctuations that the choice of traversal can generate, as well as their magnitude. The main results that can be drawn from this table are:

- A clear overall improvement in the site index, with the data $p'$ and $i$ being close to the maximum values technically achievable by this method.

| | $p'$ | $\frac{i}{\uparrow}E\vert$ | $p'^{\frac{i}{\uparrow}E\vert}$ |
|---|---|---|---|
| optimization without FBFS, pass 1 | 107884 | 0.9376 | 52341 |
| optimization without FBFS, pass 2 | 113148 | 0.9340 | 52487 |
| optimization with FBFS, pass 1 | 121963 | 0.9338 | 56165 |
| optimization with FBFS, pass 2 | 118386 | 0.9372 | 56845 |

Table 3.3: Characteristics of the different partitions after optimization

- Optimization with FBFS remains better in all cases than optimization without FBFS, even though the added value is smaller than for the raw partitions.
- The fluctuations in $p'$ and $i$ in the four cases seem to suggest that it may be difficult to determine in advance the best optimization traversal order (a problem due to the use of a greedy algorithm).

Part II

# Web Page Ranking Algorithms: PageRank

# Chapter 4
# Markov Chains

*Always buy an unowned property if it is an orange property (always block this group if you can).*

*Mr. Monopoly Strategy Wizard*

Before tackling the central topic of this part, namely PageRank(s), it seems necessary to provide a theoretical review of the techniques that we will use throughout the following pages.

Andrey Markov (1856–1922), a Russian mathematician, is known for having defined and studied memoryless discrete stochastic processes, also known as *Markov chains*. This tool from the beginning of the last century is fundamental for understanding the idea of PageRank, which is why we will briefly recall the essential results[1].

## 4.1 Definitions

A *discrete random, or stochastic, process* is a set of random variables $X_k$, with $k \in \mathbb{N}$. $X_k$ can for instance represent the position on the game board of a Monopoly player at time $k$.

Discrete Markov chains are special cases of discrete stochastic processes with discrete values whose future depends only on the present (not on the past): the states preceding the present state play no role. If we call $V$ the set of values (or states) that the variables $X_k$ can take, the mathematical characterization of a Markov chain is the equality, for all $j \in V$,

$$\forall k \in \mathbb{N}, P(X_k = j \mid X_l = i_l, l \in [\![0, k-1]\!]) = P(X_k = j \mid X_{k-1} = i_{k-1}) \quad (4.1)$$

In other words, the probability of being in state $j$ at time $k$ depends only on the value taken at time $k-1$, and not on earlier values. *A priori*, this probability may not be the same depending on the time $k$ considered. The Markov chain can therefore be defined by the transition probabilities between two states at time $k$:

[1] For more comprehensive information on Markov's work, the reader may refer to [Sal96, She88].

$$p_{i,j}^k = P(X_k = j \mid X_{k-1} = i) \tag{4.2}$$

Throughout this thesis, we will restrict ourselves to the case where the transition probabilities do not depend on the time $k$ considered. *The corresponding Markov chain is then said to be homogeneous.*

If $V$ is finite (we will then take $V = [\![1, n]\!]$), it is convenient to consider the transition matrix $A = \left(p_{i,j}\right)_{1 \leq i,j \leq n}$. $A$ is a row-stochastic matrix, meaning that $\forall i \in V, \sum_{j=1}^{n} A_{i,j} = 1$. This is due to the fact that the process is completely closed, and that if one is in state $i$ at time $k-1$, then one will be in $V$ at time $k$.

The transition matrix $A$ allows one to obtain the evolution of our process. Indeed, if we call $x^k$ the vector representing the state distribution at time $k$ ($x_j^k = P(X_k = j)$), then the following proposition gives $x^k$ as a function of $x^0$ and $A$.

**Proposition 4.1**: $x^k = \left(A^t\right)^k x^0$, where $^t$ is the transpose operator.[2]

*Proof*: It suffices to show that, for any $k \geq 1$, we have $x^k = A^t x^{k-1}$; Proposition 4.1 is then merely the application of a straightforward induction. The desired result follows from the fact that:

$$\begin{aligned} \left(A^t x^{k-1}\right)_j &= \sum_{i=1}^{n} p_{i,j} x_i^{k-1} = \sum_{i=1}^{n} P\big(X^k = j \mid X^{k-1} = i\big) P\big(X^{k-1} = i\big) \\ &= \sum_{i=1}^{n} P\big(X^k = j, X^{k-1} = i\big) = P\big(X^k = j\big) = x_j^k \end{aligned} \tag{4.3}$$

■

## 4.2 Graph side, matrix side

We have just seen that stochastic matrices are both an elegant and compact means of describing the evolution of a Markov chain, but this is not the only one. The representation in terms of a weighted directed graph is also very useful, and it is convenient to be able to switch from one to the other as needed.

[2]Throughout this thesis, we will make extensive use of the transpose operator. Why not work directly with the matrix $P^t$ and thus avoid introducing $^t$? First, because it is more comfortable to consider that a coefficient $a_{i,j}$ represents the action of $i$ on $j$ rather than that of $j$ on $i$. Second, because my background from preparatory classes makes me prefer working with column vectors rather than row vectors. Finally, because this is generally the standard notation in the existing literature.

To any matrix $M = \left(m_{i,j}\right)_{1 \leq i,j \leq n}$ of size $n$, it is possible to associate a weighted directed graph $G(M)$ with $n$ vertices, whose edges are the set of pairs $(i, j)$ such that $m_{i,j} \neq 0$, each weighted by its associated coefficient $m_{i,j}$.

Conversely, any weighted directed graph can correspond to the matrix $M$ defined by:

$$m_{i,j} = \begin{cases} \text{the weight of edge } (i,j) \text{ if it exists} \\ 0 \text{ otherwise} \end{cases} \tag{4.4}$$

In the case of a graph whose edges are unweighted, by considering them as implicitly having weight 1, one recovers the adjacency matrix.

Sometimes, the passage from the matrix viewpoint to the graph representation amounts to a simple rewriting: thus, if the characterization of a stochastic matrix[3] $M = \left(m_{i,j}\right)_{1 \leq i,j \leq n}$ is

$$\forall i \in [\![1, n]\!], \sum_{j=1}^{n} m_{i,j} = 1 \tag{4.5}$$

that of a graph $G = (V, E)$ representing a homogeneous Markov chain is

$$\forall v \in V, \sum_{w \leftarrow v} e_{v,w} = 1 \tag{4.6}$$

However, it sometimes happens that the two approaches correspond to genuinely two subtly different views of the same problem. Thus, saying that a nonnegative matrix $M$ is irreducible amounts to saying that

$$\forall (i,j) \in [\![1, n]\!]^2, \exists k \in \mathbb{N}, \left(M^k\right)_{i,j} > 0 \tag{4.7}$$

At the graph level, for the corresponding graph $G$, this amounts to saying that for every pair of vertices $(i, j)$, there exists a path (of length $k$) connecting $i$ to $j$. In other words, the irreducibility of the matrix translates into the strong connectivity of the graph.

Finally, let us mention that a Markov chain is said to be *ergodic* if the corresponding matrix is irreducible and aperiodic.

[3]When not otherwise specified, by stochastic matrix we mean a row-stochastic matrix, that is, a nonnegative matrix such that the sum of each of its rows equals 1.

## 4.3 Evolution of a homogeneous Markov chain

In order to study the long-term evolution of a Markov chain, one may ask whether convergence properties can be obtained. This is indeed guaranteed by the following theorem.

**Theorem 4.1**: Let $A$ be a stochastic matrix.

1. The spectral radius of $A$ is 1, and it is an eigenvalue.
2. If $A$ is irreducible, then there exists a unique probability vector $P$ that is a right eigenvector of $A^t$ for the eigenvalue 1, and $P$ is strictly positive, meaning that all its components are strictly positive.
3. If $A$ is irreducible and aperiodic, then all eigenvalues other than 1 have modulus strictly less than 1.

*Proof*:

1. 1 is an eigenvalue, associated with the eigenvector $\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$.

   Moreover, if one considers an arbitrary vector $x = (x_i)_{1 \leq i \leq n}$, we always have

   $$\|Ax\|_\infty \leq \|A|x|\|_\infty \leq \|x\|_\infty \tag{4.8}$$

   with the convention $|x| = (|x_i|)_{1 \leq i \leq n}$.

   This implies that every eigenvalue of $A$ is at most 1.

2. If $A$ is irreducible, then we are within the framework of the Perron-Frobenius theorem (see Appendix A, page 130), which ensures that there exists, up to scaling, a unique eigenvector[4] of $A^t$ for the eigenvalue 1. Since this vector is strictly positive, after normalization it can be considered as a probability vector.

3. By the Perron-Frobenius theorem, if $\lambda$ is an eigenvalue of $A$ satisfying $|\lambda| = 1$, then $\lambda$ is a $d$-th root of unity, where $d$ is the cyclicity of $A$. If $A$ is aperiodic, then necessarily $\lambda = 1$. All eigenvalues of $A$ other than 1 therefore have modulus strictly less than 1.

■

[4] In fact, there are two eigenvectors, a right one and a left one, the left one of $A$ being the right one of $A^t$ and vice versa.

By Theorem 4.1, it is now possible to determine the asymptotic behavior of a homogeneous Markov chain.

**Theorem 4.2**: Let $A$ be an irreducible aperiodic stochastic matrix of size $n$ representing a homogeneous Markov chain. If we call $P$ the right probability eigenvector of $A^t$ for the eigenvalue 1 (whose existence and uniqueness are guaranteed by the Perron-Frobenius theorem), then

$$\left(A^t\right)^k \underset{k\to\infty}{\longrightarrow} P \cdot \mathbf{1}_n \tag{4.9}$$

where $\mathbf{1}_n$ is the row vector of size $n$ consisting entirely of 1s.

*Proof*: This is a simple application of the power method, or Jacobi method. Indeed, since 1 is an eigenvalue of dimension 1, one can decompose $A^t$ on the eigenspace spanned by $P$ on one hand and on the space associated with the other eigenvalues on the other hand (which is not necessarily an eigenspace). Thus, there exists an invertible change-of-basis matrix $T$ such that

$$A^t = T \cdot \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & & & \\ \vdots & & Q & \\ 0 & & & \end{pmatrix} T^{-1} \tag{4.10}$$

where $Q$ is a matrix with spectral radius strictly less than 1 (the eigenvalues of $Q$ are those of $A$ except 1). The spectral radius of $Q$ implies that $Q^k$ converges geometrically to 0. $\left(A^t\right)^k$ therefore converges geometrically to:

$$\lim_{k\to\infty} T \cdot \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & & & \\ \vdots & & Q^k & \\ 0 & & & \end{pmatrix} T^{-1} = T \cdot \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & & & \\ \vdots & & 0 & \\ 0 & & & \end{pmatrix} T^{-1} \tag{4.11}$$

This matrix, which we will call $A^t_\infty$, is in fact a projection, since $\left(A^t_\infty\right)^2 = A^t_\infty$. The projection space is one-dimensional, as the matrix has rank 1. Since by passing to the limit, we have $A^t_\infty P = P$, the projection line is the one spanned by $P$. $A^t_\infty$ is also a stochastic matrix (as a limit of stochastic matrices). In particular, if $e_i$ is the certain probability vector at $i$, defined by $\left(e_i\right)_k = \delta_i^k$, then

$$A^t_\infty e_i = \begin{pmatrix} \left(A^t_\infty\right)_{1,i} \\ \vdots \\ \left(A^t_\infty\right)_{n,i} \end{pmatrix} = P \tag{4.12}$$

hence $A^t_\infty = P \cdot \mathbf{1}$. ■

The asymptotic behavior of the associated homogeneous Markov chain is then given by Corollary 4.1:

**Corollary 4.1**: Let $P_0$ be an arbitrary initial probability distribution. The state distribution at time $k$, $P_k = (A^t)^k P_0$, converges, as $k$ tends to infinity, to $P$.

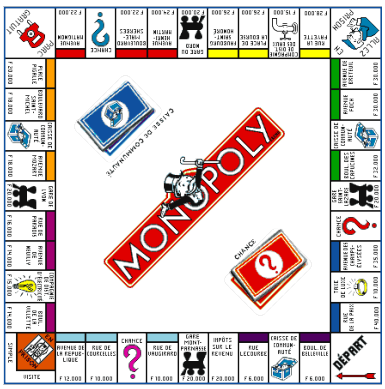
*Proof*: For any probability distribution $P_0$, we have $(A^t)^k P_0 \underset{k\to\infty}{\longrightarrow} (A^t_\infty)P_0 = P \cdot \mathbf{1} \cdot P_0 = P$. ■

For the reader wishing to become familiar with the applications of the asymptotic study of Markov chains, the following section is devoted to a brief analysis of probabilities in the game of Monopoly (registered trademark).

## 4.4 Intermezzo: Monopoly™ According to Markov

Sources: [Col, Gau96, Ste96]

A question to ask is: what are the chances of landing on a given square? If certain squares are more likely than others, one can easily see that they will be of greater strategic interest. The evolution of a player's position over successive dice rolls can be viewed as a Markov chain. Ian Stewart [Ste96] associates to each square-state 11 possible transitions corresponding to the possible outcomes of a dice roll, from 2 to 12. The probability of each transition is that of obtaining the result with two dice. Ian Stewart concludes that the stochastic matrix representing the Markov chain is circulant, and that the asymptotic probability distribution is the uniform distribution. In fact, if one looks more closely at the rules, one notices that not all




Nearly everyone knows the game of Monopoly (registered trademark), a board game in which the goal is to acquire properties, build monopolies called color groups (sets of properties of the same color), build houses, then hotels, and ultimately bankrupt all one's opponents.

There is a certain strategy to Monopoly. As in real financial life, everything is a matter of negotiations, risk-taking, and return on investment. What is the role of Markov chains in Monopoly? One of the sources of income (and of ruin!), the main one in the late game, is the obligation at each turn to pay rent to the owner (if there is one) of the property where one's token lands.

Figure 4.1: Monopoly board (registered trademark)

| No. | Name | Group | No. | Name | Group |
|---|---|---|---|---|---|
| 0 | Go | | 21 | Matignon | red |
| 1 | Belleville | brown | 22 | Chance Card | |
| 2 | Community Chest | | 23 | Malesherbes | red |
| 3 | Lecourbe | brown | 24 | Henri-Martin | red |
| 4 | Income Tax | | 25 | Gare du Nord | |
| 5 | Gare Montparnasse | | 26 | Saint-Honoré | yellow |
| 6 | Vaugirard | light blue | 27 | Bourse | yellow |
| 7 | Chance Card | | 28 | Water Works | |
| 8 | Courcelles | light blue | 29 | La Fayette | yellow |
| 9 | République | light blue | 30 | Go to Jail | |
| 10 | Just Visiting | | 31 | Breteuil | green |
| 11 | La Villette | purple | 32 | Foch | green |
| 12 | Electric Company | | 33 | Community Chest | |
| 13 | Neuilly | purple | 34 | Capucines | green |
| 14 | Paradis | purple | 35 | Gare Saint-Lazare | |
| 15 | Gare de Lyon | | 36 | Chance Card | |
| 16 | Mozart | orange | 37 | Champs-Élysées | dark blue |
| 17 | Community Chest | | 38 | Luxury Tax | |
| 18 | Saint-Michel | orange | 39 | Rue de la Paix | dark blue |
| 19 | Pigalle | orange | 40 | Jail | |
| 20 | Free Parking | | | | |

Table 4.1: List of Monopoly squares (registered trademark)

states are equivalent, and that the limiting probability distribution is not necessarily uniform.

### 4.4.1 Brief Reminder of the Rules and Notation

By convention, we consider that there are 41 squares: from the *Go* square (number 0) to the *Rue de la Paix* square (number 39), with the *Jail* square having number 40 and the *Just Visiting* square having number 10. Table 4.1 summarizes the different squares, with the name and possible color group.

A game begins on the *Go* square. At each turn, the player rolls two dice. After three consecutive doubles, the player goes to jail. If the player lands on a *Chance* or *Community Chest* square, they draw a card from the corresponding pile, and this draw is possibly followed by an immediate effect on their position. When in jail, one can get out for free by rolling doubles within the three turns following imprisonment; otherwise, one must pay to get out. One can also pay to get out before the end of the three turns.

Here is the detailed list of *Chance* cards: 1 sends to jail, 1 sends to Avenue Henri-Martin, 1 sends to Boulevard de la Villette, 1 sends to Rue de la Paix, 1 sends to Gare de Lyon, 1 sends to the Go square, 1 *Go back three spaces*. There are 9 other *Chance* cards that have no effect on position.

Here is the detailed list of *Community Chest* cards: 1 *Return to Belleville*, 1 sends to jail, 1 sends to the Go square, 1 option to draw a *Chance* card (alternative with a fine). There are 12 other *Community Chest* cards that have no effect on position.

### 4.4.2 Transition Matrix

Because of the rules, not all states have the same transitions: thus, any transition to square 30 (*Go to Jail*) must in fact be replaced by a transition to square 40 (*Jail*). Similarly, for any transition to the *Chance* or *Community Chest* squares, the possible redirections must be considered. There is also the problem of doubles: the stochastic process uses memory (number of turns in jail or number of consecutive doubles already rolled) and is therefore not a true Markov process. But since the memory is finite (3 rolls), one can reduce it to a memoryless process by considering a space with 123 states[5]: 120 states of the form $(i,j)_{i\in[0,40],j\in[0,2]}$ representing *being on square $i$ having already rolled $j$ consecutive doubles*, and 3 *jail* states representing the three turns one can spend in jail. It should also be noted that if one pays, as one often has an interest in doing early in the game, one spends only one turn in jail, and the transitions are therefore modified. One must therefore consider the transitions for a *jail* strategy and those for a *freedom* strategy[6].

One thus arrives at writing, for each of the 2 strategies considered, a stochastic matrix describing the strategy in question. Figure 4.2 thus graphically represents the matrix corresponding to the *jail* strategy.

### 4.4.3 Asymptotic Probabilities and Conclusion

Once the transition matrix $A$ is computed, thanks to Corollary 4.1, we know that it suffices to iterate $P_n = A^t P_{n-1}$ ($P_0$ being for example the uniform distribution) to obtain convergence to the asymptotic distribution. One then only needs to return to the space of $40+1$ squares to have usable results, which are shown in Figure 4.3. To complete this study, one would now need to take into account the sale prices and rents to obtain a mean time to return on investment, without forgetting to consider

[5] An alternative solution, proposed by [Col], consists of estimating for each square the probability of having arrived there by 2 consecutive doubles.

[6] Other factors can also alter the transition probabilities:

- The *Draw a Chance Card or Pay a Fine of…* card, which offers two strategies.
- The *Get Out of Jail Free* cards, which, if kept by the players, slightly increase the probabilities of drawing a displacement card.

The effect of these variations being relatively small, we allow ourselves not to take them into account here.

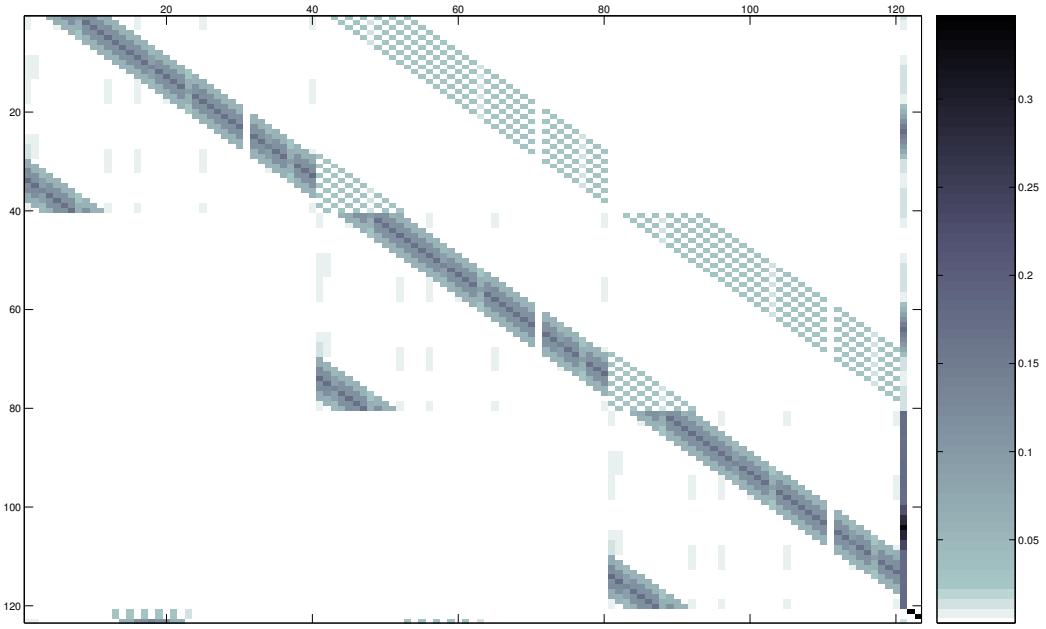



Figure 4.2: $123 \times 123$ Monopoly transition matrix. Transition matrix (*jail* strategy) in the *(square, number of doubles) + (jail, number of turns)* space

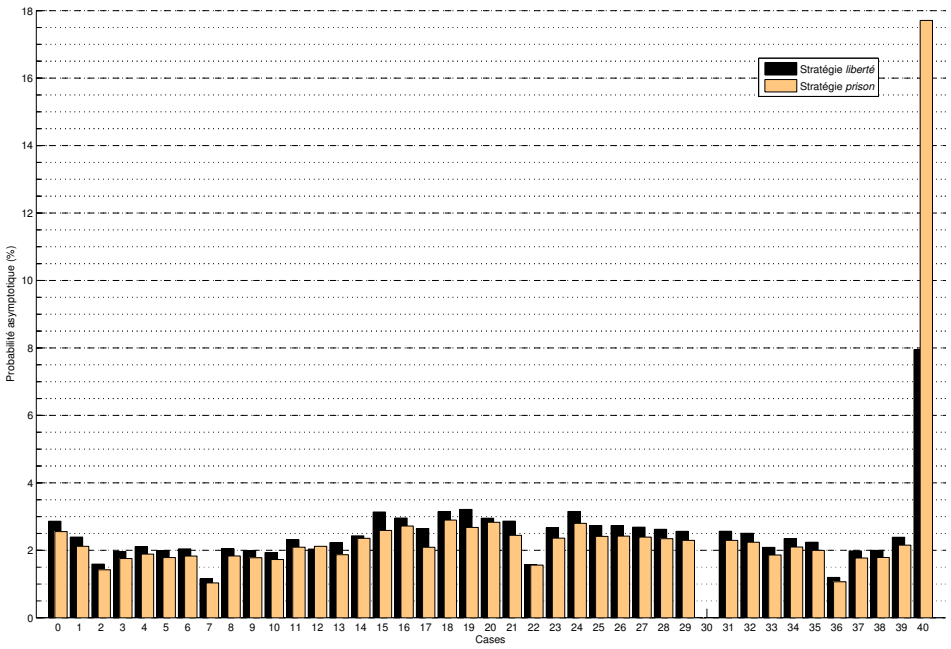



Figure 4.3: Asymptotic probabilities of the Monopoly game (registered trademark)

[7]For an idea of the results obtained, see [Col]. Note that the results correspond to the international Monopoly game, which has different cards from the French version.

purchasing power, but this is no longer within the scope of Markov matrices[7]. Let us conclude with just these few remarks:

- The *Go to Jail* square has zero probability, since one does not stay there.
- Similarly, the *Chance* and *Community Chest* squares have fairly low probability, because of the immediate displacement cards.
- The second and third quarters of the board generally have higher probabilities, because of the exit from jail. This confers a definite interest to the properties located there (the orange and red groups in particular).
- Paradoxically, one is more likely to land on the *Electric Company* by choosing to stay in jail. Why this counterintuitive result? With the *freedom* strategy, the probability of landing on it upon leaving jail is $1/36$. With the *jail* strategy, this probability becomes $1/36 + \frac{5}{6} \cdot 1/36 + \frac{25}{36} \cdot 1/36$…

## 4.5 (Sub-)Stochastic Matrices: General Case

In the following chapters, we will sometimes deal with matrices exhibiting periodicity, or that are non-irreducible, or even sub-stochastic[8], where the *or* is not necessarily exclusive. We therefore propose to study the viability of Corollary 4.1 under these different hypotheses.

### 4.5.1 Non-Irreducible Matrices

Let us first consider the case where $A = \left(a_{i,j}\right)_{1 \leq i,j \leq n}$ is stochastic but not irreducible. This means that the corresponding graph $G = (V, E)$ is not strongly connected. Let us then consider the decomposition into strongly connected components of $G$: $G = ((C_1, ..., C_k), E)$. Each component $C_c$ has exactly one of the two following properties:

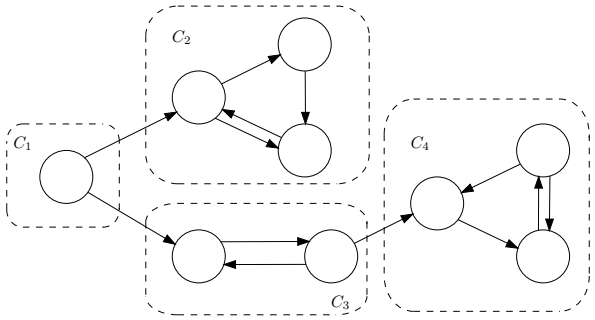



Figure 4.4: Example of a non-strongly-connected graph

[8]In this case, one can no longer *a priori* speak of an associated Markov chain.

- Either $\exists i \in C_c, j \notin C_c, a_{i,j} > 0$. The component and its states are then called transient.
- Or $\forall i \in C_c, j \notin C_c, a_{i,j} = 0$. The component and its states are then called recurrent.

If one quotients $G$ by its strongly connected components, the reduced graph gives a partial order on the components (since there are no circuits) whose recurrent components are the maxima.

For example, in the graph of Figure 4.4, there are four strongly connected components, $C_1$, $C_2$, $C_3$, and $C_4$. $C_1$ and $C_3$ are transient, while $C_2$ and $C_4$ are recurrent.

We will now reorder the states $V$ as follows: first the $k_t$ transient states (all components combined), then the $k_1$ states of a first recurrent strongly connected component, ..., up to the $k_d$ states of the last recurrent strongly connected component. In this rearrangement of states, the associated stochastic matrix (which we will continue to call $A$) can now be written:

$$A = \begin{pmatrix} T & & E & \\ & \begin{pmatrix} R_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & R_d \end{pmatrix} \\ 0 & \end{pmatrix} \tag{4.13}$$

where $T$ is a sub-stochastic, non-stochastic matrix of size $k_t$, $E$ is a nonnegative nonzero matrix of size $k_t \times \sum_{i=1}^{d} k_i$, and the $R_i$ are irreducible stochastic matrices of size $k_i$.

**Theorem 4.3**: Let $A$ be a stochastic matrix reduced according to its transient and recurrent strongly connected components. $A$ is of the form

$$A = \begin{pmatrix} T & E \\ 0 & R \end{pmatrix} \tag{4.14}$$

where $T$ is a sub-stochastic, non-stochastic matrix of size $k_t$, $E$ is a nonnegative nonzero matrix of size $k_t \times \sum_{i=1}^{d} k_i$, and

$$R = \begin{pmatrix} R_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & R_d \end{pmatrix} \tag{4.15}$$

the $R_i$ being irreducible stochastic matrices of size $k_i$.

If all the matrices $R_i$ are aperiodic, then the iterated powers of $A$ converge. More precisely, we have:

$$A^k \underset{k\to\infty}{\longrightarrow} \begin{pmatrix} 0 & F \\ 0 & R_\infty \end{pmatrix} \tag{4.16}$$

with

$$R_\infty = \begin{pmatrix} R_{1\infty} & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & R_{d\infty} \end{pmatrix} \tag{4.17}$$

$$R_{i\infty} = \lim_{k\to\infty} R_i^k = P_{i\infty} \cdot \mathbf{1}_{k_i} \tag{4.18}$$

where $P_{i\infty}$ is the probability vector of size $k_i$ satisfying $R_i^t P_{i\infty} = P_{i\infty}$, and

$$F = \left(Id_{k_t} - T\right)^{-1} E R_\infty \tag{4.19}$$

**Corollary 4.2**: For any probability distribution $P_0$ on $V$, $\left(A^t\right)^k P_0$ converges as $k$ tends to infinity, and the limit vector is a probability distribution belonging to the $d$-dimensional space spanned by the canonical embedding of the $P_{i\infty}$ into $V$.

**Remark 4.1**: When not all the $R_i$ are aperiodic, there is *a priori* no convergence. The technique that we will see in Section 4.5.2 nevertheless allows one to ensure convergence at low cost.

*Proof*: We first observe that

$$A^k = \begin{pmatrix} T & E \\ 0 & R \end{pmatrix}^k = \begin{pmatrix} T^k & \sum_{i=0}^k T^i E R^{k-i} \\ 0 & R^k \end{pmatrix} \tag{4.20}$$

**Lemma 4.1**:

$$T^k \underset{k\to\infty}{\longrightarrow} 0 \tag{4.21}$$

and the convergence is geometric. Moreover, $\left(Id_{k_t} - T\right)$ is invertible, and its inverse equals $\sum_{k=1}^{\infty} T^k$.

Indeed, let us examine the structure of $T$ more closely. We will reorder the states by strongly connected components, starting with those that, in the quotient graph $G/C$, have no incoming edges (ultra-transient components), and sorting by distance from the ultra-transient components. The matrix $T$ is then of the form:

$$T = \begin{pmatrix} T_1 & G_{1,2} & \dots & G_{1,l} \\ 0 & \ddots & G_{i,j} & \vdots \\ \vdots & \ddots & \ddots & G_{l-1,l} \\ 0 & \dots & 0 & T_l \end{pmatrix} \tag{4.22}$$

where the $T_i$ are irreducible sub-stochastic, non-stochastic matrices (by convention, the one-dimensional matrix 0 is considered irreducible). By Section 4.5.3, each matrix $T_i$ on the main diagonal has a spectral radius $\rho(T_i)$ strictly less than 1. Since the structure of $T$ is block upper triangular, the spectral radius of $T$ is $\rho(T) = \max_{1\le i\le l} \rho(T_i) < 1$. This ensures that $T^k$ converges geometrically to 0.

Since 1 is not an eigenvalue of $T$, $\left(Id_{k_t} - T\right)$ is invertible. Now, for all $k$, we have:

$$\left(Id_{k_t} - T\right) \sum_{j=0}^{k} T^j = Id_{k_t} - T^{k+1} \tag{4.23}$$

hence

$$\sum_{j=0}^{k} T^j = \left(Id_{k_t} - T\right)^{-1} \left(Id_{k_t} - T^{k+1}\right) \tag{4.24}$$

We deduce that

$$\lim_{k\to\infty} \sum_{j=0}^{k} T^j = \left(Id_{k_t} - T\right)^{-1} \tag{4.25}$$

which completes the proof of the lemma.

Let us return to our proof. It is now established that $T^k \underset{k\to\infty}{\longrightarrow} 0$. With the aperiodicity hypothesis on the $R_i$, we also have $R^k \underset{k\to\infty}{\longrightarrow} R_\infty$.

It remains to prove that

$$\sum_{i=0}^{k} T^i E R^{k-i} \underset{k\to\infty}{\longrightarrow} \left(Id_{k_t} - T\right)^{-1} E R_\infty \tag{4.26}$$

Now, we have

$$\sum_{i=0}^{k} T^i E R^{k-i} = \sum_{i=0}^{k} T^i E R_\infty + \sum_{i=0}^{k} T^i E\left(R^{k-i} - R_\infty\right) \tag{4.27}$$

The first term converges to $\left(Id_{k_t} - T\right)^{-1} E R_\infty$. As for the second, we have:

$$\left|\sum_{i=0}^{k} T^i E\left(R^{k-i} - R_\infty\right)\right| \leq \left|\sum_{i=0}^{\lfloor k/2 \rfloor} T^i E\left(R^{k-i} - R_\infty\right)\right| + \left|\sum_{i=\lfloor k/2 \rfloor + 1}^{k} T^i E\left(R^{k-i} - R_\infty\right)\right| \tag{4.28}$$

The first of the two right-hand terms tends to 0 because of the geometric convergence of $R^k$ to $R_\infty$, the second because of the (also geometric) convergence of $\sum_{i=0}^{k} T^k$. This ensures the convergence to 0 of the left-hand term. Q.E.D. ■

### 4.5.2 Periodic Matrices

If a stochastic matrix $A$ is periodic, there is *a priori* no convergence. One can for example think of the circulant matrix

$$C = \left(c_{i,j}\right)_{1\leq i,j\leq n} \quad \text{with } c_{i,j} = \begin{cases} 1 \text{ if } j \equiv (i+1)[n] \\ 0 \text{ otherwise} \end{cases} \tag{4.29}$$

The successive iterates of $C$ describe an orbit of size $n$ corresponding to the $n$-th roots of unity, and in particular there is no convergence in the classical sense, although there exists convergence in the Cesaro sense ($\frac{1}{k}\sum_{i=0}^{k} C^i$ converges).

Cesaro convergence could allow us to recover the eigenvector direction associated with the maximal positive eigenvalue, but this is not necessary: as shown by Theorem 4.4, it is possible to reduce to "classical" convergence.

**Theorem 4.4**: Let $A$ be an irreducible stochastic matrix of size $n$, possibly periodic. Let $P$ be the unique probability vector such that $A^t P = P$. For all $\alpha \in ]0,1[$, we have

$$\left(\alpha A^t + (1-\alpha)Id\right)^k \underset{k\to\infty}{\longrightarrow} P \cdot \mathbf{1}_n \tag{4.30}$$

**Corollary 4.3**: Let $P_0$ be an arbitrary probability vector. If we set $B = (\alpha A + (1-\alpha)Id)$, then

$$\left(B^t\right)^k P_0 \underset{k\to\infty}{\longrightarrow} P \tag{4.31}$$

*Proof*: The matrix $B$ defined by $B = (\alpha A + (1-\alpha)Id)$ is stochastic, irreducible, and aperiodic, due to the presence of self-loops of length 1. $\left(B^t\right)^k$ therefore converges to $Q \cdot \mathbf{1}_n$, where $Q$ is the unique probability distribution satisfying $B^t Q = Q$. Since $B^t P = (\alpha A^t + (1-\alpha)Id)P = \alpha P + (1-\alpha)P = P$, we have $P = Q$.

Q.E.D. ■

### 4.5.3 Sub-Stochastic Matrices

The case of strictly sub-stochastic matrices seems *a priori* simple to resolve:

**Theorem 4.5**: Let $A$ be a sub-stochastic, non-stochastic, irreducible matrix of size $n$. Then the iterated powers of this matrix tend to 0:

$$A^k \underset{k\to\infty}{\longrightarrow} 0 \tag{4.32}$$

*Proof*: $A$ is dominated by and not equal to an irreducible stochastic matrix. By part (d) of the Perron-Frobenius theorem (see Appendix A, page 130), its spectral radius is strictly less than 1, which ensures the result, namely, if we call $\rho$ the spectral radius, convergence dominated by a geometric sequence with ratio $\rho$. ■

**Remark 4.2**: A sub-stochastic but non-stochastic matrix corresponds to an ill-defined Markov chain, in the sense that not all possible transitions have been specified. By analogy with automata, we will speak of an incomplete Markov chain[9]. In this case, for any probability distribution $P_0$, $\left(A^t\right)^k P_0 \underset{k\to\infty}{\longrightarrow} 0$, a result that is unsatisfactory in terms of useful information. This problem is crucial in the context of PageRank computations, and the solutions for "completing" a sub-stochastic matrix will be given in more detail in Chapter 5.

**Remark 4.3**: Theorem 4.5 can in fact be applied to any sub-stochastic matrix $A$ that is dominated by and not equal to an irreducible stochastic matrix. We will call such matrices sub-irreducible matrices.

However, as we will see in Chapter 5, the study of the maximal eigenvalue of a sub-irreducible matrix and its associated eigenspace is important for the study of PageRank, which is why we will develop this a bit further.

[9]Indeed, just as with automata, it is possible to complete our Markov chain by adding a *sink* state receiving the *stochastic deficiency* of the other states, and pointing surely to itself.

[10]Although the study we are about to undertake is primarily of interest from the point of view of sub-irreducible matrices, it is in fact valid for any nonnegative matrix.

[11]For $C \subset V$, the filter of $C$, denoted $\Uparrow C$, is the set of vertices of $V$ accessible from $C$.

### Maximal Eigenvalue and Associated Eigenspace of a Nonnegative Matrix

**Theorem 4.6**: Let $A$ be a nonnegative matrix[10].

Let $(C_1, ..., C_k)$ be the decomposition into strongly connected components of the graph $G$ associated with $A$.

We define the spectral radius $\rho(C_i)$ of a component $C_i$ as the spectral radius of $A_{C_i}$. We will call a pseudo-recurrent component of $G$ any component $C_i$ satisfying:

- $\rho(C_i)$ is maximal: $\forall 1 \leq j \leq k$, $\rho(C_j) \leq \rho(C_i)$.
- the components accessible from $C_i$ have a strictly smaller spectral radius: $\forall C_j \subset \Uparrow C_i$[11], $\rho(C_j) = \rho(C_i) \Rightarrow C_j = C_i$.

Then:

1. The spectral radius of $A$ is equal to the maximum spectral radius of the strongly connected components of $G$.
2. There exists a maximal eigenvalue that is positive (it is the only one if the pseudo-recurrent components are aperiodic). The dimension of the associated eigenspace is then equal to the number $d$ of pseudo-recurrent components.
3. If $C_i$ is a pseudo-recurrent component of $G$, there exists a nonnegative eigenvector with support $\Uparrow C_i$ associated with the maximal eigenvalue.

**Corollary 4.4**: If there exists a single pseudo-recurrent component $C_p$, and if all vertices are accessible from this component ($\Uparrow C_p = V$), then the eigenspaces associated with the maximal eigenvalues[12] are one-dimensional, and there exists a nonnegative eigenvector with support $V$ associated with the maximal positive eigenvalue. We then say that $A$ is pseudo-irreducible.

*Proof*: The proof is in fact very similar to that of Theorem 4.3 for non-irreducible stochastic matrices, even though it is no longer possible to have a convergence result for the powers of $A$. Thanks to the partial order induced by the quotient graph of the strongly connected components, it is possible to make $A$ block upper triangular according to the strongly connected components: up to choosing the right permutation, one can write

$$A = \begin{pmatrix} A_{C_1} & T_{12} & \dots & T_{1k} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{(k-1)k} \\ 0 & \dots & 0 & A_{C_k} \end{pmatrix} \tag{4.33}$$

[12]The plural is used here for possible periodicities. In the aperiodic case, there is of course only one maximal eigenvalue.

where the $T_{ij}$ are the transition matrices between components $i$ and $j$.

This triangular decomposition ensures that the spectral radius of $A$ is equal to the maximum spectral radius of the different components $C_i$. In fact, by applying the Perron-Frobenius theorem to each of the strongly connected components, it turns out that there exists an eigenvalue $\lambda > 0$ equal to the spectral radius, and that its multiplicity equals the number of components with maximal radius.

If $C_i$ is pseudo-recurrent, then the multiplicity of $\lambda$ in $A^t_{\uparrow C_i}$ is 1. There therefore exists, up to scaling, a unique eigenvector $x$ associated with $\lambda$ in $A^t_{\uparrow C_i}$. Since $\Uparrow C_i$ is stable under $A^t$, the embedding of $x$ into $V$ is an eigenvector of $A^t$. By construction, we also have $A^t_{C_i} x_{C_i} = \lambda x_{C_i}$. By the Perron-Frobenius theorem applied to $A_{C_i}$, the components of $x_{C_i}$ are strictly positive up to scaling. Step by step, the equality $A^t_{\uparrow C_i} x = \lambda x$ shows that $x$ is strictly positive.

For each pseudo-recurrent component of $G$, there therefore exists a nonnegative eigenvector with support $\Uparrow C_i$ associated with $\lambda$.

It remains to show that there is no eigenvector associated with $\lambda$ outside of the space spanned by the $d$ eigenvectors associated with the pseudo-recurrent components. For this, it suffices to observe that any non-pseudo-recurrent component $C_i$ with maximal radius generates a triangular structure in the space associated with $\lambda$: if there exists a pseudo-recurrent component $C_j$ with $C_j \subset \Uparrow C_i$, then $A_{\uparrow C_i}$ contains (in a suitable basis) a block of the form $\begin{pmatrix} \lambda & \mu \\ 0 & \lambda \end{pmatrix}$, with $\mu \neq 0$, which shows that it is not possible to associate an eigenvector of $\lambda$ to $C_i$.

Q.E.D. ■

### 4.5.4 General Case: Conclusion

We have just seen that one can arrange to find a convergence algorithm even if the matrix is neither aperiodic nor irreducible. Let us simply note that in the latter case, there is no uniqueness. On the other hand, if the matrix is sub-stochastic but not stochastic, the asymptotic vector will be zero everywhere except on possible recurrent strongly connected components within which the matrix is stochastic. The canonical completion, which consists of adding a "sink" state, is not satisfactory because it does not fundamentally change the result: the "sink" state absorbs the probabilities lost at the sub-stochastic components, but the values on the states other than the *sink* state are not modified. Fortunately, the various PageRank computation algorithms that we will now study will provide us with other completion methods to reliably find a vector having all the desired properties, which will be called the PageRank vector.

# Chapter 5
# PageRank, a way to estimate the importance of web pages

*When a researcher wants to publish a paper in a specialized journal, he must go through review committees that evaluate the quality of his work according to precise criteria. To appear on television or in a newspaper, all one needs is a good story to tell.*

*Michel de Pracontal,* l'imposture scientifique

*Surfing the internet is like sex: everyone brags about doing more than they actually do. But in the case of the Internet, they brag far more.*

*Tom Fasulo*

*Omnes Viae ad Googlem ducent*

This chapter will attempt to lay the intuitive, axiomatic, and theoretical foundations of *PageRank*-type ranking methods, brought to prominence by the famous *Google* search engine [Goo98]. This painstaking *survey* work was greatly facilitated by Mohamed Bouklit, who, with the help of Alain Jean-Marie, cleared the ground before me [Bou01, BJ02], providing me with all the necessary references.

There already exist a number of *surveys* on PageRank (cf [BGS02, BGS03, LM04]), but we deemed this chapter necessary because it presents all PageRanks from a single perspective, that of the stochastic interpretation, and highlights several convergence results that cannot be found elsewhere[1].

[1] At least not applied to PageRanks.

## 5.1 A needle in a haystack...

One can find (just about) anything on the web; virtually all network users agree on this. Whether the information I am looking for exists on the web is no longer the crucial question. Nowadays, the problem has become: *How do I find the page I am looking for?*

- By knowing, one way or another, the address in question. For instance, by reading on a bag of flour the address of a site offering cooking recipes, by receiving an email from a friend recommending a web address, or by having the address in one's *Bookmarks*…
- By navigating (surfing) the web, starting from a known address. If, for example, I am looking for a page about raccoons and I know of a zoology website, starting from that site may be a good way to find the page I am looking for.
- By using a search engine. In exchange for a *query*, that is, a set of words attempting to describe more or less precisely what I am looking for, the engine will return a list of pages likely to answer this query. According to [LG99], 85% of internet users rely on search engines.

The whole challenge for search engines is to return the pages the user is actually looking for. However, the answers to a given query often number in the hundreds or even thousands[2], as shown in Table 5.1. On the other hand, users quickly lose patience, and it is estimated that 90% of users do not go beyond the first page of results.

The goal of search engines is therefore to display within the first ten to twenty results the documents that best answer the question posed. In practice, no sorting method is perfect, but their variety gives search engines the possibility of combining them to better refine their results. The main sorting methods are the following:

**Relevance ranking** Relevance ranking is the oldest and most widely used sorting method. It is based on the number of occurrences of the search terms in the pages, their proximity, their position in the text [Sal89, YL95]… Unfortunately,

| **Query** | **Google** | **Yahoo** |
|---|---:|---:|
| PageRank | 1 410 000 | 807 000 |
| Raton laveur | 18 100 | 25 300 |
| Amazon | 107 000 000 | 66 600 000 |
| Pâte à crêpes | 46 300 | 30 900 |
| Pâte à crêpe | 22 800 | 47 000 |

Table 5.1: Number of results returned by Google and Yahoo for selected queries (August 2004)

[2]The problem is in fact less critical if duplicates are eliminated — see Table 2.1, page 29 — but it remains significant.

this method has the drawback of being easy to manipulate by authors wishing to place their pages at the top of the list. To do so, it suffices to overload the page with important words, either in the header or in invisible text within the body of the page.

**URL analysis** One can also assign importance to a page based on its URL. For instance, depending on the context, URLs belonging to the *Top Level Domain* `http:.com` might be more important than others, as might URLs containing the string `http:home` [Li+00]. It has also been suggested that URLs at a low depth in the cluster tree are more important than others [CGP98].

**Inbound links** Citation counting consists of assigning pages an importance proportional to the number of known links to that page. This method has been widely used in scientometrics to evaluate the importance of publications [Pri63].

**PageRank** As we shall see, PageRank is a kind of recursive generalization of citation counting.

## 5.2 The two axioms of PageRank

PageRank, introduced by Brin *et al.* in 1998 [Pag+98], is the ranking method that made the *Google* search engine [Goo98] distinctive. It is in fact an adaptation to the Web of various methods introduced by scientometricians[3] since the 1950s. Two scientometric methods in particular should be mentioned:

**Citation counting** In 1963, Price published *Little Science, Big Science* [Pri63]. In this book, the first major work dealing with what would later become scientometrics, he proposed measuring the quality of scientific production through, among other things, a very simple technique, citation counting: one way to measure the quality of a publication is to count the number of times that publication is cited.

**Markovian modeling** Markov chains, described in Chapter 4, not only allow one to play Monopoly (registered trademark), but also to model the evolution of a population distributed among several states, provided one can estimate the transition probabilities between states. For example, Goffman proposed in 1971

[3]For several decades now, a new field of research has emerged that is devoted to the study of human intellectual production. The main branches of this meta-science are:

**Bibliometrics** defined in 1969 as "the application of mathematics and statistical methods to books, articles, and other means of communication."

**Scientometrics** it can be considered as bibliometrics specialized to the field of Scientific and Technical Information (STI). However, scientometrics more generally refers to the application of statistical methods to quantitative data (economic, human, bibliographic) characterizing the state of science.

**Informetrics** a term adopted in 1987 by the F.I.D. (International Federation of Documentation, IFD) to designate all metric activities related to information, covering both bibliometrics and scientometrics.

the study of the evolution of research in various subfields of logic using Markov chains [Gof71].

Let us now see how Brin *et al.* adapted these concepts to estimating the importance of Web pages.

### 5.2.1 An important page is pointed to by important pages

Transposed directly to Web graphs, the citation counting method amounts to saying that the importance of a page is proportional to its in-degree, that is, to the number of pages that cite it through a hyperlink:

$$I(v) = \sum_{w \to v} 1 \tag{5.1}$$

where $w \to v$ means $w$ *points to* $v$.

Although this measure can indeed be used to estimate the importance of pages [CGP98], it is partly undermined by the absence of quality control. Indeed, when a researcher wants to publish a paper in a specialized journal, he must go through review committees that evaluate the quality of his work according to precise criteria. Because of this, the mere fact of being published gives a minimum level of importance to the articles in question, and there is some guarantee that the citations a paper receives are not entirely spurious. In the case of the Web, this safeguard does not exist: because of the low intellectual and material cost of a web page, anyone can link to anything without it necessarily having any real meaning[4]. For instance, why not create a multitude of meaningless pages that cite me through hyperlinks, in order to inflate my own importance at will?

Brin *et al.* propose to counter this problem with a recursive description of importance: What is an important page? It is a page pointed to by important pages. Concretely, if a page $v$ is pointed to by $k$ pages $v_1, ..., v_k$, the importance of $v$ should be defined by:

$$I(v) = f_v(I(v_1), ..., I(v_k)) \tag{5.2}$$

It remains to define $f_v$ and solve Equation (5.2).

### 5.2.2 The random surfer: chance does things well

A romanticized view of the Web is the principle of hyperlink navigation: to find what he is looking for, the user navigates from page to page and click to click until reaching his destination. Of course, this is not what happens in practice, for both technical reasons (it is not always possible to reach a page $p$ from a page $q$) and social reasons (use of search engines, user fatigue…)[5].

[4] After all, is it not said that *you can find anything on the Web*?

[5] Cf [CP95, CM01, Mil+04, TG97, WM04].

Brin *et al.* had the idea of modeling the behavior of the clicking user by a Markov chain. All that was needed was to find the transition probabilities from one page to another. One of the simplest ways of looking at things is to consider that once on a given page, the user will click uniformly at random on one of the links contained in that page:

$$p_{v,w} = \begin{cases} \frac{1}{d(v)} \text{ if } v \to w \\ 0 \text{ otherwise} \end{cases} \quad \text{where } d \text{ is the out-degree} \tag{5.3}$$

This is the basis of the random surfer model. For Brin *et al.*, given that Web graphs reflect a deliberate and thoughtful architecture, interesting pages should be structurally easy to access, just as a city is all the more accessible by the road network the more important it is. Therefore, since the random surfer lets himself be guided by the hyperlink network, statistically, he should land on a page all the more frequently the more important it is. Hence the idea of defining the importance of a Web page by the asymptotic probability of being on that page in the random surfer model.

### 5.2.3 Consistency of the two interpretations

Now that we have defined two ways of considering the importance of a page, let us observe that they coincide and describe the same phenomenon: indeed, in the random surfer model, it is possible to estimate the asymptotic probability of being on a page $v$ as a function of those of the pages $w$ that point to $v$:

$$P(v) = \sum_{w \to v} \frac{1}{d(w)} P(w) \tag{5.4}$$

One can see that we indeed have an importance transfer relation like the one defined by Equation (5.2), and that Equation (5.4) additionally obeys a conservation principle: a given page transmits all of its importance, which is equally distributed among the different pages it points to. Probability, viewed as importance, is thus transmitted through hyperlinks in the manner of a flow.

## 5.3 The classical models

Although PageRank is often referred to in the singular, there actually exists a multitude of PageRank(s). We shall see here how, starting from the theoretical random surfer model that we have just described, several variations have been introduced in order to adapt to the reality of Web graphs. This *survey* work has already been carried out for all PageRank(s) arising from explicitly stochastic processes [BGS02, BGS03, LM04], but we also wish to take into account sub-stochastic models here.

### 5.3.1 Ideal case

In the case where the Web graph $G = (V, E)$ that we wish to study is aperiodic and strongly connected, the principles seen in Section 5.2 apply directly: indeed, the task is to find a probability distribution on $V$ satisfying:

$$\forall v \in V, P(v) = \sum_{w \to v} \frac{P(w)}{d(w)} \tag{5.5}$$

This amounts to finding the asymptotic distribution of the homogeneous Markov chain whose transition matrix is:

$$A = \left(a_{i,j}\right)_{i,j \in V} \quad \text{with } a_{i,j} = \begin{cases} \frac{1}{d(i)} \text{ if } i \to j \\ 0 \text{ otherwise} \end{cases} \tag{5.6}$$

As seen in Section 4.3, the distribution sequence

$$P_{n+1} = A^t P_n \tag{5.7}$$

initiated by an arbitrary probability distribution $P_0$[6], converges geometrically to the unique distribution $P$ satisfying

$$P = A^t P \tag{5.8}$$

that is, satisfying relation Equation (5.5).

This probability distribution $P$ is called the PageRank.

**Remark 5.1**: As specified in our definition of a Web graph, links from a page to itself are not counted. According to [Pag+98], this allows the PageRank computation to be "smoothed out."

**Remark 5.2**: If the graph is not strongly connected, by Theorem 4.3, convergence still occurs, but there is neither uniqueness (the dimension of the solution space equals the number of recurrent strongly connected components) nor a guarantee that the support of the solution equals $V$ (in particular if transient components exist). The existence of periodicity, on the other hand, may prevent convergence, but Theorem 4.4 indicates how the problem can be circumvented.

### 5.3.2 Simple renormalization

Most of the time, a Web graph is not strongly connected. In particular, there exists a non-negligible number of dangling pages, which are either pages that actually

[6]Very often, the initial probability vector is taken to be a *zap* distribution $Z$ — see Section 5.3.3.

contain no links, or simply pages that are known but not indexed. The rows of $A$ corresponding to these dangling pages therefore contain only 0s, and $A$ is thus strictly sub-stochastic. Consequently, the sequence of $P_n$ will converge[7] to a vector that is zero outside any recurrent strongly connected components on which $A$ is stochastic[8]. To avoid this problem, one could consider recursively removing all dangling pages until a stochastic matrix is obtained, with the drawback of considering a smaller graph than the original. Another approach, proposed by [Pag+98], consists in renormalizing $P_n$ at each iteration:

$$P_{n+1} = \frac{1}{\|A^t P_n\|_1} A^t P_n \tag{5.9}$$

This iterative procedure is a power method ([Ste94]), so it is known to converge to an eigenvector associated with the largest eigenvalue of $A$. Two cases must then be considered:

- If the matrix $A$ is sub-irreducible, then its maximum eigenvalue is strictly less than 1. By Theorem 4.6[9], the associated eigenspace has dimension $d$, where $d$ is the number of pseudo-recurrent components, and its support is that generated by the union of the filters of the pseudo-recurrent components. In the particular case where $A$ is pseudo-irreducible, the eigenspace is a line, and there exists an associated strictly positive eigenvector: thanks to renormalization, everything proceeds as in the case of an irreducible stochastic matrix.
- If the matrix $A$ is not sub-irreducible, $G$ contains at least one recurrent strongly connected component on which $A$ is stochastic. The maximum eigenvalue of $A$ is therefore 1, which means that renormalization will not change the initial result: the eigenvector will be a linear combination of the eigenvectors on the different strongly connected components on which $A$ is stochastic[10].

**Equivalent stochastic process**

The use of sub-stochastic matrices means that we lose the natural interpretation of the random surfer. However, it is sometimes possible to complete the matrix into an equivalent stochastic matrix, that is, to find a stochastic matrix $B$ satisfying:

- $A \leq B$
- if $A^t P = \lambda P$, with $\lambda$ maximal, then $P = B^t P$

If there exists a unique maximal probability vector $P$ for $A^t$, the simplest way to define such a matrix $B$ is to consider the stochastic defect of $A$: if $A$ is sub-stochastic, the stochastic defect of $A$ is defined as the vector $s = \mathbf{1}_n^t - A \cdot \mathbf{1}_n^t$. The matrix

$$B = A + s \cdot P^t \tag{5.10}$$

[7] Subject to aperiodicity. See Section 4.5.2, page 66 for possible periodicities.

[8] Given the way the graph is constructed, any recurrent strongly connected component not reduced to a single element induces a stochastic process.

[9] See Theorem 4.6, Theorem 4.6.

[10] In other words, the recurrent strongly connected components.

is indeed a stochastic matrix[11] greater than $A$, and it is easy to see that $B^t P$ is a probability distribution homogeneous to $P$, hence equal to $P$.

The matrix $B$ has the advantage of giving a stochastic interpretation to the simple renormalization procedure: asymptotically, the random surfer follows at each step a link according to the probabilities given by $A$. When he does not know what to do, he *zaps* to another page according to the probability distribution that is the maximal eigenvector of $A$.

On the other hand, as soon as the maximal eigenspace has dimension greater than 1, the problem is delicate, even very delicate in the case of pseudo-recurrent components whose filters have a non-zero intersection. We will therefore limit our study of equivalent stochastic processes to cases where the maximal eigenvector is unique.

### 5.3.3 Stochastic completion

In order to replace $A$ with a stochastic matrix, and to avoid establishing an uncontrolled equivalent stochastic process through simple renormalization, a natural idea is to add transitions to dangling pages. One possible method consists in modeling the use of the *Back* button, and will be the subject of Chapter 6. Another method, proposed[12] initially by [Pag+98] (in the form of Algorithm 5.1) and studied among others by [LM04], consists in defining a *default* probability distribution $Z$ on $V$, and using it to model the behavior of the random surfer when he arrives on a dangling page. Concretely, each zero row in $A$ (which therefore corresponds to a dangling page) is replaced by the row $Z^t$.

This procedure can be generalized to complete any sub-stochastic matrix: The completion of $A$ by $Z$ is then the stochastic matrix

$$\overline{A} = A + s \cdot Z^t \tag{5.11}$$

**Choice of $Z$ and interpretation**

$Z$ represents the behavior of the random surfer when $G$ does not specify where he should go, that is, all page changes that are not due to the use of hyperlinks (manually typed address, *Bookmarks*, search engine query…). Generally, the uniform distribution is chosen for $Z$:

$$\forall v \in V, Z(p) = \frac{1}{n} \tag{5.12}$$

which represents a *zap* anywhere at random on the known Web.

It has also been proposed to "personalize" $Z$, notably by [Pag+98]. For instance, it is possible to restrict to the home pages of sites. On the one hand, this avoids

[11] If $A$ is sub-stochastic and $D$ a probability distribution on $V$, by construction, $(A + s \cdot D^t)$ is always a stochastic matrix.

[12] Unknowingly? See Remark 5.3.

**Data**

- a sub-stochastic matrix $A$ with no periodicity;
- a covering *zap* distribution $Z$;
- a real number $\varepsilon$.

**Result**

A (the if $A$ is sub-irreducible) probability eigenvector $P$ of $\overline{A}^t$ associated with eigenvalue 1.

**begin**

$n \leftarrow 0$, $P_n \leftarrow Z$, $\delta \leftarrow 2\varepsilon$

**while** $\delta > \varepsilon$ :

$P_{n+1} \leftarrow A^t P_n$

$\mu \leftarrow \|P_n\|_1 - \|P_{n+1}\|_1$

$P_{n+1} \leftarrow P_{n+1} + \mu Z$

$\delta \leftarrow \|P_n - P_{n+1}\|_1$

$n \leftarrow n + 1$

**return** $P_n$

Algorithm 5.1: PageRank: stochastic completion model (after [Pag+98])

implicitly giving sites a partial importance measure proportional to the number of crawled pages (see Section 7.5, page 116). On the other hand, this obeys a certain natural intuition: when one breaks hyperlink navigation to *zap* to something else, it is likely that one will start from a home page. This intuition is confirmed by numerous studies that demonstrate the existence of pages that play the role of "hubs" for actual users [CP95, Kle98, Mil+04, WM04].

**Irreducibility of $\overline{A}$**

A probability distribution $Z$ is said to be covering if its support $\chi_Z$ satisfies $\Uparrow \chi_Z = V$. This is a natural condition to impose on any *zap* distribution, since it guarantees that all known pages are potentially accessible after a *zap*. The uniform distribution is obviously covering. The same holds for the distribution on home pages if all known pages of a site are accessible from the home page. It is also the case for the importance distribution if, and only if, all pages in $V$ have a non-zero in-degree.

**Theorem 5.1**: Let $Z$ be a covering probability distribution, and $A$ a strictly sub-stochastic matrix. The completion of $A$ by $Z$ is irreducible if, and only if, $A$ is sub-irreducible.

*Proof*: If $A$ is sub-irreducible, then from any page $v$ of $V$, it is possible to access a page with a stochastic defect $w \in \chi_s$. Indeed, let $B$ be an irreducible stochastic matrix such that $A < B$, and consider a path connecting $v \in V$ to

$w \in \chi_s$ in the graph induced by $B$. Either this path exists in $G$, and we have what we need, or it does not exist, which implies that at least one of the pages $w'$ on the path has a stochastic defect, and therefore $w' \in \chi_s$.

$\overline{A}$ is therefore irreducible, since any pair of pages is connected in the induced graph by at least one path passing through $\chi_s$.

Conversely, if $A$ is not sub-irreducible, there exists at least one stochastic strongly connected component strictly smaller than $V$. This component remains unchanged in $\overline{A}$, which proves that $\overline{A}$ is not irreducible.

Q.E.D. ■

### 5.3.4 Rank source: *zap* factor

In order to resolve the irreducibility problem, Brin *et al.* [Pag+98] propose another incorporation of the *zap* distribution $Z$. They replace $A$ by $\left(A + \alpha \cdot (Z \cdot \mathbf{1})^t\right)$, and seek to solve:

$$P = c\left(A^t + \alpha \cdot Z \cdot \mathbf{1}\right) \cdot P \tag{5.13}$$

If $\chi_Z = V$, one then has the guarantee of operating on a positive, irreducible, and aperiodic matrix, since the underlying graph is a clique[13]. The Perron-Frobenius theorem therefore ensures convergence of the iterative process to the eigenvector associated with the maximal eigenvalue. However, unless $\alpha \cdot Z \cdot \mathbf{1} \leq Z \cdot s^t$, the new matrix is neither stochastic nor even sub-stochastic, which makes the interpretation in terms of a random surfer problematic. In order to more easily give an interpretation to the iterative process, Brin and Page normalize the matrix by taking the weighted average by $d$ of A and $(Z \cdot \mathbf{1})^t$:

$$A \to \hat{A} = d \cdot A + (1 - d) \cdot (Z \cdot \mathbf{1})^t \tag{5.14}$$

The resulting matrix $\hat{A}$ preserves the (sub-)stochasticity of $A$, and it is irreducible and aperiodic. If $A$ is stochastic, $\hat{A}$ corresponds to the ideal case of Section 5.3.1[14]. The maximal eigenvector, which we shall call PageRank with *zap* factor, is then obtained by Algorithm 5.2. This vector represents the last academic version of PageRank presented by Brin and Page before Google's ranking methods became an industrial secret. It is what is generally referred to when one speaks of PageRank.

[13] If $Z$ is merely covering, irreducibility is still guaranteed, since one can go from any page to any other page *via* a page of $\chi_Z$. However, aperiodicity is no longer necessarily guaranteed (there exist simple counterexamples, for instance a branch-tree completed on its root), even if empirically, the problem does not arise for Web graphs.

[14] The cases where $A$ is strictly sub-stochastic will be studied in detail in Section 5.4.

**Data**

- a stochastic matrix $A$;
- a covering *zap* distribution $Z$;
- a *zap* coefficient $d \in ]0, 1[$;
- a real number $\varepsilon$.

**Result**

The probability eigenvector $P$ of $\hat{A}^t$ associated with the maximal eigenvalue.

**begin**

$n \leftarrow 0$, $P_n \leftarrow Z$, $\delta \leftarrow 2\varepsilon$

**while** $\delta > \varepsilon$ :

$P_{n+1} \leftarrow d \cdot A^t P_n + (1 - d) \cdot Z$

$\delta = \|P_n - P_{n+1}\|_1$

$n \leftarrow n + 1$

**return** $P_n$

Algorithm 5.2: PageRank: model with added *zap* factor (after [BP98])

**Remark 5.3**: As an aside, let us note in the founding PageRank paper ([Pag+98]) a slight confusion: Algorithm 5.1 is proposed to solve Equation (5.13). Although it is specified that the introduction of $\mu$ may have a slight impact on the influence of $Z$[15], the qualifier "slight" may be an understatement: in Equation (5.13), $Z$ ensures an aperiodic irreducible matrix. One therefore has the guarantee of a unique strictly positive eigenvector. In contrast, in Algorithm 5.1, one implicitly works on the completion of $A$ by $Z$, and we have just seen that the influence of $Z$ is negligible as soon as $A$ is not sub-irreducible[16] (Theorem 5.1). Fortunately, the confusion was resolved in subsequent articles, notably with the normalization of the *zap* factor [BP98].

**Interpretation**

When $A$ is stochastic, the dual interpretation of $\hat{A}$ is fairly straightforward:

**Importance transfer** with the *zap* factor, pages transmit only a fraction $d$ of their importance. In return, each page $p$ is assigned a Minimum Insertion PageRank (MIPR) equal to $(1 - d) \cdot Z(p)$.

**Random surfer** at each step of the stochastic process described by $\hat{A}$, the random surfer will click at random on one of the outgoing links, with probability $d$, or *zap* with probability $(1 - d)$ somewhere on the graph according to the distribution $Z$.

[15] "The use of [$\mu$] may have a small impact on the influence of [$Z$].", *op. cit.*

[16] Recall that it suffices for this to have two pages that point only to each other.

### 5.3.5 Choice of $d$

Before going further, it is appropriate to discuss the choice of the parameter $d$ and the reasons that led to this choice. To begin with, let us state a universal and unalterable empirical reality: $d$ equals $0,85$, give or take $0,05$. Since the beginnings of PageRank, $0,85$ has always been the reference value, and to my knowledge, practical PageRank computations following the rank source model seen in Section 5.3.4 always use a $d$ between $0,8$ and $0,9$.

**Convergence/graph alteration tradeoff**

By Theorem 5.2, if $A$ is stochastic, which can be assumed by performing a completion if necessary, the eigenvalues of $\hat{A}$ other than 1 are less than $d$ in absolute value[17]. This guarantees algorithms Algorithm 5.2 and Algorithm 5.3 a geometric convergence with ratio at most $d$. One therefore has an interest in choosing $d$ as small as possible… except that the smaller $d$ is, the greater the influence of the *zap*, which is a component external to the intrinsic Web graph. A small $d$ alters, or even distorts, the underlying graph. Choosing the largest $d$ that guarantees reasonable convergence therefore seems like a good tradeoff. Now, technical limitations mean that the number of iterations achievable by a search engine like *Google* is on the order of a hundred[18]. $d = 0,85$ offers a precision of $10^{-8}$ after 114 iterations, $10^{-11}$ after 156 iterations, and therefore appears to be heuristically the desired tradeoff. Indeed, as we shall see in Section 5.5, $10^{-8}$ corresponds to the differentiation threshold for a Web graph of one million pages, while $10^{-11}$ is the differentiation threshold for one billion pages.

**Theorem 5.2**: Let $A$ be a stochastic matrix. If $x$ is an eigenvector of $\hat{A}^t$ associated with $\lambda \neq 1$, then $x$ is an eigenvector of $A^t$ and $\hat{A}^t x = dA^t x$. In particular, $|\lambda| \leq d$.

*Proof*: Since $\mathbf{1}$ is a left eigenvector of $\hat{A}^t$ associated with 1, we have

$$\begin{aligned} \mathbf{1}\hat{A}^t x &= \mathbf{1} \cdot x \\ &= \lambda \mathbf{1} \cdot x \end{aligned} \tag{5.15}$$

Since $\lambda \neq 1$, $\mathbf{1} \cdot x = 0$, hence

$$\begin{aligned} \hat{A}^t x &= \lambda x \\ &= (d \cdot A^t + (1-d) \cdot (Z \cdot \mathbf{1}))x \\ &= dA^t x \end{aligned} \tag{5.16}$$

■

[17] If $A$ has more than one recurrent strongly connected component, $d$ is an eigenvalue.

[18] Indeed, PageRank must be recomputed periodically, and with several billion pages to process, each iteration takes a non-negligible amount of time.

**Remark 5.4**: In [Kam+03a] there is a proof of the fact that every eigenvalue other than 1 (which is simple for $\hat{A}$) is less than $d$. In [LM04], it is additionally shown that the secondary eigenvalues of $\hat{A}$ are equal to $d$ times those of $A$ (the multiplicities of 1 being counted as secondary), and the authors claim that their proof is more compact than that of [Kam+03a]. Theorem 5.2 additionally shows that the secondary eigenvectors of $\hat{A}^t$ are those of $A^t$, and we claim that our proof is more compact than that of [LM04]. All that remains is to find a theorem more precise than Theorem 5.2, with an even more compact proof…

## 5.4 Rank source and sub-stochastic matrices

In the rank source model seen in Section 5.3.4, we saw that while adding a *zap* factor associated with a covering distribution $Z$ guarantees the irreducibility of $\hat{A}$, the stochasticity of $\hat{A}$ remains that of $A$. When $A$ is sub-stochastic, one must therefore adapt, and we shall see in this section the main possible solutions.

### 5.4.1 Hybrid model: *zap* factor and renormalization

$\hat{A}$ is pseudo-irreducible (since it is irreducible). A first possible method for obtaining a fixed point is therefore to apply the simple renormalization method (cf Section 5.3.2) to the matrix $\hat{A}$.

The interpretation in terms of a random surfer is the same as in the case where $A$ is stochastic, except that the stochastic defect of $\hat{A}$ must be completed by a zap according to the distribution defined by the maximal probability eigenvector associated with $\hat{A}$.

### 5.4.2 Completion and rank source: $\mu$-compensation

Since stochastic completion is a relatively simple way of assimilating any sub-stochastic matrix to a stochastic matrix, it is interesting to hybridize the stochastic completion method with a *zap*-type method or virtual page addition (see Section 5.6). This avoids having to renormalize at each iteration, and provides greater consistency in terms of stochastic interpretation. Moreover, if one chooses a completion distribution equal to the *zap* distribution $Z$, one obtains a very simple algorithm (Algorithm 5.3): the $\mu$-compensation algorithm. The interpretation is equally simple: at each step of the stochastic process described by $\hat{\overline{A}}$, the random surfer will click at

**Data**
- a (sub-)stochastic matrix $A$;
- a *zap* and completion distribution $Z$ that is covering;
- a *zap* coefficient $d \in ]0,1[$;
- a real number $\varepsilon$.

**Result**
The probability eigenvector $P$ of $\hat{\overline{A}}^t$ associated with the maximal eigenvalue.

**begin**
$n \leftarrow 0$, $P_n \leftarrow Z$, $\delta \leftarrow 2\varepsilon$
**while** $\delta > \varepsilon$ :
  $P_{n+1} \leftarrow d \cdot A^t P_n$
  $\mu \leftarrow \|P_n\|_1 - \|P_{n+1}\|_1$
  $P_{n+1} \leftarrow P_{n+1} + \mu Z$
  $\delta \leftarrow \|P_n - P_{n+1}\|_1$
  $n \leftarrow n + 1$
**return** $P_n$

Algorithm 5.3: PageRank: $\mu$-compensation (after [Kam+03b])

random on one of the outgoing links (if any exist), with probability $d$. In all other cases, he will zap according to $Z$.

### 5.4.3 Non-compensated PageRank

The non-compensated PageRank algorithm consists in computing $P_{n+1} = dA^t P_n + (1-d)Z$, exactly as in Algorithm 5.2, without worrying about renormalization. This yields a strictly positive vector, which can be used to define a PageRank, as shown by Theorem 5.3 and the interpretation that follows.

**Theorem 5.3**: Let $A$ be a sub-stochastic matrix, $d$ a *zap* factor, $0 < d < 1$, and $Z$ a covering probability distribution for the graph induced by $A$.

The sequence $P_{n+1} = dA^t P_n + (1-d)Z$ converges geometrically, with ratio less than or equal to $d$, to a unique fixed point $P$, regardless of the initial vector $P_0$. $P$ is a strictly positive vector, and for any completion $\overline{A}$ of $A$, if $\overline{P}$ is the probability distribution associated with $\hat{\overline{A}}$, then $P \leq \overline{P}$, with a totally strict inequality unless $A$ is stochastic (i.e. $A = \overline{A}$).

*Proof*: Since $\forall X \in \mathbb{R}^n$, $\|A^t X\|_1 \leq \|X\|_1$, the mapping $X \to dA^t X + (1-d)Z$ is $d$-Lipschitz. It therefore has a unique fixed point toward which any sequence

$X_{n+1} = dA^t X_n + (1-d)Z$ converges geometrically, with ratio less than or equal to $d$[19].

This fixed point $P$ satisfies $P = dA^t P + (1-d)Z$, and is therefore equal to:

$$P = (1-d)\sum_{k=0}^{\infty} (dA^t)^k Z \tag{5.17}$$

In particular, it is strictly positive, since as $Z$ is covering, for every $w$ in $V$, there exists $v$ in $\chi_Z$, $k$ in $\mathbb{N}$ such that $\left((dA)^k\right)_{v,w} > 0$, and therefore

$$P(w) \geq (1-d)\left((dA)^k\right)_{v,w} Z(v) > 0 \tag{5.18}$$

■

### 5.4.4 Comparison: $\mu$-compensated or non-compensated algorithms?

There is a very strong link between the $\mu$-compensated algorithm and the non-compensated algorithm: they yield the same result, as shown by Theorem 5.4.

**Theorem 5.4**: Let $A$ be a sub-stochastic matrix, $Z$ a covering distribution. If $\overline{P}$ is the PageRank obtained by $\mu$-compensation (cf Algorithm 5.3), and $P$ the non-compensated PageRank, fixed point of the mapping $X \to dA^t X + (1-d)Z$, then $P$ is homogeneous to $\overline{P}$.

*Proof*: By passing to the limit, it is easy to see that $\overline{P}$ satisfies

$$\overline{P} = dA^t\overline{P} + \mu Z \quad \text{where } \mu = 1 - \left\|dA^t\overline{P}\right\|_1 \tag{5.19}$$

We deduce

$$\overline{P} = \mu\sum_{k=0}^{\infty} (dA^t)^k Z \tag{5.20}$$

Equation (5.17) and Equation (5.20) give us $(1-d)\overline{P} = \mu P$. ■

**Convergence**

The tests we have been able to perform show that with the choice of $Z$ as the initial distribution, there is no significant difference between the convergence of the $\mu$-compensation algorithm and the non-compensated one. In both cases, convergence is very fast during the first iterations, and then stabilizes toward a convergence with ratio $d$ (cf main loop of Figure 5.1).

[19]Note in passing that in the case where $A$ is stochastic, we have here a very simple proof of convergence with ratio $d$, but Theorem 5.2 nonetheless provides more information…

**Iteration speed**

The $\mu$-compensation must compute the parameter $\mu$ at each iteration, while the non-compensated version does not. Does this have a significant influence on performance?

- If the matrix $A$ does not fit in memory, the limiting factor in computing an iteration is the multiplication of $A^t$ by $P_n$. The time used for compensation is then negligible, and the iteration speed has no influence on the choice between the two algorithms.
- On the other hand, if $A$, or even simply the adjacency matrix, fits in memory, the computation and incorporation of $\mu$ takes a duration comparable to that of computing $A^t P$. In fact, any norm computation adds a non-negligible overhead to the computation of an iteration, and even the computation of the convergence parameter $\delta$ considerably reduces performance. This leads to the SpeedRank algorithm (5.4), which computes the non-compensated PageRank by roughly estimating the number of iterations needed for good convergence. Our experiments revealed a speed gain of more than 300% on iterations between SpeedRank and Algorithm 5.3, which more than compensates for the slight overestimation of the number of iterations needed to converge.

In terms of performance, the $\mu$-compensated algorithm is therefore to be avoided when working on "small" graphs. One may be surprised in passing that Kamvar *et al.* use $\mu$-compensation in their BlockRank algorithm [Kam+03b], which is precisely based on the decomposition of PageRank on small graphs. For specialists in PageRank computation optimization (cf [Kam+03a]), it is curious to overlook a speed gain of 300%…

**Leaf-stripping and re-pluming**

In practice, PageRank algorithms are rarely applied to the entire graph $G = (V, E)$. A technique known as "leaf-stripping and re-pluming" is very often used: the PageRank is first computed on the graph $(R, E_R)$, where $R$ is the set of vertices of $V$ having at least one outgoing link. $(R, E_R)$ is called the stripped graph, or rake. Once good convergence is achieved, "re-pluming" is performed: we return to the graph $G$ and carry out a few PageRank iterations with the PageRank on $R$ as the initial estimate.

The problem is that the PageRank on the stripped graph is not necessarily a good estimate of the PageRank on $G$, as shown in Figure 5.1: because of re-pluming, the convergence factor $\delta$ is virtually reset. If one wants to once again reach the convergence condition ($\delta < \varepsilon$), almost as many iterations are needed as starting from the distribution $Z$.

The leaf-stripping and re-pluming method is nonetheless useful if one limits the number of iterations in the re-pluming phase: since the main loop operates on the rake, the iterations are faster. As for the final vector, although it is not a stationary vector, it lies halfway between the PageRank on the rake and that on $G$, and the fact that this vector is used in practice seems to indicate that the ranking it produces is worthy of interest.

**Data**

- The adjacency matrix $M$ of an arbitrary graph $G = ((R \cup S), E)$, $R$ being the set of vertices with non-zero out-degree;
- a covering probability distribution $Z$;
- a *zap* coefficient $d \in ]0, 1[$;
- a real number $\varepsilon$.

**Result**

With a precision of at least $\varepsilon$, a vector $P$ homogeneous to the $\mu$-compensated PageRank on $G$.

**begin**

$P \leftarrow Z$, $Y \leftarrow (1-d)Z$

$C : v \rightarrow \begin{cases} \frac{d}{d(v)} \text{ if } v \in R \\ 0 \text{ otherwise} \end{cases}$

**for** $i$ ranging from 1 to $\frac{\ln(\varepsilon)}{\ln(d)}$ :

  $P \leftarrow M^t(C \otimes P) + Y$ ▷ $\otimes$: element-wise product

**return** $P$

Algorithm 5.4: SpeedRank: fast PageRank computation when the adjacency matrix fits in main memory

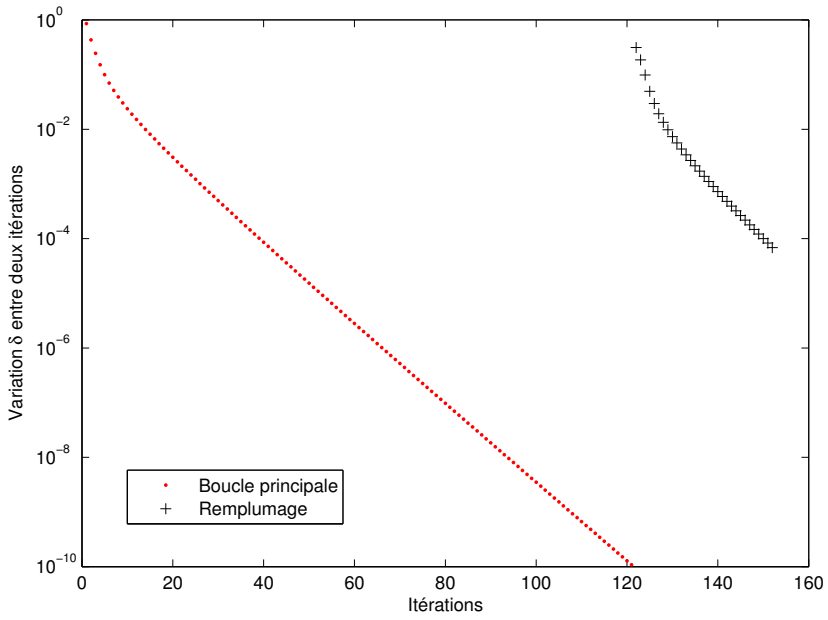



Figure 5.1: Convergence of $\mu$-compensated and non-compensated PageRanks; the re-pluming problem

## 5.5 Convergence in 1-norm and convergence of the ranking

In all the PageRank algorithms we have presented, as in all those we are about to present, we use as a convergence criterion the convergence in 1-norm of a sequence $P_n$ of positive vectors. Thus, when a rank source associated with a *zap* factor $d$ is used and the convergence criterion $\varepsilon$ is met, we know that the error with respect to the limit vector is at most $\frac{\varepsilon}{1-d}$. However, only the ranking induced by $P$ is of *a priori* interest, since the main purpose of PageRank is to provide an importance ordering on the Web pages under consideration[20].

### 5.5.1 Normalized Kendall distance

A first solution is to compare at each iteration the induced rankings, and to stop when there is no more change. One can also define a distance on rankings and replace convergence in 1-norm with convergence on rankings. A fairly classical distance on rankings is the symmetric difference distance, or Kendall distance[21]: if $\sigma_1$ and $\sigma_2$ are two rankings, presented as permutations, then the Kendall distance between these two permutations is the minimum number of inversions of two adjacent elements needed to go from one to the other. It can be shown that this distance is translation-invariant and that the distance from a permutation $\sigma$ to the identity is:

$$\mathrm{dist}(\sigma, \mathrm{id}) = \sum_{i<j} \chi_{\sigma(i)>\sigma(j)} \tag{5.21}$$

The distance between two permutations $\sigma_1$ and $\sigma_2$ is therefore $\mathrm{dist}(\sigma_1, \sigma_2) = \mathrm{dist}(\sigma_1 \circ \sigma_2^{-1}, Id)$.

Since the greatest possible distance $\mathrm{dist}_{\max}$ between two permutations of size $n$ is that between two reversed rankings, namely $\frac{n(n-1)}{2}$, one may, if one wants a convergence criterion independent of the size of the ranking, consider the Kendall distance normalized by $\mathrm{dist}_{\max}$.

### 5.5.2 PageRank density

By a simple order-of-magnitude argument, it is possible to establish a link between $\varepsilon$ and the convergence of the ranking. The starting point is the study of the relationship between a page's rank and its PageRank. Figure 5.2 shows this relationship for two PageRank models that we shall study in more detail: the standard $\mu$-compensated

[20] In reality, things are slightly different. The ranking returned for a given query is presumably the result of combining several importance estimates, with relevance and PageRank being the main ones. Knowing the quantitative PageRank of pages can then be of interest.

[21] Thanks to François Durand and his master's thesis report [Dur04] for introducing me to the Kendall distance.

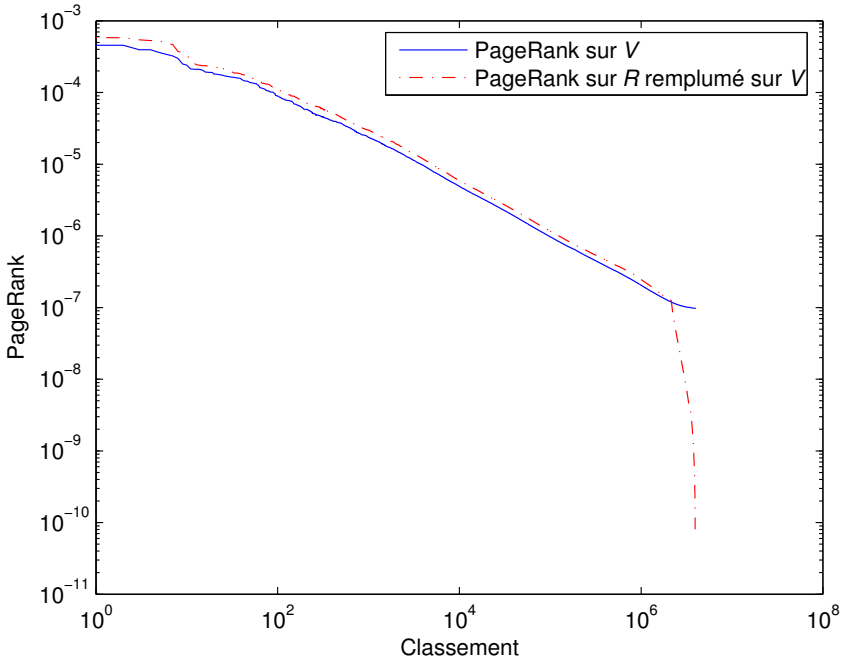



Figure 5.2: Relationship between a page's rank and its PageRank

PageRank with uniform *zap* over $V$, and the $\mu$-compensated PageRank with leaf-stripping/re-pluming technique and uniform *zap* over $R$. The *zap* factor $d$ is of course $0,85$.

The regularity of the curves[22] leads us to consider the mesoscopic density of pages at a given PageRank: we seek to know what is the number $dN$ of pages whose PageRank lies between $p$ and $p + dp$. We place ourselves at the mesoscopic scale, that is, we assume $dp \ll p$ and $dN \gg 1$. Experimentally, we found that the mesoscopic hypothesis was entirely realistic on graphs with more than one million vertices. We also observed that there exists, for each PageRank model, a function $\rho$, relatively independent of the Web graph under consideration[23], such that, if $n$ is the number of pages in the graph,

$$\frac{dN}{dp} \approx n^2 \rho(np) \tag{5.22}$$

$\rho$ is the normalized mesoscopic density (independent of the graph size $n$) typical of the PageRank model under consideration. Figure 5.3 shows experimental measurements of $\rho$ corresponding to the two models studied here.

[22]For each of the two PageRanks studied here, we have plotted only a single curve, but experimentally, the other samples studied generate extremely similar curves.

[23]In the case of PageRank with leaf-stripping/re-pluming, this holds for a given proportion of dangling pages. Empirically, this constant is often a crawl invariant.

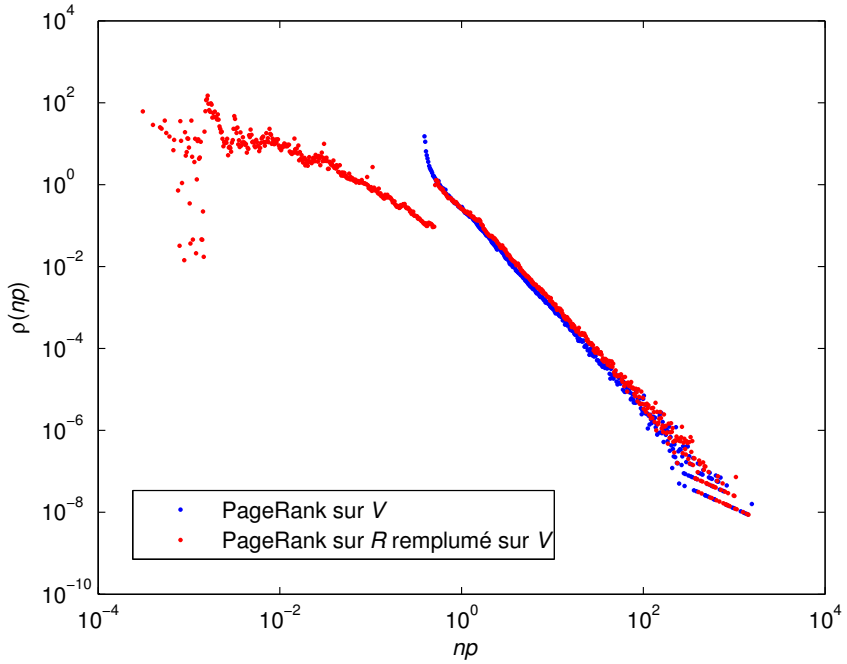



Figure 5.3: Normalized mesoscopic page density as a function of PageRank

## 5.6 Models with virtual page

Adding a *zap* factor is sometimes called the maximal irreducibility method, in the sense that if $\chi_Z = V$, then the underlying graph becomes a clique. This method is often criticized for being too intrusive and distorting the structure of the Web graph. The completion method, by adding $|\chi_Z| \cdot |\chi_s|$ fictitious links, can also be considered intrusive.

An alternative is to employ so-called minimal irreducibility methods[24], that is, to add a virtual page that will play the role of a "hub." This type of method is used among others by [APC03, Tom03].

### 5.6.1 Virtual *zap* page

The principle of the virtual *zap* page is simple: add an $(n+1)^{\text{th}}$ page, pointed to by and pointing to all other pages, and controlled by $d$ and $Z$.

Formally, if $A$ is a stochastic matrix (possibly completed), one considers the matrix

$$\tilde{A} = \begin{pmatrix} dA & (1-d)\mathbf{1}^t \\ Z^t & 0 \end{pmatrix} \tag{5.23}$$

The corresponding asymptotic probability vector is obtained by Algorithm 5.5.

[24] Cf [Tom03].

**Data**

- a stochastic matrix $A$;
- a covering *zap* distribution $Z$;
- a *zap* coefficient $d \in ]0,1[$;
- a real number $\varepsilon$.

**Result**

The probability eigenvector $(P, v)$ of $\tilde{A}^t$ associated with the maximal eigenvalue.

**begin**

Choose a positive vector $P_0$ and a positive real number $v_0$

$n \leftarrow 0$, $\delta \leftarrow 2\varepsilon$

**while** $\delta > \varepsilon$ :

$$P_{n+1} = d \cdot A^t P_n + v_n \cdot Z$$
$$v_{n+1} = (1-d)\|P_n\|_1$$
$$\delta = \|P_n - P_{n+1}\|_1$$
$$n \leftarrow n+1$$

**return** $P_n$

Algorithm 5.5: PageRank: model with virtual page

### 5.6.2 On the usefulness of the virtual page

Theorem 5.5 shows that *a priori*, as soon as $d > 0,5$, the use of a virtual *zap* page changes strictly nothing compared to adding a *zap* factor, both in terms of the asymptotic vector[25] and of convergence (the eigenvalues other than 1 are less than $\max(d, 1-d)$ in absolute value).

**Theorem 5.5**: Let $A$ be a stochastic matrix. Since $\tilde{A}$ is stochastic, irreducible, and aperiodic, we know that 1 is a singular and dominant eigenvalue. More precisely, if $(\lambda, (P, v))$ satisfies $\tilde{A}^t(P, v) = \lambda(P, v)$, then one of the following 3 cases holds:

- Either $\lambda = 1$, and $(P, v)$ is then homogeneous to $\left(\hat{P}, (1-d)\right)$, where $\hat{P}$ is the probability distribution such that $\hat{P} = \hat{A}^t\hat{P}$.
- Or $\lambda = d - 1$.
- Or $\lambda$ is neither 1 nor $d-1$. $P$ is then an eigenvector of $A^t$, $v$ is zero, and we have $\tilde{A}^t(P, 0) = (dA^tP, 0)$. In particular, $|\lambda| \leq d$.

[25] Indeed, up to a weight of $(1-d)$ on the virtual page, the maximal eigenvectors are identical.

## 5.7 Managing physical resources

Before closing this chapter, I would like to highlight some fundamental technical considerations. The reader may have noticed that none of the presented algorithms ever explicitly involves the matrices whose maximal eigenvector is being computed ($\overline{A}$, $\hat{A}$, $\widehat{\overline{A}}$, and $\tilde{A}$). This is a general phenomenon: if one wants all the algorithm's constants to fit in main memory[26], a minimum of thought is required. Indeed, one must remember that $A$ is a sparse matrix containing $n \cdot \overline{d}$ non-zero elements, where $\overline{d}$ is the average degree. $\overline{d}$ being relatively constant (between 7 and 11 depending on whether unvisited pages are taken into account and whether link filtering is applied), this amounts to approximately linear size in $n$, which is both a lot given the sizes of the graphs under consideration, and a minimum if one wants to use the structure of the Web. The implicit matrices generated by completion and use of the *zap* factor are far from sparse, and their size can be $n^2$. We see here the value of using rewriting tricks in PageRank algorithms so as to never involve a matrix less sparse than $A$.

Personally, I perform my PageRank experiments with simply the adjacency matrix, stored as a logical sparse matrix (*logical sparse matrix*) and a few vectors of size $n$: $D : i \to d(i)$, $Z$, $s$, $T$… Algorithms Algorithm 5.4 and Algorithm 5.6 give examples of the effective treatment of the algorithms. It is thus possible to handle up to 8 million vertices on a home PC with 1 GB of main memory, while keeping all constants in memory and performing a minimum of operations[27]. To go further, one must use transparent graph compression techniques[28].

[26]It is possible, even necessary for very large graphs, to perform PageRank computations without loading all constants into main memory and with optimized disk accesses [APC03], but seeking to minimize the size of constants remains very important when one wants to implement a PageRank algorithm.

[27]This may seem modest when one knows that one GB of main memory can store the PageRank of 125 million pages in double-precision floating point. However, one should always remember the colossal gain obtained when the adjacency matrix is in main memory.

[28]See for instance the Webgraph project [BV].

**Data**

- an adjacency matrix $M$ of an arbitrary graph;
- a covering probability distribution $Z$;
- a *zap* coefficient $d \in ]0, 1[$;
- a real number $\varepsilon$.

**Result**

The probability eigenvector $P$ of $\hat{\underline{A}}^t$ associated with the maximal eigenvalue.

**begin**

$n \leftarrow 0$, $P_n \leftarrow Z$, $\delta \leftarrow 2\varepsilon$

$T \leftarrow \frac{1}{M\mathbf{1}_n^t + \mathbf{1}_n^t}$ ▷ $1/\cdot$: element-wise inverse

**while** $\delta > \varepsilon$ :

> $P_{n+1} \leftarrow d \cdot M^t(T \otimes P_n)$ ▷ $\otimes$: element-wise product
>
> $\mu \leftarrow \|P_n\|_1 - \|P_{n+1}\|_1$
>
> $P_{n+1} \leftarrow P_{n+1} + \mu Z$
>
> $\delta \leftarrow \|P_n - P_{n+1}\|_1$
>
> $n \leftarrow n + 1$

**return** $P_n$

Algorithm 5.6: Hybrid algorithm: virtual completion page and $\mu$-compensation

# Chapter 6
# BackRank

– *You've got to come back with me!*
– *Where?*
– *Back to the future.*

Back to the Future

*True novelty always arises from a return to the sources.*

*Edgar* Morin, Amour, poésie, sagesse

We have just seen the general theoretical principles governing the various PageRank algorithms. In particular, it has become apparent that the problem of stochastic completion and that of PageRank diffusion can, and indeed should, be treated independently. In this chapter, which is an extension of a series of articles co-authored with Mohamed Bouklit [BM03, MB04], we shall study a way of performing stochastic completion while refining the random surfer model: taking into account the user's ability to go back at any point during navigation using the *Back* button of their browser[1]. As we shall see, the PageRank algorithm resulting from this model has numerous advantages over classical PageRank variants.

## 6.1 On the importance of the *Back* button

In 1995, Catledge and Pitkow published a study showing that the *Back* button accounted for 41% of all recorded navigation actions (52% of actions being hyperlink usage) [CP95]. Another study, by Tauscher and Greenberg, dating from 1997, reported 50% hyperlinks and 30% *Back* [TG97]. Finally, let us cite a more recent study (2004) by Milic *et al.* [Mil+04], whose main results are presented in Table 6.1.

From all these studies, a certain decline in the use of the *Back* button between 1995 and 2004 seems to emerge, but according to [CM01], one must take into account

[1] I apologize in advance to purists, but I confess to preferring the term *Back* over more verbose alternatives.

| **Navigation mode** | **%** | **Navigation mode** | **%** |
|---|---|---|---|
| Hyperlinks | 42,5% | *Bookmarks* | 2,9% |
| *Back* button | 22,7% | Manually typed URL | 1,6% |
| New session | 11,2% | Start page | 1,1% |
| Other methods[2] | 10,7% | *Refresh* button | 0,5% |
| Forms | 6,6% | *Forward* button | 0,2% |

Table 6.1: Statistical study of real surfers' navigation modes (after [Mil+04])

the changes that occurred during this interval[3], as well as changes in the experimental protocols used.

The other interesting piece of information these figures give us is that in all cases, the use of the *Back* button comes very clearly in second place, behind the use of hyperlinks which remains the preferred navigation mode. In particular, one should note that according to all studies, for every 2 clicks on a hyperlink, there is at least 1 use of the *Back* button. Yet, the use of the *Back* button has no equivalent in "classical" PageRank(s), whereas quantitatively less significant navigation modes such as manually typed URLs and *Bookmarks* can be taken into account in the computation algorithms through the *zap* vector.

Even though it is not really known to date whether a model closer to the real surfer necessarily yields a better PageRank, the importance of the *Back* button in real users' behavior seemed to us a sufficient motivation to study the possibilities of incorporating it into a new PageRank model.

## 6.2 Previous and contemporary work

Kleinberg, in the HITS algorithm [Kle98], also uses the inlink matrix, and thus works on a model where following hyperlinks backward is implicitly taken into account. Nevertheless, the purpose of the HITS algorithm is not to model the *Back* button but to view the Web graph as a superposition of "hubs" and "authorities," the former pointing to the latter.

On the other hand, Fagin *et al.*, in [Fag+00] propose a very comprehensive study of the influence of introducing backward steps in a random walk, and propose a stochastic model of the random surfer that takes the *Back* button into account. In this

[2]Dynamic loading, automatic redirections, address auto-completion…

[3]On the scale of the Web, if 9 million pages represent a twig, 9 years represent an eternity. During this interval, browser ergonomics evolved (*Bookmarks*, URL auto-completion, history browsing…), and user behavior was also modified by the omnipresence of search engines. Thus, rather than going back, some users prefer to re-enter in their search engine the query that led them to the page they came from.

model, an initial *classical* Markov chain is considered, to which an infinite-capacity history of visited pages is appended. To each page $v$ is associated a probability $\alpha(i)$ of going back, provided the history is non-empty. The main results obtained are:

- Depending on the back probabilities, the process can be transient (asymptotically, the starting page is "forgotten" and eventually ceases to influence the probability distribution) or ergodic (backward steps bring the surfer infinitely often back to the starting page), with a phase transition between the two.
- In the particular case where the back probability $\alpha$ does not depend on the page, if $\alpha < 0,5$, the process converges to the same distribution as the *classical* Markov chain (without the possibility of going back).
- The other cases are characterized, different types of convergence are considered, and the limit values (when they exist) are given.

The main drawback of Fagin *et al.*'s model is the prohibitive cost[4] of computing asymptotic distributions.

In [Syd04], Sydow proposes to reduce the complexity by limiting the history size. He thus introduces a model where only the last visited page is kept in memory (the *Back* button cannot be used twice in a row), and where the *Back* probability is uniform[5]. The resulting algorithm exhibits performance comparable to that of a standard PageRank, in terms of convergence speed and resource requirements. The ranking obtained from the asymptotic probability distribution remains similar to that of a standard PageRank, with significant differences in the ranking of dominant pages.

Compared to Sydow's algorithm, our approach to incorporating the *Back* button has the advantage of eliminating many problems associated with classical PageRank variants (stochastic completion, leaf trimming and restoration, convergence speed).

## 6.3 Modeling the *Back* button

Adopting an approach similar to [Syd04][6], we choose to introduce the possibility of going back with a limited history. Potentially, to reduce a stochastic process with memory $m$ to a memoryless Markov chain, one may need to consider the set of all paths of length $m$ in $G$. In the case where $m = 1$, which is the one we shall study, the canonical working space is the set $E$ of hyperlinks. We shall introduce two intuitive models of the *Back* button for the case where $m = 1$, and show that for one of them,

[4]With Web graphs, any algorithm whose complexity exceeds $n$, or at most $n \log n$, is considered prohibitive.

[5]Since the history is limited, the limit distribution is not necessarily equal to the distribution of the classical model, even for $\alpha < 0,5$.

[6]Unless it was Sydow who adopted an approach similar to ours, the research having been conducted independently, and each having discovered the other's work at the thirteenth *WWW* conference.

it is possible to reduce the working space from $E$ to $V$. We shall also see that this latter case is compatible with the use of a *zap* factor.

### 6.3.1 Reversible model

In this model, we assume that at each step, the user can either choose an outgoing link from the current page, or press the *Back* button, which will take them back to the previous step. The *Back* button is treated as a hyperlink like any other. In particular, the probability of using the *Back* button is the same as that of clicking on any given link, if at least 1 exists, and equals 1 in the absence of outgoing links. We believe this approach has two advantages over the uniform back probability chosen in [Syd04]:

- Intuitively, it seems logical that the use of the *Back* button should depend on the available choices on the current page, that is, on the number of outgoing links. This is partially confirmed by the behaviors observed in [Mil+04].
- Just like the virtual completion page introduced in Section 5.6, the *Back* button we have just modeled provides an escape even from pages without links, and creates a form of stochastic completion.

Let us finally note that the term *reversible* is due to the fact that two consecutive uses of the *Back* button cancel each other out.

Formally, the stochastic process we have just introduced can be described as follows: for each $v \in V$, we define $P_n(v)$ as the probability of being at $v$ at time $n$. We also introduce the term $P_n^{rb}(w, v)$, for $w, v \in V$, the probability of having moved from $w$ to $v$ between times $n-1$ and $n$. $P_n^{rb}(w, v)$ is nonzero whenever $(w, v)$ or $(v, w)$ is in $E$, and there is a very simple relationship between $P_n$ and $P_n^{rb}$:

$$P_n(v) = \sum_{w \leftrightarrow v} P_n^{rb}(w, v) \tag{6.1}$$

where $w \leftrightarrow v$ means $w$ *pointed to by or pointing to* $v$. One notes that the use of the *Back* button requires working on the undirected graph induced by $G$.

Similarly, it is possible to write an equation describing the terms $P_{n+1}^{rb}(w, v)$: if $(w, v) \in E$, to go from $w$ to $v$ between times $n$ and $n+1$, one can either follow the link from $w$ to $v$ (this requires being at $w$ at time $n$ and choosing among $d(w)+1$ possibilities), or use the *Back* button (this requires having gone from $v$ to $w$ between times $n-1$ and $n$, and choosing among $d(w)+1$ possibilities). If $(w, v) \notin E$, but $(v, w) \in E$, then the transition from $w$ to $v$ can only be performed *via* the *Back* button. There is no transition in other cases. We can therefore write:

$$P_{n+1}^{rb}(w, v) = \begin{cases} \frac{1}{d(w)+1}(P_n(w) + P_n^{rb}(v, w)) & \text{if } (w, v) \in E \\ \frac{P_n^{rb}(v,w)}{d(w)+1} & \text{if } (v, w) \in E \text{ and } (w, v) \notin E \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

Using Equation (6.1) and Equation (6.2), it is possible to carry out an iterative computation of $P_n$, which can for example be initiated with the distribution

$$P_0^{rb}(w,v) = \begin{cases} \frac{1}{|E|} \text{ if } (w,v) \in E \\ 0 \quad \text{otherwise.} \end{cases} \tag{6.3}$$

If $G' = (V, E')$ is the undirected graph induced by $G$, it thus seems necessary to use $|V| + |E'|$ variables, compared to $|V|$ for standard PageRank. By substituting Equation (6.1) into Equation (6.2), it is possible to reduce the number of variables to $|E'|$, but this remains very large. This led us to consider the *irreversible Back* model[7].

### 6.3.2 Irreversible model

In this new model, we assume that it is not possible to use the *Back* button twice in a row: it is grayed out after use, and one must first follow at least one real link before being able to use it again. Although more complex in appearance, this model has definite advantages over the reversible model:

- It is closer to the behavior of the *Back* button in real browsers, which indeed becomes disabled when the history is exhausted. The use of the *Back* button after history exhaustion in the reversible model is more akin to the use of the *Forward* button in real browsers. And, if we are to believe [Mil+04] (see Table 6.1), the role of the *Forward* button remains relatively negligible.
- It is compatible with the effective introduction of a *zap* factor (see Section 6.3.3).
- Computing the asymptotic distribution, even with a *zap* factor, requires no more resources than in the case of a classical PageRank (see Section 6.4.1).
- The use of the *Back* button inevitably introduces a kind of "greenhouse effect" at dead-end pages, which can be problematic (an effect similar to that of recurrent strongly connected components described in the previous chapter). The irreversible model reduces this effect compared to the reversible model. Consider the example of Figure 6.1: a page 1 has 2 links, one to a dead-end page 2, the other to an escape page from which the entire graph is reachable. If the random surfer ends up at 2, they will necessarily return to 1 via the *Back* button. Going back to 2 then creates the beginning of a greenhouse effect, and the return to 2 occurs with probability $\frac{2}{3}$ in the reversible model (2 favorable outcomes out of 3), compared to $\frac{1}{2}$ in the irreversible case: the probability of remaining "stuck" at 2 is lower in the irreversible model.

In order to compute the evolution in such a model, we shall introduce:

$\boldsymbol{P_n^{hl}(w,v)}$ probability of going from page $w$ to page $v$ between times $n-1$ and $n$ using a hyperlink.

$\boldsymbol{P_n^{ib}(v)}$ probability of having arrived at page $v$ at time $n$ by using the *Back* button.

[7] It may be possible to reduce the number of variables to $|V|$, as we shall do with the irreversible model, but we have not yet formalized this reduction, and therefore leave the reversible *Back* model at a purely descriptive stage.

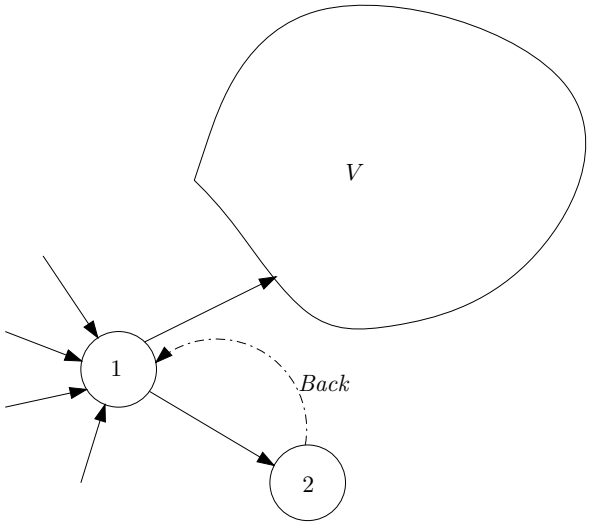



Figure 6.1: *Back* button and greenhouse effect: role of (ir)reversibility

It is possible to describe $P^{hl}_{n+1}(w,v)$ as a function of the situation at time $n$: to follow the link from $w$ to $v$, either one arrived at $w$ by following a real link, and must choose among $d(w)+1$ possibilities, or one arrived at $w$ via the *Back* button, which grayed it out and limits the possibilities to $d(w)$. We thus have, for $(w,v) \in E$,

$$P^{hl}_{n+1}(w,v) = \frac{\sum_{u \to w} P^{hl}_n(u,w)}{d(w)+1} + \frac{P^{ib}_n(w)}{d(w)} \tag{6.4}$$

One will note that since $w \to v$, there is no ambiguity in dividing by $d(w)$. One will also note that the destination vertex $v$ does not appear in any way in the expression for $P^{hl}_{n+1}(w,v)$. If $R$ is the set of pages with at least one link, and $S$ the set of dead-end pages, we shall therefore henceforth speak of $P^{hl}_n(w)$, with $w \in R$, rather than $P^{hl}_{n+1}(w,v)$, with $(w,v) \in E$.

Similarly, to have arrived at a page $v$ via the *Back* button, one must previously have gone from $v$ to a page $w$ pointed to by $v$ using a hyperlink, then have chosen the *Back* button among the $d(w)+1$ possibilities. In particular, one cannot have arrived at a page in $S$ via the *Back* button, and $P^{ib}_n(v)$ is zero for $v \in S$. For $v \in R$, we can write:

$$P^{ib}_{n+1}(v) = \sum_{w \leftarrow v} \frac{P^{hl}_n(v)}{d(w)+1} \tag{6.5}$$

where $w \leftarrow v$ means $w$ *pointed to by* $v$.

If we define the *Back*-attractiveness of a vertex $v$ belonging to $R$ by

$$a(v) = \sum_{w \leftarrow v} \frac{1}{d(w)+1} \tag{6.6}$$

it is then possible to rewrite Equation (6.5) and Equation (6.4) as follows:

$$P^{ib}_{n+1}(v) = a(v) P^{hl}_n(v) \tag{6.7}$$

$$P_{n+1}^{hl}(v) = \frac{1}{d(v)+1} \sum_{w \to v} P_n^{hl}(w) + \frac{P_n^{ib}(v)}{d(v)} \tag{6.8}$$

By combining Equation (6.7) and Equation (6.8), we obtain Equation (6.9)

$$P_{n+1}^{hl}(v) = \frac{1}{d(v)+1} \sum_{w \to v} P_n^{hl}(w) + \frac{a(v)}{d(v)} P_{n-1}^{hl}(v) \tag{6.9}$$

And what about PageRank in all this? The probability of being at a page $v$ is the sum of the probability of having arrived there via the *Back* button and that of having arrived there by following a link. By setting, by convention, $P_n^{hl}(v) = 0$ for $v \in S$, we have:

$$\begin{aligned} P_n(v) &= P_n^{ib}(v) + \sum_{w \to v} P_n^{hl}(w) \\ &= a(v) P_{n-1}^{hl}(v) + \sum_{w \to v} P_n^{hl}(w) \end{aligned} \tag{6.10}$$

### 6.3.3 Incorporating the *zap* factor

Adding the irreversible *Back* button guarantees a stochastic process regardless of the initial graph, with the sole condition that the support of the initial distribution contains no dead-end page. The problem of short-distance dead ends, such as page 2 in Figure 6.1, is solved, but irreducibility problems remain. Worse still, the *Back* button transforms medium- and long-distance dead ends into rank sinks (see Figure 6.2).

It is therefore necessary to introduce *zap* into the process. We have chosen a classical *zap*, with a factor $d$ and a distribution $Z$. We require that "zapping" deactivates the *Back* button: after a *zap*, it is not possible to go back. This has the practical advantage of not having to consider all the transitions implicitly generated

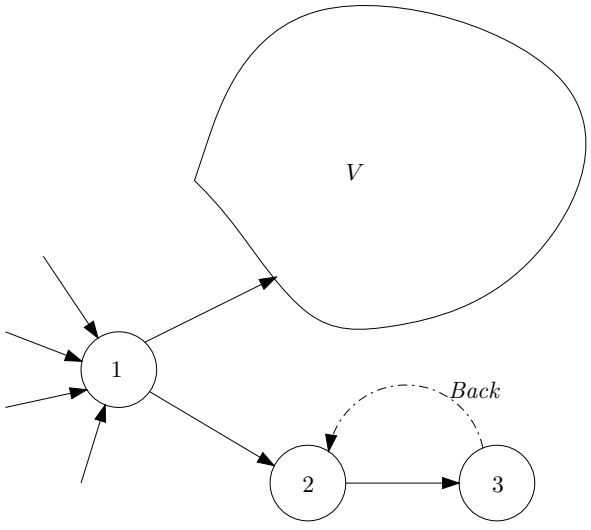



Figure 6.2: Recurrent strongly connected component created by the *Back* button

by the *zap*, namely $(V \times \chi_Z) \subset E$. This choice can also be interpreted at the surfer modeling level: according to Table 6.1, many real *zaps* actually correspond to starting a new session, and therefore to a history reset to 0.

Now that the framework is defined, it is possible to rewrite $P_n^{hl}(v)$, for $v \in R$:

$$P_{n+1}^{hl}(v) = d\left(\frac{1}{d(v)+1}\sum_{w\to v} P_n^{hl}(w) + \frac{P_n^{ib}(v)}{d(v)}\right) \tag{6.11}$$

Similarly, we can rewrite the probability $P_n^{ib}(v)$ of being at time $n$ on a page $v$ with an empty history. By setting, by convention, $P_n^{hl}(v) = 0$ for $v \in S$, we have:

$$P_{n+1}^{ib}(v) = d \cdot a(v) P_n^{hl}(v) + (1-d)Z(v) \tag{6.12}$$

And here, a problem arises: what happens if $\chi_Z \cap S \neq \emptyset$? We will have a nonzero probability of being on a page in $S$ with no possibility of going back. In this situation, with probability $(1-d)$, our surfer will "zap" elsewhere, and with probability $d$... they will not know what to do!

We consider two methods to work around this problem:

- Choose $Z$ such that $\chi_Z \cap S \neq \emptyset$. Since the only condition required of a *zap* distribution is that it be covering, this is entirely possible. A distribution over home pages will suffice, provided no home page is a single dead-end page. One can also choose the uniform distribution over $R$ defined by

$$Z(v) = \begin{cases} \frac{1}{|R|} \text{ if } v \in R \\ 0 \quad \text{otherwise.} \end{cases} \tag{6.13}$$

- Complete the stochastic process on the fly. We do know the probability of not knowing what to do between times $n$ and $n+1$: $d\sum_{v\in S} P_n^{ib}(v)$. It then suffices to redistribute this probability, for example as *zap* according to $Z$. Equation Equation (6.12) then becomes:

$$P_{n+1}^{ib}(v) = d \cdot a(v) P_n^{hl}(v) + \left[1 - d + d\sum_{v\in S} P_n^{ib}(v)\right] Z(v) \tag{6.14}$$

## 6.4 Practical algorithm: BackRank

We have just defined a stochastic process, which thanks to the *zap* factor is irreducible and aperiodic. The Perron-Frobenius theorem therefore applies[8] and allows us to assert that successive iterations of Equation (6.12) and Equation (6.11) will converge

[8]Note in passing that we do not need to explicitly write the associated transition matrix, which is a square matrix of size $|V| + |E|$. It suffices to know that this matrix exists and that it implicitly governs our process.

to a unique fixed point (up to renormalization). The initial conditions can for example be a distribution according to $Z$ with an empty history ($P_0^{ib} = Z$ and $P_0^{hl} = 0$).

### 6.4.1 Optimization

We shall assume here that we have chosen $Z$ such that $Z(v) = 0$ if $v \in S$.

From Equation (6.12) and Equation (6.11), $P_{n+1}^{hl}(v)$ can be defined recursively, for $v \in R$, by:

$$P_{n+1}^{hl}(v) = \frac{d}{d(v)+1} \sum_{w \to v} P_n^{hl}(w) + \frac{d}{d(v)} \big( d \cdot a(v) P_{n-1}^{hl}(v) + (1-d) Z(v) \big) \quad (6.15)$$

Equation (6.15) is a two-term recurrence, which is known to converge to a fixed point. Since only the fixed point is of interest, we can replace $P_{n-1}^{hl}$ with $P_n^{hl}$ while maintaining convergence to the same fixed point (Gauss-Seidel method). We thus obtain Equation (6.16).

$$P_{n+1}^{hl}(v) = \frac{d}{d(v)+1} \sum_{w \to v} P_n^{hl}(w) + \frac{d}{d(v)} \big( d \cdot a(v) P_n^{hl}(v) + (1-d) Z(v) \big) \quad (6.16)$$

Note in passing that the iterations are performed over vertices with nonzero out-degree: there is an implicit leaf trimming, similar to what is done for standard PageRank (see Section 5.4.4.3).

After the vector $P_n^{hl}$ has converged to a vector $P^{hl}$, it remains to perform the "restoration" to obtain the asymptotic presence distribution $P$:

$$\begin{aligned} P(v) &= \sum_{w \to v} P^{hl}(w) + P^{ib}(v) \\ &= \sum_{w \to v} P^{hl}(w) + d \cdot a(v) \cdot P^{hl}(v) + (1-d) Z(v) \end{aligned} \quad (6.17)$$

We now have all the building blocks of *BackRank* (Algorithm 6.1). Note in passing that there is no need to perform renormalization, since there is convergence to a fixed point.

### 6.4.2 Results

Once an algorithm is completely defined, it must be put to the test. In order to make comparisons, we needed a baseline $X$, and we chose the $\mu$-compensated PageRank defined in Algorithm 5.3. It is indeed the simplest PageRank guaranteeing full stochastic control, apart from the uncompensated PageRank of course[9]. The leaf trimming and restoration technique was added for fairness toward BackRank, which performs it implicitly, and also for realism (this is the method that appears to be

[9]We kept the compensation to make comparisons simpler, since both competitors thus produce a probability distribution.

**Input**

- an adjacency matrix $M$ of an arbitrary graph $G = (V, E)$, with $V = (R \cup S)$, where $R$ is the set of pages with links, $S$ that of dead-end pages;
- a covering *zap* distribution $Z$ over $R$;
- a *zap* coefficient $d \in ]0, 1[$;
- a real number $\varepsilon$.

**Output**

The PageRank vector $P$ of the irreversible *Back* model with *zap*.

**begin**

$D \leftarrow M\mathbf{1}_n^t$

$T \leftarrow 1./(D + \mathbf{1}_n^t)$ ▷ $1/\cdot$ : element-wise inverse

$A \leftarrow M \cdot T$

$T \leftarrow T_R$

$D \leftarrow D_R$

$C \leftarrow 1./D$

$B \leftarrow d \cdot C \otimes A_R$ ▷ $\otimes$ : element-wise product

$Y \leftarrow d(1-d)C \otimes Z$

$N \leftarrow M_R$

$R_0 = dC \otimes Z$

$\delta \leftarrow 2\varepsilon$, $n \leftarrow 0$

**while** $\delta > \varepsilon$:

> $R_{n+1} \leftarrow T \otimes (N^t R_n) + B \otimes R_n$
>
> $R_{n+1} \leftarrow dR_{n+1}$
>
> $R_{n+1} \leftarrow R_{n+1} + Y$
>
> $\delta \leftarrow \|R_{n+1} - R_n\|_1$
>
> $n \leftarrow n + 1$

$R_n \leftarrow R_n$ over $V$ (zero on $S$)

**return** $P = M^t R_n + dA \otimes R_n + (1-d)Z$

Algorithm 6.1: BackRank: PageRank model with irreversible *Back* button

used in practice for *off-line* computations on very large graphs, and BackRank is designed to handle very large graphs). For both algorithms, we used $d = 0.85$ (unless stated otherwise) and $Z$ is the uniform distribution over the rake $R$.

[10]The algorithms run under Matlab.

The test was conducted on a crawl of 8 million URLs which, for technical reasons[10], is split into two samples of 4 million URLs.

**Convergence**

One of the first viability criteria for a PageRank-type algorithm is its convergence speed. Figure 6.3 compares, on a semi-logarithmic scale, the value of the convergence parameter $\delta$ after $n$ iterations. For BackRank, the condition $\delta < 10^{-10}$ is reached after 87 iterations in both samples, whereas for PageRank, it takes between 126 and 127 iterations. We measured the ratio of the observed geometric decrease at $10^{-10}$, and found 0.844 for standard PageRank (convergence is always less than $d$, but tends asymptotically toward $d$) compared to 0.809 for BackRank. We conjecture that this gap, which explains BackRank's performance, is due to the fact that unlike PageRank, BackRank implicitly uses a partial Gauss-Seidel method. It has been shown that using a Gauss-Seidel-type method considerably improves PageRank convergence [Ara+01].

We should clarify that the algorithms used are, up to rewriting[11], exactly those described in this work. In particular, the power methods employed are truly power methods. For a more complete study of BackRank's convergence performance, one would need to investigate how it behaves under the many convergence acceleration methods that exist [Ara+01, Hav99, Kam+03a].

To conclude this convergence study, let us note that on the samples considered, one iteration of BackRank takes on average 2 seconds compared to 3 seconds for PageRank. This difference is due to the fact that all computation constants, including the adjacency matrix, fit in memory. The $\mu$-compensation thus has a non-negligible cost within an iteration. In fact, even compared to an uncompensated PageRank with $\delta$ estimation, BackRank incurs only a small overhead, on the order of 5%.

**Restoration**

The finalization of the PageRank computation, which we call restoration, is a phase that is rather poorly described. According to [Pag+98], after convergence of PageRank on the graph restricted to vertices with nonzero out-degree, dead-end pages are added back to the process, *without affecting things significantly*[12]. But in order to assign an importance to these new pages, at least one PageRank iteration must be performed, which requires modifying most of the constants. In our experiment, we chose to perform four iterations on the full graph after convergence on the trimmed graph. Besides slower iterations (6 seconds, given that approximately 50% of the pages in the samples were dead-end pages), we observe, as in Section 5.4.4.3 page Section 5.4.4.3, the reinitialization of the $\delta$ parameter (see Figure 6.3): after four iterations, we are at the same level as after eight iterations of the main loop, that is, far from a stationary state…

[11] Algorithm 5.3 was rewritten following the model of Algorithm 5.5.

[12] "without affecting things significantly," *op. cit.*

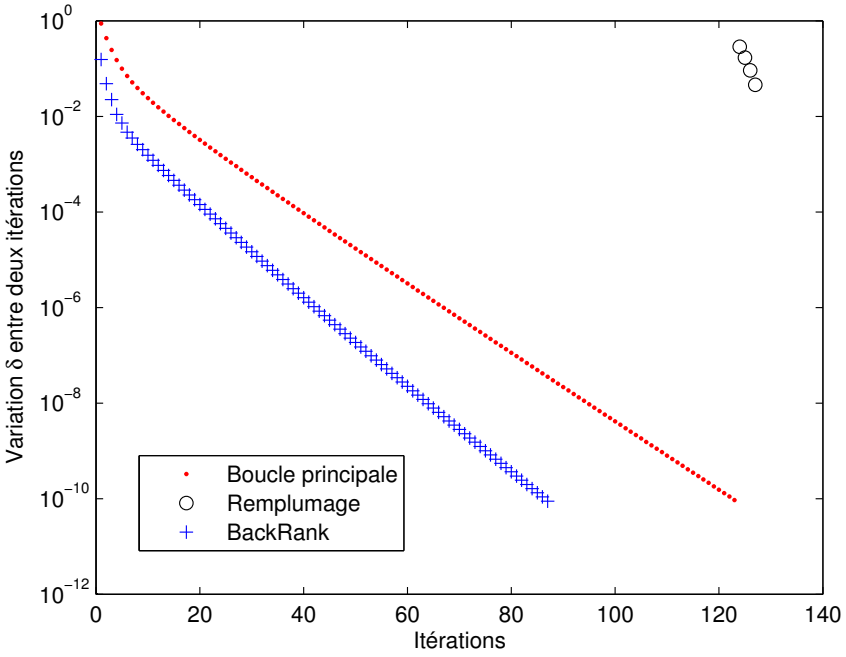



Figure 6.3: Compared convergence of BackRank and a standard PageRank

For BackRank, restoration is much less problematic, since it reduces to applying Equation (6.17) *once and only once*. There is therefore no need to start a new series of iterations. Let us nonetheless specify, for honesty's sake, that since $\delta$ refers to $P^{hl}$ and not $P$, the variations in $P$ can be larger than $\delta$. Experimentally, it indeed appears that when $\delta$ is around $10^{-10}$, the variations at the level of $P$ are on the order of $10^{-9}$ (slightly lower in fact). This result is nevertheless more than acceptable, especially in light of the variations of $10^{-1}$ generated by the PageRank restoration.

**Ranking**

Technical performance is only half of the evaluation criteria for a PageRank-type algorithm. One must also test the relevance of the importance ranking induced by the obtained probability distribution. A first approach consists of quantitatively comparing the results returned by BackRank and those returned by the reference PageRank. Figure 6.4 presents such a comparison, showing the percentage of pages common to the top $n$ pages returned by BackRank and PageRank respectively. We observe significant fluctuations among the very top pages. As the number of pages considered increases, the overlap stabilizes, reaching a relatively stable value of 0.845 when the number of pages considered approaches 1% of the sample size (40000 pages). This tends to prove that BackRank yields results fairly close to those returned by PageRank, with some specificities.

This 1% stabilized overlap of 0.845 for both samples intrigued us. Upon closer examination, we noticed that the 1% overlap rate is a variable that depends only on $d$ (at least on our samples). Figure 6.5 shows this relationship between $d$ and the 1% overlap for $0 \leq d \leq 1$. One will note in particular that:

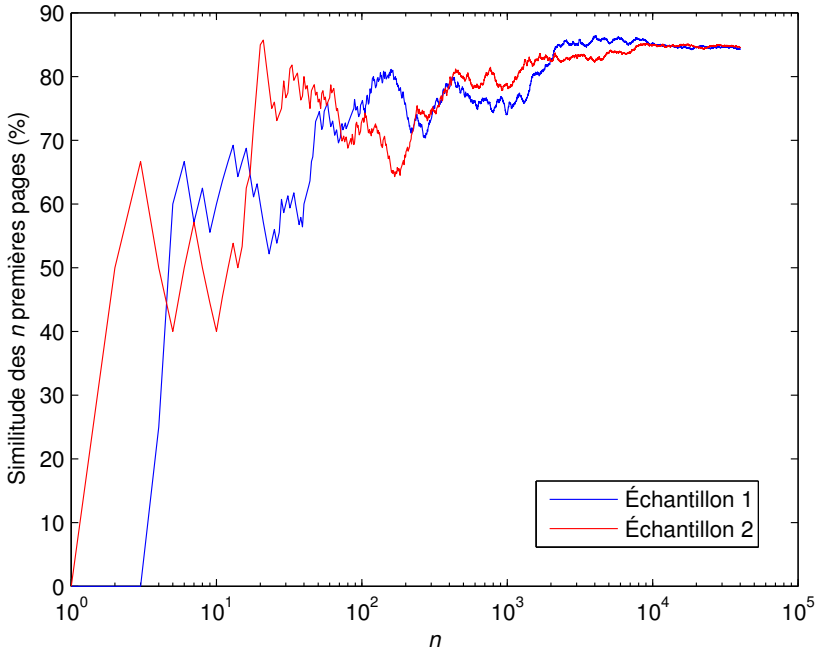



Figure 6.4: Overlap measurements between the top $n$ pages of BackRank and PageRank

- The overlap is a decreasing function of $d$. This echoes the idea, discussed in Section 5.3.5, of the smoothing role of the *zap* factor.
- In particular, the overlap tends toward 1 as $d$ tends toward 0, meaning that BackRank tends, just like PageRank, toward the in-degree distribution (see Section 5.3.5).
- The overlap tends toward 50% as $d$ tends toward 1, which seems to indicate that BackRank is intrinsically half different from (or half similar to?) standard PageRank. One should however not forget that when $d = 1$, the ranking is ossified by rank sinks and no longer necessarily meaningful. Upon verification of a few URLs, this 50% overlap indeed only means that BackRank does not get trapped in exactly the same sinks as PageRank.

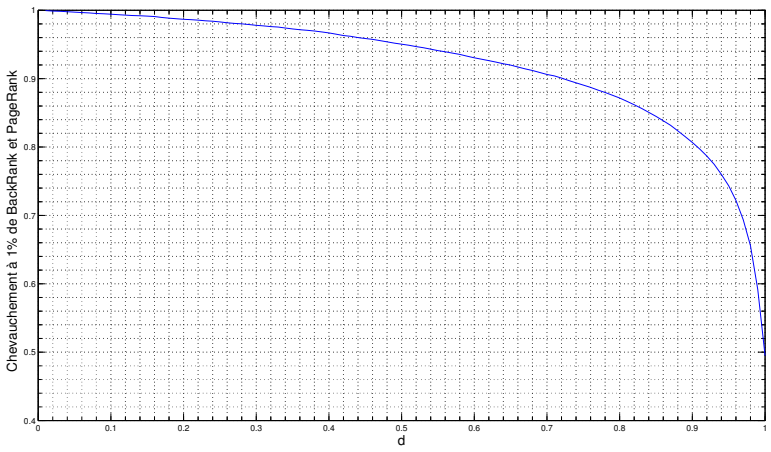



Figure 6.5: Influence of the *zap* factor on the 1% overlap

```
http://messagerie.business-village.fr:80/svc/jpro/search
http://server.moselle.cci.fr:80/Fichier/index.html
http://messagerie.business-village.fr:80/svc/jpro/aide
http://backstage.vitaminic.fr:80/add_artist.shtml
http://backstage.vitaminic.fr:80/
http://vosdroits.service-public.fr:80/Index/IndexA.html
http://emploi.cv.free.fr:80/index.htm
http://server.moselle.cci.fr:80/Fichier/listenaf.html
http://ec.grita.fr:80/isroot/fruitine/blank.html
http://www.adobe.fr:80/products/acrobat/readstep.html
```

Listing 6.1: Sample 1: the 10 most important pages according to BackRank

We are aware that we are only providing here a few indicators of the quality of BackRank's ranking. Strictly speaking, a complete validation would require incorporating BackRank into a search engine and conducting a series of satisfaction tests on a control population. In lieu of this, we shall use the reader as a control population, who can compare in lists Listing 6.1, Listing 6.2, Listing 6.3, and Listing 6.4 the top 10 pages returned by BackRank and PageRank on the two samples considered. One will note for example that the main page of CNRS is ranked by BackRank, but not by PageRank (in fact, it is $16^{\text{th}}$), whereas for the Ministry of Education, the opposite is true (page also ranked $16^{\text{th}}$, but by BackRank this time).

```
http://backstage.vitaminic.fr:80/
http://backstage.vitaminic.fr:80/add_artist.shtml
http://forums.grolier.fr:8002/assemblee/nonmembers/
http://server.moselle.cci.fr:80/Fichier/listenaf.html
http://server.moselle.cci.fr:80/Fichier/index.html
http://messagerie.business-village.fr:80/svc/jpro/search
http://www.adobe.fr:80/products/acrobat/readstep.html
http://mac-texier.ircam.fr:80/index.html
http://mac-texier.ircam.fr:80/mail.html
http://bioscience.igh.cnrs.fr:80//current/currissu.htm
```

Listing 6.2: Sample 1: the 10 most important pages according to PageRank

```
http://www.fcga.fr:80/
http://www.moselle.cci.fr:80/Fichier/index.html
http://www.lhotellerie.fr:80/Menu.htm
http://www.ima.uco.fr:80/
http://www.info-europe.fr:80/europe.web/document.dir/actu.dir/
http://www.moselle.cci.fr:80/Fichier/listenaf.html
http://www.machpro.fr:80/sofetec.htm
http://www.cnrs.fr:80/
http://www.quartet.fr:80/ce/
http://www.gaf.tm.fr:80/espace-pro.htm
```

Listing 6.3: Sample 2: the 10 most important pages according to BackRank

```
http://www.moselle.cci.fr:80/Fichier/index.html
http://www.moselle.cci.fr:80/Fichier/listenaf.html
http://www.lhotellerie.fr:80/Menu.htm
http://www.machpro.fr:80/sofetec.htm
http://www.education.gouv.fr:80/default.htm
http://www.infini.fr:80/cgi-bin/lwgate.cgi
http://www.proto.education.gouv.fr:80/cgi-bin/ELLIB/Lire/Q1
http://www.dma.utc.fr:80/~ldebraux/Genealogie/Whole_File_Report.html
http://www.ldlc.fr:80/
http://www.ldlc.fr:80/contact.shtml
```

Listing 6.4: Sample 2: the 10 most important pages according to PageRank

# Chapter 7
# Fine Decomposition of PageRank

*I shall compose until decomposition*

*Serge Gainsbourg*

*It's a psychofrakulator. It creates a cloud of radically fluctuating free-deviant chaotrons which penetrate the synaptic relays. It's concatenated with a synchronous transport switch that creates a virtual tributary. It's focused onto a biobolic reflector. And what happens is that hallucinations become reality and the brain is literally fried from within.*

Mystery Men

The purpose of this chapter, which is an extension of [MV03b] and [MV04], is to study how PageRank behaves with respect to the site structure presented in Part I, Section 3.3. We will show that there exists a natural decomposition of PageRank into two terms, internal PageRank and external PageRank. This decomposition provides a better understanding of the roles played by internal and external links. A first application is an algorithm for estimating the local PageRank within a website. We will also show some quantitative results on the possibilities available to a website for increasing its own PageRank.

More precisely, Section 7.1 briefly presents the various existing contributions regarding the use of the Web's site structure to compute PageRank. Section 7.2 specifies the hypotheses and conventions that will be used. The notions of internal and external PageRank will be introduced in Section 7.3, and applied to a theoretical decomposed PageRank algorithm in Section 7.4. Finally, after taking advantage in Section 7.5 of the study of local PageRank variations to introduce the *zap* factor into our model, we will give in Section 7.6 algorithms applicable to real-world graphs.

## 7.1 Previous and contemporaneous work

Among all published studies on PageRank, only a few attempt to take advantage of the site structure. [Kam+03b] gives a method for quickly computing a good approximation of PageRank using a partition into websites.

Bianchini *et al.* decompose PageRank into internal links, incoming links, outgoing links, and leaks [BGS02, BGS03]. This decomposition allows, among other things, a certain understanding of how a website can change its own PageRank, while providing stability results for PageRank in the face of changes in the internal link structure of a website.

Finally, Arasu *et al.* showed that a PageRank computation on the quotient graph over servers converged faster than on pages, and even faster when taking into account link multiplicity (see [Ara+01]).

Compared to these works, our approach consists of using, as in [BGS02, BGS03], an exact flow decomposition of PageRank, with more flexible assumptions on the *zap* distribution. From this decomposition, we derive an exact semi-distributed PageRank computation algorithm, which we hybridize with the algorithm proposed in [Kam+03b] in order to obtain a fast semi-distributed algorithm with few approximations.

## 7.2 Hypotheses

We have seen that a structural definition of a website could be a set of pages tightly interconnected by hyperlinks. The block structure of the adjacency matrix (see Figure 3.5, page 42 and Figure 3.6, page 43) allows us to hope for many methods to decompose a Web graph into websites, and Section 3.3, page 36 gives us one. For the remainder of this chapter, we will therefore assume that our Web graph is equipped with a partition that allows it to be decomposed into websites, denoted $\mathcal{S} = (S_1, ..., S_k)$, with $k > 1$.

As a first step, we will place ourselves in the ideal case where the Web graph $G = (V, E)$ under consideration is strongly connected and aperiodic. We will also assume the absence of self-loops.

As was seen in Section 5.3.1, if we consider the matrix $A$ defined by

$$A = \left(a_{v,w}\right)_{v,w \in V}, \quad \text{with } a_{v,w} = \begin{cases} \frac{1}{d(v)} & \text{if } v \to w \\ 0 & \text{otherwise.} \end{cases} \tag{7.1}$$

we know that there exists a unique probability distribution, denoted $P$, satisfying

$$P = A^t P \tag{7.2}$$

We will seek to highlight the connections between $P$ and $\mathcal{S}$.

## 7.3 Internal PageRank, external PageRank

### 7.3.1 Notation

For $v$ in $V$, we denote by $S(v)$ the element of $\mathcal{S}$ such that $v \in S(v)$. We also define $\delta_{\mathcal{S}} : V \times V \to \{0, 1\}$ as follows:

$$\delta_{\mathcal{S}}(v, w) = \begin{cases} 1 \text{ if } S(v) = S(w) \\ 0 \text{ otherwise.} \end{cases} \tag{7.3}$$

We will call $A_{\mathcal{S}}$ the restriction of $A$ to the elements internal to the components of $\mathcal{S}$:

$$A_{\mathcal{S}} = \left(a_{v,w}\delta_{\mathcal{S}}(v, w)\right)_{v,w \in V} \tag{7.4}$$

We also need to define the internal degree $d_i$ (resp. external degree $d_e$) of a vertex $v$ as its out-degree in the subgraph induced by $S(v)$ (resp. induced by $\{v\} \cup (V \setminus S(v))$).

We are now in a position to define the notions of internal and external PageRank, and to relate them to the PageRank $P$ defined by Equation (7.2).

- The **incoming internal PageRank** $P_{ei}$ (resp. **incoming external PageRank** $P_{ee}$) of a page $v$ in $V$ is the probability, when the random surfer is in steady state, of arriving at $v$ from a page in $S(v)$ (resp. from a page in $V \setminus S(v)$). From this definition, Equation (7.5) and Equation (7.6) are derived:

$$P_{ei} = A_{\mathcal{S}}^t P \tag{7.5}$$

$$P_{ee} = (A - A_{\mathcal{S}})^t P = P - P_{ei} \tag{7.6}$$

- The **outgoing internal PageRank** $P_{si}$ (resp. **outgoing external PageRank** $P_{se}$) of a page $v$ in $V$ is the probability, when the random surfer is in steady state, of going from $v$ to a page in $S(v)$ (resp. to a page in $V \setminus S(v)$). Equation (7.7) and Equation (7.8) formalize this definition:

$$P_{si} = (A_{\mathcal{S}} \cdot \mathbf{1}_n^t) \times P \tag{7.7}$$

$$P_{se} = ((A - A_{\mathcal{S}}) \cdot \mathbf{1}_n^t) \times P = P - P_{si} \tag{7.8}$$

### 7.3.2 Conservation laws

One can define a PageRank (possibly internal, outgoing, …) on a site $S$ as the sum of the PageRanks of its pages: $P(S) = \sum_{v \in S} P(v)$. With this convention, we can state the internal and external conservation laws of a site:

**Theorem 7.1**: Let $S$ be a site. The incoming external and outgoing external PageRanks of $S$ are equal:

$$P_{ee}(S) = P_{se}(S) \quad \text{(external conservation law)} \tag{7.9}$$

The same holds for the incoming internal and outgoing internal PageRanks:

$$P_{ei}(S) = P_{si}(S) \quad \text{(internal conservation law)} \tag{7.10}$$

*Proof*: Let us begin by proving the internal conservation law (Equation (7.10)):

$$\begin{aligned} P_{ei}(S) &= \sum_{v \in S} P_{ei}(v) = \sum_{v \in S} \sum_{w \to v, w \in S} \frac{P(w)}{d(w)} \\ &= \sum_{w \in S} \sum_{v \leftarrow w, v \in S} \frac{P(w)}{d(w)} = \sum_{w \in S} P_{si}(w) \\ &= P_{si}(S) \end{aligned} \tag{7.11}$$

Next, Equation (7.6) and Equation (7.8) allow us to write:

$$P = P_{ee} + P_{ei} = P_{se} + P_{si} \tag{7.12}$$

Equation (7.12) combined with the internal conservation law Equation (7.10) gives us the external conservation law:

$$P_{se}(S) = P_{ee}(S) + P_{ei}(S) - P_{si}(S) = P_{ee}(S) \tag{7.13}$$

■

The external conservation law Equation (7.9) shows us that a site returns, through the outgoing external PageRank, exactly the PageRank it receives (the incoming external PageRank). As Lavoisier said, *nothing is lost, nothing is created, everything is transformed*. If we consider the PageRank flow on the quotient graph $G/\mathcal{S}$, there is therefore conservation of flow (see Figure 7.1). This observation is the basis of the decomposed PageRank computation.

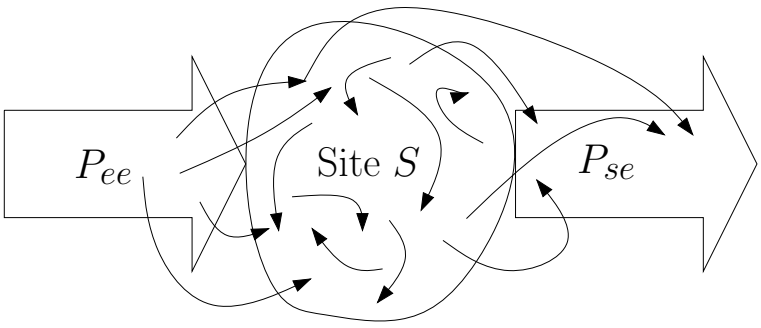



Figure 7.1: External PageRank conservation law: $P_{ee}(S) = P_{se}(S)$

**Remark 7.1**: Another way, perhaps simpler, of proving Theorem 7.1 would have been to consider PageRank directly as a stationary flow. It is then obvious that the flow on any subset $S$ of $V$ is also stationary. We preferred the matrix approach because it is the one we will continue to use hereafter, even though we will always try to interpret the results in terms of flow whenever possible.

## 7.4 Decomposition of the PageRank computation

### 7.4.1 Relationship between external PageRank and PageRank

Starting from Equation (7.5) and Equation (7.6), we can write that $A_{\mathcal{S}}^t \cdot P = P - P_{ee}$, and therefore that $P_{ee} = (\mathrm{Id} - A_{\mathcal{S}}^t)P$, where Id is the identity matrix.

**Lemma 7.1**: The matrix $(\mathrm{Id} - A_{\mathcal{S}}^t)$ is invertible.

*Proof*: It suffices to show that $A_{\mathcal{S}}$ is sub-irreducible. This will prove that its spectral radius is strictly less than 1 (Theorem 4.5, Remark 4.3), and therefore that $(\mathrm{Id} - A_{\mathcal{S}}^t)$ is invertible.

We proceed by contradiction: if $A_{\mathcal{S}}$ is not sub-irreducible, there exists at least one stochastic strongly connected component in the transition graph associated with $A_{\mathcal{S}}$. This component is necessarily internal to a site since there are no external links. It therefore also exists in $A$, which can then only be irreducible if the component is all of $V$, which is impossible (we assumed that $A$ was irreducible and that $\mathcal{S}$ contained at least two sites). ■

Lemma Lemma 7.1 then allows us to express $P$ as a function of the incoming external PageRank $P_{ee}$:

$$P = (\mathrm{Id} - A_{\mathcal{S}}^t)^{-1} P_{ee} \tag{7.14}$$

To compute the PageRank of a site $S$, it therefore suffices to know its internal structure, through the matrix $A_S$, and the incoming external PageRank it receives from the others.

**Remark 7.2**: The matrix $(\mathrm{Id} - A_{\mathcal{S}}^t)^{-1} = \sum_{k=0}^{\infty} (A_{\mathcal{S}}^t)^k$, which like $A_{\mathcal{S}}$ is a block diagonal matrix, can be interpreted as the transition matrix of all possible internal paths. Indeed, for $v, w \in V$, $(A_{\mathcal{S}})^k_{v,w}$ represents the probability of going from $v$ to $w$ via a path of length $k$ that follows only internal links (in particular, $(A_{\mathcal{S}})^k_{v,w} = 0$ if $S(v) \neq S(w)$.)

### 7.4.2 Transition matrix of external PageRank

We want to formalize the intuition of a site-to-site PageRank propagation given by the external PageRank conservation law (see Figure 7.1), and find a description of the relationships between the different components of $P_{ee}$. By combining Equation (7.6) and Equation (7.14), we obtain:

$$P_{ee} = (A - A_{\mathcal{S}})^t P = (A - A_{\mathcal{S}})^t (\mathrm{Id} - A_{\mathcal{S}}^t)^{-1} P_{ee} \tag{7.15}$$

We can thus define the transition matrix of external PageRank:

$$A_e^t = (A - A_{\mathcal{S}})^t (\mathrm{Id} - A_{\mathcal{S}}^t)^{-1} \tag{7.16}$$

This matrix possesses some very interesting properties:

**Lemma 7.2**: The matrix $A_e$ is stochastic.

*Proof*: $A_e$ is obviously nonnegative, so it suffices to show that the sum of each column of $A_e^t$ equals 1. To this end, we begin by rewriting $A_e^t$:

$$\begin{aligned} A_e^t &= \sum_{k=0}^{\infty} \left( A^t (A_{\mathcal{S}}^t)^k - (A_{\mathcal{S}}^t)^{k+1} \right) \\ &= A^t + \sum_{k=1}^{\infty} \left( A^t (A_{\mathcal{S}}^t)^k - (A_{\mathcal{S}}^t)^k \right) \\ &= A^t + A^t M - M, \quad \text{with } M = \sum_{k=1}^{\infty} (A_{\mathcal{S}}^t)^k \end{aligned} \tag{7.17}$$

Consider the sum $s_w$ of the column of $A^t M$ associated with page $w$:

$$s_w = \sum_{u \in V} \sum_{v \in V} A^t_{u,v} M_{v,w} = \sum_{v \in V} \left( \sum_{u \in V} A^t_{u,v} \right) M_{v,w} = \sum_{v \in V} M_{v,w} \tag{7.18}$$

Thus, the sum of each column of $A^t M - M$ is zero, which shows that $A_e$ is stochastic, since the sum of each column $w$ equals $\sum_{v \in V} A^t_{v,w} = 1$. ∎

**Lemma 7.3**: Let $V_{\text{int}}$ be the set of pages with no incoming external link, and $V_{\text{ext}}$ the set of pages with at least one incoming external link. If we reorder the pages according to $(V_{\text{int}}, V_{\text{ext}})$, then $A_e$ can be written as

$$A_e = \begin{pmatrix} 0 & T \\ 0 & \tilde{A}_e \end{pmatrix} \tag{7.19}$$

where $\tilde{A}_e$ is an irreducible stochastic matrix.

*Proof*: The columns of $(A - A_{\mathcal{S}})$ corresponding to pages of $V_{\text{int}}$ are zero. The same is therefore true of those of $A_e$, which shows that $A_e$ can be written in the form

$$A_e = \begin{pmatrix} 0 & T \\ 0 & \tilde{A}_e \end{pmatrix} \tag{7.20}$$

$\tilde{A}_e$ is stochastic, since $A_e$ is. It remains to show that it is irreducible. Consider two vertices $v$ and $w$ in $V_{\text{ext}}$, and a path $\mathcal{C} = v, ..., w$ leading from $v$ to $w$ in $G$. Let $i_0, i_1, ..., i_{k-1}, i_k$ be the subsequence obtained by keeping in $\mathcal{C}$ only the vertices of $V_{\text{ext}}$ ($i_0 = v$ and $i_k = w$). Then, $i_0, i_1, ..., i_{k-1}, i_k$ is a path in the graph induced by $\tilde{A}_e$. Indeed, between $i_{l-1}$ and $i_l$, there exists a subpath of $\mathcal{C}$ consisting of an internal path within $S(i_{l-1})$, followed by an external jump leading to $i_l$. By the definition of $A_e$, we therefore have

$$\tilde{A}_{e_{(i_{l-1}, i_l)}} = A_{e_{(i_{l-1}, i_l)}} > 0 \tag{7.21}$$

$i_0, i_1, ..., i_{k-1}, i_k$ is thus indeed a path in the graph induced by $\tilde{A}_e$, which shows that $\tilde{A}_e$ is irreducible.

Q.E.D. ■

$A_e$ therefore has a unique PageRank, which is zero on $V_{\text{int}}$[1] and equals the PageRank of $\tilde{A}_e$ on $V_{\text{ext}}$. Subject to aperiodicity, it can be computed iteratively. Only the coefficients of $\tilde{A}_e$ are needed to compute this PageRank. Although we have not conducted extensive research to estimate the size of $V_{\text{ext}}$, the few analyses we have been able to perform, both on crawls and on server logs (in particular those of INRIA), seem to indicate that one can expect $|V_{\text{ext}}| \leq 0.1\ |V|$.

### 7.4.3 Theoretical decomposed PageRank computation

From Equation (7.14) and Equation (7.15), we can establish a theoretical semi-distributed method for computing PageRank.

[1]This result is natural: a page with no incoming external link cannot receive incoming external PageRank.

- Each site $S$ computes, from its block $A_S$ of the internal transition matrix $A_{\mathcal{S}}$, its block $\left(\mathrm{Id} - A_S^t\right)^{-1}$ of the matrix $\left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1}$.
- The different rows of $\tilde{A}_e$ can then be reconstructed and centralized.
- The external PageRank $P'_e$ associated with $A_e$ is then computed (here an aperiodicity assumption on $\tilde{A}_e$ is needed).
- Each site S obtains its own PageRank $P'(v)$, $v \in S$, by applying $P'_e(v)$, $v \in S$, to its matrix $\left(\mathrm{Id} - A_S^t\right)^{-1}$.

**Lemma 7.4**: The vector $P'$ thus obtained is homogeneous to the PageRank $P$ associated with $G$.

*Proof*: Since $A$ is irreducible, it suffices to show that $A^t P'$ equals $P$:

$$\begin{aligned} A^t P' &= A^t \left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1} P'_e \\ &= (A^t - A_{\mathcal{S}}^t)\left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1} P'_e + A_{\mathcal{S}}^t \left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1} P'_e \\ &= A_e^t P'_e + \left(\left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1} - \left(\mathrm{Id} - A_{\mathcal{S}}^t\right)\left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1}\right) P'_e \qquad (7.22) \\ &= P'_e + \left(\left(\mathrm{Id} - A_{\mathcal{S}}^t\right)^{-1} - \mathrm{Id}\right) P'_e \\ &= P'_e + P' - P'_e = P' \end{aligned}$$

Q.E.D. ■

# 7.5 Intermezzo: modifying one's own PageRank

Before tackling the central piece of this chapter, the FlowRank algorithm, we want to show that our decomposition of PageRank explains to what extent a website can modify its own PageRank, which will be an opportunity to gently introduce the *zap* factor into our model.

The results we are about to present make sense if one accepts the idea that a website can only modify its external PageRank with great difficulty, whereas the situation is entirely different for internal PageRank. Indeed, PageRank exchanges between websites are closely monitored by Google, which does not hesitate to penalize websites that exchange links for the sole purpose of increasing their external PageRank. Such PageRank factories, called *link farms* or *nurseries*, are generally assigned a PageRank of zero, and thus end up ranked behind all other pages[2].

[2]Note that this policy has been the subject of numerous lawsuits between Google and search engine optimization companies. Despite suspicions regarding Google's impartiality when it comes

### 7.5.1 Amplification coefficient

Consider a site $S \in \mathcal{S}$, its PageRank $P(S)$ and its incoming external PageRank $P_{ee}(S)$. We define the amplification coefficient $\alpha$ of $S$ as the ratio between PageRank and incoming external PageRank:

$$\alpha(S) = \frac{P(S)}{P_{ee}(S)} \tag{7.23}$$

Since $P = (\mathrm{Id} - A_S^t)^{-1} P_{ee}$, $\alpha(S)$ depends only on the internal structure of $S$ and on the distribution of external PageRank over $S$[3].

Knowledge of $S$ alone gives us an estimate of $\alpha(S)$:

**Lemma 7.5**: A bound on the amplification coefficient $\alpha(S)$ is

$$\frac{1}{1-\omega} \leq \alpha(S) \leq \frac{1}{1-\Omega} \tag{7.24}$$

with $\omega = \min_{v \in S} \frac{d_i(v)}{d(v)}$ and $\Omega = \max_{v \in S} \frac{d_i(v)}{d(v)}$.

*Proof*: If we consider the vector space associated with $S$, for every elementary vector $e_v$, $v \in S$, we have $\|A_S(e_v)\|_1 = \frac{d_i(v)}{d(v)}$, and therefore $\omega \|X\|_1 \leq \|A_S X\|_1 \leq \Omega \|X\|_1$ for every vector $X > 0$ defined on $S$.

We deduce the first inequality of Equation (7.24):

$$P(S) = \sum_{v \in S} P(v) = \| \sum_{k \in \mathbb{N}} \left(A_S^t\right)^k (P_{ee}) \|_1 \geq \sum_{k=0}^{\infty} \omega^k \|P_{ee}\|_1 = \frac{1}{1-\omega} P_{ee}(S) \tag{7.25}$$

as well as the second:

$$P(S) = \sum_{v \in S} P(v) = \| \sum_{k \in \mathbb{N}} \left(A_S^t\right)^k (P_{ee}) \|_1 \leq \sum_{k=0}^{\infty} \Omega^k \|P_{ee}\|_1 = \frac{1}{1-\Omega} P_{ee}(S) \tag{7.26}$$

■

The consequence of Equation (7.24) is that as soon as $\Omega = 1$, nothing prevents a site from amplifying PageRank arbitrarily. In the limiting case where $\omega = 1$ (a site with no outgoing external link, for example a commercial site that does not want the user to go elsewhere), there is infinite amplification and a short-circuit phenomenon. We recover the well-known rank sink phenomenon (see [Pag+98]), seen this time from the perspective of amplification: a set of pages with no outgoing link will absorb and accumulate all the incoming external PageRank until the flow is exhausted.

to defining a nursery, Google has never been found guilty: a search engine ranks its results as it sees fit.

[3]Note that this distribution can to some extent be influenced by modifications to the structure of $S$. But as we have seen, overly large variations can be a sign of a link farm.

Fortunately, over-amplification is controlled by the *zap* factor.

### 7.5.2 Introduction of the *zap* factor

From now on, we will leave the ideal setting where $G$ is strongly connected and aperiodic to consider an arbitrary Web graph $G$. In particular, we must choose which PageRank model to adopt, and our choice naturally fell on the non-compensated PageRank with *zap* factor. Thanks to Theorem 5.4, we know that this is a model strictly equivalent to the $\mu$-compensated PageRank generally used, with the difference that it allows working with a constant *zap* flow.

$P$ is therefore now the unique vector satisfying $P = dA^tP + (1-d)Z$, where $Z$ is a covering distribution and $d$ is the *zap* factor.

We also need to label the *zap* flow. We could split it into external and internal flow, depending on whether the *zap* takes us out of our current site or not, but we judged it more appropriate to consider the *zap* flow as entirely external, and to separate the external flow into external click flow and external *zap* flow. We will continue to reserve the terms incoming external PageRank and outgoing external PageRank for the external click flow, and by analogy with electricity we will define two new PageRank flows related to *zap*: the **induced PageRank**, denoted $P_{\text{ind}}$, which is the probability[4] of arriving at a page by *zap*, and the **dissipated PageRank**, denoted $P_{\text{dis}}$, which is the probability[5] of leaving a page by *zap*.

We now have a bestiary of six PageRanks, or rather six flows, which are summarized in Table 7.1 (recall: $s$ is the stochastic defect, defined by $s = \mathbf{1}^t - A \cdot \mathbf{1}^t$).

Note in passing that $P = P_{ei} + P_{ee} + P_{\text{ind}} = P_{si} + P_{se} + P_{\text{dis}}$.

The internal and external conservation laws still hold. This time we will not attempt to prove them by matrix calculation, and will content ourselves with justifying them by the fact that we are dealing with a stationary flow. In particular, the external conservation law on a site $S$ now reads:

$$P_{ee}(S) + P_{\text{ind}}(S) = P_{se}(S) + P_{\text{dis}}(S) \tag{7.27}$$

| **flow** | **incoming** | **outgoing** |
|---|---|---|
| internal | $P_{ei} = dA_S^tP$ | $P_{si} = d(A_S\mathbf{1}^t) \times P$ |
| external (click) | $P_{ee} = d(A - A_S)^tP$ | $P_{se} = d\big((A - A_S)\mathbf{1}^t\big) \times P$ |
| external (*zap*) | $P_{\text{ind}} = (1-d)Z$ | $P_{\text{dis}} = (1-d)P + ds \times P$ |

Table 7.1: The six PageRank flows in the non-compensated model

[4]Even though the non-compensated model means that we should no longer speak of probability, we will sporadically allow ourselves to continue using this term, even though it is more correct to speak of flow.

[5]See preceding footnote.

This equation gives us an interesting result: if a site $S$ has a PageRank greater than $Z(S)$, its outgoing external PageRank is less than its incoming external PageRank, with equality if, and only if, $P(S) = Z(S)$ and $s(S) = 0$. This means that a site $S$ can only hope to have a PageRank greater than the default PageRank $Z(S)$ on condition that it gives less than what it receives.

### 7.5.3 *Zap* and amplification coefficient

With the introduction of the *zap* factor, Equation (7.14) now becomes

$$P = (\mathrm{Id} - dA_S^t)^{-1}(P_{ee} + P_{\mathrm{ind}}) \tag{7.28}$$

The bound seen in Section 7.5.1 still holds upon replacing $A$ by $dA$ and $P_{ee}$ by the total incoming external PageRank $P_{ee} + P_{\mathrm{ind}}$, and setting by convention $\frac{d_i(v)}{d(v)} = 0$ if $d(v) = 0$. We thus obtain Lemma 7.6.

**Lemma 7.6**: The amplification factor $\alpha'$ defined by $\alpha'(S) = \frac{P(S)}{P_{ee}(S)+P_{\mathrm{ind}}(S)}$ satisfies

$$\frac{1}{1 - d\omega} \leq \alpha'(S) \leq \frac{1}{1 - d\Omega} \tag{7.29}$$

*Proof*: We proceed exactly as in the proof of Lemma 7.5. If we consider a fixed site $S$, and if we restrict $P_{ee}$ and $P_{\mathrm{dis}}$ to their values on $S$, the first inequality is obtained by writing

$$\begin{aligned} P(S) = \sum_{v \in S} P(v) &= \| \sum_{k \in \mathbb{N}} (dA_S^t)^k (P_{ee} + P_{\mathrm{ind}}) \|_1 \\ &\geq \sum_{k=0}^{\infty} (d\omega)^k (\|P_{ee}\|_1 + \|P_{\mathrm{ind}}\|_1) \\ &\geq \frac{1}{1 - d\omega}(P_{ee}(S) + P_{\mathrm{ind}}(S)) \end{aligned} \tag{7.30}$$

and the second similarly:

$$\begin{aligned} P(S) = \sum_{v \in S} P(v) &= \| \sum_{k \in \mathbb{N}} (dA_S^t)^k (P_{ee} + P_{\mathrm{ind}}) \|_1 \\ &\leq \sum_{k=0}^{\infty} (d\Omega)^k (\|P_{ee}\|_1 + \|P_{\mathrm{ind}}\|_1) \\ &\leq \frac{1}{1 - d\Omega}(P_{ee}(S) + P_{\mathrm{ind}}(S)) \end{aligned} \tag{7.31}$$

■

**Numerical value**

For a real website, it is entirely possible to have $\omega = \Omega = 0$ (a site with no internal link), or on the contrary $\omega = \Omega = 1$ (a site with no external link, and all of whose pages have at least one link). Thus, the amplification coefficient can vary between 1 (the site derives no benefit from the PageRank it receives) and $\frac{1}{1-d}$ (maximum utilization of received PageRank). Since $d$ is a universal constant equal to $0,85$, we conclude that for a fixed total incoming external PageRank, a site's PageRank can vary depending on its structure by a factor of $\frac{20}{3}$. For example, a very poorly structured site can, by restructuring itself, achieve a new PageRank equal to approximately 666%[6] of the former PageRank.

**Robustness of PageRank**

Bianchini *et al.* [BGS02, BGS03] show that the effect a site can produce on the Web is controlled by the PageRank of that site. More precisely, if we consider a dynamic graph between two instants $t$ and $t+1$, they proved that:

$$\sum_{v \in V} |P_t(v) - P_{t+1}(v)| \leq \frac{2d}{1-d} \sum_{s \in S} P_t(s) \tag{7.32}$$

This result can also be deduced from Lemma 7.6: if a site $S$ changes between $t$ and $t+1$, the largest possible relative variation is the one where we go from $\alpha'(S) = 1$ to $\alpha'(S) = \frac{1}{1-d}$. This modification of the site's structure, which corresponds to creating a rank sink, can only be done at the expense of external PageRank, and therefore of incoming external PageRank, so we necessarily have $P_{(ee)_t} \geq P_{(ee)_{t+1}}$, giving a variation of at most $\frac{d}{1-d}P(S)$. Since Bianchini *et al.* work here in a compensated model (the sum of PageRanks is constant), a variation of $\frac{d}{1-d}P(S)$ in $S$ generates the same variation outside $S$, which gives us inequality Equation (7.32).

### 7.5.4 Amplification of a given page

For a website, the interest of PageRank is above all to be visible to users. In particular, the administrator of a site will probably be less interested in a high PageRank across the entire site than in a very high PageRank on a few pages, or even on a single page. It is therefore better to concentrate one's PageRank on a general homepage rather than distribute it among several specialized pages. We will therefore consider the following problem: consider a site $S$ of $n+1$ pages fed by an incoming external PageRank $P_{ee}$. How can we maximize the PageRank of a given page $v_0 \in S$?

The answer is not difficult once one notices that the optimal structure is the one where all pages of $S$ other than $v_0$ point to $v_0$ (and only $v_0$) and $v_0$ points to at least one other page of $S$. $v_0$ thus recovers, up to dissipation, all the PageRank of the other pages, and recovers its own PageRank up to a factor of $d^2$, which is the maximum possible in a graph where, let us recall, self-loops are not taken into account. We then have

[6] This lovely number is further proof of the necessity of having $0,85$ as the value of $d$.

$$P(v_0) = \frac{P_{ee}(v_0)}{1-d^2} + \frac{Z(v_0)}{1+d} + d \sum_{v \in S, v \neq v_0} \left( \frac{P_{ee}(v)}{1-d^2} + \frac{Z(v)}{1+d} \right) \tag{7.33}$$

In the particular case where $Z$ is the uniform distribution, we thus obtain

$$P(v_0) \leq \frac{P_{ee}(S)}{1-d^2} + \frac{1+nd}{(1+d)|V|} \tag{7.34}$$

with equality if, and only if, all the incoming external PageRank is concentrated on $v_0$, that is, $P_{ee}(S) = P_{ee}(v_0)$.

From all this, we deduce some strategies for improving the PageRank of a page $v_0$, which corroborate the recommendations found on many websites devoted to PageRank improvement:

- Ask the administrators of other sites to always point, as far as possible, to the main page rather than to a specific page. Optionally, set up scripts that redirect to the homepage any access from a page external to the site[7].
- Always remember to include links back to the homepage, and limit the depth of the site (and thus the dissipation) as much as possible. In the particular case of frame-based sites, include `<noframe>` tags inside which the star structure is explicitly represented.

Let us conclude with a few remarks valid when $Z$ is the uniform distribution:

- With the optimal strategy, a site consisting simply of two pages pointing to each other has a PageRank that is at least equal to the average PageRank, even if the incoming external PageRank is zero: $P(v_0) \geq \frac{1}{|V|}$.
- If $1 \ll n \leq |V|$ (for example a site that dynamically generates pages pointing to $v_0$, such as a database query site with links other than forms), the ratio $\frac{P(v_0)}{P_{\text{moyen}}}$ is approximately $\frac{d}{1+d}n$. It is therefore possible to increase one's PageRank on $v_0$ linearly. In practice, this is valid provided that Google's robots take the trouble to explore all the pages[8], and that the sole purpose of all these pages is not to increase one's PageRank[9].

## 7.6 Real-world case: FlowRank and BlowRank

The objective of this section is to adapt the theoretical computation seen in Section 7.4.3 to real-world situations.

[7]This recommendation should be taken with caution: the way Google handles redirections is not very clear. Moreover, this can make navigation less ergonomic for the user.

[8]To my great regret, Google has not yet finished exploring the *page that points to all pages* (see Section 1.4 page Section 1.4), which explains why it does not yet have a maximal PageRank…

[9]Otherwise, beware the penalty.

### 7.6.1 Equilibrium relations with the *zap* factor

Now that we have had time to become familiar with the induced and dissipated flows, we can rewrite the equations seen in the course of Section 7.4.

With the introduction of the *zap* factor, Equation (7.14), which describes the link between incoming external PageRank and PageRank, becomes as we have seen

$$P = (\mathrm{Id} - dA_S^t)^{-1}(P_{ee} + (1-d)Z) \tag{7.35}$$

The equilibrium relation for external PageRank, the equivalent of Equation (7.15), is obtained by combining Equation (7.35) with the definition of $P_{ee}$. We thus obtain:

$$\begin{aligned} P_{ee} &= dA_e^t P_{ee} + Z_e, \quad \text{with} \\ A_e^t &= (A - A_S)^t (\mathrm{Id} - dA_S^t)^{-1} \quad \text{and} \\ Z_e &= d(1-d)A_e^t Z \end{aligned} \tag{7.36}$$

**Lemma 7.7**: The spectral radius of $A_e$ is less than 1.

*Proof*: The proof is similar to that of Lemma 7.2: we show indeed that $A_e$ is substochastic (in the broad sense). We will show for this that the spectral radius of $dA_e$ is less than $d$. Since $A_e$ is nonnegative, it suffices, by the Perron-Frobenius theorem, to show that for every nonnegative vector $X$, $\|dA_e^t X\|_1 \leq d\,\|X\|_1$. To this end, we begin by rewriting $dA_e^t$:

$$\begin{aligned} dA_e^t &= \sum_{k=0}^{\infty} \Big( dA^t (dA_S^t)^k - (dA_S^t)^{k+1} \Big) \\ &= dA^t + \sum_{k=1}^{\infty} \Big( dA^t (dA_S^t)^k - (dA_S^t)^k \Big) \\ &= dA^t + dA^t M - M, \quad \text{with } M = \sum_{k=1}^{\infty} (dA_S^t)^k \end{aligned} \tag{7.37}$$

Now consider a nonnegative vector $X$. Since $dA_e^t X$, $dA^t X$, $dA^t MX$, and $MX$ are all nonnegative vectors, we have

$$0 \leq \|dA_e^t X\|_1 = \|dA^t X\|_1 + \|dA^t MX\|_1 - \|MX\|_1 \tag{7.38}$$

Since the spectral radius of $A^t$ is less than 1, we have $\|dA^t MX\|_1 \leq \|MX\|_1$, hence

$$\|dA_e^t X\|_1 \leq \|dA^t X\|_1 \leq d\,\|X\|_1 \tag{7.39}$$

Q.E.D. ∎

**Remark 7.3**: The inequality $\|dA^t MX\|_1 \leq \|MX\|_1$ used in the proof of Lemma 7.7 is very crude. In practice, one can therefore legitimately expect the spectral radius of $A_e$ to be smaller than 1, and thus that Equation (7.36) allows finding $P_{ee}$ with a convergence rate smaller than $d$. The results presented by Arasu *et al.* (see [Ara+01]) support this and allow us to hope for very fast convergence in practice.

**Application: estimating the PageRank of a website**

For the administrator of a website $S$, being able to estimate the importance of its pages without calling on external help or crawling the indexable Web can be of considerable interest. For example, this importance could be incorporated into an internal search engine. Now, according to Equation (7.35), it suffices for this to be able to estimate the incoming external PageRank on $S$.

Indeed, if we define the function $\text{SpeedRank}(M, X)$, inspired by Algorithm 5.4, as a function that returns, for $M$ nonnegative with spectral radius strictly less than $\frac{1}{d}$ and $X$ nonnegative, the vector $P$ satisfying $P = dM^t P + X$, with precision $\varepsilon$, then

$$P_S = \text{SpeedRank}\Big(A_S, \big((P_{ee})_S + (1-d)Z_S\big)\Big) \tag{7.40}$$

To estimate this incoming external PageRank, two local approaches are possible:

- We have seen that PageRank is supposed to, to a certain extent, try to represent the behavior of real users (see Section 5.2.2). One can then take, as an estimate of the incoming external PageRank, the number of real clicks from outside the site to a page of the site, measured from the analysis of the Web server logs.
- Thanks to, or because of, the factor $d$, the ranking by in-degree is an approximation of the ranking by PageRank (see Section 5.3.5). By counting the number of external references associated with each page, obtained again through an analysis of the Web server logs, one obtains another estimate of $P_{ee}$.

Once one has an estimate of $(P_{ee})_S$, it must be balanced against $Z_S$. One way, among many others, to achieve this balancing is to take a weighted average of the two vectors. For example, one could return as a normalized estimate of PageRank on $S$

$$P_S = \text{SpeedRank}\left(A_S, \left(d\frac{(P_{ee})_S}{\|(P_{ee})_S\|_1} + (1-d)\frac{Z_S}{\|Z_S\|_1}\right)\right) \tag{7.41}$$

Note in passing that the rank source is then normalized to 1 regardless of the size of the site $S$ under consideration. Since the goal is merely to estimate the relative importance of pages within $S$, this poses no problem.

### 7.6.2 A first approach: FlowRank

We now have all the data in hand to construct the FlowRank algorithm. Let us first observe that thanks to the *zap* factor, $P_{ee}$ is completely defined by Equation (7.36), whereas Equation (7.15) only guaranteed obtaining a vector up to a scalar multiple. In order to avoid explicitly inverting the matrices $(\mathrm{Id} - dA_S)$, for $S \in \mathcal{S}$, we will resort to the function SpeedRank defined previously. Thanks to this function, all the values we need to know can be computed:

- $(A - A_{\mathcal{S}})^t$ SpeedRank$(A_{\mathcal{S}}, e_v)$, where $e_v$ is the vector equal to 1 on $v$, 0 elsewhere, gives the column of $A_e^t$ associated with $v$. This computation can be limited, in both input and output, to the pages of $V_{\text{ext}}$. We thus obtain the $V_{\text{ext}} \times V_{\text{ext}}$ matrix that we previously called $\tilde{A}_e$, and which we will continue by abuse of notation to call $A_e$. Note also that this computation can be performed locally within each site $S(v) \in \mathcal{S}$.
- $Z_e$ equals $d(1-d)(A - A_{\mathcal{S}})^t$ SpeedRank$(A_{\mathcal{S}}, Z)$. This computation can also be distributed across all sites. Note that $Z_e$ is zero outside $V_{\text{ext}}$, so we can restrict ourselves to considering $\tilde{Z}_e$, the restriction of $Z_e$ to $V_{\text{ext}}$.
- Once $\tilde{A}_e$ and $\tilde{Z}_e$ have been computed, it is possible to compute $\tilde{P}_{ee}$, the restriction of $P_{ee}$ to $V_{\text{ext}}$: $\tilde{P}_{ee} = \text{SpeedRank}\big(\tilde{A}_e, \tilde{Z}_e\big)$.
- The non-compensated PageRank is then given by $P = \text{SpeedRank}(A_{\mathcal{S}}, (P_{ee} + (1 - d)Z))$. Once again, this step can be performed at the site level.

All these operations are summarized in Algorithm 7.1.

The FlowRank algorithm has numerous advantages:

- By breaking down the SpeedRank computations at the site level, it becomes possible to store the matrix in RAM, which allows extremely fast computation, all the faster since SpeedRank performs no norm computation.
- Apart from the computation of $P_{ee}$, which is global, all other computations can be performed in a decentralized manner at the site level. It is thus possible to parallelize a large portion of the computations.
- To take into account the update of a given site, it is not necessary to rerun the entire algorithm (unlike the case of a fully centralized algorithm). It suffices to rerun the SpeedRank computations at the level of the site in question, and to update $P_{ee}$. Using the previous values as initial values for SpeedRank can make the operation very fast.

But it also has drawbacks. Indeed, $2k + 1 + |V_{\text{ext}}|$ SpeedRank computations must be performed. Even though SpeedRank, as its name suggests, is very fast, this number remains high. Likewise, the computation of $P_{ee}$ is a SpeedRank on a $|V_{\text{ext}}| \times |V_{\text{ext}}|$ matrix. If this matrix does not fit in RAM, much of the practical benefit of the decomposed computation is lost. To solve these problems, we will draw inspiration from a competing algorithm, the *BlockRank* algorithm.

**Data**

- a Web graph $G = (V, E)$;
- a site partition of $G$, $\mathcal{S} = (S_1, ..., S_k)$;
- a covering probability distribution $Z$;
- a *zap* coefficient $d \in ]0, 1[$;
- a real number $\varepsilon$;
- a function SpeedRank, inspired by Algorithm 5.4, such that $P = \text{SpeedRank}(M, X)$ satisfies $P = dM^t P + X$, with precision $\varepsilon$.

**Result**

The non-compensated PageRank vector associated with $(G, Z, d)$.

**begin**

$A_e \leftarrow 0_{|V_{\text{ext}}| \times |V_{\text{ext}}|}$

**for** each site $S$ in $\mathcal{S}$:

  Compute $A_S$, the $|S| \times |S|$ submatrix of $A_{\mathcal{S}}$

  Compute $\overline{A}_S$, the $|S| \times |V_{\text{ext}}|$ submatrix of $(A - A_{\mathcal{S}})$

  $Z_{S_e} \leftarrow d(1-d)\overline{A}_S^t$ SpeedRank$(A_S, Z_S)$

  **for** each page $v$ in $S \cap V_{\text{ext}}$:

    $(A_e^t)_{*,v} \leftarrow \overline{A}_S^t$ SpeedRank$(A_S, e_v)$

  $Z_e \leftarrow \sum_{S \in \mathcal{S}} Z_{S_e}$

  $P_{ee} \leftarrow \text{SpeedRank}(A_e, Z_e)$

  **for** each site $S$ in $\mathcal{S}$:

    $P_S \leftarrow \text{SpeedRank}\Big(A_S, (P_{ee})_S + (1-d)Z_S\Big)$

**return** $(P_S)_{S \in \mathcal{S}}$

Algorithm 7.1: FlowRank: decomposed computation of non-compensated PageRank

### 7.6.3 Distributed algorithm of Kamvar *et al.*: BlockRank

In parallel with our work on the block structure of Web graphs [MV02, MV03a] and the possible applications to PageRank [MV03b, MV04], Kamvar *et al.* carried out very similar research. In [Kam+03b], they use a site decomposition to propose a semi-distributed algorithm for computing an estimate of PageRank: BlockRank. This algorithm is based on the computation of a local PageRank, which in our notation is the PageRank on $A_{\mathcal{S}}$, and of a site PageRank, based on a substochastic BlockRank matrix, denoted $B$, defined on the quotient graph $G/\mathcal{S}$. The estimate of the PageRank of a page $v$ is then given by the product of the local PageRank of $v$ by the PageRank of $S(v)$, also called BlockRank[10]. Although FlowRank and BlockRank resemble each other at first glance, there are important differences that we wish to highlight:

[10]For full details, see [Kam+03b].

- Up to the errors introduced by $\varepsilon$, FlowRank gives the non-compensated PageRank[11], and not an approximation. Thus, the problem of underestimation of the PageRank of main pages encountered with the BlockRank algorithm [Kam+03b] does not arise. There is of course a price to pay in terms of algorithm complexity.
- The keystone of FlowRank is non-compensated PageRank, and the SpeedRank algorithm that follows from it, whereas BlockRank stays with the idea of $\mu$-compensated PageRank, which is three times slower on "small" graphs.
- While the local PageRank within a site $S$ equals, up to normalization, $\text{SpeedRank}(A_S, Z_S)$, the global FlowRank matrix, $\tilde{A}_e$, and the BlockRank one, $B$, have almost nothing in common. For example, $B$ has transitions from a site to itself. One will also note that whereas FlowRank uses an external *zap* $Z_e$ adapted to $G$, the PageRank on $B$ uses a uniform *zap*.

### 7.6.4 Hybrid algorithm: BlowRank

The main advantage of BlockRank over FlowRank is the reduction from $|V_{\text{ext}}|$ to $k = |\mathcal{S}|$ made possible by the approximations. This gain applies both to the number of local computations and to the size of the global matrix. We are therefore tempted to reduce $\tilde{A}_e$ to a $k \times k$ matrix. To do this, we must reduce the external flow information to a scalar per site $S$, for example by replacing $(P_{ee})_S$ with $P_{ee}(S)$. We must then define how the incoming external PageRank is injected inside each site. We will therefore assume that each site $S$ is equipped with a probability distribution that estimates the distribution of incoming external PageRank. This distribution, which we denote $D_S$, may be the uniform distribution on $S$, or more finely a distribution on the entry pages of the site, which are the most likely to be pointed to.

We can now consider the hybrid BlowRank algorithm (7.2), which differs from FlowRank by an imposed reorganization of the external flow at the entrance of each site: everything happens as if at the entrance of each site, the incoming external PageRank (by click) were collected and redistributed according to $D_S$.

We thus obtain an algorithm that requires only $2 \cdot k$ local calls to SpeedRank, plus one global call on a $k \times k$ matrix, which places it in terms of performance at the same level as BlockRank, while the approximations made are smaller, yielding a PageRank less perturbed relative to the global model.

[11] Let us recall once again that from a theoretical standpoint, there is strictly no difference between non-compensated PageRank and the $\mu$-compensated PageRank traditionally used.

**Data**

- a Web graph $G = (V, E)$;
- a site partition of $G$, $\mathcal{S} = (S_1, ..., S_k)$, each site $S$ being associated with a probability distribution $D_S$;
- a covering probability distribution $Z$;
- a *zap* coefficient $d \in ]0, 1[$;
- a real number $\varepsilon$;
- a function SpeedRank, inspired by Algorithm 5.4, such that $P = \text{SpeedRank}(M, X)$ satisfies $P = dM^tP + X$, with precision $\varepsilon$.

**Result**

An estimate of the non-compensated PageRank vector associated with $(G, Z, d)$.

**begin**

$B_e \leftarrow 0_{k \times k}$

**for** each site $S$ in $\mathcal{S}$:

$A_S \leftarrow \left(A_{v,w}\right)$, for $v$ and $w$ in $S$

$\left(\overline{A}_S\right)_{v,T} \leftarrow \sum_{w \in T}^{w \leftarrow v,} A_{v,w}$, for $v \in S$ and $T \neq S$

$Z_{S_i} \leftarrow \text{SpeedRank}(A_S, (1-d)Z_S)$

$Z_{S_e} \leftarrow d\overline{A}_S^t Z_{S_i}$

$B_S \leftarrow \text{SpeedRank}(A_S, D_S)$

$\left(B_e^t\right)_{*,S} \leftarrow \overline{A}_S^t B_S$

$Z_e \leftarrow \sum_{S \in \mathcal{S}} Z_{S_e}$

$P_{ee} \leftarrow \text{SpeedRank}(B_e, Z_e)$

**for** each site $S$ in $\mathcal{S}$:

$P_S \leftarrow P_{ee}(S)B_S + Z_{S_i}$

**return** $(P_S)_{S \in \mathcal{S}}$

Algorithm 7.2: BlowRank: BlockRank-style approximation of the FlowRank algorithm

# Chapter 8
# Conclusion

> *The Internet is the meeting place of researchers, but also of every crackpot, every voyeur, and every piece of gossip on earth.*
>
> *Alain* Finkielkraut

After these few years of doctoral work, it appears to me that my understanding of Web graphs, PageRanks, and the mechanisms governing them has grown considerably since my tentative beginnings when, following a lead laid out by Daniel Krob, I was trying to detect *hot spots* on the Web; an understanding that I have endeavoured to share with the reader throughout this work. This conclusion summarises the work already accomplished, as well as what remains to be done.

—

The first chapter presented that human achievement which is the Web. Defining the latter generated more problems than solutions, and faced with the many facets of the electronic web, we had to limit our field of study to what we called the indexable Web.

We then considered *crawlers*, those trawlers of the Web that ceaselessly traverse the indexable Web in order to populate databases, and we studied the sizes of the largest of these databases, namely those of search engines. In particular, we highlighted the precautions that were necessary when dealing with figures announced by commercial search engines, which sometimes seem fanciful.

We then dwelt on several aspects of the graph structure induced by hyperlinks: the bow-tie structure, and the misinterpretations that are sometimes made of it; the site-level structure, which is closely tied to the URL decomposition tree. We set aside certain structural aspects that have been extensively studied elsewhere, such as the distribution of in-degrees and out-degrees [AJB99, AJB00, BA99] or the small-world structure of Web graphs [Ada99, Kle00, PLH01]. These aspects, although interesting, fall outside the scope of this thesis.

—

The second part got to the heart of the matter: PageRank-type ranking methods. We began in Chapter 4 with some reminders on Markov chains, their representation using stochastic matrices, and the application of the Perron-Frobenius theorem to

stochastic matrices to find stationary distributions. We also studied some convergence results for sub-stochastic matrices. All the results presented in this chapter have most certainly been presented many times in the literature, but we chose to re-derive them in order to introduce a formalism extensively used throughout the remainder, for example to describe sub-stochastic matrices. The wheel was certainly reinvented more than once in this chapter, but we thereby ensured that the size of said wheels was the right one.

In Chapter 5, we defined the general principles governing PageRank, stated the problem, and catalogued the most classical algorithms. We aimed to distinguish ourselves from other surveys on the subject [BGS02, BGS03, Bou01, BJ02, LM04] by offering a different perspective, notably through a unification of the dual interpretation — stochastic and flow-based — associated with the various algorithms. Some problems raised in this chapter would have deserved a more thorough study, which I hope to have the opportunity to carry out in the near future. This is the case, for instance, for the link between the convergence of a PageRank ranking and that of its distribution, which is addressed in Section 5.5.

The last two chapters then provided the opportunity to present new PageRank algorithms. The reader will have discovered the *BackRank* algorithm, which models the possibility for a surfer to go back during navigation. We showed that this algorithm is no more complicated to implement than a classical PageRank algorithm, while offering numerous advantages in terms of convergence and the handling of dangling pages. Finally, after abandoning the stochastic interpretation of PageRank in favour of a flow decomposition, we introduced two PageRank computation algorithms based on the strongly clustered structure of Web graphs: an exact algorithm, *FlowRank*; and an approximate algorithm, inspired by the *BlockRank* algorithm proposed in [Kam+03b]: *BlowRank*. The decomposition on which these algorithms are based opens the way to a broad range of semi-distributed and local PageRank estimation methods.

The study of these algorithms is far from complete; they still need to be tested on very large graphs and, why not, incorporated into a real search engine. At the time of writing, I am conducting experiments with Mohamed Bouklit on the graphs available on the *WebGraph* project website [BV], the largest of which has 118 million vertices. The results, very promising, point in the same direction as those presented in this thesis.

# Appendix A
# Perron-Frobenius Theorem

In 1907, Oskar Perron (1880-1975) published a theory of strictly positive matrices, which Georg Ferdinand Frobenius (1849-1917) extended in 1908, 1909, and 1912 to the case of nonnegative matrices[1]. The Perron-Frobenius theorem, which summarizes this theory, is in a sense the cornerstone of most convergence algorithms for stochastic matrices, and in particular of PageRank algorithms. We therefore thought it worthwhile to include a proof in the appendix, since in addition to guaranteeing the convergence of the algorithms, the concept of flow is inherent to the proof (strict inequality propagation lemma).

**Theorem A.1** (Perron-Frobenius): Let $A = \left(a_{i,j}\right)_{1 \leq i,j \leq n}$ be a nonnegative square matrix of size $n$, irreducible. Then, there exists an eigenvalue of $A$, denoted $r$, strictly positive, which has the following properties:

(a) There exist a left eigenvector and a right eigenvector associated with $r$ that are strictly positive.
(b) For any eigenvalue $\lambda$ of $A$, $|\lambda| \leq r$.
(c) The eigenspace associated with $r$ has dimension 1.
(d) For any nonnegative matrix $B$ less than or equal to $A$, and for any eigenvalue $\beta$ of $B$, $|\beta| \leq r$, with equality only if $A = B$.
(e) If $\frac{1}{r}A$ has cyclicity $d$, then the eigenvalues of $A$ with modulus $r$ are exactly $r \cdot \omega^j, j = 1, ..., d$, where $\omega = e^{\frac{2\pi i}{d}}$, and the associated eigenspaces have dimension 1.

*Proof*: To prove the Perron-Frobenius theorem, we will need the following lemma, which we shall call the strict inequality propagation lemma.

**Lemma A.1**: If $A$ is a nonnegative irreducible matrix of size $n$, and if $x$ and $y$ are two nonnegative vectors such that $x \leq y$, with at least one component for which there is strict inequality, then $\sum_{k=0}^{n-1} A^k x < \sum_{k=0}^{n-1} A^k y$.

[1] Helmut Wielandt – 1910-2001 – presented a simpler approach to the problem in 1950, which is the one used nowadays.

Indeed, let $i$ be a component for which there is strict inequality. Since the matrix $A$ is irreducible, for each component $j$, there exists $k \in [0, n-1] \cap \mathbb{N}$ such that $\left(A^k\right)_{j,i} > 0$. We deduce that $A^k x \leq A^k y$, with strict inequality at component $j$. By using the operator $\sum_{k=0}^{n-1} A^k$, we ensure that each component will "benefit" by propagation from the strict inequality present at $i$. In other words, $\sum_{k=0}^{n-1} A^k x < \sum_{k=0}^{n-1} A^k y$.

With this lemma in hand, we can now proceed to the proof itself.

(a) Consider the set $\Gamma$ of nonnegative vectors in $\mathbb{R}^n$ with norm 1 (we use the 1-norm: if $x = (x_i)_{1 \leq i \leq n}$, $\|x\|_1 = \sum_{i=1}^{n} |x_i|$).

For each $\gamma$ in $\Gamma$, we set

$$\mu_\gamma = \sup\{x \in \mathbb{R} : x\gamma \leq (A\gamma)\} \tag{A.1}$$

Clearly, $\mu_\gamma$ is a nonnegative real number, since it is bounded below by 0 and above by $n\|A\gamma\|_\infty$. Now consider $r = \sup_{\gamma \in \Gamma} \mu_\gamma$. $r$ is a (finite) strictly positive real number, since we have the bound:

$$0 < \frac{\min_{1 \leq i \leq n} \sum_{1 \leq j \leq n} a_{i,j}}{n} \leq r \leq n\|A\|_\infty \tag{A.2}$$

To show that $r$ is an eigenvalue, consider a sequence $(\gamma_n)_{n \in \mathbb{N}}$ of elements of $\Gamma$ such that $\mu_{\gamma_n}$ converges to $r$. Since $\Gamma$ is compact, it is possible (by the Bolzano-Weierstrass theorem) to extract a subsequence $\left(\gamma_{\varphi(n)}\right)_{n \in \mathbb{N}}$ that converges to a vector $\gamma_\infty \in \Gamma$.

$\gamma_\infty$ is a (right) eigenvector of $A$, associated with the eigenvalue $r$. Indeed, we have $r\gamma_\infty \leq A\gamma_\infty$. If equality does not hold, then by Lemma A.1,

$$r\left(\left(\sum_{k=0}^{n-1} A^k\right)\gamma_\infty\right) < A\left(\left(\sum_{k=0}^{n-1} A^k\right)\gamma_\infty\right) \tag{A.3}$$

which contradicts[2] the maximality of $r$.

We therefore have $r\gamma_\infty = A\gamma_\infty$, which proves that $\gamma_\infty$ is a (right) eigenvector of $A$, associated with $r$. The strict positivity of $\gamma_\infty$ is guaranteed by the strict inequality propagation lemma.

By reasoning with $A'$, we find an eigenvalue $s$ associated with a strictly positive left eigenvector of $A$. To prove (a), it suffices to show that $r = s$. Without loss of generality, by possibly interchanging $A$ and $A'$, assume $r \leq s$. It then suffices to take $x \neq 0$ such that $Ax = sx$[3] and to observe that $s|x| = |Ax| \leq A|x|$, which forces $s \leq r$, hence equality.

[2] Up to normalizing $\left(\sum_{k=0}^{n-1} A^k\right)\gamma_\infty$…

[3] $s$ is an eigenvalue of $A$, so there exists an associated right eigenvector.

(b) The proof is the same as for showing that $r = s$. If $\lambda x = Ax$, with $x \neq 0$, then $|\lambda||x| = |\lambda x| = |Ax| \leq A|x|$, hence $|\lambda| \leq r$.

(c) The fact that $r$ is a simple eigenvalue is also proved using the strict inequality propagation lemma. Indeed, if the eigenspace has dimension greater than or equal to 2, there exists an eigenvector $\gamma_\diamond$ not collinear with $\gamma_\infty$. The vector $\gamma_\diamond + \left(\max_{1\leq i\leq n}\left(-\frac{\gamma_{\diamond i}}{\gamma_{\infty i}}\right)\right)\gamma_\infty$ is also an eigenvector associated with $r$. By construction, it is nonzero, nonnegative, and at least one of its components is zero. By propagation of strict inequality, $n-1$ iterations of $A$ will make this component strictly positive, which is a contradiction and proves that $r$ is a simple root.

(d) Same proof as (b): if $\beta x = Bx$, with $x \neq 0$, then $|\beta||x| = |\beta x| = |Bx| \leq B|x| \leq A|x|$, hence $|\beta| \leq r$.

Equality case: $|\beta| = r$ forces $|\beta||x| = r|x| = B|x| = A|x|$. $|x|$ is therefore strictly positive, since it is collinear with $\gamma_\infty$. $(A - B)$ is a nonnegative matrix satisfying $(A - B)|x| = 0$, hence $(A - B) = 0$.

(e) Without loss of generality, by considering $\frac{1}{r}A$, we may assume that $r = 1$. Consider the following equivalence relation on the components: two components $i$ and $j$ are equivalent if there exists a component $k$ such that $a_{k,i}$ and $a_{k,j}$ are strictly positive, or a component $l$ such that $a_{i,l}$ and $a_{j,l}$ are strictly positive. Then, the cyclicity of $A$ equals the number of equivalence classes among the components. Indeed, the cyclicity of $A$ corresponds to the cyclicity on the underlying graph (the vertices corresponding to the components and the edges to the nonzero coefficients $a_{i,j}$). Since the graph is strongly connected (because the matrix is irreducible), this cyclicity is found by placing in the same equivalence class the successors and predecessors of each component.

Let $I_0$, ..., $I_{d-1}$ be the equivalence classes for the predecessor/successor relation, arranged in order of succession.

- If $\lambda$ is a $d$-th root of unity, then $\lambda$ is an eigenvalue of $A$, and an associated eigenvector is $x = (x_i)_{1\leq i\leq n}$, with

$$x_i = \lambda^{-k}\gamma_{\infty i} \quad \text{si } i \in I_k \tag{A.4}$$

Indeed, for $i$ in $I_k$, we have

$$\begin{aligned}(Ax)_i &= \sum_{j=1}^{n} a_{i,j}x_j = \sum_{j=1}^{n} a_{i,j}\lambda^{-k+1}\gamma_{\infty j} \\ &= \lambda^{-k+1}\sum_{j=1}^{n} a_{i,j}\gamma_{\infty j} = \lambda\lambda^{-k}\gamma_{\infty i} = \lambda x_i\end{aligned} \tag{A.5}$$

- Let $\lambda$ be an eigenvalue of modulus 1, and $x$ an associated eigenvector. We then write:

$$|x| = |Ax| \leq A|x| \tag{A.6}$$

In fact, equality must hold, otherwise by the strict inequality propagation lemma, 1 would no longer be the maximal eigenvalue. Note in passing that $|x|$ is collinear with $\gamma_\infty$. For complex numbers, the absolute value of a sum equals the sum of the absolute values only if all (nonzero) elements have the same phase. Since the coefficients of $A$ are nonnegative, this implies that all predecessors of a given component $x_i$ under $A$ have the same phase. Moreover, by construction, the successors also have the same phase. We can deduce that all components in the same equivalence class have the same phase. Furthermore, if $\varphi(k)$ is the phase of class $I_k$, then $\frac{\varphi(k-1)}{\varphi(k)} = \lambda$. By cyclicity, and using the convention $I_d = I_0$, we have:

$$1 = \frac{I_0}{I_d} = \prod_{k=1}^{d} \frac{I_{k-1}}{I_k} = \prod_{k=1}^{d} \lambda = \lambda^d \tag{A.7}$$

$\lambda$ is therefore a $d$-th root of unity.

Finally, note that given everything stated above, the vector $x$ is collinear with the vector $y$ defined by

$$y_i = \lambda^{-k} \gamma_{\infty i} \quad \text{si } i \in I_k \tag{A.8}$$

The eigenspace associated with $\lambda$ therefore has dimension 1.

Q.E.D.

■

# Appendix B
# A Short Study of PageRank on the INRIA Website

Starting from a snapshot of the graph of the website `http://www.inria.fr`, we applied a PageRank-type algorithm to determine which pages received the highest ranking. Several interesting observations emerged:

- The justification for removing, in the PageRank computation, the leaves of the graph (by leaf, we mean a node with zero out-degree).
- It proved essential to remove self-links (links from a page to itself), as this causes a resonance phenomenon. On the INRIA graph, before performing this modification, the PageRank was largely dominated by

  `http://www.inria.fr/DR:/multimedia/Bsv-fra.html`,

  which combines the roles of a self-referencing page and the root of a sink.
- The choice of the weight of the "random click" turns out to be crucial: if it is too small, sinks or near-sinks will absorb all the PageRank. If it is too large, the iterative aspect of PageRank will vanish, and the ranking will roughly correspond to a ranking by in-degree. In the case of INRIA, a random click probability of 10% at each iteration appears to be a good compromise.

**Results** It seems quite interesting to analyse the top ten URLs returned by our PageRank algorithm (see Table 2.1). One can observe that, while naturally correlated with the in-degree ranking, it differs from it significantly (compare Table 2.1 and Table 2.2).

In terms of relevance, the pages returned by our PageRank appear to be well chosen overall (homepage in first place, "index" or "site map" type pages), with the notable exception of two pages:

`http://www.inria.fr/rapportsactivite/RA94/RA94.kw.html` and

`http://www.inria.fr/rapportsactivite/RA94/RA94.pers.html`.

Upon verification, and as one might expect, these two pages turn out to be the two main nodes of a near-sink, namely `rapportsactivite/RA94/`. These two pages,

| URL (http://www.inria.fr/...) | Local PR | Google PR | In-deg |
|---|---|---|---|
| index.fr.html | 25, 3 | 9/10 | 608 |
| rapportsactivite/RA94/RA94.kw.html | 18, 7 | 6/10 | 327 |
| actualites/index.fr.html | 18, 6 | 8/10 | 367 |
| fonctions/plan.fr.html | 18, 4 | 8/10 | 297 |
| valorisation/index.fr.html | 18, 2 | 8/10 | 302 |
| travailler/index.fr.html | 18, 2 | 8/10 | 312 |
| recherche/index.fr.html | 18, 2 | 8/10 | 297 |
| publications/index.fr.html | 17, 9 | 8/10 | 294 |
| inria/index.fr.html | 17, 9 | 8/10 | 229 |
| rapportsactivite/RA94/RA94.pers.html | 17, 6 | 6/10 | 320 |

Table 2.1: The top ten URLs of www.inria.fr according to a local PageRank. Comparison with Google.

having both a high in-degree and being located in a near-sink, appear very difficult to filter out using only a local PageRank.

**Comparison with Google** Google assigns a ranking of 9/10 to the INRIA homepage and 8/10 to the other top ten pages of the local PageRank, with the exception of

`http://www.inria.fr/rapportsactivite/RA94/RA94.kw.html` and

`http://www.inria.fr/rapportsactivite/RA94/RA94.pers.html`,

which receive a score of 6/10. Two main observations:

- The two `RA94` pages had a local PageRank nearly equal to that of the other pages, with the exception of the homepage. Google's global PageRank managed to isolate them. A plausible explanation is the likely existence of numerous links

| URL (http://www.inria.fr/...) | In-deg |
|---|---|
| index.fr.html | 608 |
| index.en.html | 391 |
| actualites/index.fr.html | 367 |
| rapportsactivite/RA94/RA94.kw.html | 327 |
| rapportsactivite/RA94/RA94.pers.html | 320 |
| travailler/index.fr.html | 312 |
| valorisation/index.fr.html | 302 |
| fonctions/recherche.fr.html | 299 |
| fonctions/annuaire.fr.html | 297 |
| fonctions/plan.fr.html | 297 |

Table 2.2: The ten URLs with the highest in-degree

from external pages to the "index" type pages, whereas it is very likely that very few external pages point to `RA94`.

- The score of 6/10 assigned to

  `http://www.inria.fr/rapportsactivite/RA94/RA94.kw.html` and

  `http://www.inria.fr/rapportsactivite/RA94/RA94.pers.html`.

  remains high, certainly higher than one would wish. Many homepages of websites considered more noteworthy do not achieve this score.

# Appendix C
# Persistence and diffusion of files in peer-to-peer networks

The inability to finish downloading files is a frequent situation in peer-to-peer file-sharing networks. While this problem sometimes seems inherently linked to the nature of the network, it appears that even *intelligent* download networks can end up in a situation where all parts of the file are available except one.

We propose a simple and scalable file-sharing model that can apply to all block-based downloading protocols, such as BitTorrent. Simulations show that the *missing block* case can occur even for popular files, and provide some theoretical leads.

These new results enable a different approach to download problems in file-sharing networks, and new strategies are proposed.

## 1 Introduction

A P2P file-sharing network is an interface that permits the exchange of data between users arriving and departing independently. P2P issues can be classified in three categories:

- dynamical management of subjacent overlay networks [Har+03, Rat+01, Sto+01, ZKJ01];
- publication and search of shared content [CS02, KRR01];
- downloading protocols.

The last domain, downloading protocols, seems less studied than the two others. However, countless people have downloading issues every day. In this paper we focus on the "unfinished download" case.

For example, imagine you want to download your favorite linux distribution. For more efficiency, you use three of the most popular file-sharing softwares, KaZaA, MLDonkey MlDonkey [LP03] and BitTorrent [Coh03]. Downloads start, no problem to be seen, you leave for a week. When you come back, none of your downloads is finished, downloading is null, all you got are those messages:

**KaZaA**

```
415312kb/714204kb downloaded;
    more sources needed
```

**eDonkey**

```
64/65 parts downloaded;
    last seen complete: a long time ago
```

**Bittorrent**

```
99,7% downloaded; connected to 0 seeds;
    also seeing 0.997 distributed copies
```

The KaZaA message is not so surprising knowing that KaZaA peers only upload finished downloads. As long as there exists a user sharing the whole file, downloads go on[1], but once this (these) user(s) quit(s), all is over. The cases of eDonkey and Bittorrent are more interesting. To allow partially downloaded content to be shared, these protocols broke files into smaller blocks. Experience shows it is not just bad luck if the download stagnates at the last block.

In the next section, we introduce the approach and the different assumptions used in our model. Section 3 shows interesting results coming from simulations of the model. In Section 4 we present some stability results for simulations of Section 3. This highlights a frequent issue in real file-sharing networks, called *missing block* issue. Section 5 compares the different upload strategies and gives characteristics of safe states, where the data can potentially survive forever…

# 2 Model

The whole point of this article is to study the sharing of a single file in a totally connected overlay network, which we call *torrent*. We suppose the implicite existence of a server that organizes the users but does not possess the file itself. This fits well Bittorrent protocol, as users must connect to a so-called *tracker* to participate. *Peers* are users that want to download the file, and that can potentially share partial content. *Seeds* are users that share the whole file. To resume, a torrent is made up of $S$ seeds and $P$ peers trying to get a file split in $K$ blocks.

Study of such strategies is often complex due to subjacent prisoner's dilemma. The problem is indeed to minimize the downloading time for each user and to maximize the probability that the file stays in the network even for low request frequencies. The former is an individual optimization but the latter benefits to the whole community, and both optimizations are made with detriment of the other. To simplify the problem we will make the following assumptions:

[1]Note that as we will see, downloads in KaZaA are less efficient than in the two others which allow partially downloaded content to be shared.

- Most models are download-oriented: the peers try to download blocks they need from peers or seeds they know. We choose an upload approach, where peers and seeds decide which block they upload and to whom it is uploaded. Even if it does not exactly reflect the reality, we think it makes strategy clearer without altering practical activity.
- Everyone that can upload a block to somebody will do so. Once again, reality is more complex ([Coh03] uses a choking algorithm to stimulate the exchanges), but we have a good approximation. Peers and seeds often choose a maximum upload bandwidth and a maximum number of connections and stay stuck to these maxima. As we show in Section 4, this leads to an interesting result, called "torpor" where upload can sometimes severely injure a torrent.

# 3 First simulations

In this section, we suppose $S$ and $P$ are fixed. It can be interpreted as the worst case of peer behavior: a given number of philanthropic seeds are opposed to greedy peers that leave as soon as their download is finished and are instantly replaced by other greedy peers waiting in a queue. We want to study how that sort of torrent depends on the seeds. In other word we want to investigate the chance that the networks keeps distributed copies of the file even if no seed is present.

## 3.1 Strategies

The state of a torrent at a given moment can be represented by a logical $T = \left(t_{p,k}\right)_{1\leq p\leq P, 1\leq k\leq K}$ matrix where:

$$t_{p,k} = \begin{cases} 1 \text{ if user } p \text{ possesses the block } k \\ 0 \text{ otherwise} \end{cases} \tag{C.1}$$

Anytime a block upload is finished, according to our assumptions, the uploader must choose another couple (peer, block) and begin another upload. Strategies in our model resume in strategies in the choice method. We propose the following strategies:

**Globally random strategy** In GRS, each uploader chooses a couple $(p, k)$ at random.

**Block then peer decomposition** The uploader chooses the block it will upload, then the peer that will receive it.

**Peer then block decomposition** Inverse as above.

Decompositions strategies vary with the sub-choice method. We propose two basic methods: uniformly random choice, that is self-explicit, and positive discrimination choice, where you choose the rarest block or the poorest peer in term of download

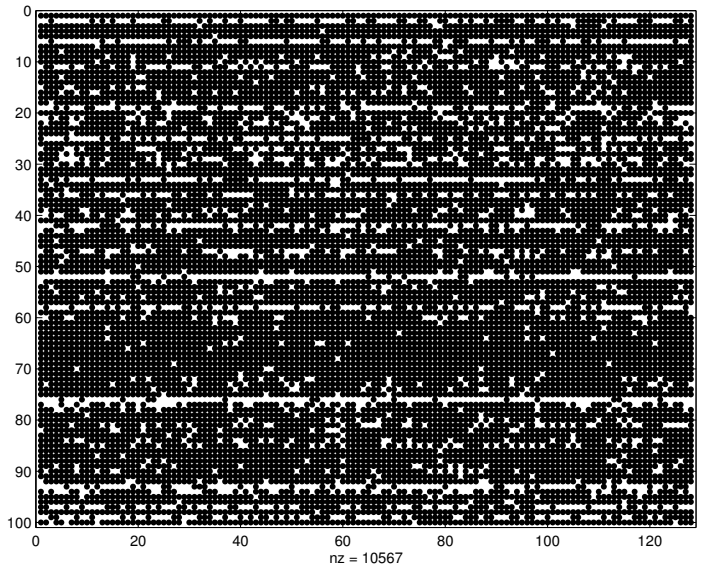



Figure 3.1: Stable state in a globally random upload strategy

progress[2]. For simplicity, we call decomposition strategies by initials. For example, BRPD strategy means a **B**lock is **R**andomly chosen, then the **P**eer is selected using positive **D**iscrimination.

## 3.2 Results

We choose $S = 1$ and $P = 100$ and $K = 120$. Everyone has the same upload bandwidth, so we can say the time to upload one block is the time unit. The ratio download/upload $r$ is infinite[3]: the number of blocks a peer can get in a time unit is not bounded.

Simulations using the GRS starting with $P$ empty peers (the deployment phase) tend towards two different stationary states, that we will call *safe* state and *torpor* state. Both states are roughly equiprobable. In safe state, seeds are unnecessary and the peers suffice themselves for themselves to keep the torrent alive. In torpor, there is a block possessed only by the seed (most of the time no peer possesses it), all the peers are waiting forever for the missing block and contaminating the newcomers. If the seed quits in a torpor state, the torrent dies. Figure 3.1 shows the matrix $T$ in a typical stable state and Figure 3.2 shows a typical torpor state (lines represent peers; line 0 stands for the seed).

Other strategies converge towards either a stable state or a torpor state:

- BRPR and PRBR converge towards a torpor state.
- other strategies tends towards a safe state.

[2]We remark that a decomposition strategy that uses random in both choices is not equivalent to the globally random strategy.

[3]Actually, for most people using ADSL, this ratio is 4 or 8. Later, we will give results for such ratios.

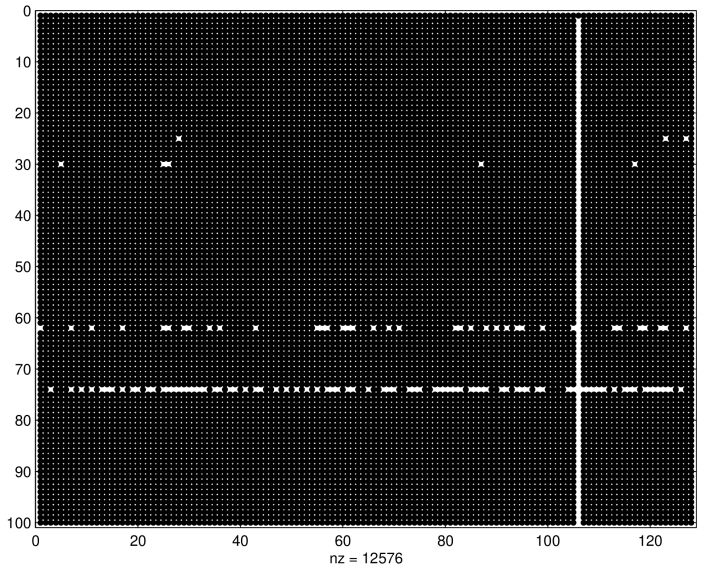



Figure 3.2: “torpor” state in a globally random upload strategy

We note that all the safe states are not identical. Parameters like the average block density or the download speed can vary, as discussed in Section 5.

# 4 “Torpor” characteristics

Torpor is a dangerous state where the torrent can not live without seed. This is why we want to deepen the whereabouts of this state.

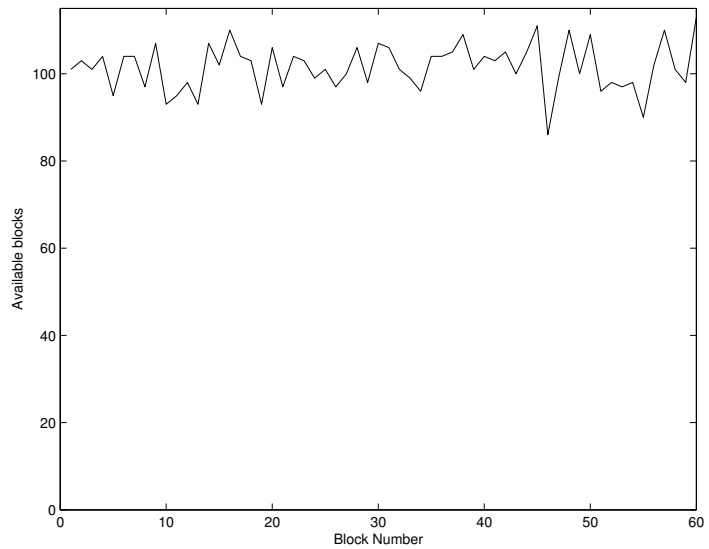



Figure 3.3: Population of each block in a stable state

## 4.1 *Torpor* emergence

The appearance of torpor in the deployment of a torrent is intuitively easy to understand. All safe states imply all the blocks having on average the same density (as shown by Figure 3.3). On the contrary, torpor means having a very rare block and the others almost totally spread. In the growing phase, the obligation of upload leads to a geometric progression of every block uploaded by the seed(s). This leads to strong irregularities of the block distribution in the earlier times. If this irregularities are not correctly smoothed by the subjacent upload strategy, the death of the torrent can occur during the earlier cycles. This is why BRPR and PRBR strategies always lead to torpor and positive discriminant strategies always lead to safe states: the first ones do nothing about the block distribution while the last ones are all about equity. We can then wonder why a safe state can be reached using GRS. Intuitively, the answer is that GRS is partially positive discriminant: the probability for a block $k$ to be downloaded is basically proportional to the number of peers needing this file, and the probability for a peer $p$ to receive a block is roughly proportional to the number of blocks it needs.

## 4.2 *Torpor* robustness

In *torpor* state, we call "patients" peers having all the blocks but the missing one. Because of the contagion of the patients, a torpor can be irremediable even using the smartest strategy, assuming peers that are able to upload do so. For example, if $S = 1$ and $r = +\infty$, a *torpor* will be stable as long as $P \geq K$. The time to "heal" a patient is then greater or equal than the time to contaminate a newcomer. This result spreads with $r$ and $S$ unspecified, and a sufficient (but not necessary) condition for stability, independent from the strategy, is:

$$P \geq S + S(K-1)\left(1 + \frac{1}{r}\right) \tag{C.2}$$

*Proof*: A torpor is necessary stable when the contaminating power of patients is greater than the "healing" power of seeds. The number of newcomers patients can contaminate in a time unit is $\frac{P'}{K-1}$, where $P'$ is the number of patients. Seeds can heal at most $S$ patients per time unit. Healed patients (newcomers) need $\frac{K-1}{r} + 1$ time units to be recontaminated.

As long as $\frac{P'}{K-1} \geq S$, the torpor is stable (note that $P'$ can vary at each time unit). The healing bound implies $P - P' \leq S\left(\frac{K-1}{r} + 1\right)$. These two inequalities lead to Equation (C.2). ■

With the assumption *everyone that can upload a block to somebody will do so*, patients will always perform good strategies to contaminate other peers. On the other hand, the healing strategy of seeds must be highly precise to be efficient (each seed must heal a patient with each time unit). Thus we can say that Equation (C.2) is a precise stability bound for strategies using positive discrimination first (whether it is on the

peers or the blocks), while torpor stability in random oriented strategies is much more important.

## 4.3 The missing block in real networks

Although our model is upload-oriented, it captures a phenomena that occurs in real peer-to-peer sharing networks. For now, we can only give intuitive reasons for that. In eDonkey networks, users trying to get a file are waiting in a queue, and once their turn comes, they are granted one (or more) blocks. As far as we know, the choice of the block is random (except possibly for the first and the last block, for previewing purpose). The queue system is *a priori* independent from the number of blocks users possess (although it may be possible there is a slightly negative discrimination due to the fact that "old" peers are more likely to have at the same time many blocks and good positions in queues). Thus we would say eDonkey transferts can be seen as a RPRB strategy. Knowing how this strategy is sensible to torpor phenomenons, we may have an interpretation of torpor in eDonkey.

For Bittorrent networks, apparition of torpor seems more surprising, as the strategy is globally a block-oriented positive discrimination. Then again, we can only rely on empiric observations to suppose torpor can happen when $P$ tends toward 0 (after a first rush, the torrent becomes less attractive). In future work, we will introduce a flexible processus for arriving of newcomers and try to verify this hypothesis. However, equation Equation (C.2) shows that when a torpor occurs, healing can be fastidious or even impossible. A frequent strategy for the original seed of a torrent (the user that first offered the file to be shared) when downloads are stopping is to re-seed, that is to reintegrate the torrent the time for new seeds to appear, then to leave. The problem is real patients are often almost as miserly as the patients of our model. That means they stay in the torrent a few moments after the download is complete, then leave. Then we can imagine the following situation: after a torpor, the original seed decides to reintegrate the torrent. Rapidly, many patients become "seeds", so the original seed quits believing the torrent is alright. But if the torpor is not completely healed, the odds are high for the remaining patients to contaminate the torrent again once the "seeds" are gone.

**Importance of the last block**

We can wonder if it is that important if there is a block missing. With Error-Correcting Codes (ECC), it is possible to share files that can be completed without all the blocks. But ECC alone can not solve the problem for long: knowing a missing block is acceptable, peers will start to behave consequently. So ECC allow a torrent to function in torpor. We just consider a virtual torrent with $K - 1$ blocks. And what happen if this virtual torrent goes in torpor?

The conclusion of this section is that the missing block is a real issue in P2P networks today, and that a real optimization of the transferts strategies can be beneficial.

# 5 Efficiency of upload strategies

We now want to compare the different strategies seen in Section 3.1 on other domains than stability, such as average download speed, original diffusion, density, and robustness to *very greedy peers*.

## 5.1 Average download speed

The global download speed is bounded by the sum of upload bandwidths. For a $(S, P)$ torrent with uniform upload bandwidth $U$, the average download speed can not be greater than $\overline{D}_{\max} = \frac{S+P}{P}U$. Simulations show that whenever a safe state is reached, average download speed tends towards this limit.

If the torrent goes in torpor, average download speed can be lessened: it is a good approximation to say that only seeds can trigger the end of a download. Thus if there is one seed and if the theoretical minimum average time between 2 finished download is inferior to a time unit, peers are going to stay idle (without uploading) part of the time. More precisely, with $S$ seeds, the average download speed is bounded by $\min\left(\frac{S \cdot K}{P}\overline{D}_{\max}, \overline{D}_{\max}\right)$.

## 5.2 Speed of Deployment

The deployment is the very dangerous phase of a torrent. If the original seed leaves before it ends, the game is over. Moreover, the original seed often wants to minimize its upload time. For both reason, deployment must be fast. The minimum time for deployment is the time for the original seed to upload each block one time, that is $K$ time units. It is achieved if a positive discrimination on blocks is used.

**Linear strategy**
Sharing a file linearly (from the first to the last block) like KaZaA protocol is not a good idea from a torpor point of view. Even if peers do not quit as soon as they get their last block, the last blocks of the file are very likely to miss sooner or later. The quality of deployment in a linear strategy is also far from optimal, whatever the peer strategy is:

- If the original seed allow every peers to download its block, deployment will take $P \cdot K$ time unit (assuming all peers are treated equally).
- The original seed can restrain the peers allowed to download from it to a subset of size $Q < P$, so time for deployment will be $Q \cdot K$ time units.
- In KaZaA networks where you cannot share a file until you completely possess it, achieving the minimum deployment times implies to upload to only one peer. If the original seed quits after that, you come back to the original state, except the original seed is know a new seed whose reliability is unknown.

**Random block strategies**

Strategies where blocks are chosen at random are not really better. In fact, the more $K$ is important, the more the original seed has to upload, as shown by the following theorem:

**Theorem C.1**: Given a set $F = \{1, ..., k\}$, the mean time to choose one block of each in a random choice repeated in $F$ is equivalent to $k \ln(k)$.

*Proof*: let $T_k$ be the stopping time (in number of draws) when in the sequence of number chosen each symbol $1, 2, ..., k$ is at least one time. Then

$$\mathbb{E}(T_k) = \sum_{\omega} T_k(\omega) \tag{C.3}$$

Let's look the evolution of the process:

1. a first number is chosen (with probability 1)
2. with probability $(1/k)^{i_1} \frac{k-1}{k}$ it is chosen $i_1$ times before other one is taken.
3. with probability $(2/k)^{i_2} \frac{k-2}{k}$ one of both is chosen $i_2$ times before an other one is taken.
4. ...
5. with probability $(1-1/k)^{i_{k-1}} \frac{1}{k}$ one of the $k-1$ last ones is chosen $i_{k-1}$ times before the last one is taken.

for one of those trajectories $T_k = k + i_1 + i_2 + ... + i_{k-1}$. Thus

$$\begin{aligned}
\mathbb{E}(T_k) &= \sum_{i_1 \dots i_{k-1}} \left( (k + i_1 + i_2 + ... + i_{k-1}) \cdot (1/k)^{i_1} \frac{k-1}{k} ... (1-1/k)^{i_{k-1}} \frac{1}{k} \right) \\
&= \frac{k-1}{k} ... \frac{1}{k} \sum_{i_1 \dots i_{k-2}} (1/k)^{i_1} ... (1-2/k)^{i_{k-2}} \cdot \\
&\quad \left( \sum_{i_{k-1}} (k + i_1 + i_2 + ... + i_{k-1}) (1-1/k)^{i_{k-1}} \right) \\
&= \frac{(k-1)!}{k^{k-1}} \sum_{i_1 \dots i_{k-2}} (1/k)^{i_1} ... (1-2/k)^{i_{k-2}} \cdot \\
&\quad \left( (k - 1 + i_1 + i_2 + ... + i_{k-2}) \frac{1}{1-(1-1/k)} + \sum_{i_{k-1}} (1 + i_{k-1}) (1-1/k)^{i_{k-1}} \right)
\end{aligned} \tag{C.4}$$

Notice that $\sum_{i \in \mathbb{N}} (i+1) X^i = \frac{d}{dX} \left( \frac{1}{1-X} \right)(X) = \left( \frac{1}{1-x} \right)^2$ it also true in $\mathbb{R}$ for $X = x$ with $|x| < 1$. Thus

$$\begin{aligned}\mathbb{E}(T_k) &= 1 + \frac{1}{1-1/k} + ... + \frac{1}{1-(1-1/k)} \\ &= k(1/k + 1/(k-1) + ... + 1/1) \\ &= k(\ln(k) + \gamma + \varepsilon(k))\end{aligned} \tag{C.5}$$

with $\varepsilon(k)$ tending towards 0 and $\gamma$ is the Euler constant. ■

## 5.3 Density

In Bittorrent, the number of distributed copies seen and the average download are often considered as indicators of wealth. As said before, each block in a safe state has roughly the same density, so the two parameters are basically proportional (see Figure 3.3). From this point of view, peer-oriented discriminant strategies seem to be better, as most of the earlier blocks are possessed very fast. But as we see in next paragraph, having the biggest density is not always a good thing.

## 5.4 Robustness to *very greedy peers*

*Very greedy peers* (VGP) are peers wanting to get their file as soon as possible and that are likely to trick the torrent to do so.

**VGP and positive discrimination**

Positive peer discrimination helps new users to get blocks faster. It increases the probability that the uploaded blocks can be uploaded many times. By construction this leads to stable states where the repartition of blocks has a small standard deviation (in PDB(DR) it is close to 0). Intensity of users arrivals is maximal, because users can upload almost every time. The problem is that the peers wait a very long time at the end to get their last block (see Figure 3.4 for download speed during download progress), and that it is profitable to cheat by announcing that you have to download a number of blocks larger than your real one. With $K = 60$ and $P = 120$ (parameters of Figure 3.4), declaring 2 missing files instead of 1 increases the average download speed by 86, thus increasing the average effective download speed of the last block by 43.

**VGP robust strategies**

Let $f(k)$ be the average speed download of peers possessing $k$ blocks. A strategy is robust regarding VGP if asking for blocks you already have is not benefic.

A random-block strategy is VGP-robust if $f$ has the following property:

$$\frac{f(k)}{K-k} \text{ increases} \tag{C.6}$$

*Proof*: A peer possessing $k_1$ blocks and declaring $k_2 < k_1$ blocks will have a average download speed of $f(k_2)$ with a proportion $\frac{K-k_1}{K-k_2}$ of useful blocks if the

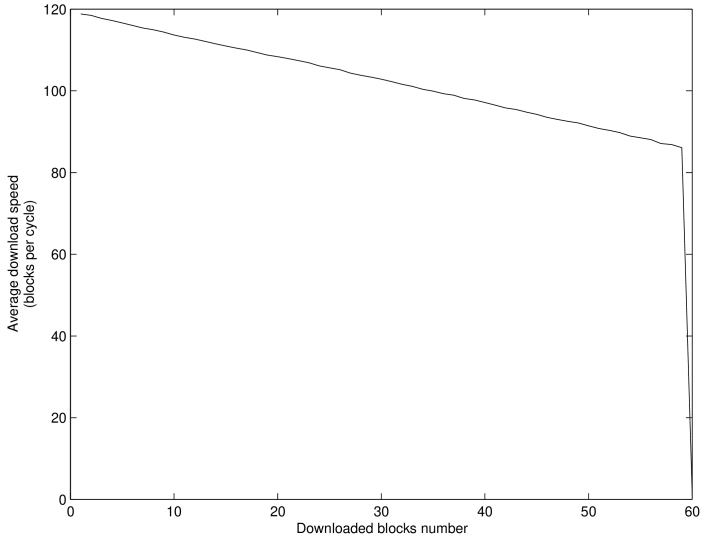



Figure 3.4: Average download speed in the download progress in a positive peer discrimination strategy

block strategy is random. Thus lying is not profitable if $f(k_1) > \frac{K-k_1}{K-k_2} f(k_2)$. This is the case if the function $k \to \frac{f(k)}{K-k}$ increases. ■

We remark that this result stay true in a discriminant-block strategy if the state is safe, because all the block have roughly the same density. In a torpor state with seeds, this result is false: lying allow the VGP to get the missing block faster than expected, with heavy cost for the torrent (one of the few who possesses the missing block leaves sooner).

We verify easily that positive peer-discrimination does not fulfill Equation (C.6) for $k = K - 1$. Contrary, in GRS, the download speed (in a safe state) is basically an affine function of $k$ (see Figure 3.5) that fulfill Equation (C.6). This linearity can be intuitively explained if we arbitrary change the number of blocks $k$ a given peer $p_0$ possesses (all other parameters being unchanged). The average speed download for $p_0$ will be obviously proportional to the $K - k$ missing blocks. Unlinearities noted in Figure 3.5 comes from random fluctuations and retroactions that have been neglected.

The conclusion of the study of VGP is that positive peer-oriented strategies are not robust, and that it is better to sacrifice some density and homogeneity (see Figure 3.6 for a typical download distribution in GRS) to gain robustness. Especially when that does not harm the global download speed, as we have seen.

# 6 Future work

We think it is possible to study the stability of GRS and the probability of torpor during deployment using a mean field theory. That should give a base to purpose new strategies. We also want to improve our model using variable upload bandwidth

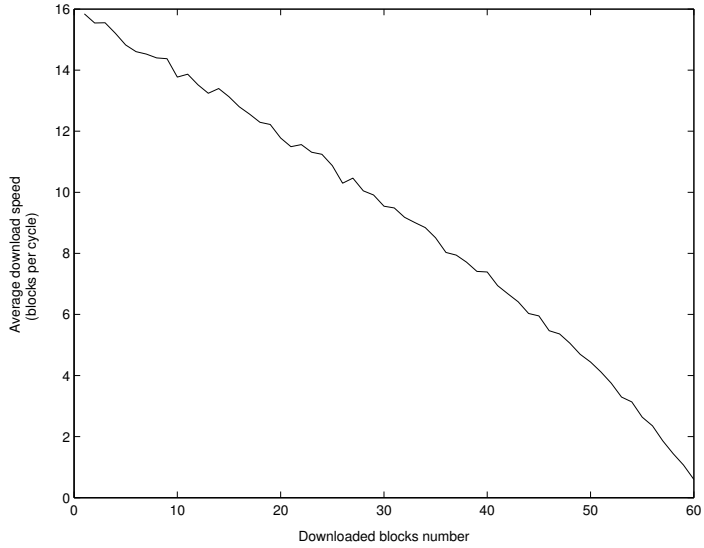



Figure 3.5: Average download speed in the download progress: GRS-stable case

and departing and arriving of peers ([SGG02] gives precious information about real distributions). This will allow to make our model more accurate and to verify some hypothesis such as the apparition of torpor during the decline of the torrent and the difficulty to reseed. Lastly, we think about analyzing logs from eDonkey servers and Bittorrent trackers to validate our model.

# 7 Conclusion

We gave a rather precise and intuitive survey of missing block issues. We showed that a block-oriented discriminant strategy is more efficient than a random strategy, so we could say Bittorrent behaves better that eDonkey from this point of view.

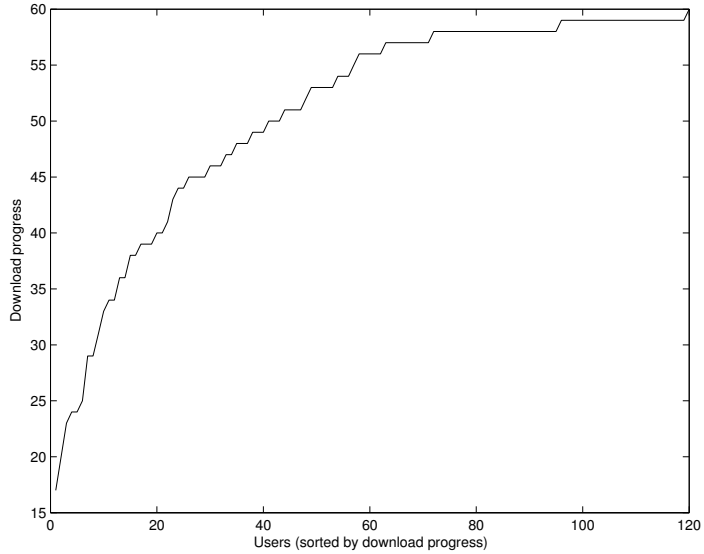



Figure 3.6: Distribution of download progress among users in GRS

Lastly we saw that peer-oriented discrimination is less important, it can even be damageable regarding to bad social behavior. These results could be used to enhance existing protocols. For example, a tracker aware of torpor issue could anticipate it and reseeding could be more effective.

# Bibliography

[Mat01] F. Mathieu, "Structure supposée du graphe du Web," in *Première journée Graphes Dynamiques et Graphes du Web*, 2001.

[MV02] F. Mathieu and L. Viennot, "Structure intrinsèque du web," RR-4663, 2002. [Online]. Available: http://www.inria.fr/rrrt/rr-4663.html

[MV03] F. Mathieu and L. Viennot, "Local Structure in the Web," in *12th international conference on the World Wide Web*, 2003a.

[BM03] M. Bouklit and F. Mathieu, "Effet de la touche Back dans un modèle de surfeur aléatoire : application à PageRank," in *Journées Francophones de la Toile 2003*, 2003.

[MB04] F. Mathieu and M. Bouklit, "The effect of the back button in a random walk: application for pagerank," in *Alternate track papers & posters of the 13th international conference on World Wide Web*, ACM Press, 2004, pp. 370–371.

[MV03] F. Mathieu and L. Viennot, "Aspects locaux de l'importance globale des pages web," in *5es rencontres francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL'2003)*, 2003b.

[MV04] F. Mathieu and L. Viennot, "Local Aspects of the Global Ranking of Web Pages," RR-5192, 2004. [Online]. Available: http://www.inria.fr/rrrt/rr-5192.html

[MR04] F. Mathieu and J. Reynier, "File Sharing in P2P: Missing Block Paradigm and Upload Strategies," RR-5193, 2004. [Online]. Available: http://www.inria.fr/rrrt/rr-5193.html

[Gen04] Genitrix, "Une histoire de réseaux."

[Ber00] M. K. Bergman, "The Deep Web: Surfacing Hidden Value." [Online]. Available: http://www.brightplanet.com/pdf/deepwebwhitepaper.pdf

[Sul00] D. Sullivan, "Invisible Web Gets Deeper." [Online]. Available: http://searchenginewatch.com/sereport/article.php/2162871

[SP01] C. Sherman and G. Price, *The Invisible Web: Uncovering Information Sources Search Engines Can't See*. Independent Publishers Group, 2001.

[GLM02] J.-L. Guillaume, M. Latapy, and F. Mathieu, "Tout le Web accessible en quelques clics." [Online]. Available: http://www.liafa.jussieu.fr/~fmathieu/arbre.php

[BB98] K. Bharat and A. Broder, "A technique for measuring the relative size and overlap of public Web search engines," in *Proceedings of the seventh international conference on World Wide Web 7*, Elsevier Science Publishers B. V., 1998, pp. 379–388.

[Hen+99] M. R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork, "Measuring index quality using random walks on the Web," *Computer Networks*, vol. 31, no. 11–16, pp. 1291–1303, 1999.

[Dah00] M. Dahn, "Counting angels on a pinhead: Critically interpreting web size estimates," *Online*, vol. 24, no. 1, pp. 35–40, 2000.

[BMM94] T. Berners, L. Masinter, and M. Mc Cahill, "RFC-1738 : Uniform Resource Locators (URL)." [Online]. Available: http://www.ietf.org/rfc/rfc1738.txt

[Bro+00] A. Broder *et al.*, "Graph structure in the web," in *Proc. 9th International World Wide Web Conference*, 2000, pp. 309–320.

[Fie+99] R. Fielding *et al.*, "Hypertext Transfer Protocol — HTTP/1.1."

[Kam+03] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub, "Extrapolation methods for accelerating PageRank computations," in *Proceedings of the Twelfth International World Wide Web Conference*, 2003a.

[Mur00] B. H. Murray, "Sizing the Internet," 2000.

[LG98] S. Lawrence and C. L. Giles, "Searching the World Wide Web," *Science*, vol. 280, pp. 98–100, 1998.

[LG99] S. Lawrence and C. L. Giles, "Accessibility of Information on the Web," *Nature*, vol. 400, no. 6740, pp. 107–109, 1999.

[EMT04] N. Eiron, K. S. McCurley, and J. A. Tomlin, "Ranking the Web Frontier," in *Proc. 13th International World Wide Web Conference*, 2004, pp. 309–318.

[APC03] S. Abiteboul, M. Preda, and G. Cobena, "Adaptive on-line page importance computation," in *Proc. 12th International World Wide Web Conference*, 2003, pp. 280–290.

[CGP98] J. Cho, H. García-Molina, and L. Page, "Efficient crawling through URL ordering," *Computer Networks and ISDN Systems*, vol. 30, no. 1–7, pp. 161–172, 1998.

[Cra04] T. C. Craven, "Google, the Source," *College & Research Libraries*, vol. 65, no. 4, pp. 306–312, 2004.

[Sea] SearchEngineShowdown, "http://www.searchengineshowdown.com/."

[CF02] C. Cooper and A. Frieze, "Crawling on web graphs," in *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, ACM Press, 2002, pp. 419–427.

[LM04] A. N. Langville and C. D. Meyer, "Deeper inside pagerank," 2004.

[RG03] S. Raghavan and H. Garcia-Molina, "Representing web graphs."

[Ara+01] A. Arasu, J. Novak, A. Tomkins, and J. Tomlin, "PageRank Computation and the Structure of the Web: Experiments and Algorithms."

[Kam+03] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub, "Exploiting the Block Structure of the Web for Computing PageRank."

[GL02] J.-L. Guillaume and M. Latapy, "Le Graphe du Web," *Tangente*, vol. Hors-Série, no. 12, pp. 12–15, 2002.

[Dil+01] S. Dill, S. R. Kumar, K. S. McCurley, S. Rajagopalan, D. Sivakumar, and A. Tomkins, "Self-similarity in the Web," *The VLDB Journal*, pp. 69–78, 2001.

[Bra97] D. Brake, "Lost in Cyberspace," *New Scientist*, 1997.

[Ran+01] K. Randall, R. Stata, R. Wickremesinghe, and J. Wiener, "The link database: Fast access to graphs of the Web," *Research Report 175, Compaq Systems Research Center, Palo Alto, CA*, 2001.

[GLV02] J. Guillaume, M. Latapy, and L. Viennot, "Efficient and simple encodings for the web graph," in *Proceedings of the 11th international conference on the World Wide Web*, 2002.

[Kle98] J. M. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," in *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998, pp. 668–677.

[BBM03] T. Bennouas, M. Bouklit, and F. de Montgolfier, "Un modèle gravitationnel du Web," in *Actes de ALGOTEL03 5ème Rencontres Francophones sur les aspects Algorithmiques des Télécommunication*, 2003.

[O'N+02] E. O'Neill, B. Lavoie, R. Bennett, A. Dyer, and S. Worthington, "Web Characterization Project." [Online]. Available: http://wcp.oclc.org/

[Net04] Netcraft, "http://www.netcraft.com/."

[Who04] WhoIs, "http://www.whois.net/."

[Bri+98] S. Brin, R. Motwani, L. Page, and T. Winograd, "What can you do with a Web in your Pocket?," *Data Engineering Bulletin*, vol. 21, no. 2, pp. 37–47, 1998.

[Ada99] L. A. Adamic, "The Small World Web," in *Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries*, Springer-Verlag, 1999, pp. 443–452.

[Li+00] W.-S. Li, O. Kolak, Q. Vu, and H. Takano, "Defining logical domains in a web site," in *Proceedings of the eleventh ACM on Hypertext and hypermedia*, ACM Press, 2000, pp. 123–132.

[FCE95] Q. Feng, R. F. Cohen, and P. Eades, "How to draw a planar clustered graph," in *Computing and Combinatorics*, Springer-Verlag, 1995, pp. 21–31.

[Bri03] M. Brinkmeier, "Communities in Graphs," in *Proceedings of FCT 2003*, 2003.

[Sol01] Soleil Levant, "http://hipercom.inria.fr/soleil/."

[She88] O. B. Sheynin, "A A. Markov's Work on Probability," *Archive for History of Exact Science*, vol. 39, pp. 337–377, 1988.

[Sal96] L. Saloff-Coste, "Lectures on finite Markov Chains," in *Lecture Notes on Probability Theory and Statistics*, Springer Verlag, 1996, pp. 301–413.

[Ste96] I. Stewart, "Monopoly revisited," *Scientific American*, vol. 175, pp. 116–119, 1996.

[Col] T. Collins, "Probabilities in the Game of Monopoly." [Online]. Available: http://www.tkcs-collins.com/truman/monopoly/monopoly.shtml

[Gau96] P. Gaucher, "Le Monopoly pour les Nuls." [Online]. Available: http://www.pps.jussieu.fr/~gaucher/monopoly/monopoly.html

[Goo98] Google, "http://www.google.com/."

[Bou01] M. Bouklit, "Quelques méthodes de classement des moteurs de recherche basée sur la structure du graphe du web," 2001.

[BJ02] M. Bouklit and A. Jean-Marie, "Une analyse de PageRank, une mesure de popularité des pages web," in *Actes ALGOTEL'02*, 2002.

[BGS02] M. Bianchini, M. Gori, and F. Scarselli, "PageRank: A Circuital Analysis."

[BGS03] M. Bianchini, M. Gori, and F. Scarselli, "Inside pagerank," *ACM Transactions on Internet Technology*, 2003.

[Sal89] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[YL95] B. Yuwono and D. L. Lee, "Search and Ranking Algorithms for Locating Resources on the World Wide Web," in *Proceedings of the 12th International Conference on Data Engineering*, IEEE Computer Society, 1995, pp. 164–171.

[Pri63] D. J. d. S. Price, *Little Science, Big Science*. Columbia University Press, 1963.

[Pag+98] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," 1998. [Online]. Available: http://google.stanford.edu/~backrub/pageranksub.ps

[Gof71] W. Goffman, "A Mathematical Method for Analyzing the Growth of a Scientific Discipline," *J. ACM*, vol. 18, no. 2, pp. 173–185, 1971.

[CP95] L. D. Catledge and J. E. Pitkow, "Characterizing Browsing Strategies in the World Wide Web," *Computer Networks and ISDN Systems*, vol. 27, no. 6, pp. 1065–1073, 1995.

[TG97] L. Tauscher and S. Greenberg, "How people revisit web pages: empirical findings and implications for the design of history systems," *International Journal of Human Computer Studies*, vol. 47, no. 1, pp. 97–137, 1997.

[CM01] A. Cockburn and B. McKenzie, "What Do Web Users Do? An Empirical Analysis of Web Use," *International Journal of Human Computer Studies*, vol. 54, pp. 903–922, 2001.

[WM04] L. Wang and C. Meinel, "Behaviour Recovery and Complicated Pattern Definition in Web Usage Mining," in *ICWE*, LNCS, 2004, pp. 531–544.

[Mil+04] N. Milic-Frayling, R. Jones, K. Rodden, G. Smyth, A. Blackwell, and R. Sommerer, "Smartback: supporting users in back navigation," in *Proceedings of the 13th international conference on World Wide Web*, ACM Press, 2004, pp. 63–71.

[Ste94] W. J. Stewart, *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[BP98] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1–7, pp. 107–117, 1998.

[Dur04] F. Durand, "Expériences sur la décomposition de PageRank," 2004.

[Tom03] J. A. Tomlin, "A new Paradigm for Ranking Pages on the World Wide Web," in *Proceedings of the Twelfth International World Wide Web Conference*, 2003.

[BV] P. Boldi and S. Vigna, "Projet WebGraph." [Online]. Available: http://vigna.dsi.unimi.it/papers.php#BoVWFI

[Fag+00] R. Fagin *et al.*, "Random walks with 'back buttons' (extended abstract)," 2000.

[Syd04] M. Sydow, "Random surfer with back step," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, ACM Press, 2004, pp. 352–353.

[Hav99] T. Haveliwala, "Efficient computation of PageRank," 1999.

[AJB99] R. Albert, H. Jeong, and A.-L. Barabasi, "Diameter of the World Wide Web," *Nature*, vol. 401, pp. 130–131, 1999.

[BA99] A.-L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, 1999.

[AJB00] R. Albert, H. Jeong, and A.-L. Barabasi, "Scale-Free Characteristics of Random Networks: The Topology of the World-Wide Web," *Physica A*, vol. 281, pp. 69–77, 2000.

[Kle00] J. M. Kleinberg, "The Small-World Phenomenon: An Algorithmic Perspective," in *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, ACM Press, 2000, pp. 163–170.

[PLH01] A. R. Puniyani, R. M. Lukose, and B. A. Huberman, "Intentional Walks on Scale Free Small Worlds," 2001.

[Sto+01] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the 2001 ACM SIGCOMM conference*, 2001, pp. 149–160.

[ZKJ01] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph, "Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing," UCB/CSD-1–1141, 2001.

[Rat+01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addressable Network," in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, ACM Press, 2001, pp. 161–172.

[Har+03] N. J. A. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman, "SkipNet: A Scalable Overlay Network with Practical Locality Properties," in *Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems*, 2003.

[KRR01] J. Kangasharju, J. Roberts, and K. W. Ross, "Object Replication Strategies in Content Distribution Networks," *Computer Communications*, vol. 25, no. 4, pp. 376–383, 2001.

[CS02] E. Cohen and S. Shenker, "Replication strategies in unstructured peer-to-peer networks."

[LP03] F. Le Fessant and S. Patarin, "MLdonkey, a Multi-Network Peer-to-Peer File-Sharing Program," RR-4797, 2003.

[Coh03] B. Cohen, "Incentives Build Robustness in BitTorrent."

[SGG02] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," in *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, 2002.

## Résumé

L'application des mesures d'importance de type PageRank aux graphes du Web est le sujet de cette thèse, qui est divisée en deux parties. La première introduit une famille particulière de grands graphes, les graphes du Web. Elle commence par définir la notion de Web indexable, puis donne quelques considérations sur les tailles des portions de Web effectivement indexées. Pour finir, elle donne et utilise quelques constatations sur les structures que l'on peut observer sur les graphes induits par ces portions de Web.

Ensuite, la seconde partie étudie en profondeur les mesures d'importance *à la* PageRank. Après un rappel sur la théorie des chaînes de Markov est présentée une classification originale des algorithmes de PageRank, qui part du modèle le plus simple jusqu'à prendre en compte toutes les spécificités liées aux graphes du Web. Enfin, de nouveaux algorithmes sont proposés. L'algorithme *BackRank* utilise un modèle alternatif de parcours du graphe du Web pour un calcul de PageRank plus rapide. La structure fortement clusterisée des graphes du Web permet quant à elle de décomposer le PageRank sur les sites Web, ce qui est réalisé par les algorithmes *FlowRank* et *BlowRank*.



## Mots clés

Graphes — Web — Arbre de décomposition — Matrices sous-stochastiques
PageRank — Surfeur aléatoire — Algorithmes — Retour arrière – Flots d'importance



## Abstract

The purpose of this thesis is to apply PageRank-like measures to Web graphs. The first part introduces the Web graphs. First we define the notion of indexable Web, then we give an insight on how big the effective crawls really are. Finally, we notice and use some of the structures that exist on the portions of the Web known as Web graphs.

Then, the second part study deeply the PageRank algorithms. After a remainder on Markov chains theory is given an original classification of PageRank algorithms. From a basic model, we incorporate all the specificities needed to cope with real Web graphs. Lastly, new algorithms are proposed. BackRank uses an alternative random surfer modeling leading to a faster computation. The highly clustered structure of Web graphs allows a PageRank decomposition according to Web sites, and is the reason for introducing the algorithms FlowRank and BlowRank.



## Keywords

Graphs — WWW — Decomposition tree — Substochastic matrix
PageRank — Random walk — Algorithms — Backoff process — Importance flows



Lirmm — 161 rue Ada — 34392 Montpellier Cedex 5 — France