Graphes du Web, Mesures d'importance à la PageRank

Fabien Mathieu

ACADÉMIE DE MONTPELLIER

UNIVERSITÉ **M**ONTPELLIER **II**

— **S**CIENCES ET **T**ECHNIQUE DU **L**ANGUEDOC —

THÈSE

présentée à l'Université des Sciences et Techniques du Languedoc
pour obtenir le diplôme de doctorat

SPÉCIALITÉ	: **INFORMATIQUE**
Formation Doctorale	: **Informatique**
École Doctorale	: **Information, Structures, Systèmes**

GRAPHES DU WEB
MESURES D'IMPORTANCE À LA PAGERANK

par

Fabien MATHIEU

Soutenue le 8 décembre 2004 devant le jury composé de :

M. Serge ABITEBOUL, Directeur de recherche, INRIA . . . . . rapporteur
M. Vincent BLONDEL, Professeur, UCL (Belgique) . . . . . invité
M. Pierre FRAIGNIAUD, Directeur de recherche, CNRS . . . . . rapporteur
M. Michel HABIB, Professeur, LIRMM, Montpellier . . . . . directeur de thèse
M. Alain JEAN-MARIE, Directeur de recherche, INRIA . . . . . président
M. Laurent VIENNOT, Chargé de recherche, INRIA . . . . . co-encadrant

Table des matières

Remerciements

Alors que je m'apprête à transférer ce mémoire sur le serveur de l'université de Montpellier II, le figeant ainsi pour les siècles et les siècles à venir, force m'est de constater qu'il n'aurait pû être sans le concours de nombreuses personnes, auxquelles je tiens à rendre un hommage mérité.

Tout d'abord, même si ce n'est guère protocolaire, merci à celui sans qui Fabien Mathieu ne serait guère plus que la somme de deux prénoms. Merci Papa, c'est grâce à toi que cette thèse existe, et il est naturel que je te la dédie.

Merci aussi à celui que j'ai connu alors que je terminais plus ou moins mon DEA, et que la thèse était encore pour moi quelque chose d'étrange et venu d'ailleurs. Sans Laurent Viennot pour me remettre les pieds sur terre, je serais peut-être encore en train de chercher la transformée de Fourier du graphe du Web. Je ne sais toujours pas comment il a fait pour gérer un thésard comme moi, têtu et vivant à l'heure de New-York, mais c'est grâce à sa patience que je peux aujourd'hui écrire ces lignes, et je lui en serai éternellement reconnaissant. Merci Laurent — le chercheur —, et merci Laurent — l'ami.

Je remercie bien sûr aussi Michel Habib, qui a accepté de me diriger en toute connaissance de cause et m'a accordé toute sa confiance. Son soutien sans faille m'a permis de pratiquer l'ubiquité méditerrano-parisienne pendant ces quelques années. Puisse-t-il à son tour la pratiquer avec la même liberté que j'ai pu avoir grâce à lui.

Merci aux rapporteurs de cette thèse de m'avoir accordé leur temps et leur attention, et d'avoir réussi à supporter les délais que je leur ai imposé : merci à Serge Abiteboul, qui a su me révéler les points faibles de mon mémoire et m'a permis d'en corriger quelques-uns ; merci à Pierre Fraigniaud d'avoir risqué jusqu'à sa moralité pour les besoins de la science.

Je remercie également les autres membres du jury, Alain Jean-Marie, dont les corrections m'ont été bien utiles, et Vincent Blondel, dont l'intérêt pour mon mémoire m'a particulièrement touché. Merci aussi à François Baccelli, qui a hélas dû s'excuser au dernier moment.

Merci à tous les chercheurs, webmestres ou simples internautes pour toutes les données qu'ils ont bien voulu me communiquer par mail quand j'en avais besoin. La liste est longue, mais je n'en oublie pas pour autant leur contribution, bien au contraire.

Évidemment, rien n'aurait été possible sans le soutien de ces innombrables travailleurs de l'ombre que j'ai eu l'occasion de réduire en esclavage, particulièrement au moment de la ré-

daction. Fidèle garde rapprochée, ils ont relu sans relâche les prototypes successifs jusqu'à la présente version. Si il y a moins de vingt fotes par page, c'est grâce à eux.

L'équipe montpelliéraine d'abord. Fabien (l'autre), dont le code source du mémoire m'a permis de passer de 0 à 100 pages en moins de temps qu'il n'en faut pour écrire « copier/coller » avec un marteau-piqueur. Force m'est de constater que ces remerciements n'auraient pû être sans son concours, et je tiens à lui rendre un hommage mérité. Mohamed, de son côté, a été là pour moi à chaque que moi je n'y étais pas, et je n'oublie pas non plus que sans ses talents de programmeur, je n'aurais jamais passé la barre des 100 millions de pages dans les délais impartis.

L'équipe parisienne ensuite. Julien, qui n'a pas son pareil pour faire une mousse au chocolat au sel de rinçage pour 20 personnes, et qui a eu la gentillesse, avec Anamaria, de me prêter un camp de base pour l'été 2004. En parlant de camp de base, un grand merci à Fanfan. Son lit est trop dur, mais on y fait de beaux rêves. Et je n'oublie bien sûr pas Mathilde, qui a su surmonter son handicap (être blonde et littéraire) et me donner de précieux conseils stylistiques.

J'ai également une pensée toute particulière pour Danielle Croisy, de l'INRIA, bien que je ne m'explique pas comment elle a fait pour supporter tous ces ordres de mission en retard.

Merci à tous les autres dont je n'ai pas oublié le soutien. *Thank you very many* à Mickey et Thibaut d'être passés faire un tour au LIRMM en ce mois de décembre 2004. Merci à toute la Halle Gorithme, avec ses lapins, ses crawlers, ses pizzas et ses huîtres.

Enfin, un grand merci à ma sœur Nathalie, ma tante Marie-Paule, ma nièce Marion et mon neveu Quentin. À force de bourlinguer aux quatre coins de l'Hexagone et d'ailleurs, on finit par apprécier à sa juste valeur de savoir qu'il existe un lieu où l'on se sent chez soi.

MERCI À TOUS
ET BONNE LECTURE !

Introduction

La méthode scientifique, qui choisit, explique, et ordonne, admet les limites qui lui sont imposées par le fait que l'emploi de la méthode transforme son objet, et que par conséquent la méthode ne peut plus se séparer de son objet.

HEISENBERG

Les moteurs de recherche aujourd'hui

DEPUIS quelques années, l'Internet, et le Web en particulier, ont subi de profondes transformations liées à de multiples changements d'ordres de grandeur. Toujours plus de données sur toujours plus de machines sont accessibles à toujours plus d'internautes. L'économie électronique s'est également développée, et ce qui n'était hier pour les entreprises commerciales qu'une simple vitrine expérimentale est devenu, souvent au prix d'essais malheureux, un secteur économique à part entière. Ces mutations ont généré de nouveaux comportements aussi bien au niveau des internautes que des administrateurs de site.

Du côté des internautes, le problème qui est apparu consiste à se repérer au milieu des milliards de pages disponibles. L'utilisateur *lambda* veut avoir les moyens d'accéder à toutes les ressources offertes par le Web, et les méthodes habituelles (navigation par hyperliens à partir d'un portail, connaissance de la bonne adresse par des moyens extra-Web) ne suffisent plus.

Du côté des sites se pose le problème symétrique de la visibilité. Un site, si bien conçu soit-il, n'a de valeur que s'il est fréquenté, tel l'arbre qui tombe dans la forêt. Ce problème de la visibilité a de multiples facettes, qui sont les mêmes que pour la visibilité dans le monde réel. La visibilité du site personnel de Monsieur Durand est pour lui l'assurance de pouvoir être connu ou contacté par les personnes qui le désirent. La visibilité d'un site commercial représente de l'argent. Être plus visible que ses concurrents permet d'acquérir une nouvelle clientèle à leurs dépens, ce qui, dans un marché encore limité mais en pleine expansion, est une condition quasi-nécessaire de

survie.

Si les deux problèmes de l'accessibilité et de la visibilité sont symétriques, ils ne font pas forcément bon ménage. Si un internaute veut manger des pommes, sa principale préoccupation va être de trouver des pages Web qui parlent de pommes, voire qui en vendent. De son côté, le site d'un amateur de poires va vouloir se faire connaître de l'internaute mangeur de pommes, si possible avant qu'il n'ait trouvé un site de pommes, pour essayer de le faire changer d'avis et de le convertir à la poire.

Les moteurs de recherche ont pour but premier d'assurer l'accessibilité d'un maximum de pages Web aux internautes. Concrètement, parmi le plus grand choix possible de pages, le moteur doit renvoyer les pages répondant aux besoins de l'internaute, ce besoin étant exprimé au travers d'une *requête*. Mais ils sont de fait devenus aussi le principal média de visibilité pour les sites.

Ce placement stratégique, à la croisée de l'internaute et du site, fait du moteur de recherche la pièce maîtresse du Web actuel, et il n'est pas étonnant de voir que la société *Google*, qui possède — à ce jour — le quasi monopole du secteur des moteurs de recherche, est maintenant cotée en bourse.

Connaître le Web

Une des premières qualités pour un moteur de recherche est d'avoir une base de données conséquente, autant d'un point de vue quantitatif que d'un point de vue qualitatif. Dans la partie I nous nous proposons de mettre en évidence la problématique que posent ces bases de données constituées par les moteurs de recherche.

Nous commençons tout d'abord par tenter de définir le Web, ce qui, comme nous allons le voir lors du chapitre 1, n'est pas aussi facile qu'on pourrait le penser de prime abord. Ceci fait, nous pourrons au cours du chapitre 2 étudier ces sous-ensembles du Web que sont les *crawls*. Les crawls sont naturellement munis d'une structure de graphe que nous détaillerons lors du chapitre 3. Nous essaierons en particulier de mettre en perspective le modèle du nœud papillon, dont les conclusions sont trop souvent mal interprétées, et nous montrerons l'existence d'une très forte organisation en sites des pages Web liée à l'arbre de décomposition des URLs. Nous verrons en particulier que la structure canonique des sites permet de calculer en temps linéaire une bonne approximation d'une certaine décomposition en sites d'un graphe du Web.

Trouver les bonnes pages

Plus que la taille de sa base de donnée, c'est la façon dont un moteur de recherche va s'en servir qui va déterminer son efficacité. Alors que pour une requête donnée, la base de données va souvent contenir quelques dizaines de milliers de pages susceptibles de fournir une réponse adaptée, l'internaute va rarement aller au delà des 10 ou 20 premières réponses renvoyées par le moteur. Il est donc nécessaire d'opérer un tri parmi les réponses possibles afin d'extraire, de manière automatique, les quelques pages les mieux adaptées à une requête donnée.

Les moteurs de recherche de première génération utilisaient des méthodes de pertinence lexicale et sémantique afin de réaliser ce tri : les premières pages renvoyées étaient celles qui, du point de vue de l'analyse sémantique, se rapprochaient le plus de la requête. Mais assez vite, le classement par pertinence a été dénaturé par le besoin de visibilité des sites. Beaucoup de sites se sont mis à étoffer le contenu sémantique de leurs pages à l'aide de techniques plus ou moins rusées et plus ou moins honnêtes, par exemple en surchargeant la page de texte blanc sur fond blanc. Très souvent, au lieu d'être renvoyé sur les pages les plus pertinentes, l'internaute se retrouvait alors sur des pages qui n'ont rien à voir avec sa requête, mais dont la volonté d'être visible est très grande.

Pour contrer ces techniques, de nouvelles méthodes de classement ont été introduites, dans le but d'inférer l'importance des pages Web à partir de la structure de graphe formée par les pages connues. C'est l'une de ces familles de méthodes de classement, les PageRanks, qui sera étudiée dans la partie II.

Le chapitre 4 rappellera quelques bases théoriques sur les chaînes de Markov, et développera plus particulièrement certains liens entre la théorie des graphes et celles des processus stochastiques nécessaires à une bonne compréhension du PageRank. Le chapitre 5 sera l'occasion de présenter un bestiaire quasi-exhaustif des algorithmes classiques de PageRank. Nous présenterons en particulier une vision du PageRank unifiée en terme d'interprétation stochastique, et mettrons en évidence les point communs, différences, avantages et inconvénients des différentes méthodes.

Enfin, au cours des chapitres 6 et 7, nous présenterons quelques algorithmes de PageRank originaux dont nous pensons qu'ils peuvent apporter des améliorations significatives aux algorithmes existants.

L'algorithme *BackRank*, conçu au départ pour modéliser l'utilisation de la touche *Back*, possède quelques effets secondaires intéressants, comme une convergence plus rapide que les algorithmes classiques et l'absence de problèmes liés à l'existence de pages sans lien.

FlowRank est un algorithme qui essaie de prendre en compte de manière fine le rôle des pages internes, des pages externes et du flot de *zap* dans un calcul exact du PageRank. *BlowRank* est une adaptation pratique de *FlowRank* inspirée d'un autre algorithme de calcul décomposé du PageRank.

Tous ces algorithmes sont autant d'outils potentiels ayant pour but de faciliter l'arbitrage entre accessibilité et visibilité, mais ils permettent aussi de se faire une meilleure idée des mécanismes qui entrent en jeu lorsque l'on veut modéliser un comportement stochastique sur le Web.

Publications

Voici la liste à ce jour des publications de mes travaux. [Mat01, MV02, MV03b] correspondent aux travaux sur les graphes du Web qui sont à la base de la partie I de ce mémoire. [BM03, MB04] sont quant à eux le point de départ du chapitre 6, tout comme [MV03a, MV04] pour le chapitre 7. Les travaux de classification des PageRanks du chapitre 5 ainsi que les analyses détaillées des algorithmes *BackRank* et *BlowRank* sont trop récents et n'ont pas encore

fait l'objet de publication. Enfin, [MR04] est un rapport de recherche sur une modélisation des problèmes de téléchargement dans les réseaux pair-à-pair fait conjointement avec Julien Reynier. C'est un sujet prometteur, mais qui a encore besoin de mûrir, c'est pourquoi je l'ai simplement mis en annexe (Annexe C page 159).

Tous ces documents sont téléchargeables sur
`http://www.lirmm.fr/~mathieu`

[Mat01] F. MATHIEU. Structure supposée du graphe du Web. Première journée Graphes Dynamiques et Graphes du Web, décembre 2001. http://www.liafa.jussieu.fr/~latapy/gdgw.html

[MV02] F. MATHIEU et L. VIENNOT. Structure intrinsèque du Web. Rapport Tech. RR-4663, INRIA, 2002. http://www.inria.fr/rrrt/rr-4663.html

[MV03a] 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. http://dept-info.labri.u-bordeaux.fr/algotel03/

[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. http://www.antsearch.univ-tours.fr/jft2003/

[MV03b] F. MATHIEU et L. VIENNOT. Local Structure in the Web. In *12th international conference on the Worl Wide Web*, 2003 http://www2003.org/cdrom/papers/poster/p102/p102-mathieu.htm

[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. http://www.www2004.org/proceedings/docs/2p370.pdf

[MV04] F. MATHIEU et L. VIENNOT. Local aspects of the Global Ranking of Web Pages. Rapport Tech. RR-5192, INRIA, 2004. http://www.inria.fr/rrrt/rr-5192.html

[MR04] F. MATHIEU et J. REYNIER. File Sharing in P2P : Missing Block Paradigm and Upload Strategies. Rapport Tech. RR-5193, INRIA, 2004. http://www.inria.fr/rrrt/rr-5193.html

Première partie

Structures du Web

Chapitre 1

Qu'est-ce que le Web ?

Je parlerai par exemple de l'Esprit et du « démiurge » en me gardant bien de définir trop précisément ce que j'entends par là : parce que je ne sais pas clairement... Dans cet état affreux de notre ignorance, faut-il vraiment définir si précisément ?

Rémy CHAUVIN*, la Biologie de l'Esprit*

1.1 Genèse du Web

Sources : *Une histoire de réseaux [Gen04]*.

VERS le début des années 1960, alors que dans le ciel passait une musique (un oiseau qu'on appelait Spoutnik), les États-Unis décidèrent de mettre au point une forme de réseau décentralisé capable de résister à une attaque nucléaire qui détruirait un ou plusieurs de ses centres nerveux. C'était l'année soixante-deux.

Quelques années plus tard, ce projet est devenu *ARPANET*. Nous sommes en 1969 et quatre universités américaines sont reliées par ce réseau d'un concept nouveau. À partir de ce moment, ce réseau n'a cessé de grossir et d'évoluer, jusqu'à devenir ce qu'on appelle aujourd'hui Internet[1], c'est-à-dire un formidable réseau de réseaux.

1. Le terme Internet a été introduit, semble-t-il, en 1974, par Vincent Cerf et Bob Kahn. Notons au passage que internet et Internet ne veulent pas dire la même chose ! L'un est un nom commun désignant une structure de méta-réseau, l'autre est un nom propre pour LE méta-réseau utilisant le protocole TCP/IP.

En 1972, la charte *AUP* (*A*cceptable *U*se *P*olicy) interdit à toute structure commerciale de se connecter sur le réseau.

En 1984, le cap de 1000 machines est franchi et le *CERN* (Centre européen de recherche nucléaire) rejoint Internet. Six ans plus tard, en 1990, alors que le nombre d'ordinateurs connectés atteint les 300 000, le plus grand site Internet au monde est celui du CERN, futur berceau du Web, un vaste ensemble mondial de documents dits hypertextes et hypermédias distribués sur Internet.

Cette même année 1990, l'AUP cesse d'exister, ouvrant la voie à ce qui deviendra quelques années plus tard la *Bulle Internet*.

En 1991, Tim Berners-Lee du CERN introduit le concept de World Wide Web (www), que l'on traduit parfois en français par la *toile*. Le World Wide Web est la partie d'Internet où la méthode de navigation est l'HyperTexte, et le protocole *http* (*H*yper*T*ext *T*ransfert *P*rotocol).

La philosophie d'*http* réside dans les liens dits *hypertextes* qui relient les pages entre elles et qui permettent de naviguer quand on les sélectionne. On parle du « Web » — avec une majuscule — même s'il s'agit en réalité du « World Wide Web » ou « W3 ».

1.2 Une définition du Web

À l'origine, comme nous venons de le voir, le *Web* était caractérisé à la fois par un protocole (le *http*) et un langage (le *html*, *H*yper*T*ext *M*arkup *L*anguage), le premier servant à diffuser des « pages » (des fichiers) écrites dans le second, et interprétées du côté client par des navigateurs Web (*Browsers*).

De nos jours, on peut se poser des questions sur la validité de cette double caractérisation :

- Le *html* est un langage facile à apprendre permettant d'écrire des documents structurés et reliés entre eux par des hyperliens. De fait, il n'est plus utilisé aujourd'hui exclusivement à travers *http*, et on le retrouve sur de nombreux autres supports (CD-ROM, DVD-ROM…) à des fins aussi nombreuses que variées (documentation, enseignement, encyclopédie, aide…).
- Afin de délivrer du contenu multimedia, le *http* est prévu pour servir tout type de fichier. Images, sons, vidéos, textes sous divers formats, archives, exécutables… sont ainsi accessibles grâce au protocole *http*, dans un esprit plus ou moins éloigné de la conception originelle de navigation hypertexte. Cette tendance au « tout-http » aboutit à des situations parfois paradoxales. Ainsi, pour transférer des fichiers (même gros), il existe une certaine tendance à délaisser le protocole adéquat (le *ftp*, *F*ile *T*ransfert *P*rotocol) au profit de *http*. Comme le montre le tableau 1.1[2], il semble que les fichiers traditionnellement transférés par *ftp* le soient maintenant par pair-à-pair, mais aussi par *http* (par exemple, la plupart des serveurs de téléchargements de *sourceforge.net* sont sous *http*).

2. Un grand merci à Philippe Olivier et Nabil Benameur, de pour m'avoir

- Certains documents qui ne sont pas des documents *html* permettent une navigation par hyperliens : documents propriétaires (Acrobat PDF, Flash…) ou nouveaux langages (*WML*, *XML*…).

Année	*http*	*ftp*	*p2p*
2001	13 %	10 %	35 %
2002	14 %	2 %	50 %
2004	20 %	*négligeable*	65 %

TAB. 1.1 – *É volution des trafics* http, ftp *et* p2p *(en pourcentage du volume)*

On entrevoit la difficulté de trouver une « bonne » définition du Web. Par un souci de simplicité plus que tout autre chose, tout au long de ce mémoire, nous continuerons à définir le Web selon sa double caractérisation initiale. Ainsi, nous appellerons *Web* l'ensemble des documents écrits en *html* et disponibles sur Internet par le protocole *http*. Nous sommes conscients du caractère extrêmement restrictif de cette définition, mais préférons travailler sur un ensemble bien défini et largement étudié plutôt que de rechercher une exhaustivité dont il n'est pas dit qu'elle puisse être atteinte.

1.3 Accessibilité du Web

À l'intérieur du Web que nous venons de définir se pose maintenant le problème de la visibilité et de l'accessibilité des pages. Que pouvons-voir du *Web* et comment y accéder ? Plusieurs structurations du Web basées sur ces questions de visibilité, d'accessibilité et d'indexabilité on été proposées.

1.3.1 Profondeurs du Web

Michael K. Bergman, en 2000, propose la métaphore de la profondeur pour distinguer les différents *Webs* [Ber00]. On distingue ainsi :

The Surface Web : la surface du *Web*, selon Bergman, est constituée de l'ensemble des pages statiques et publiquement disponibles.

The Deep Web : à l'inverse, le *Web* profond consiste en des sites web dynamiques et des bases de données accessibles par interface Web.

Cette vision du *Web* reste assez manichéenne. Danny Sullivan [Sul00] propose une troisième sorte de *Web*, *The Shallow Web*[3], constitué par exemple de pages dynamiques publiquement disponibles, comme celles de l'*I*nternet *M*ovies *D*ata*B*ase (`http://www.imdb.com`), ou encore celles du site `http://citeseer.ist.psu.edu/`.

3. Pour rester fidèle à l'analogie de Bergman, on pourrait traduire *Shallow Web* par *espace proche*. Je préfère le terme de *marécage*, qui traduit assez bien l'effet de cette zone du Web sur les crawlers.

1.3.2 Visibilité du Web

Chris Sherman et Gary Price proposent dans leur ouvrage *The Invisible Web* [SP01] une approche basé sur la visibilité par les grands moteurs de recherche. L'équivalent du *Surface Web* est chez Sherman et Price l'ensemble des pages indexées par les moteurs de recherche. Selon eux, le reste du Web se décompose alors en 4 catégories :

The Opaque Web : les pages qui pourraient être indexées par les moteurs mais qui ne le sont pas (limitation d'indexation du nombre de pages d'un site, fréquence d'indexation, liens absents vers des pages ne permettant donc pas un crawling).

The Private Web : les pages webs disponibles mais volontairement exclues par les webmasters (mot de passe, metatags ou fichiers dans la page pour que le robot du moteur ne l'indexe pas).

The Proprietary web : pages accessibles seulement pour les personnes autorisées (intranets, système d'identification...). Le robot ne peut donc pas y accéder.

The Truly Invisible Web : contenu qui ne peut être indexé pour des raisons techniques. Par exemple, format inconnu par le moteur, pages générées dynamiquement (incluant des caractères comme ? et &)...

1.3.3 Web accessible

Chacune des deux approches que nous venons de voir a ses avantages et ses inconvénients. S'il est vrai que la définition de Bergman est assez séduisante (d'un côté, un Web statique accessible par clics, de l'autre un Web dynamique atteignable uniquement par requêtes), elle ne décrit pas de manière réaliste le Web actuel. L'approche de Sherman et Price, à savoir une discrimination selon l'indexabilité par les moteurs de recherche, est plus souple, mais n'associe à mon avis pas toujours les bonnes causes aux bons effets[4].

Une troisième approche, utilisée par [BB98, HHMN99, Dah00], fait une sorte de synthèse des modèles cités à l'instant. C'est le modèle du Web *accessible* :

Définition 1

Le Web accessible est l'ensemble des pages Web susceptibles d'être pointées par un hyperlien. Plus précisément :

- *Ceci est équivalent à considérer comme faisant partie du Web accessible toute page accessible simplement en tapant la bonne adresse (également désignée par le terme URL, Uniform Ressource Locator [BMC94]) dans un navigateur.*
- *Les pages dynamiques ne possédant pas de variable cachée, c'est-à-dire dont les éventuelles variables sont transmises dans l'URL, font partie du Web accessible.*

4. Par exemple, l'ensemble des pages dynamiques accessibles depuis `http://www.liafa.jussieu.fr/~fmathieu/arbre.php`[GLM02] devrait logiquement appartenir au Web *opaque*, alors que la catégorisation de Sherman et Price semble les destiner à être complètement invisible...

Par convention, nous allons exclure certaines pages de notre définition du Web accessible :

- Les pages d'erreur renvoyées par un serveur (erreurs *4xx* par exemple) ;
- Les pages dont l'accès par des robots est interdit par un `robots.txt` ;
- Les pages protégées par login et/ou mot de passe (même si login et mot de passe peuvent être donnés dans l'URL).

Cette définition a bien évidemment elle aussi ses défauts. Par exemple, elle ne prend pas du tout en compte le problème des doublons (comment considérer deux pages au contenu strictement identique — par exemple deux URLs correspondant au même fichier physique ?), ni celui de la dynamicité temporelle des pages et de leur contenu (que signifie l'accessibilité de la page d'accueil d'un journal ? L'accessibilité d'une page qui renvoie l'heure et la date ?). En toute rigueur, nous devrions ainsi parler de Web accessible à un instant t, et accepter d'identifier une page à son URL malgré les redondances inévitables. Même ainsi, beaucoup de zones d'ombre persistent. Ainsi, certains administrateurs mal intentionnés ne renvoient pas le même contenu selon que le demandeur de la page soit humain ou non, afin de tromper les moteurs de recherche ; d'autres renvoient des pages adaptées au navigateur utilisé par l'internaute ; d'autres encore vérifient que l'internaute est bien passé par la page d'accueil avant de naviguer à l'intérieur du site, et le renvoie à la page d'accueil si ce n'est pas le cas ; enfin, à cause des problèmes de routage, il est tout à fait possible qu'à un instant t, un serveur soit tout à fait visible d'une adresse IP et inaccessible d'une autre. En plus de l'instant t doivent donc être aussi définis l'adresse d'où est émise la requête ainsi que toutes les informations transmises dans la requête *http*.

1.4 Intermezzo : la page qui pointait vers toutes les pages

Au cours d'une discussion avec Jean-Loup Guillaume et Matthieu Latapy, dans ce haut lieu de la recherche scientifique qu'est la salle de la machine à café, alors que nous polémiquions abondamment sur le modèle du *nœud papillon* proposé par Broder *et al.* [BKM+00][5], une idée farfelue est tombée du gobelet : et si nous mettions en place une page Web capable de pointer vers toutes les autres ?

Après avoir été d'abord écrite par Matthieu Latapy, dont j'ai repris le code, c'est une version un peu améliorée de la page initiale qui est maintenant disponible sur `http://www.liafa.jussieu.fr/~fmathieu/arbre.php` [GLM02].

Le principe de cette page est celui d'une machine à écrire, ou plutôt d'une machine à cliquer. Pour atteindre une page donnée, il suffit de cliquer ses lettres une à une, l'adresse obtenue étant gardée en mémoire grâce à une variable passée en paramètre dans l'URL. La page vérifie dynamiquement si cette adresse a un sens (si elle fait partie du Web indexable), et si c'est le cas, un hyperlien est inséré vers cette adresse[6]. La figure 1.1 est une capture d'écran de la *page qui pointe vers toutes les pages* à l'œuvre. Aux limitations de longueur d'URL près (et aux bugs

5. Voir la section 3.2 page 36.

6. La *page qui pointe vers toutes les pages* donne aussi le PageRank de Google (note de 0 à 10) quand il est

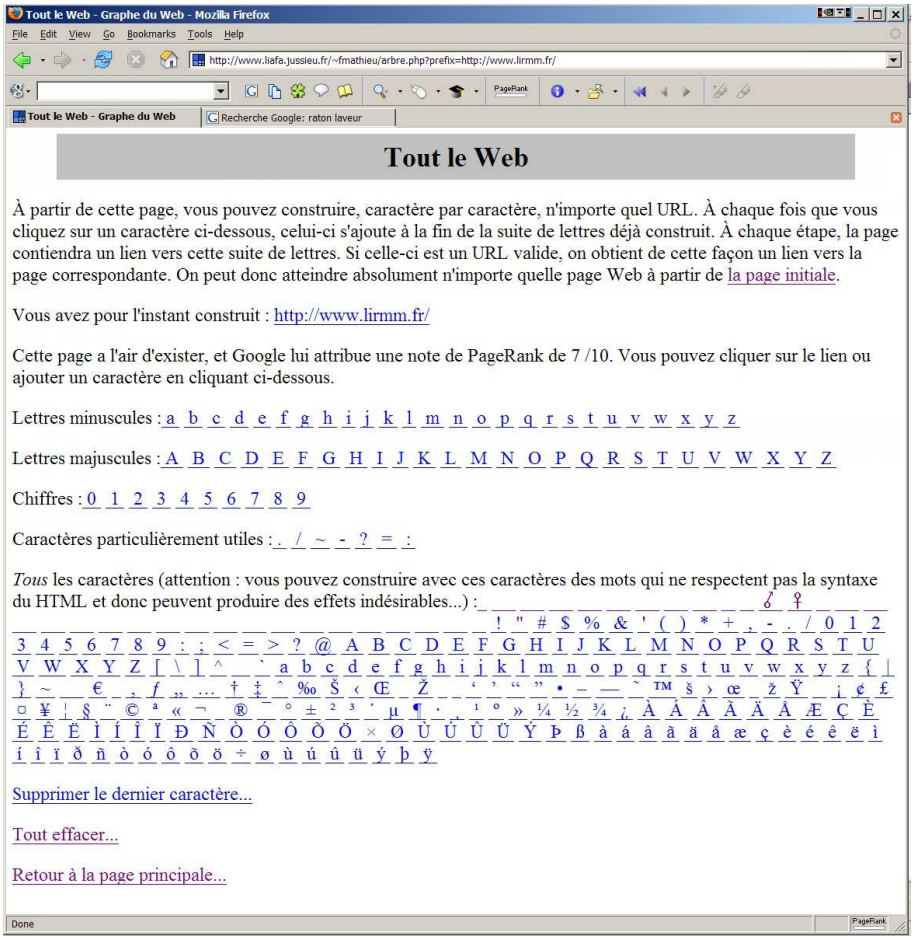


FIG. 1.1 – *Capture d'écran de* la page qui pointe vers toutes les pages *en pleine action*

de gestion des caractères d'échappement qui n'ont pas encore été corrigés), le rôle de la page, à savoir être reliée par hyperliens à virtuellement toute page du Web indexable, est atteint.

Cette page est avant tout ludique, mais elle permet de prendre ses distances quant à bon nombre d'idées reçues :

- Le but premier de cette page était de porter une petite pique contre l'interprétation qui est généralement faite du modèle du nœud papillon, et il a été je pense atteint.
- Elle permet de prendre avec un minimum de recul toutes les affirmations sur la structure du Web. Après tout, je peux affirmer connaître une partie du Web d'environ 256^{n-55} pages, à un facteur 10^{10} près, où n est le nombre maximal de caractères que peut contenir une URL[7]. Ce résultat, de loin supérieur à toutes les estimations de la taille du Web avancées

disponible. Elle ne fait pas encore le café, hélas.

7. Le protocole HTTP 1.1 ne spécifie pas de taille maximale d'URL (cf [FGM+99]). n est donc a priori aussi

jusqu'à présent (voir section 2.2), permet aussi d'affirmer quelques statistiques étonnantes :

- Le degré moyen du *Web que je connais* est de $257 - \epsilon$, ϵ étant une perturbation due aux *vraies* pages. De plus, contrairement à tout ce que l'on pensait, la distribution des degrés ne suit pas une loi de puissance, mais ressemble bien à un Dirac.
- Il existe une composante fortement connexe dans le *Web que je connais*, et toute page du *Web que je connais* appartient de manière quasi-certaine à cette composante.

Bien sûr, il ne faut pas prendre ces résultats pour argent comptant ! Je serais le premier à juger suspicieux un article affirmant que le *Web* fait plus d'un gogol de pages[8], ou que les distributions à queues lourdes n'existent pas. La *page qui pointe vers toutes les pages* n'est qu'une *idée du samedi soir*[9] qui a été mise en application et j'en suis parfaitement conscient. Mais elle a l'avantage de nous montrer en pleine lumière l'immensité du Web accessible (et en particulier l'impossibilité de l'indexer[10]) et de nous inciter à bien comprendre que tous les résultats signifiants que l'on peut avoir sur le *Web* portent en fait sur des crawls qui représentent une part infime du *Web indexable* en terme de nombre de pages.

Pour en conclure avec cet *intermezzo*, précisons qu'en terme de théorie de l'information, nous doutons que la *page qui pointe vers toutes les pages* et ses pages dynamiques valent plus que les quelques lignes de code qui se cachent derrière.

grand que l'on veut. Dans les faits, chaque serveur a une limite fixée, de l'ordre de quelques kilo-octets. D'ailleurs, il fut un temps où envoyer une URL de plus de 8 ko était une façon efficace de planter un serveur *IIS*...

8. Le gogol est un nombre valant 10^{100}. Il n'honore pas l'écrivain russe Nicolas Gogol (1809-1852), mais a été choisi en 1938 par le neveu du mathématicien américain Edward Kassner (1878-1955), Milton, alors âgé de 9 ans. Notons qu'en anglais, gogol se dit... googol ! Le nom du célèbre moteur de recherche est directement dérivé du nom inventé par Kassner, il existe d'ailleurs un contentieux entre Google et les héritiers de Kassner. Enfin, signalons que le nombre représenté par un 1 suivi d'un gogol de 0 s'appelle gogolplex.

9. J'emprunte ce concept d'*idée du samedi soir* à Michel de Pracontal. C'est « le genre d'idées dont les chercheurs discutent parfois pendant leurs heures de loisirs, sans trop les prendre au sérieux » (L'IMPOSTURE SCIENTIFIQUE EN DIX LEÇONS, leçon 7).

10. Pour fixer les idées, en faisant un parcours en largeur de la *page qui pointe vers toutes les pages* à raison de 10 pages par seconde, il faudrait environ 100 millions d'années pour arriver à écrire `http://free.fr`. Quant à l'adresse `http://www.lirmm.fr`, l'âge de l'univers est largement insuffisant pour qu'un robot puisse la trouver uniquement en parcourant la *page qui pointe vers toutes les pages*.

Chapitre 2

Chalutiers et tailles du Web

Le monde est de taille moyenne.
Michel HOUELLEBECQ*, Lanzarote*

La taille ne fait pas tout. La baleine est en voie d'extinction alors que la fourmi se porte bien.
Bill VAUGHAN

Il faut cacher la profondeur. Où ça ? À la surface.
Hugo von HOFMANNSTHAL

MAINTENANT que nous avons défini un cadre sur lequel travailler (le Web accessible), la question de savoir quels objets nous allons étudier se pose. D'un point vue mathématique, le Web accessible peut être infini dénombrable — selon la RFC 2616 [FGM+99] — ou fini — car il n'existe qu'un nombre fini de serveurs et que chacun possède une taille limite d'URL. Il répond en tout cas à une certaine définition physicienne de l'infini, qui est celle que nous utiliserons ici : *grand devant le nombre d'atomes de l'univers*. Le Web accessible contient en effet beaucoup de nids de pages quasi-infinis. La page qui pointe vers toutes les pages, bien sûr, mais aussi des serveurs mal configurés, ou même des sites commerciaux qui essaient, en créant une *infinité* de pages, d'augmenter leur PageRank (voir partie II section 7.5, page 136). Pour donner une exemple concret, lors de la présentation de leur article *Extrapolation methods for accelerating PageRank computations* [KHMG03b], les auteurs ont raconté comment une de leur expérience a été perturbée par un site allemand qui occupait 90 % des pages qu'ils avaient indexées. Vérification faite, le site en question était un site pornographique qui espérait grâce à une infinité de pages étroitement reliées entre elles avoir un bon classement dans les moteurs de recherche (principe des *Free For All*, aussi appelés *farm links* ou *pouponnières*).

En conclusion, il est impossible d'indexer *tout* le Web accessible, de même que parler de sa taille n'a plus de sens[1]. En l'absence d'accès physique aux serveurs, les seules données tangibles que nous ayons à notre disposition sont donc les *crawls*, c'est-à-dire les zones du Web effectivement découvertes, et éventuellement indexées, par des entreprises commerciales (moteurs de recherche) ou des institutions publiques. L'objet de ce chapitre sera de définir précisément les *crawls*, de donner des ordres de grandeurs, et de voir dans quelle mesure il est possible de comparer des *crawls* ou de mesurer une certaine qualité.

2.1 Les chalutiers du Web

Les *crawleurs*, ou *spiders*, ou *agents*, sont les chalutiers qui parcourent le Web afin de générer ces morceaux de Web appelés *crawls*. Il y a deux principaux types de crawleurs : les crawleurs statiques et les crawleurs dynamiques. Le principe de tous les crawleurs statiques est le même : à partir d'un ensemble de pages initial, ils analysent les hyperliens contenus dans ces pages, essaient de récupérer les pages pointées par ces hyperliens, et ainsi de suite, pour récupérer ainsi une partie sans cesse croissante des pages du Web accessible. Chaque page n'est *a priori* récupérée qu'une seule fois, et on arrête le procédé quand on estime le moment voulu (suffisamment de pages récupérées, croissance devenue négligeable…). Le crawl *statique* que l'on obtient alors est constitué de :

- un ensemble de pages connues, qui est la réunion de l'ensemble initial et de l'ensemble des pages découvertes ;
- un ensemble de pages indexées, sous-ensemble de celui des pages connues constitué des pages effectivement parcourues par le crawl ;
- un ensemble d'hyperliens issus des pages indexées et pointant vers les pages connues.

Un crawleur dynamique, lui, n'est pas prévu pour s'arrêter, mais pour parcourir perpétuellement le Web et revisiter périodiquement les pages qu'il a parcouru [APC03].

Ce que nous appelons crawleur est en fait l'ensemble des moyens logiciels et matériels permettant de réaliser un crawl. Qu'est-ce qui va faire qu'un crawleur va produire un certain crawl et pas un autre ?

- La date à laquelle le crawl démarre.
- L'ensemble des pages de départ.
- La stratégie d'exploration des nouvelles pages (dans quel ordre récupérer quelles pages ?).
- Les limitations techniques : bande passante, capacité de stockage et de traitement, temps disponible…

1. On remarquera que si, il y a quelques années, estimer la taille du Web était un sujet *à la mode* (voir [BB98, Mur00, LG98, LG99, Ber00]), plus personne ne s'y aventure depuis deux ans, et le point de vue évoqué ici semble aujourd'hui partagé (voir notamment [Dah00, EMT04]).

Ceci est valable pour tous les types de crawleurs, même si asymptotiquement, seules les stratégies d'exploration et les limitations techniques jouent un rôle important pour les crawleurs dynamiques.

2.1.1 Création de l'ensemble des pages initial

Il va de soi que le choix d'un bon ensemble de départ est fondamental pour l'efficacité d'un crawleur. Pour des raisons commerciales, il est difficile d'obtenir les ensembles de départ des grands moteurs de recherche. Beaucoup de gens s'accordent cependant pour estimer que les grands moteurs de recherche utilisent :

- un sous-ensemble *bien choisi* de l'ensemble des pages obtenues par les crawls précédents [CGMP98]. Les pages d'accueil des *annuaires*, en particulier, semblent être des candidats de choix pour faire partie de l'ensemble initiale [Cra] ;
- des nouvelles pages soumises par l'un des nombreux procédés de *référencement* existants (parfois payants) ;
- il est possible que soient également utilisées quelques techniques plus ou moins rusées pour découvrir des nouvelles pages, comme analyser les logs des serveurs Web, ou essayer de remonter l'arbre des URLs des pages « isolées » (rumeurs lues sur différentes pages du site `http://www.webrankinfo.com`).

Pour conclure avec le choix de l'ensemble initial, signalons que c'est cet ensemble qui détermine en grande partie la structure de nœud papillon d'un *graphe du Web* (voir section 3.2 page 36).

2.1.2 Stratégies d'exploration

Un crawleur idéal qui pourrait récupérer et traiter une infinité de pages par unité de temps n'aurait pas de souci pour obtenir tout le Web théoriquement atteignable par hyperliens à partie de l'ensemble initial. Toutes les méthodes de parcours exhaustif (en largeur ou en profondeur par exemple) aboutiraient à un même résultat optimal. Mais les crawlers réels sont limités par leur bande passante, leur capacité de traitement ainsi que le nombre de requêtes par serveur et par minute, qui doit tenir compte autant des règles de politesse que du risque d'être banni.

Toutes ces contraintes font que, à un instant donné, on peut considérer qu'il existe une limite finie au nombre de pages qu'un moteur de recherche précis peut indexer, et il semble que ce nombre ait toujours été, quelque soit le moteur de recherche considéré, assez inférieur aux estimations finies du Web Indexable [Ber00, LG98, LG99, HHMN99, BB98].

Chaque moteur possède donc une barrière physique au nombre de pages qu'il peut indexer. Par exemple, pour et d'après Google, cette barrière est d'environ 4 milliards de documents. La stratégie d'exploration (couplée avec le choix de l'ensemble initial) va décider quels seront ces 4 milliards de pages, et donc dans une large mesure quelle sera la qualité[2] d'un crawl. Un moteur

2. Au sens vague…

de recherche dont les agents se perdraient dans les pouponnières et autres *pages qui pointent vers toutes les pages* aurait sûrement peu de succès. C'est pourquoi certains moteurs, comme *AltaVista* et *Teoma*, évitent la pratique du *Deep Crawl*[3], alors que d'autres, comme Google, répertorient les pouponnières dans une liste noire.

La stratégie peut aussi modifier négativement la barrière d'un moteur de recherche, si elle ne cherche pas à optimiser les ressources disponibles. Par exemple, à cause de la limite du nombre de requêtes par site, la bande passante prise par l'exploration d'un serveur est souvent minime devant la bande passante disponible, d'où la quasi-nécessité d'explorer plusieurs serveurs en parallèle. Pour les mêmes raisons, il vaut mieux toujours avoir un serveur rapide parmi les serveurs que l'on est en train de questionner.

Toutes ces contraintes « physiques » prises en compte, on distingue principalement deux philosophies dans les stratégies d'exploration connues.

- Traditionnellement, les moteurs de recherche employaient semble-t-il une exploration de type *parcours en largueur*, les pages découvertes en premier étant explorées prioritairement (sous réserve des contraintes décrites ci-dessus) [BKM+00].
- Des méthodes d'exploration inspirées des marches aléatoires se développent également (voir [HHMN99] et, dans une certaine mesure, la stratégie *greedy* de [APC03]). Ces méthodes présentent l'avantage de récupérer plus facilement les pages importantes au sens du PageRank [HHMN99], voire d'estimer le PageRank de manière dynamique [APC03]. Les liens entre PageRank et marche aléatoire seront développés lors de la partie II.

2.2 Tailles et évolutions des crawls

Maintenant que le concept de crawl a été grossièrement expliqué, nous allons donner quelques statistiques sur les ordres de grandeur considérés lorsque l'on parle de crawl pour les principaux moteurs de recherches.

2.2.1 Selon les organisateurs

Ces données ont pour la plupart été obtenues sur le site `http://searchenginewatch.com/`.

La figure 2.1 présente les tailles revendiquées des cinq moteurs de recherche milliardaires à la date du 2 septembre 2003. Si on étudie l'évolution respective des différents moteurs de recherche, on peut distinguer plusieurs grandes périodes de bouleversement, qui correspondent à des périodes de compétition intense entre les moteurs. Ces *guerres du Crawl* sont représentées en détail figure 2.2. Chacune de ces guerres s'est soldée pour le ou les vainqueurs par le franchissement d'une barre psychologique.

Entre décembre 1997 et juin 1999 a lieu la première *guerre du Crawl*, à l'issue de laquelle AltaVista dépasse la barre des 150 millions, talonné de près par NorthernLight (figure 2.2(b)).

3. Le *Deep Crawl* consiste à essayer de récupérer le maximum de pages web d'un site donné.

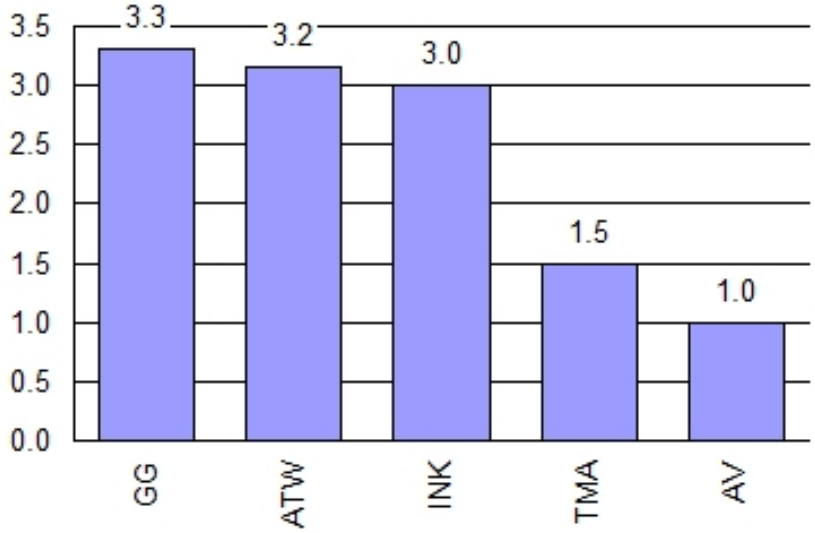


FIG. 2.1 – *Nombre de pages revendiquées par différents moteurs de recherche à la date du 2 septembre 2003*

La deuxième *guerre du crawl* (septembre 1999 – juin 2000) est marquée par un match serré entre Altavista et AllTheWeb, qui se conclut par la victoire écrasante de... Google, qui va se développer jusqu'à dépasser le milliard et demi de pages revendiquées, ce qui impose un nouveau standard (figure 2.2(c)).

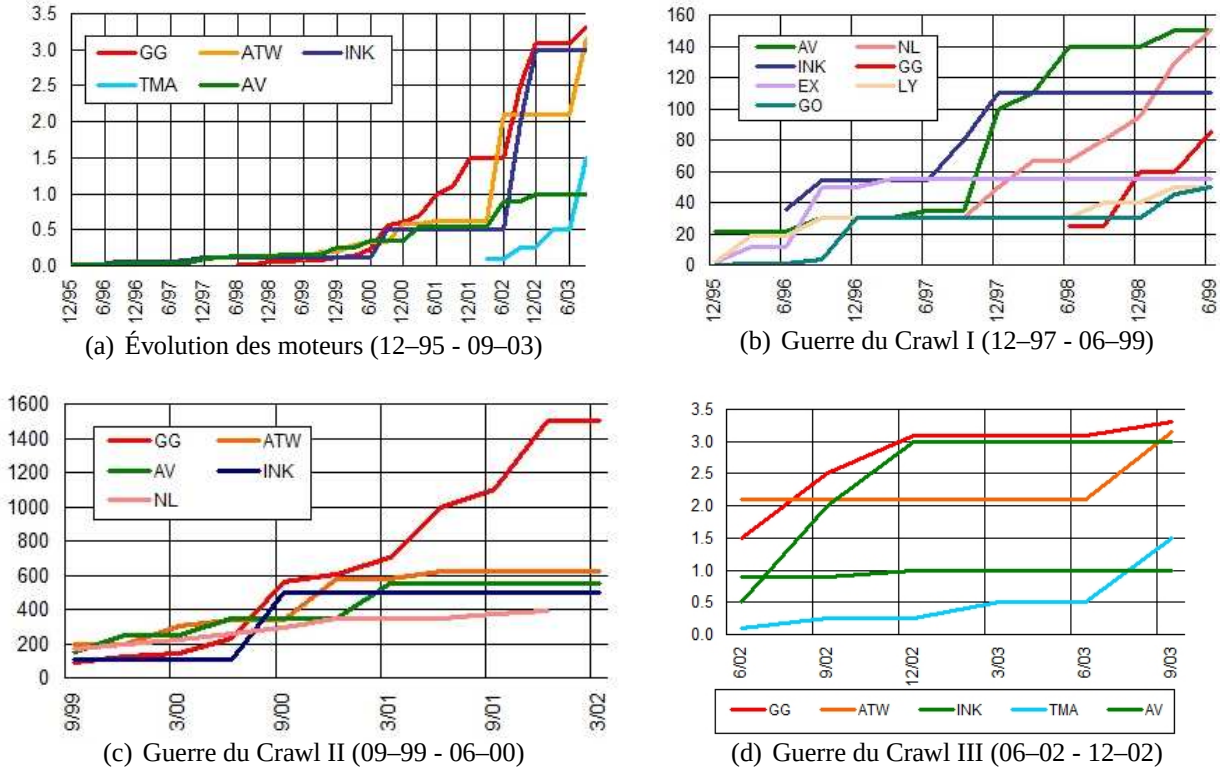
La troisième guerre (juin 2002 – décembre 2002) débute par un court passage d'AllTheWeb en tête de course. Google et Inktomi ripostent presque aussitôt, et en l'espace de six mois franchissent le cap des 3 milliards de pages web. Quelques mois plus tard, AllTheWeb se remet au diapason (figure 2.2(d)).

Actuellement (été 2004), Google affirme indexer plus de 4 milliards de pages, mais le champ de bataille s'est déplacé. La bataille entre Google et Yahoo (qui sous-traitait avec Google jusqu'en février 2004, et qui depuis utilise la base de donnée d'Inktomi) essaie de se faire plus subtile, le but étant d'avoir les résultats « les plus compréhensibles et les plus pertinents ». Yahoo préparerait d'ailleurs sa propre version du PageRank de Google, le WebRank.

2.2.2 Selon la police

Tous les chiffres que nous venons de voir sont ceux annoncés par les moteurs de recherche eux-mêmes. Comme nous sommes dans un contexte de concurrence commerciale, il est légitime de s'interroger sur la fiabilité de ces chiffres, avec le problème de définir une méthodologie adéquate.

Le site *SearchEngineShowdown* [Sea], administré par Greg Notess, propose une méthode d'estimation grossière de la taille effective des moteurs de recherche. Le principe est d'effectuer



(a) Évolution des moteurs (12–95 - 09–03)

(b) Guerre du Crawl I (12–97 - 06–99)

(c) Guerre du Crawl II (09–99 - 06–00)

(d) Guerre du Crawl III (06–02 - 12–02)

FIG. 2.2 – *Guerres du Crawl. Légende : AV=AltaVista, ATW=AllTheWeb, EX=Excite, GG=Google, GO=GO/Infoseek, INK=Inktomi, LY=Lycos, NL=Northern Light, TMA=Teoma. Unité : 1 million de pages (figure 2.2(b) et 2.2(c)) ou 1 milliard de pages (figure 2.2(a) et 2.2(d)).*

plusieurs requêtes parallèlement sur les différents moteurs de recherche que l'on veut comparer, et de postuler que le nombre de réponses renvoyées par chaque moteur est en première approximation proportionnel au nombres d'URLs connues par le moteur de recherche. Si l'on connaît la taille réelle d'un des moteurs, on peut alors par une simple règle de trois estimer la taille réelle des autres moteurs. Or, en décembre 2002, il était possible de connaître la taille exacte du moteur AllTheWeb simplement en soumettant la requête `url.all:http`[4].

La figure 2.3 compare les tailles revendiquées et estimées par la méthode de Greg Notess de plusieurs moteurs de recherche. On peut facilement distinguer trois catégories de moteur de recherche, qui montrent autant les éventuelles exagérations des moteurs que le biais introduit par les requêtes :

Les réalistes : *Google, AllTheWeb* et *WiseNut* ont des valeurs estimées et revendiquées extrê-

4. Merci à Greg Notess, webmaster du site SearchEngineShowdown de m'avoir communiqué cette méthode, même si elle ne marche plus aujourd'hui. La requête *url.all :http* ne marche en effet qu'avec les moteurs utilisant la technologie Fast, laquelle n'est plus utilisée par les grands moteurs depuis le 1^{er} avril 2004 (et ce n'est pas un canular...).

ment proches, et tendent à prouver que la méthode d'estimation présente une certaine fiabilité.

Les sites qui voulaient trois milliards : Les estimations des taille de HotBot et MSN Search sont en nette contradiction avec les valeurs revendiquées. Il s'agit de moteurs basés sur la technologie Inktomi. Selon Inktomi, tous les ordinateurs formant leur base de données ne sont pas accessibles en même temps, et donc on ne peut jamais accéder à la totalité de la base. De plus, il est possible que les partenaires d'Inktomi (HotBot et MSN Search dans le cas présent) n'utilisent qu'une partie de la base de données, pour des raisons pratiques ou commerciales. Pour Greg Notess, ces estimations reflètent donc la portion de la base de données réellement disponibles au moment où les requêtes ont été faites. Il est évident que le biais de la méthode de Notess nous empêche de faire des affirmations catégoriques, mais l'on ne peut s'empêcher de soupçonner le chiffre de trois milliards d'être une réponse publicitaire aux trois milliards de Google (rappel : nous sommes à la fin de la troisième *guerre du Crawl*).

Les modestes : AltaVista, Teoma, NLResearch et Gigablast semblent revendiquer beaucoup moins que leur taille réelle, ce qui ne semble guère logique. D'après Greg Notess, l'explication de ce paradoxe vient de la façon de comptabiliser les pages connues, mais non-indexées. En effet, les sites que nous avons appelés *réalistes* semblent revendiquer leurs URLs connues, indexées ou non, alors que les *modestes* se restreignent aux pages effectivement indexées. Quand on sait que les pages non-indexées représentent au moins 25 % des pages connues, mais moins de 1 % des réponses aux requêtes[5] (sources : *SearchEngineShowdown*), on a un début d'explication du phénomène.

2.2.3 remarques personnelles

Avant de continuer plus loin, je voudrais mettre en avant un constat personnel de l'extrême volatilité des chiffres avancés par les moteurs de recherche. Par exemple, alors que le nombre d'URLs revendiquées par Google est de quatre milliards, la requête « the »[6] promet 5 840 000 000 pages. D'un autre côté, il est en théorie possible de connaître le nombre de pages connues pour un domaine précis (par exemple `.fr`) en employant une requête du type `site:tld`. En faisant cette opération sur l'ensemble des TLDs, on devrait logiquement tomber sur près de six milliards de pages, voire plus. Cependant, l'expérience ne nous en donne que 749 485 183 (à 0,26% près, *Google* ne donnant que 3 chiffres significatifs). Il faut conclure de ces deux tests soit qu'au moment des expériences, Google indexait au moins 5 840 000 000 pages et au plus 749 485 183, soit que plus de 5 milliards de pages indéxées n'appartiennent pas à des TLDs existant, soit que les chiffres annoncés par Google doivent être pris avec un minimum de réserve.

5. Il existe cependant des requêtes ne renvoyant que des pages non indexées. Je recommande ainsi, sous *Google*, la requête `site:.xxx`, qui renvoyait, en août 2004, 765 pages non-indexées (car non existantes, le TLD .xxx n'existant pas lors de l'expérience), dont seulement 495 pertinentes…

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

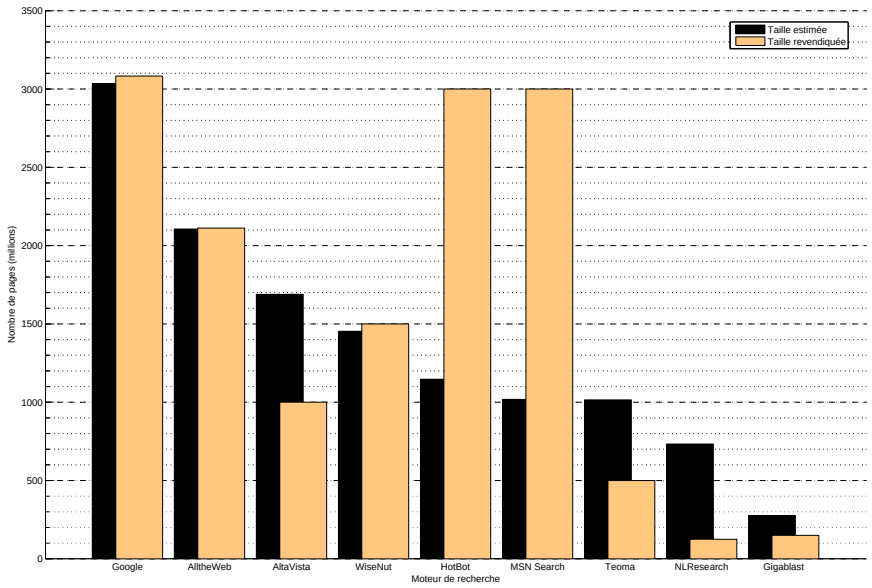


FIG. 2.3 – *Tailles estimées et revendiquées de différents moteurs de recherche au 31 décembre 2002 (source : [Sea])*

Le nouveau moteur de recherche de *Yahoo* donne lui aussi des résultats parfois étranges. Ainsi, j'ai pu constater que le nombre de résultats pouvait dépendre du rang à partir duquel on désirait voir les résultats. Ainsi, alors que la requête "`pâte à crêpes`" donne 7480 réponses si on affiche les 20 premières pages, ce nombre tombe à 7310 si on demande les pages 981 à 1000, avec une décroissance régulière. Expérimentalement, la différence peut atteindre 5% du chiffre annoncé.

Il faut enfin ne pas oublier de parler du problème des doublons. En effet, la plupart des grands moteurs considèrent pour leurs index et algorithmes que `www.domain.com/`, `domain.com/`, et `www.domain.com/index.html` sont des pages différentes. Le regroupement est fait *a posteriori*, en utilisant des méthodes de hashage pour identifier les pages identiques (que *Google* appelle pudiquement *pages à contenu similaire*). En résumé, les doublons sont comptés dans le nombres de pages trouvées, mais ne sont pas renvoyés dans les réponses (sauf demande explicite de l'utilisateur). Pour fixer les idées, le tableau 2.1 présente les résultats de quelques requêtes pour lesquelles il est possible de connaître le nombre de pages hors doublons[7]. Par convention, nous avons choisi de définir le nombre de pages revendiquées par *Yahoo* comme celui revendiqué lorsque l'on demande les dernières pages.

7. Comme *Yahoo* et *Google* ne renvoient que les 1000 premières réponses à une requête, il n'est possible de compter les doublons que lorsque le nombre de pages uniques est inférieur à 1000. Il suffit alors de demander les 10 dernières réponses pour avoir les informations voulues.

	Pages revendiquées		Pages uniques	
Requête	Google	Yahoo	Google	Yahoo
Anticonstitutionnellement	752	795	442	485
Raton laveur	18200	23500	801	> 1000
"Pâte à crêpe"[a]	2690	1630	591	489
"Pâte à crêpes"[a]	7900	5950	745	971
Webgraph	9040	8180	603	759

TAB. 2.1 – *Pages revendiquées ou pages à contenu similaire ? Quelques exemples à la date du 28 août 2004.*

[a] Notons que d'après le *Petit Robert*, il faut écrire *pâte à crêpe* et non *pâte à crêpes*.

2.3 Mesures sur des crawls

L'étude et la comparaison des différents crawls est un sujet délicat à traîter. D'un côté, il y a les crawls commerciaux issus de moteurs de recherche, qui ne peuvent guère être accessibles qu'au travers de requêtes publiques faites sur les moteurs eux-mêmes, lesquelles requêtes sont très souvent bridées. De l'autre, les laboratoires de recherche offrent souvent des tailles plus réduites et des méthodes de crawl extrêmement variées. À cause de ces disparités, il peut être difficile de comparer différents résultats, voire de savoir s'ils sont comparables. L'objet de cette section est de répertorier différentes méthodes d'évaluation des crawls, dans le cas de données privées et celui de données publiques.

La méthode d'estimation des tailles réelles proposée par Greg Notess (cf section 2.2.2) peut également servir de mesure de qualité de crawl. En dénombrant les réponses faites à 25 requêtes représentatives, on mesure en effet en quelque sorte la capacité d'un moteur à répondre à ces requêtes, transformant en quelque sorte le biais des requêtes en mesure de qualité.

2.3.1 Techniques de chevauchement (overlap)

Ce système de comparaison par requêtes est utilisé de manière plus poussé dans la méthode de comparaison par chevauchement, introduite par Bharat et Broder en 1998 [BB98] et reprise par Lawrence et Giles [LG98, LG99].

Le principe du chevauchement est le suivant : arriver à mesurer entre deux moteurs le chevauchement de leurs index respectifs. Par exemple, si on peut déterminer qu'une fraction p d'un index A est dans B, et si une fraction q de B est dans A, alors on peut en déduire que $|A \cap B| = p\,|A| = q\,|B|$, et ainsi avoir le rapport de taille entre les index :

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

Tout le problème est d'estimer p et q en l'absence d'index librement accessible au public.

C'est là qu'intervient l'échantillonnage par requête : des requêtes créées par un générateur aléatoire le plus uniforme possible, sont soumises à un moteur. À chaque fois, un des réponses parmi les 100 premières est choisie au hasard, et on vérifie, par des méthodes plus ou moins strictes, si cette réponse est dans l'index de l'autre moteur.

Par rapport à la méthode de Notess, deux autres sources de biais (en plus du biais dû au choix des requêtes et des biais standards d'échantillonnage) sont introduites :

Le biais de classement En choisissant uniquement les échantillons parmi les 100 premières réponses (contrainte imposée par les moteurs de recherche eux-même), les classements (les importances) des pages considérées sont statistiquement plus élevés que la moyenne.

Le biais de vérification Il n'y a pas de méthode parfaite pour contrôler l'appartenance d'une page donnée à un index. Les différentes méthodes proposées par Bharat et Broder sont basées sur l'analyse d'une requête censée définir la page et les critères de validation vont de *le moteur renvoie la bonne page* à *le moteur revoie un résultat non vide*.

Les mesures par chevauchement ont eu leur heure de gloire au cours des guerres du Crawl[8] (voir section 2.2), mais si elles ont l'avantage d'introduire un formalisme rigoureux, le coût en biais et la relative complexité de la méthode font que la méthode de Notess reste très compétitive.

2.3.2 Échantillonnage par marche aléatoire

Une variante intéressante de la méthode d'échantillonnage de Bharat et Broder a été proposée par Henzinger *et al.* [HHMN99] : plutôt que de mesurer un moteur à l'aide d'un autre, le principe est d'utiliser comme mètre-étalon un crawl personnel issu d'une marche aléatoire. Cette marche aléatoire ne fournit pas un échantillonnage uniforme des pages Web[9], mais un échantillonnage qui, aux biais statistiques près, obéit à la distribution de probabilité associée à la marche aléatoire. Cette distribution sur les pages Web, plus connue sous le nom de PageRank (Voir la partie II, va permettre, en mesurant à l'aide des techniques de chevauchement la fraction des échantillons connus du moteur, d'estimer la quantité de PageRank contenue dans les pages connues du moteur.

Le biais spécifique de cette méthode est que derrière l'échantillonnage par marche aléatoire se cache un crawl implicite, le crawl que l'on obtiendrait en laissant tourner la marche aléatoire suffisamment longtemps. On mesure donc le PageRank du moteur que l'on veut tester par rapport au PageRank sur ce crawl virtuel. En particulier, toute divergence entre les stratégies de crawl de la marche aléatoire et du moteur sont néfastes pour l'évaluation de ce dernier : si le moteur a indexé des parties du Web que la marche aléatoire, pour diverses raisons techniques, n'a jamais effleurées, ces parties ne compteront pas dans l'évaluation. À l'inverse, si certains sites sont volontairement évités par les moteurs (des pouponnières par exemple, voir page 25), cette omission sera sanctionnée dans l'évaluation.

8. De là à dire que certains articles ont surtout servi à montrer qu'AltaVista avait la plus grosse basse de donnée, il n'y a qu'un pas que je franchirai section 3.2…

9. Choisir une page Web *au hasard* est impossible, le Web indexable étant infini.

Chapitre 3

Graphes et structures du web

La vie est un papillon éphémère arborant les ailes du paradoxe.
BENOÎT GAGNON

Les sites ne changent pas d'aspect comme les hommes changent de visage.
TACITE

Un bon site Web est toujours en construction.
ANONYME

MAINTENANT que les concepts de Web et de crawl ont été précisé, nous allons pouvoir étudier ce qu'il est possible d'en dire. Ce chapitre étant consacré aux structures, il convient de placer une structure canonique sur les crawls que l'on veut étudier. Une des structures les plus simples et les plus naturelles dans le cas des crawls du Web est la structure de graphe. Les graphes du Web, dont une définition est donnée section 3.1, sont un sujet d'étude fréquent. Après avoir analysé de manière critique l'un des modèles les plus connus, le modèle du nœud papillon (section 3.2), nous mettrons en évidence dans le reste du chapitre l'importance de la structure en sites dans les graphes du Web.

3.1 Graphes du Web : définition

Considérons un crawl C. Ce crawl est constitué d'un ensemble d'URLs, dont certaines ont été visitées, et d'autres simplement détectées à travers les hyperliens des pages visitées. Il contient aussi l'ensemble des hyperliens des pages visitées. On appellera graphe du crawl C le graphe orienté $G = (V, E)$, où V est l'ensemble des pages du crawl, visitées ou non, et tel qu'un arc

$e \in E$ relie une page i à une page j si, et seulement si, i contient un hyperlien pointant vers j. Par convention, les ancres[1] sont ignorées, et les liens multiples ne rajoutent rien au graphe (si une page i possède deux hyperliens pointant vers j, le graphe aura toujours un seul arc de i vers j). De même, les liens d'une page vers elle-même ne seront pas pris en compte.

Par abus de langage, nous utiliserons souvent le terme *graphe du Web* pour désigner un tel graphe G. Il conviendra de toujours se rappeler qu'il s'agit toujours en fait de graphes de crawls.

3.2 Le nœud papillon revisité

La structure en nœud papillon du graphe du Web est aujourd'hui admise par beaucoup de monde, et est très souvent évoquée par des articles scientifiques [CF02, APC03, LM04, RGM03, ANTT01, LM04, KHMG03a] ou de vulgarisation [GL02], de manière parfois totalement hors de propos[2]. Je considère pour ma part que le modèle du nœud papillon s'applique mal aux graphes dynamique en général, et aux graphes du Web en particulier. Cette section, dont les idées viennent de [Mat01], va essayer de justifier cette prise de position en essayant de distinguer mythe et réalité afin de dégager le véritable intérêt de l'article *Graph Structure in the Web*.

3.2.1 Le modèle du nœud papillon

Graph Structure in the Web [BKM+00] est un article qui a été présenté lors de la 9ième conférence internationale sur le World Wide Web. L'article est basé sur l'analyse de deux crawls fournis par le moteur de recherche AltaVista, un crawl de 203 millions de pages daté de mai 1999 et un de 271 millions de pages d'octobre 1999. L'analyse donne plusieurs résultats sur la structure du graphe du web, le plus novateur étant la découverte d'une structure en nœud papillon :

- Environ un quart des pages considérées appartiennent à une seule et même composante fortement connexe, appelée noyau ou SCC (pour *S*trongly *C*onnected *C*omponent).
- Un autre quart est formé de pages à partir desquelles on peut rejoindre le noyau, l'inverse n'étant pas vrai. C'est la composante IN.
- À l'inverse, près d'un quart des pages sont accessibles à partir du noyau, la réciproque étant fausse. Ces pages forment la composante OUT.
- Les pages qui sont accessibles à partir de IN, ou qui permettent d'accéder à OUT, et qui n'appartiennent ni à IN, ni à OUT, ni à SCC, forment les tentacules (TENDRILS) et représentent aussi près d'un quart des pages.

1. Les ancres (*anchors*) permettent de pointer vers une partie spécifique d'une page Web.

2. Par exemple, il est affirmé dans [LM04] et [KHMG03a] que Arasu *et al.* utilisent le modèle du nœud papillon pour accélérer leur calcul de PageRank [ANTT01]. Vérifications faites, Arasu *et al.* citent bien le modèle du nœud papillon, et proposent une méthode de calcul de PageRank basée sur… l'écriture de A sous une forme triangulaire par blocs déduite de la décomposition en composantes fortement connexes (cf théorème 6). En particulier, le seul apport du modèle du nœud papillon est l'existence d'un bloc diagonal plus gros que les autres (la SCC).

- Les quatre quarts que nous venons d'évoquer forment en fait 90% des pages du crawl, soit la plus grande composante connexe. Les 10% restant sont des composantes déconnectées du reste et de taille moindre.

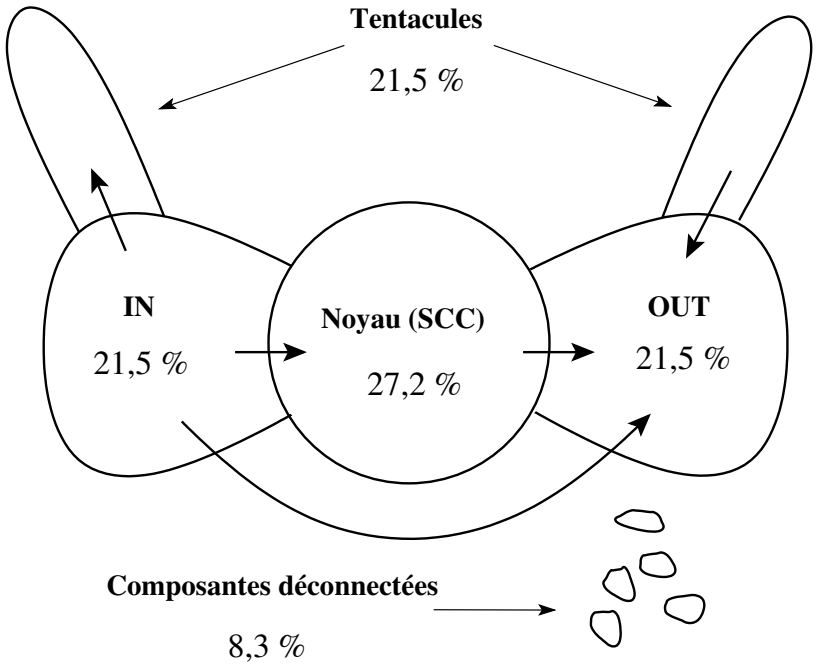


FIG. 3.1 – *La structure en nœud papillon*

Cette structure est couramment représentée par un schéma similaire à celui de la figure 3.1. La forme de nœud papillon obtenue à partir des composantes IN, SCC et OUT donne son nom au modèle (*bow tie*).

De cet article est resté auprès de beaucoup de monde l'idée que ce découpage en quantité à peu près égales en composante IN, SCC, OUT et tentacules est caractéristique du Web : c'est le modèle du nœud papillon.

3.2.2 Faiblesses du modèle du nœud papillon

Existe-t-il encore ?

Le principal inconvénient du modèle du nœud papillon, c'est que selon toute vraisemblance, il ne représente plus la réalité. Le graphe du Web, c'est-à-dire le graphe du Web accessible, ne peut pas avoir une structure de nœud papillon équilibrée, puisqu'il contient entre autres la *page qui pointe vers toutes les pages* (cf section 1.4 page 21). Il faut donc considérer que ce modèle

s'applique aux graphes issus de gros crawls du Web (c'est d'ailleurs l'opinion des auteurs[3]). Seulement, on constate que :

- Aujourd'hui, la plupart des grands crawls sont l'œuvre de moteurs de recherche.
- Ces moteurs recensent les sites qu'il découvrent (par crawl ou par référencement) dans des annuaires (*directories*).
- Les annuaires font de toutes évidence partie du noyau du Web.

Que peut-il donc se passer pour une page qui, pour un crawl donné, fait partie de la composante IN, ou des tentacules, ou même des composantes déconnectées :

- Si la page ou un de ses ancêtres est indexé par les annuaires, elle se retrouve *de facto* soit dans le noyau, soit dans OUT.
- Une page fait partie du crawl parce que un de ses ancêtres faisait partie de l'ensemble initial de crawl. Si aucun ancêtre de la page n'a été mis à l'annuaire, on peut douter qu'un ancêtre soit conservé dans l'ensemble initial de crawl. Conséquence, au prochain crawl, la page disparaîtra.

En conclusion, oui, il peut y avoir une composante IN, ou des composantes déconnectées, mais ces composantes sont forcément transitoires, à cause même des relations entre crawlers et annuaires. Il faut également tenir compte du fait que les gros crawls actuels possèdent une part non négligeable de pages non-indexées (au moins 25 % d'après une étude de 2002 [Sea], 75 % d'après un article de 2004 [EMT04]), qui viennent automatiquement grossir la composante OUT. Pour toutes ces raisons, j'ai tendance à douter très fortement que plus de 2 (3 ?) milliards de pages indexées par Google ne fassent partie ni du noyau, ni de OUT.

Un modèle peu robuste

Une question que l'on peut se poser est de savoir si les paramètres considérés sont pertinents pour l'étude des graphes du Web. D'un côté, nous avons les crawls, qui essaient de parcourir tant bien que mal un Web dynamique aussi bien spatialement que temporellement, et qui ont souvent un *overlap* très faible entre eux [BB98]. De l'autre, des définitions comme *IN : ensemble des pages qui accèdent au noyau, mais ne sont pas accessibles de celui-ci*. Est-il raisonnable de faire cohabiter des crawls très variables et des définitions peu robustes ? Un exemple concret : imaginons qu'un individu peu scrupuleux s'empare de l'ensemble des pages initiales des crawls d'AltaVista et mette les liens vers ces pages sur sa page Web. Si cette page est dans le noyau, cela signifie l'écroulement de toutes les composantes sur SCC et OUT[4].

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

4. C'est d'ailleurs ce phénomène d'écroulement qui tend à se produire aujourd'hui avec les annuaires.

Contrairement à ce qu'affirme les auteurs[5], il suffit donc d'une page, pas si grosse que ça[6], pour mettre à mal le modèle du nœud papillon. On est en fait en présence d'une lacune méthodologique : en face de données sur lesquelles l'incertitude et la dynamique est très grande (les graphes du Web), cela n'a pas de sens de considérer des variables extrêmement dépendantes des conditions initiales comme des constantes universelles. Pour les même raisons, considérer le diamètre de la composante connexe n'est guère pertinent ; effectivement, sur les 2 crawls d'AltaVista étudiés, il reste à peu près constant (aux alentours de 500), mais en y regardant de plus près, des indices semblent montrer que le diamètre de 500 est dû à une unique guirlande, peut-être une page mal configurée ou un piège à robots qui aurait échappé aux filtres[7].

Une expérience non reproduite

À ma connaissance, l'expérience de [BKM+00] n'a pas été reproduite sur des grands crawls depuis. Les seules expériences réelles menées sur le modèle du nœud papillon portent donc sur deux crawls propriétaires d'AltaVista, peu espacés et donc vraisemblablement basés sur la même technologie (ce qui peut expliquer certains artefacts comme la constance du diamètre). De plus, si on compare la taille et les dates de ces deux crawls (mai 1999, 203 millions de pages, octobre 1999, 271 millions de pages) avec les figures 2.2(b) et 2.2(c) (page 30), on arrive à la conclusion que les crawls utilisés étaient certainement expérimentaux.

Dans [DKM+01], on trouve pourtant que la structure en nœud papillon est une figure fractale de la structure du Web, c'est-à-dire que l'on retrouve à l'intérieur des sites et des communautés le même modèle de nœud papillon que sur le graphe tout entier. En y regardant de plus près, on s'aperçoit que la structure globale de nœud papillon est admise, et que les proportions de IN, OUT, et SCC observés dans les sous-graphes sont extrêmement variables, ce qui va dans le sens des remarques précédentes.

3.2.3 Le modèle du nœud papillon dans un contexte sociologique

Pour bien comprendre le succès du modèle du nœud papillon, il est important de se replacer dans le contexte. Nous sommes au début de l'année 2000. La *Bulle Internet* est encore en pleine croissance, et des articles apparaissent indiquant que les moteurs de recherche ne font plus face à la croissance du Web [Bra97, LG99, Ber00]. La société AltaVista a remporté la première *Guerre du Crawl* (voir page 30 ainsi que les conclusions de [BB98]) en dépassant les 150 millions de

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. Il semble que l'ensemble de départ des crawls d'Altavista de l'époque était constitué de quelques centaines de pages. Une page de *bookmarks* peut sans problème contenir quelques centaines d'hyperliens.

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.*

pages indexées, mais il lui manque l'assurance que cette augmentation de taille va lui permettre de rattraper l'évolution du Web. Dans cette optique, *Graph Structure in the Web* tombe à point nommé. En effet, quelle est la morale de l'article ?

- Il existe une structure globale du Web (avec un grand W).
- Cette structure est invisible si on considère une portion trop petite du Web, mais il existe un seuil au-delà duquel la structure d'un crawl est la structure du Web.
- Les crawls d'AltaVista sont assez grands pour posséder la structure du Web.

En somme, *Graph Structure in the Web* répond à une attente à la fois existentielle (Peut-on comprendre le Web ?) et commerciale (Notre crawl est le Web.). On pourrait presque le voir comme un mécanisme issu des lois de l'offre et de la demande. À la limite, la question de savoir si ses bases scientifiques sont solides ou non devient secondaire, ce qui est peut-être un début d'explication au succès de cet article.

3.2.4 Conclusion

Le modèle du nœud papillon a très certainement eu un sens au moment de sa découverte, mais ce sens n'est pas *le Web a une structure de nœud papillon*. Il exprime plutôt une situation particulière où à cause de la *Bulle Internet*, les nouvelles pages arrivaient trop vite pour que le Web puisse les assimiler. Mais aujourd'hui, deux phénomènes font que l'on est plus dans ce cas : d'une part, la *Bulle* a éclaté. La création de nouveaux sites se fait rare, et on assiste plus à un développement de sites existants qu'à l'apparition d'une nuée de sites non répertoriés (cf section 3.3.2), ce qui est un début d'explication au fait que la structure en nœud papillon se retrouve dans les graphes des sites [DKM+01]. D'autre part, les moteurs de recherche ne font plus seulement qu'indexer le Web, ils en modifient continuellement la structure par l'intermédiaire des annuaires. Ces deux effets conjugués au manque de robustesse du modèle font qu'il est peu probable que les grands crawls commerciaux aient encore la même proportion de IN, tentacules et autres composantes déconnectées que celles trouvées dans *Graph Structure in the Web*.

On retiendra quand même que Broder *et al.* ont mis en évidence l'existence d'un important noyau de pages toutes connectées les unes aux autres, ainsi que le comportement protectionniste de certains sites commerciaux qui s'incluent totalement dans la composante OUT. On retiendra également que la proportion de composante IN dans le graphe d'un crawl est une certaine indication de la difficulté d'un grand graphe structuré à s'adapter à une expansion rapide.

3.3 Rôle des serveurs et des sites dans la structure du graphe du Web

La notion de site est fondamentale dans l'organisation du Web. Par exemple, beaucoup de moteurs de recherche ont pour politique de ne renvoyer qu'un nombre limité de pages par site

(avec une option *plus de pages venant du même site* disponible). Une bonne décomposition en sites a de nombreuses applications, qui vont des méthodes de calcul du PageRank [KHMG03a, MV04] à des méthodes de compression efficaces des crawls [RSWW01, GLV02].

3.3.1 Serveurs Web, sites et communautés

Les concepts de serveur Web, site et communauté sont souvent rencontrés dans la littérature, avec des définitions parfois contradictoires. Nous allons expliciter ici ce que nous entendrons par serveur, site et communauté.

Serveur réel Un serveur Web réel se définit comme une machine physique précise connectée à Internet et identifiée par une adresse IP, qui renvoie un code de réponse 200 et une page Web en retour d'une requête *http* demandant la racine (*/*) sur le port 80[8]. Le site Web physique associé au serveur consiste en toutes les pages hébergés à l'adresse IP considérée.

Serveur virtuel Les adresses IP n'étant pas illimitées, des solutions ont dû être envisagées pour économiser les adresses. Une de ces solutions est l'emploi de serveurs virtuels. Les serveurs virtuels, introduits avec la norme *http 1.1*, permettent à une même adresse IP de se comporter comme plusieurs serveurs différenciés par leur adresse DNS (*Hostname*). Ce procédé est par exemple utilisé par *free* pour l'hébergement des pages personnelles de ses clients, ou au Japon, où les ressources en adresses IP sont très limitées. Du point de vue de l'utilisateur, un serveur virtuel est indicernable d'un serveur réel, si ce n'est qu'on ne peut pas remplacer l'adresse DNS par l'adresse IP.

Site logique Un site Web logique est constitué d'un ensemble de pages fortement reliées par hyperliens. Intuitivement, un site se caractérise par une certaine homogénéité de style (imposée par le *Webmaster*) et une navigation aisée au sein des pages du site. L'étude des sites sera l'objet des sections 3.3.3 et 3.3.4.

Communauté La notion de communauté est en quelque sorte orthogonale à la notion de site. Une communauté est un ensemble de pages reliées par un intérêt commun. La recherche de communautés est au cœur de l'algorithme *HITS* de Kleinberg [Kle98] ainsi que du projet *CosmoWeb* [BBM03].

3.3.2 Évolution du nombre des serveurs

Il est plus facile d'estimer le nombre de serveurs Web (réels et virtuels) que le nombre de pages Web.

Pour dénombrer les serveurs réels, il « suffit » de faire une requête *http* sur toutes les adresses IP possibles. Or, si on oublie les adresses IPv6, qui restent encore relativement négligeables, il

8. Le port 80 est le port standard du protocole http. D'autres ports sont plus ou moins couramment utilisés, comme le port 443 pour le http sécurisé ou le port 8080 pour un serveur secondaire, mais nous n'en tiendrons pas compte ici.

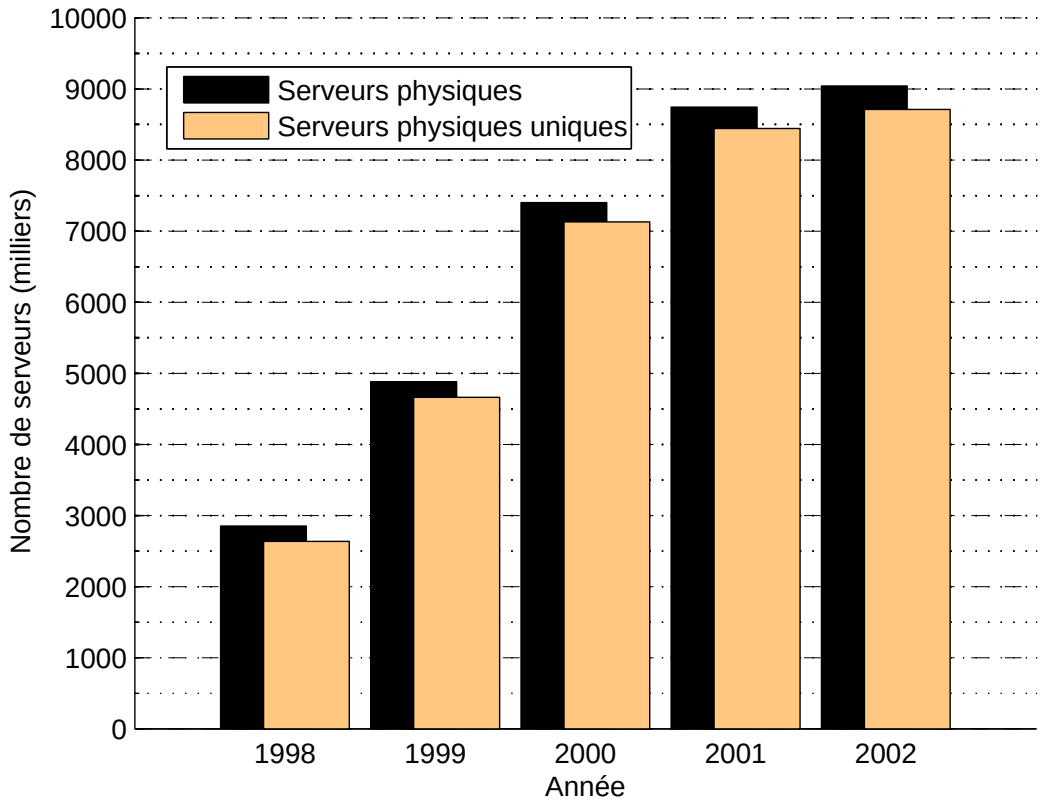


FIG. 3.2 – *Serveurs Web physiques (adresses IP)*

n'y a que 2^{32} adresses possibles, soit environ quatre milliards[9]. Ce travail a été effectué par le *Web Characterization Project* [OLB+02], sur la période 1998-2002, et les résultats concernant le nombre de serveurs sont représentés figure 3.2. Le nombre de serveurs physiques uniques est le nombre de serveurs physiques diminué des doublons, c'est-à-dire des serveurs renvoyant exactement le même contenu qu'un serveur précédemment testé.

La principale remarque que l'on peut faire quant à ces chiffres est l'observation d'une stagnation du nombre de serveurs physiques depuis 2001. Cette stagnation peut s'expliquer par la fin de la *bulle Internet*. En effet, le Web a cessé d'être une nouvelle technologie, et la plupart des acteurs universitaires, commerciaux et sociaux susceptibles de s'intégrer au Web l'ont déjà fait. D'après le *Web Characterization Project*, il ne se crée presque pas, voire plus de serveurs Web, la croissance se fait individuellement au niveau des serveurs.

9. Il y en a en fait moins si on enlève certaines adresses *réservées* - les adresses militaires par exemple - qui ne sont pas censées héberger de serveur Web accessible. L'IANA (*I*nternet *A*ssigned *N*umbers *A*uthority) est l'organisme chargé de gérer les tranches d'adresses disponibles ou non. On peut ainsi éliminer environ 48 % des adresses disponibles.

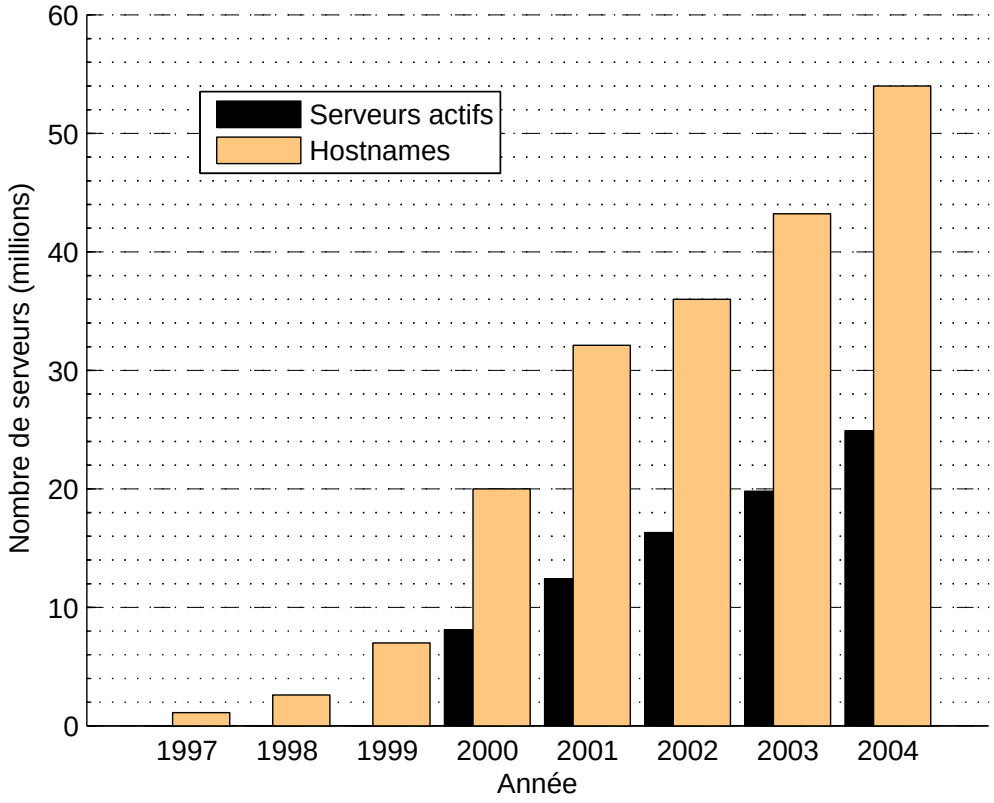


FIG. 3.3 – *Serveurs Web virtuels (noms de domaines)*

L'étude des serveurs virtuels a été réalisée par le site *Netcraft* [Net04]. Elle consiste à répertorier l'ensemble des adresses DNS déclarées (comme le site *WhoIs* [Who04]), et à tester par une requête *http* l'existence d'un serveur sur chacune des adresses obtenues.

L'évolution des serveurs Web virtuels, d'après *Netcraft* [Net04], montre une progression constante à peu près linéaire du nombre de serveurs Web virtuels actifs au cours des 4 dernières années (cf figure 3.3). Ce n'est en tout cas à priori pas une progression géométrique.

3.3.3 Approche intuitive et visuelle de la notion de site structurel

Le concept de site est selon nous fondamental dans la structure du Web. Ses applications sont nombreuses :

- Évaluation plus précise du degré d'aboutissement d'un crawl (pour chaque site, quelle est la quantité de pages visitées ?)
- Distinguer des liens purement navigationnels de liens plus pertinents (liens de confiance en d'autres sites).

- Permettre à certains moteurs de recherche de ne renvoyer qu'une ou deux pages par site, afin d'éviter une monopolisation des réponses à une requête.
- Affinement du facteur *zap* dans les calculs de PageRank (voir partie II, section 5.3, ainsi que [BMPW98, HHMN99])
- Algorithmes de PageRank basés sur la structure de site (l'algorithme FlowRank, présenté partie II, chapitre 7, ainsi que son « concurrent » BlockRank [KHMG03a])

La plupart du temps, les sites sont approximés par les serveurs ([Ada99]). Mais la réalité peut parfois être plus complexe. Des gros sites peuvent reposer sur plusieurs serveurs, alors qu'inversement un serveur peut héberger plusieurs sites. Par exemple, on aurait envie de regrouper www.microsoft.com et download.microsoft.com et de les considérer comme faisant partie d'un même site. À l'inverse, tous les hébergeurs de sites personnels n'utilisent pas les domaines virtuels, une grande partie met les sites dans les répertoires d'un serveur unique.

Toutes ces considérations nous amènent à nous poser la question de la définition d'un site logique. Les internautes (humains) n'ont en général aucun problème pour savoir ce qu'est un site, ce qui constitue en quelque sorte une définition empirique, mais n'est guère rigoureux et ne se prête guère à l'automatisation. Notons que les sites référencés à la main par les *annuaires* rentrent dans cette définition.

Le Web Characterization Project [OLB+02] propose quant à lui la définition sémantique suivante :

> *[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.*
>
> [Un site Web logique est] un ensemble de pages similaires qui, par leur réunion, forment un objet composite délivrant une information pertinente. Cette information pertinente implique que l'objet en question réponde à un besoin d'information non-trivial.

À cette caractérisation sémantique, Li, Kolak, Vu et Takano superposent dans [LKVT00] une approche structurelle pour définir proposent la notion de *domaine logique* :

> *A logical domain is a group of pages that has a specific semantic relation and a syntactic structure that relates them.*
>
> Un domaine logique est un groupe de pages qui possèdent une relation sémantique spécifique et une structure syntaxique qui les relie.

Se basant sur cette définition, ils définissent un ensemble de règles (sémantiques et structurelles) pour identifier des sites.

Plus modestement, notre approche est de voir ce qu'il est possible de réaliser en ne considérant que le côté structurel des pages. Nous nous limitons donc à d'une part la structure de graphe induite par les hyperliens, d'autre part la structure en arbre des URLs qui fait apparaître des clusters naturels sur les graphes des crawls. En effet, très souvent, un site logique obéit à des règles

hiérarchiques au niveau des URLs (Uniform Resource Locators [BMC94]), la plus commune étant l'existence d'un préfixe commun caractéristique.

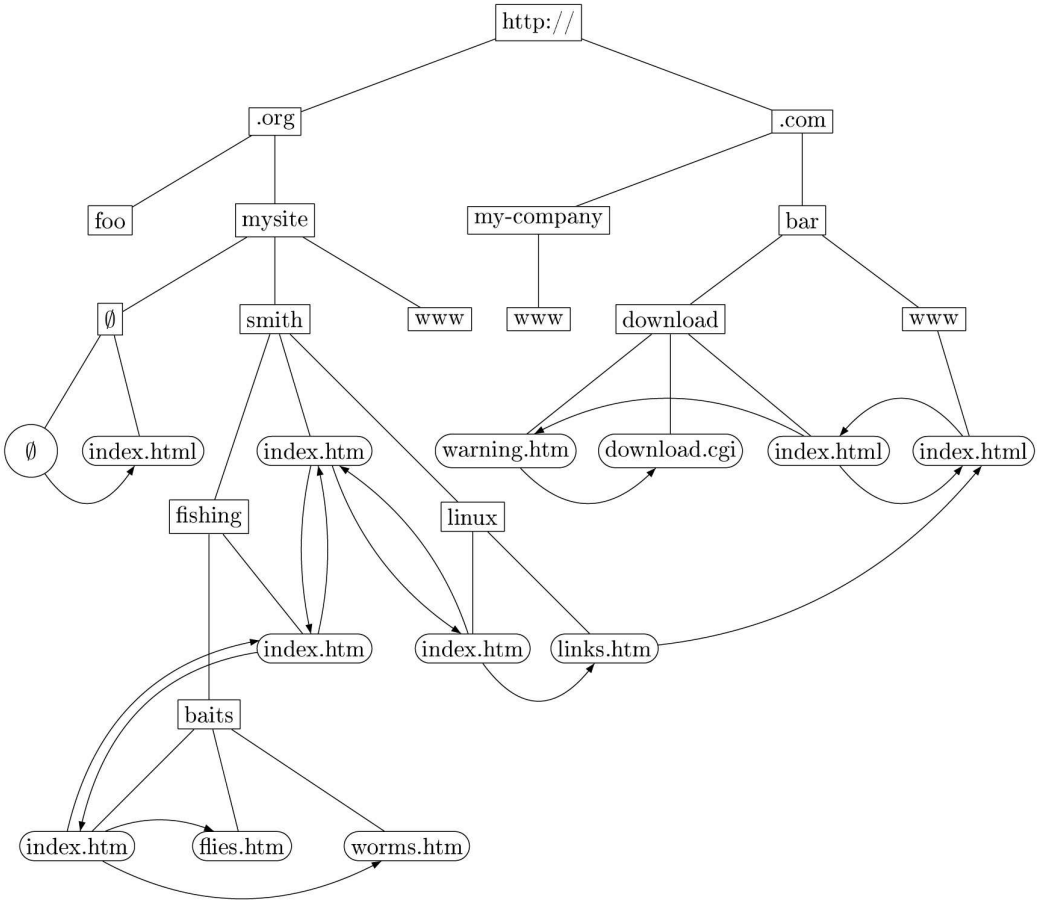


FIG. 3.4 – *Les URLs : un arbre de décomposition naturel pour les graphes du web*

Cette structure d'arbre de décomposition[10] sur les graphes (*clustered graphs*) a été introduite par Feng *et al.* dans [FCE95] comme un outil de représentation des grands graphes. Sa principale utilisation est donc de permettre de dessiner des graphes de manière à laisser apparaître l'éventuelle hiérarchie existante, et on emploie surtout les arbres de décomposition dans les domaines où des structures de diagrammes implicites ou explicites existent[11].

Tout le problème est alors de trouver pour un graphe donné l'arbre de décomposition qui offre la meilleure représentation structurelle [Bri02]. Dans le cas des graphes du Web, l'arbre des URLs offre un arbre de décomposition intrinsèque.

10. Merci à Fabien de Montgolfier de m'avoir rappelé le terme *arbre de décomposition.*
11. On pensera par exemple aux arbres de décomposition modulaire.

Definition

Soit $G(V, E)$ un graphe. Un arbre de décomposition de G est la donnée d'un arbre T dont les feuilles correspondent aux sommets V. Chaque nœud interne n de T définit un cluster (ensemble de sommets) $V_T(n)$. Ce cluster est constitué de l'ensemble des sommets de V correspondant aux feuilles du sous-arbre de T de racine n. Par exemple, le cluster associé à la racine de T est l'ensemble V tout entier.

Un bon exemple de l'utilisation des arbres de décomposition est la modélisation des relations humaines. Il existe en effet un graphe assez naturel des relations humaines, où les sommets sont des personnes et où une personne A pointe vers une personne B si A connaît B). Un arbre de décomposition qui peut complèter de manière pertinente ce graphe est celui de la localisation géographique : monde, pays, (état), ville, quartier, rue,...

Graphes du Web et arbre de décomposition des URLs

Les URLs offrent une structure naturelle d'arbre de décomposition sur les graphes du Web. L'ensemble des serveurs[12] peut être représenté par un arbre dont la racine est `http`, les nœuds de profondeur 1 les TLDs[13], suivis des noms de domaines proprement dits et éventuellement des sous-domaines. Chaque serveur, qui est une feuille de l'arbre des noms de domaine, héberge une hiérarchie de pages identique à la structure du système de fichiers physique correspondant. La réunion des arbres des fichiers dans l'arbre des serveurs donne l'arbre de décomposition des URLs.

Par exemple, l'URL `http://smith.mysite.org/linux/index.html` se décompose en *http*, *org*, *mysite*, *smith*, *linux* et enfin *index.html*, formant ainsi un chemin qui part de la racine jusqu'à la feuille correspondante dans l'arbre de décomposition (voir figure 3.4).

On peut éventuellement s'interroger sur la pertinence de commencer la décomposition par le domaine de premier niveau (TLD). En effet, il existe des macro-sites qui, pour avoir une meilleure visibilité, se déploient sur plusieurs TLDs (`.com` et les ccTLDs des pays d'implantation, de manière assez classique). Nous préférons quand même conserver le tri par TLD, d'une part parce que cela correspond à la philosophie initiale des noms de domaines, d'autre part parce qu'il existe aussi des noms de domaine pour lesquels le TLD change beaucoup de choses. Le lecteur majeur et averti pourra par exemple constater une différence sémantique certaine entre le contenu du site `http://www.france2.fr` et celui de `http://www.france2.com`...

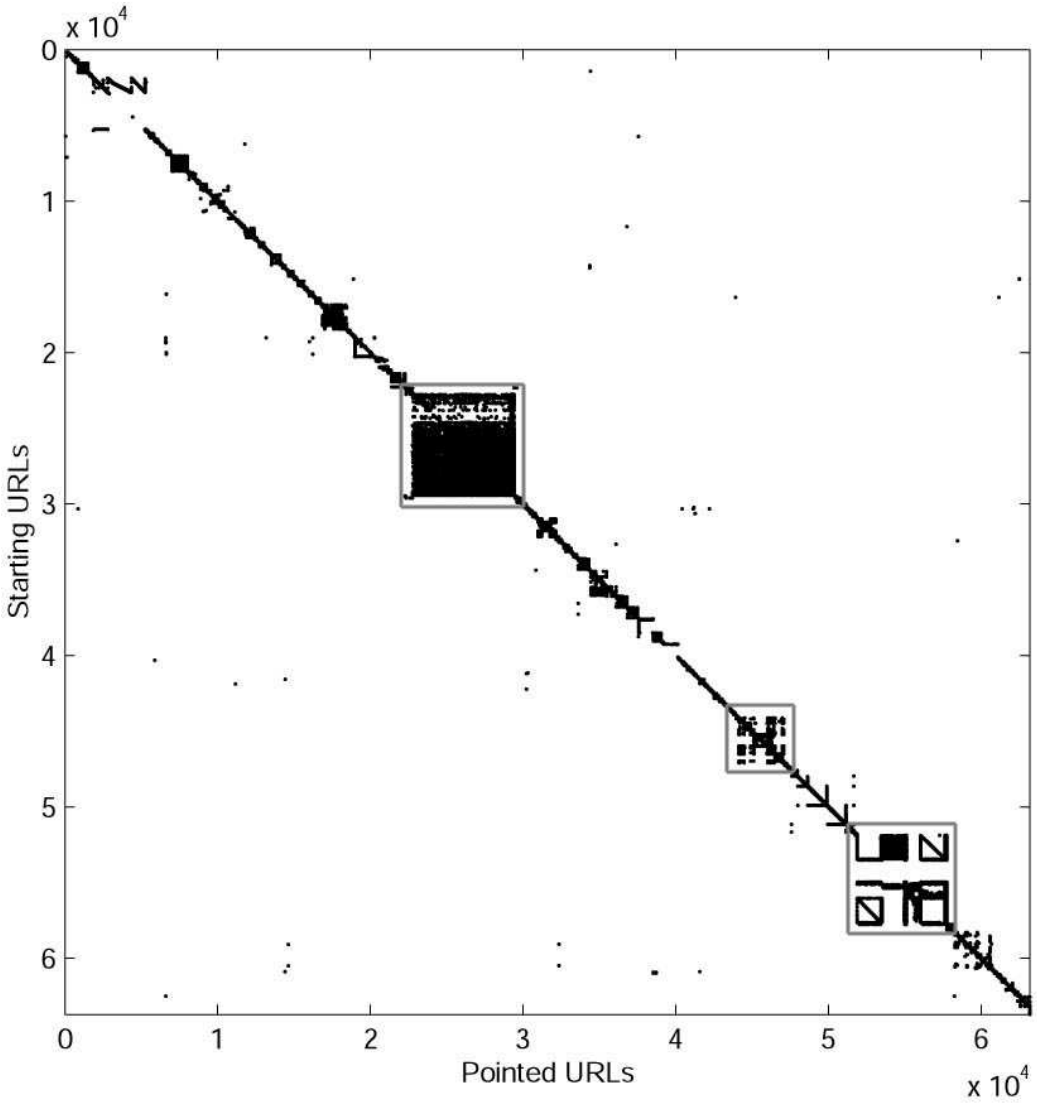


FIG. 3.5 – *Matrice d'adjacence de* 6.10^4 *de pages parmi un crawl de* 8 *millions de pages de* .fr

Voir la structure des sites

Étant donné que d'une manière générale, les webmasters essaient d'organiser leurs sites, on peut s'attendre à ce que le concept de site Web soit intimement relié à celui l'arbre de décom-

12. Pour simplifier, nous nous limiterons aux serveurs identifiés par leur nom de domaine DNS et utilisant le port 80.

13. TLD : *T*op *L*evel *D*omain, ou domaine de premier niveau. Les TLDs sont divisés en deux catégories : les domaines génériques de premier niveau (gTLD) et les domaines de premier niveau qui sont des codes de pays (ccTLD). Les TLDs sont gérés par l'IANA (`http://www.iana.org`).

position des URLs. Nous en avons eu confirmation en observant la représentation graphique de la matrice d'adjacence M du graphe d'un crawl d'environ 8 millions d'URLs *triées dans l'ordre lexicographique* de .fr fait en juin 2001 dans le cadre de l'action de recherche coopérative *Soleil Levant* [Lev01].

Nous avons représenté une petite partie de ce crawl (60 000 pages) figure 3.5, et quelques zooms sur des sous-parties intéressantes figure 3.6. La première constatation est que la matrice d'adjacence peut de toute évidence se décomposer en deux termes, $M = D + S$, où D est une matrice diagonale par blocs, et où S est une matrice (très) creuse.

Les sites (et les sous-sites) apparaissent en effet comme des carrés qui coincident avec les nœuds de l'arbre des URLs. Avec un peu d'habitude, on arrive même à deviner la structure profonde des sites d'après l'aspect du *carré* correspondant. Par exemple, les pages à fort degré sortant (typiquement, la carte du site) se traduisent par des lignes horizontales, alors que celle à fort degré entrant (les pages d'acceuil) sont caractérisées par des lignes verticales. Des carrés « bruités », c'est-à-dire avec des points présentant une structure pseudo-aléatoire (cf figure 3.6(c)), sont souvent le signe de pages de documentation interactives (des dictionnaires par exemple). Remarquons enfin que la structure $D + S$ peut présenter un caractère récursif, comme le montre le bloc de la figure 3.6(d).

3.3.4 Proposition d'algorithme de partitionnement en sites

Nous allons essayer, avec seulement la connaissance du graphe et des URLs d'un crawl, de générer le plus simplement possible une partition du graphe qui reflète la notion de site. Notre approche diffère donc de celle employée par [LKVT00], qui emploie également un ensemble de règles sémantiques pour définir les sites.

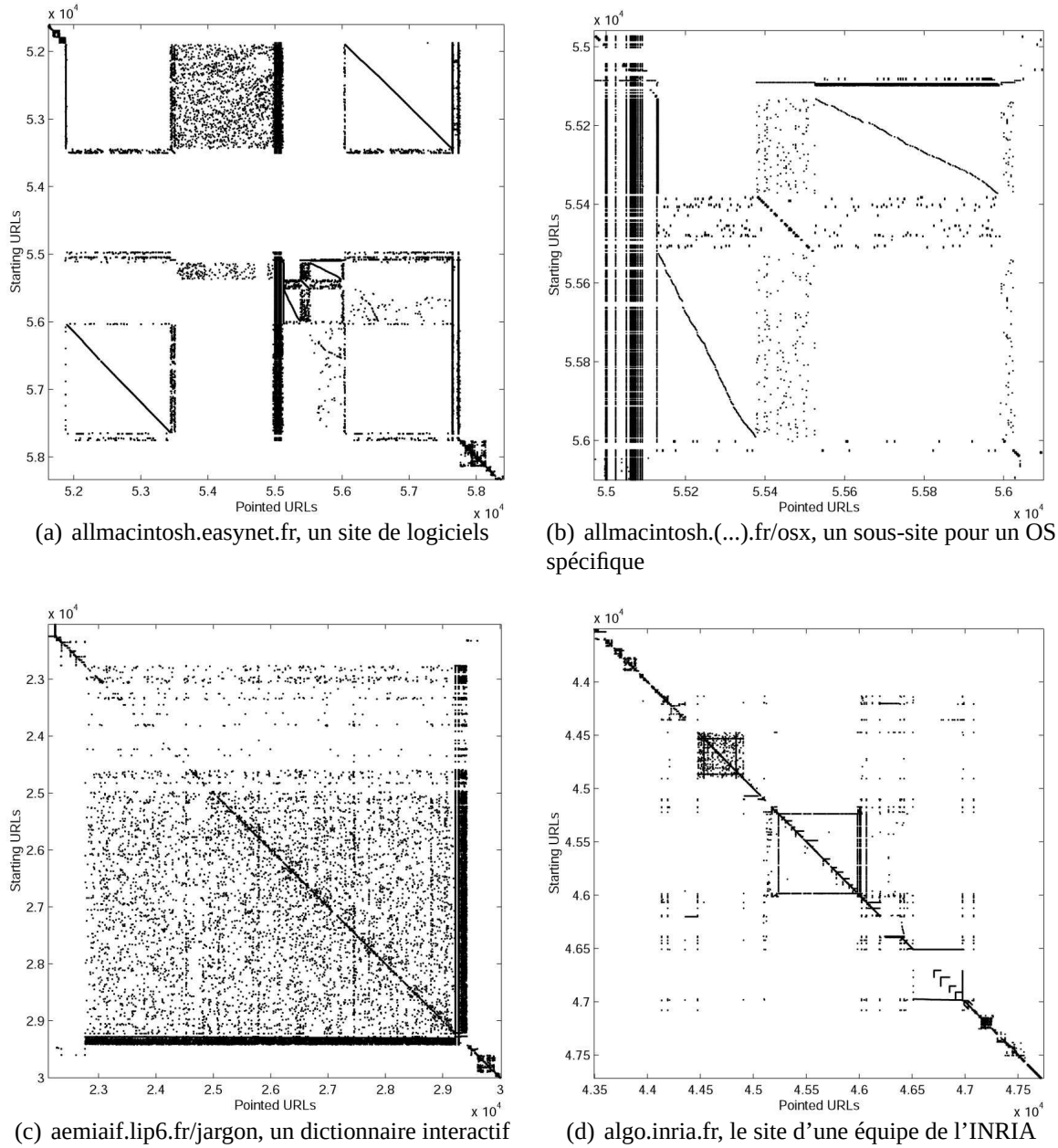
Modèle formel

Structurellement, nous pensons qu'un site se définit comme un ensemble de pages étroitement reliées entre elles par des liens navigationnels, comme semble le confirmer les représentations de la matrice d'adjacence (cf figure 3.5).

Cela nous amène à essayer de définir une fonction qui mesure la qualité d'une partition en site par rapport au taux de liens navigationnels. Cette fonction, $f_G : \mathbb{S}$ partition de $G \rightarrow f(\mathbb{S}) \in \mathbb{R}$ doit atteindre un extremum lorsque $\mathbb{S}$ approche (dans un sens à préciser) ce que nous aimerions appeler partition en sites. Une fois une telle fonction trouvée, la méthode standard pour partitionner G consiste à construire une partition initiale des URLs (adaptée à la situation considérée), puis à effectuer des ajustements locaux pour essayer d'optimiser $f(\mathbb{S})$.

Choix de la fonction f_G

Comme le facteur le plus important selon nous pour estimer une partition en sites est le nombre de liens navigationnels, il apparaît logique de chercher une fonction f_G qui dépende ex-



(a) allmacintosh.easynet.fr, un site de logiciels

(b) allmacintosh.(...).fr/osx, un sous-site pour un OS spécifique

(c) aemiaif.lip6.fr/jargon, un dictionnaire interactif

(d) algo.inria.fr, le site d'une équipe de l'INRIA

FIG. 3.6 – *Approche visuelle de la structure de site : zoom sur des clusters de la figure 3.5*

plicitement de i, nombre de liens internes. Il faut également tenir compte du nombre de sites : en effet, celui-ci n'est pas fixé à l'avance (sinon, on serait dans le cadre classique d'algorithmes de type *mincut* ou *maxcut*). Pour éviter d'avoir comme extremum la partition triviale G, et pour essayer d'avoir une décomposition fine, il nous semble également important d'essayer de maxi-

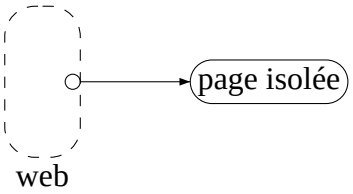


FIG. 3.7 – *Page isolée*

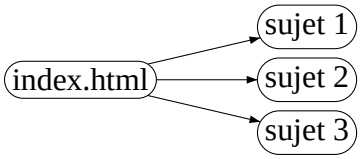


FIG. 3.8 – *Site structuré*

miser le nombre de sites p. Nous avons donc deux paramètres globaux, le nombre de sites p et le nombre de liens internes i, et la fonction f que nous recherchons doit vérifier à la fois $(\frac{\partial f}{\partial p}) > 0$ et $(\frac{\partial f}{\partial i}) > 0$. Plusieurs formules simples sont a priori possibles :

- $f : (p, i) \rightarrow \alpha p + i$, $\alpha \in \mathbb{R}$.
  Cette solution ne sera pas retenue ici, principalement parce que le choix de α nécessite pour être judicieux d'être égal au nombre moyen de liens par page, ce qui laisse peu de jeu. En effet, si α est plus grand que ce nombre, on risque de se ramener à maximiser p (on a intérêt à isoler des pages), alors que s'il est plus petit, pour les mêmes raisons, seul i va compter.
- $f : (p, i) \rightarrow p.i^{\alpha}$, $\alpha \in \mathbb{R}$.
  Cette solution présente l'avantage d'être plus souple pour ce qui est du rapport entre p et i.
- $f : (p, i) \rightarrow p^{\frac{i}{|E|}}$, où $|E|$ est le nombre total de liens.
  Cette dernière solution est délibérément dépourvue de paramètre car on a ainsi une quantité qui a une interprétation intuitive : pour une partition dépourvue de liens externes, on aurait $f(p, i) = p$. D'une manière générale, f reste du même ordre de grandeur que p en s'en approchant d'autant plus que la cohésion interne est grande. Nous appellerons donc la quantité $p^{\frac{i}{|E|}}$ l'indice de site, qui peut s'interpréter comme le nombre de sites isolés équivalent.

Problème des pages isolées

Le cas des pages de degré entrant 1 et de degré sortant nul pose un problème. Si elles peuvent parfois représenter un site isolé réduit à une page unique (figure 3.7), on constate expérimentalement que dans la plupart des cas ce sont des pages crawlées, mais non visitées (voir section 2.1) ou des pages terminales de sites structurés (figure 3.8). D'un point de vue structurel, et en l'absence de toute autre considération, il nous semble légitime de s'assurer que les pages de degré entrant 1 et de degré sortant nul soient rattachées au site de la page parente. Cela se traduit, au niveau de la fonction $f(p, i)$ introduite précédemment, par l'inégalité suivante :

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

Cette inégalité, si on essaie de l'imposer à notre fonction f (ce qui est nécessaire si l'on veut que $f(p,i)$ suffise à réaliser la partition), pose le problème suivant : toute partition de paramètre (p,i) telle que

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

est moins performante que la partition triviale :

$$f(p,i) < f(1,|E|)$$

Hélas, la relation 3.2 est vérifiée dès que le graphe est connexe. D'une manière générale, pour un graphe composé de k composantes connexes et de $|E|$ arêtes, on aura forcément pour toute partition de paramètre (p,i) l'inégalité

$$f(p,i) \leq f(k,|E|)$$

Bilan : Il n'existe aucune fonction universelle $f(p,i)$ maximale pour une partition en site non triviale respectant les pages terminales.

Deux approches sont alors envisageables pour contourner ce problème, ainsi que les autres du même genre que l'on pourrait rencontrer en essayant d'affiner notre modèle :

- Continuer à chercher à maximiser $f(p,i)$ en imposant des règles externes, ce qui reviendrait par exemple à travailler sur un sous-ensemble de l'ensemble des partitions du graphe. C'est l'approche employée pour la partie structurelle de la définition de domaines logiques utilisée par [LKVT00].
- Remplacer p ou i par de nouvelles variables qui prennent en charge le problème implicitement. C'est cette approche que nous allons étudier.

Altération des variables p et i

Une première idée consisterait à pondérer le poids des arêtes de telle sorte que les liens vers des pages isolées soient plus difficiles à rompre. Ainsi, une idée assez classique et naturelle est de prendre un poids inversement proportionnel au degré entrant (cf [Kle98]). Hélas, la nouvelle variable i' introduite présente de nombreux inconvénients, surtout si elle est l'unique altération introduite :

En premier lieu, on peut considérer le cas de deux sites (au sens intuitif) reliés entre eux par une arête unique (figure 3.9). Si l'algorithme est conçu pour ne pas séparer les pages terminales, c'est-à-dire s'il respecte la relation 3.1, il sera obligé de fusionner les deux sites, ce qui n'est pas forcément l'effet désiré.

En outre, si l'on regarde les deux exemples de sites 3-reliés de la figure 3.10, on aimerait que la décision de fusion des sites, toutes choses étant égales par ailleurs, soit la même dans les deux

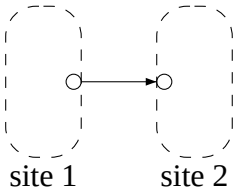


FIG. 3.9 – *Sites 1-reliés*

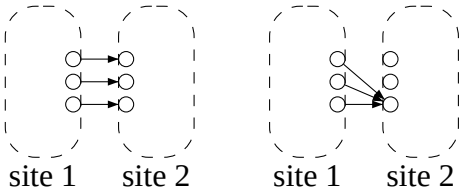


FIG. 3.10 – *Exemples de sites 3-reliés*

cas. Avec des poids sur les arêtes dépendant du degré entrant, cela sera difficile, puisque la force de liaison varie du simple au triple entre les deux cas.

Une altération par pondération des partitions suivant leur taille est une autre solution. D'une manière générale, si l'on considère une partition $V = \bigcup_{i=1}^{p} P_i$ et une fonction de pondération $h : \mathbb{N}^* \to \mathbb{R}$, on peut définir la nouvelle variable $p' = \Sigma_{i=1}^{p} h(|P_i|)$. La solution la plus simple, que nous retiendrons ici, consiste à affecter un poids nul aux partitions monoatomiques (ce qui revient en fait à travailler sur (p, i) en imposant comme règle l'interdiction des partitions monoatomiques). Le choix d'autres fonctions h ne sera pas traité ici, mais on peut espérer en choisissant la bonne fonction h un certain contrôle sur la distribution des partitions, par exemple éviter la formation de trop gros sites en affaiblissant h sur les grandes valeurs.

Fonction f_G choisie

Compte tenue de toutes les remarques que nous venons de faire, nous choisissons de prendre comme fonction d'évaluation la fonction f qui à une partition $\mathcal{S} = (S_1, \ldots, S_p)$ de G associe $p'^{\frac{i}{|E|}}$, où i est le nombre de liens internes et $p' = \Sigma_{i=1}^{p} h(|S_i|)$, avec $h = \chi_{[2,+\infty[}$.

Estimation d'une partition par site à l'aide de l'arbre de décomposition des URLs

Maintenant que nous possédons une fonction f d'évaluation d'une partition se pose la question de réaliser une partition efficace. Une méthode intuitive se base sur l'hypothèse suivante, confortée par l'observation des figures 3.5 et 3.6 : L'architecture URL des pages reflète l'architecture des sites, et donc les blocs sont *a priori* des sous-arbres ou union de sous-arbres de l'arbre du Tree-Graph. Au lieu d'avoir à choisir une partition parmi toutes celles possibles, on pourra se limiter pour le choix des candidats aux candidats coïncidant avec la structure d'arbre.

Cette solution n'est pas parfaite, et il y aura toujours des cas litigieux. Ainsi, dans des configurations en poupée russe, le choix du niveau de partition va être très lié au choix des fonctions f et h. Par contre, des configurations en cinq de dé (un site encastré dans un autre au niveau de la matrice d'adjacence, voir figure 3.11), difficiles à démêler automatiquement avec seulement la matrice d'adjacence (qui porte pourtant plus d'information que le graphe seul), ne poseront pas

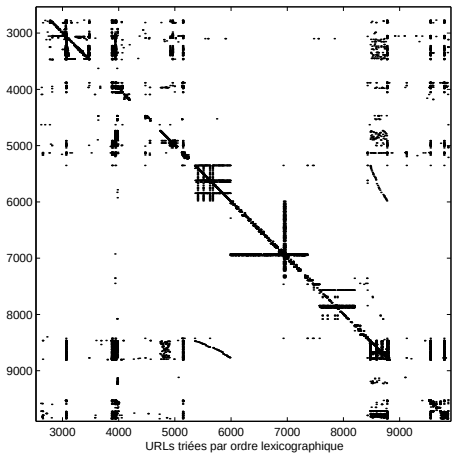


FIG. 3.11 – *Le cinq de dés : un bloc est intercalé au milieu d'un autre*

de problèmes avec l'arbre de décomposition pour peu que les sites coïncident avec l'arbre (ce qui est le cas sur tous les exemples testés).

Nous allons donner un exemple d'algorithme simple pour effectuer une partition intelligente et rapide des URLs à l'aide de l'arbre-graphe : le parcours en largeur filtré (algorithme 1).

Définition 2
On appellera cône lexicographique d'une URL le cluster associé, dans l'arbre de décomposition des URLs, au nœud interne père de la feuille correspondant à l'URL en question.

Les avantages de ce découpage initial sont les suivants :

- On évite d'amalgamer comme un seul site deux ensembles très proches dans l'arbre mais n'ayant aucun lien direct dans le graphe. Ainsi, les sites dits « persos » ne sont pas fusionnés sous le seul label du fournisseur d'espace (free, voila, multimania…) ; de même, la page d'un chercheur dans un laboratoire ne fera partie du site du laboratoire que si elle est référencée par ce dernier, ce qui nous semble satisfaisant.
- À l'inverse, des sites hébergés sur plusieurs serveurs voisins (www.inria.fr et www-rocq.inria.fr par exemple) sont traités comme une seule et même entité.

Il reste hélas des cas que l'algorithme que nous proposons ne prend pas en charge. Ainsi, certains sites sont à cheval sur plusieurs serveurs n'ayant aucun lien de parenté dans l'arbre lexicographique tel que nous l'avons défini (site perso partagé entre plusieurs hébergeurs, site commercial se déclinant en `.com`, `.fr`, `.de`...).

Algorithme 1: Algorithme de Parcours en Largeur Filtré (plf)

Données
Un ensemble d'URLs ainsi que le graphe G associé.
Résultat
Une partition en sites de G, chaque site possédant une page d'entrée.
début
  Aucun sommet n'est marqué
  tant que *Il reste un sommet non marqué*
  faire
    Choisir un sommet non marqué parmi ceux de plus faible hauteur dans l'arbre de décomposition
    Effectuer un parcours en largeur sur le graphe à partir de ce sommet en ne tenant pas compte des arcs sortant du cône lexicographique
    Étiqueter l'ensemble des sommets obtenus d'après le sommet de départ, qui sera appelé *page d'entrée* du site
    si *Un sommet faisant déjà partie d'un site est rencontré*[14]
    alors
      Fusionner les deux sites
    fin
  fin
fin

Résultats

Nous allons comparer ici la partition obtenu par parcours en largeur filtré aux partitions quotientées par serveur, par répertoire à profondeur 1 et par répertoire à profondeur 2 (ce qui correspond typiquement à des coupures à hauteur 3, 4 et 5 dans l'arbre-graphe). L'étude a été faite sur le crawl de 8212791 URLs de `.fr` daté du 15 juin 2001. Par abréviation, nous appellerons ces partitions plf, serveur, 1-rép et 2-rép.

Observons d'abord comment se répartissent les différentes partitions. La figure 3.12 montre les distributions des tailles des différentes partitions. On observe que quelque soit la méthode utilisée (plf, serveur, 1-rép ou 2-rép), les partitions obtenues se répartissent suivant une loi de puissance[15], c'est-à-dire que le nombre de blocs de taille x est de l'ordre de $\frac{C}{x^\alpha}$. Les valeurs des exposants (estimés par régression linéaire) sont indiquées dans le tableau 3.1.

La loi de puissance est une loi considérée comme caractéristique des réseaux sociaux et des

14. cette situation ne se rencontrant que si les deux sommets sont frères.

15. Par abus de langage, nous appellerons souvent loi de puissance tout phénomène qui, représenté sur une échelle *loglog*, donne une courbe formant approximativement une droite. Le fait que le nombre d'échantillons considérés soit fini, et la sensibilité des estimations aux valeurs extrêmes fait qu'il serait plus juste de se contenter de parler de phénomène à *queue lourde* (*heavy tail*).

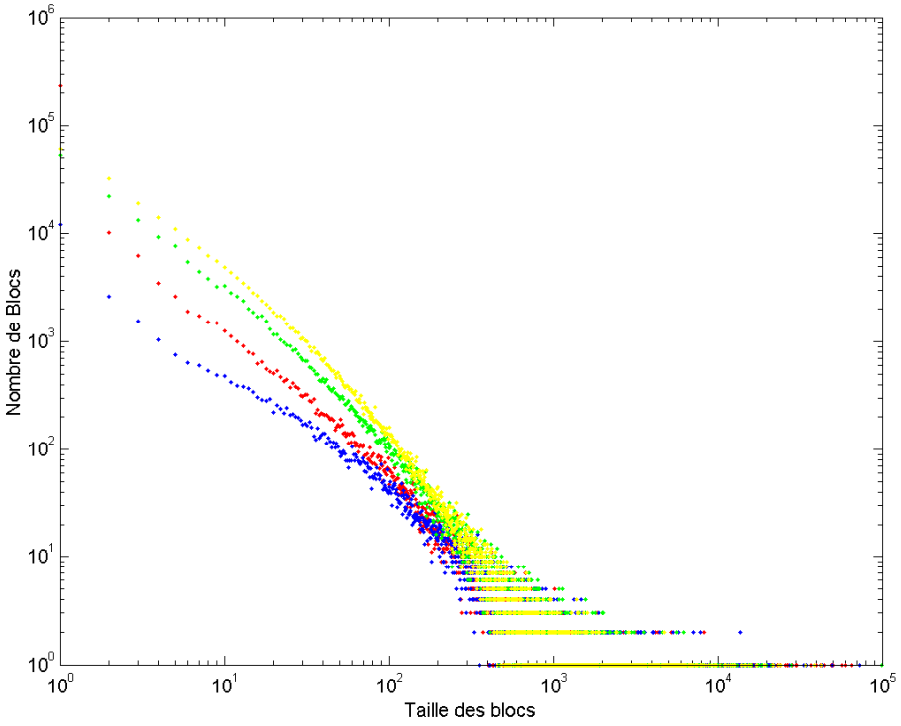


FIG. 3.12 – *Distribution des partitions en fonction de leur taille, sur une double échelle logarithmique*
Légende : plf en rouge, serveur en bleu, 1-rép en vert, 2-rép en jaune

Partition	serveur	1-rép	2-rép	plf
α	1.32	1.75	1.89	1.59

TAB. 3.1 – *Coefficients des loi de puissance des différentes partitions*

activités humaines. Le fait de rencontrer une telle loi dans tous les cas étudiés, à défaut de prouver quoi que ce soit, montre que du seul point de vue de la répartition des pages (sans tenir compte des hyperliens), les quatre répartitions étudiées semblent être compatibles avec la structure humaine du Web.

Remarquons également l'importance des « sites » de taille 1 (les données de la figure 3.12 viennent des partitions avant toute optimisation), et cela plus particulièrement pour la partition plf. L'explication est assez logique : les partitions de taille 1 contiennent entre autres toutes les erreurs qui n'ont pas été supprimées du crawl : erreur 4xx, erreur dans l'écriture de l'URL… La partition plf isole plus d'erreurs que les autres par construction, puisqu'une URL isolée dans le cône lexicographique d'un site sera isolée par plf, à l'opposé des autres partitions.

Enfin, comparons la qualité du découpage en site des différentes partitions, à l'aide de l'indice

	p	p'	$\frac{i}{\lvert E\rvert}$	$p'^{\frac{i}{\lvert E\rvert}}$
serveur	39305	27241	0.9517	16629
plf	289539	58094	0.9218	24624
1-rép	182942	137809	0.7851	10831
2-rép	250869	189698	0.7012	5025

TAB. 3.2 – *Tableau des caractéristiques des différentes partitions avant optimisation*

de site défini par la fonction f. L'optimisation est dans un premier temps minimale : les sites de taille 1 sont fusionnés avec le site avec lequel ils sont le plus liés (le plus souvent, la liaison est unique).

Les résultats sont reportés dans le tableau 3.2. On remarquera que plf, dont les résultats pour p' comme pour i sont à mi-chemin entre serveur et 1-rép, obtient de loin le meilleur indice de site.

Optimisation des partitions

Les résultats précédents proviennent de partition qui ne cherchent pas *a priori* à maximiser l'indice de site. On peut se contenter de ces résultats, si l'on voit simplement l'indice de site comme une indication de qualité, mais on peut aussi chercher à le maximiser. Nous allons donc voir ce que donnent une optimisation locale par mélange, sans plf et avec plf. Le principe de cette optimisation est simple : pour chacun des sites définis par la partition que l'on veut optimiser, on tente d'améliorer l'indice de site en le remplaçant par la décomposition locale (serveur, 1-rép,...) qui donne le meilleur indice de site. Il s'agit donc d'un algorithme glouton, et le résultat obtenu, qui est un maximum local, peut dépendre de l'ordre dans lequel les sites sont traités.

	p'	$\frac{i}{\lvert E\rvert}$	$p'^{\frac{i}{\lvert E\rvert}}$
optimisation sans plf, parcours 1	107884	0.9376	52341
optimisation sans plf, parcours 2	113148	0.9340	52487
optimisation avec plf, parcours 1	121963	0.9338	56165
optimisation avec plf, parcours 2	118386	0.9372	56845

TAB. 3.3 – *Tableau des caractéristiques des différentes partitions après optimisation*

Les résultats sont présentés dans le tableau 3.3. Deux parcours différents des partitions ont été effectués, pour se rendre compte des fluctuations que le choix du parcours peut générer, ainsi que de leur importance. Les principaux résultats que l'on peut tirer de ce tableau sont :

- Un net regain général de l'indice de site, les données p' et i étant proches des valeurs maximales techniquement possibles par cette méthode.

- L'optimisation avec plf reste meilleure dans tous les cas que l'optimisation sans plf, même si la valeur ajoutée est plus faible que pour les partitions brutes.
- Les fluctuations de p' et de i dans les quatre cas semble montrer qu'il risque d'être difficile de déterminer à l'avance le meilleur parcours d'optimisation (problème dû à l'utilisation d'un algorithme glouton).

Deuxième partie

Algorithmes de classement de pages web : les « PageRank »

Chapitre 4

Les chaînes de Markov

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

MR. MONOPOLY *Strategy Wizard*

AVANT d'attaquer l'objet central de cette partie, à savoir le(s) PageRank(s), il nous semble nécessaire de faire un rappel théorique des techniques que nous allons utiliser tout au long des pages à venir.

Andreï Markov (1856-1922), mathématicien russe, est connu pour avoir défini et étudié les processus stochastiques discrets sans mémoire, également connu sous le nom de *chaînes de Markov*. Cet outil du début du siècle dernier est fondamental pour comprendre l'idée du PageRank, c'est pourquoi nous allons brièvement rappeler les résultats essentiels[1].

4.1 Définitions

On appelle *processus aléatoire, ou stochastique, discret* un ensemble de variables aléatoires X_k, avec $k \in \mathbb{N}$. X_k peut par exemple représenter la position sur le plateau de jeu d'un joueur de Monopoly à l'instant k.

Les chaînes de Markov discrètes sont des cas particuliers de processus stochastique discret à valeur discrète dont le futur ne dépend que du présent (pas du passé) : les états précédant l'état présent ne jouent aucun rôle. Si on appelle V l'ensemble des valeurs (ou états) que peuvent prendre les variable X_k, la caractérisation mathématique d'une chaîne de Markov est l'égalité, pour tout $j \in V$,

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

1. Pour une information plus complète sur l'œuvre de Markov, on pourra se reporter à [She88, SC96].

En d'autres termes, la probabilité d'être dans l'état j à l'instant k ne dépend que de la valeur prise à l'instant $k-1$, et pas des valeurs antérieures. *A priori*, cette probabilité peut ne pas être la même selon l'instant k considéré. La chaîne de Markov peut donc se définir par les probabilités de transition entre deux états à un instant k :

$$p_{i,j}^k = P(X_k = j | X_{k-1} = i)$$

Dans tout ce mémoire, nous nous restreindrons au cas où les probabilités de transitions ne dépendent pas de l'instant k considéré. *La chaîne de Markov correspondante est alors dite homogène.*

Si V est fini (nous prendrons alors $V = [\![1, n]\!]$), il est commode de considérer la matrice des transitions $A = (p_{i,j})_{1 \leq i,j \leq n}$. A est une matrice stochastique par ses lignes, c'est-à-dire que $\forall i \in V, \sum_{j=1}^{n} A_{i,j} = 1$. Ceci est dû au fait que le processus est complètement fermé, et que si l'on est dans l'état i à l'instant $k-1$, alors on sera dans V à l'instant k.

La matrice de transition A permet d'obtenir l'évolution de notre processus. En effet, si on appelle x^k le vecteur représentant la distribution des états à l'instant k ($x_j^k = P(X_k = j)$), alors la proposition suivante donne x^k en fonction de x^0 et de A.

Proposition 1
$x^k = (A^t)^k x^0$, *où* ^{t} *est l'opérateur de transposition.*[2]

Preuve : Il suffit de montrer que, pour un $k \geq 1$ quelconque, on a $x^k = A^t x^{k-1}$; la proposition 1 n'est alors que l'application d'une récurrence immédiate. Le résultat voulu vient du fait que :

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

□

4.2 Côté graphe, côté matrice

Nous venons de voir que les matrices stochastiques sont un moyen à la fois élégant et compact de rendre compte de l'évolution d'une chaîne de Markov, mais ce n'est pas le seul. La

2. Dans tout ce mémoire, nous allons user et abuser de l'opérateur de transposition. Pourquoi ne pas travailler directement sur la matrice P^t et ainsi ne pas faire intervenir ^{t} ? D'une part parce qu'il est plus confortable de considérer qu'un coefficient $a_{i,j}$ représente l'action de i sur j que celle de j sur i. D'autre part parce que mes restes de classes préparatoires font que je préfère travailler sur les vecteurs colonne plutôt que sur les vecteurs ligne. Enfin parce que c'est généralement la notation d'usage dans la littérature existante.

représentation en terme de graphe orienté pondéré est également très utile, et il est confortable de pouvoir passer de l'une à l'autre suivant les besoins.

À toute matrice $M = (m_{i,j})_{1\leq i,j\leq n}$ de taille n, il est possible d'associer un graphe $G(M)$ orienté pondéré à n sommets, dont les arêtes sont l'ensemble des couples (i, j) tels que $m_{i,j} \neq 0$, pondérées chacune par leur coefficient $m_{i,j}$ associé.

Réciproquement, à tout graphe orienté pondéré peut correspondre la matrice M définie par :

$$m_{i,j} = \begin{cases} \text{le poids de l'arête } (i,j) \text{ si elle existe.} \\ 0 \text{ sinon.} \end{cases}$$

Dans le cas d'un graphe dont les arêtes sont non pondérées, en les considérant comme implicitement de poids 1, on retrouve la matrice d'adjacence.

Parfois, le passage de la vision matricielle à la représentation sous forme de graphe se limite à une simple réécriture : ainsi, si la caractérisation d'une matrice stochastique[3] $M = (m_{i,j})_{1\leq i,j\leq n}$ est

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

celle d'un graphe $G = (V, E)$ représentant une chaîne de Markov homogène est

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

Il arrive cependant que les deux approches correspondent vraiment à deux visions subtilement différentes d'un même problème. Ainsi, dire qu'une matrice positive M est irréductible revient à dire que

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

Au niveau du graphe G correspondant, cela revient à dire que pour tout couple de sommets (i, j), il existe un chemin (de longueur k) reliant i à j. En d'autres termes, le caractère irréductible de la matrice se traduit par la forte connexité du graphe.

Enfin, signalons qu'une chaîne de Markov est dite *ergodique* si la matrice correspondante est irréductible et apériodique.

4.3 Évolution d'une chaîne de Markov homogène

Afin d'étudier l'évolution à long terme d'une chaîne de Markov, on peut se demander s'il est possible d'avoir des propriétés de convergence. Or, ceci est acquis grâce au théorème suivant.

3. Lorsque cela n'est pas précisé, nous entendons par matrice stochastique une matrice stochastique par ses lignes, c'est-à-dire une matrice positive telle que la somme de chacune de ses lignes soit égale à 1.

Théorème 1 *Soit A une matrice stochastique.*

(a) Le rayon spectral de A est 1, et c'est une valeur propre.

(b) Si A est irréductible, alors il existe un unique vecteur de probabilité P vecteur propre de A^t pour la valeur propre 1, et P est strictement positif, c'est-à-dire que toutes ses composantes sont strictement positives.

(c) Si A est irréductible et apériodique, alors toutes les valeurs propres autres que 1 sont de module strictement inférieur à 1.

Preuve :

(a) 1 est valeur propre, associée au vecteur propre $\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$.

De plus, si l'on considère un vecteur $x = (x_i)_{1 \leq i \leq n}$ quelconque, nous avons toujours

$$\|Ax\|_\infty \leq \|A\,|x|\|_\infty \leq \|x\|_\infty ,$$

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

Ceci implique que toute valeur propre de A est inférieure à 1.

(b) Si A est irréductible, alors on entre dans le cadre du théorème de Perron-Frobenius (voir l'annexe page 151), qui assure qu'il existe, à homothétie près, un unique vecteur propre[4] de A^t pour la valeur propre 1. Ce vecteur étant strictement positif, on peut après normalisation le considérer comme un vecteur de probabilité.

(c) D'après le théorème de Perron-Frobenius, si λ est une valeur propre de A vérifiant $|\lambda| = 1$, alors λ est une racine $d-$ième de l'unité, où d est la cyclicité de A. Si A est apériodique, on a alors forcément $\lambda = 1$. Toutes les valeurs propres de A autres que 1 ont donc une norme strictement inférieure à 1.

□

Grâce au théorème 1, il est maintenant possible de connaître le comportement asymptotique d'une chaîne de Markov homogène.

Théorème 2 *Soit A une matrice stochastique irréductible apériodique de taille n représentant une chaîne de Markov homogène. Si on appelle P le vecteur de probabilité propre droit de A^t pour la valeur 1 (son existence ainsi que son unicité sont garanties par le théorème de Perron-Frobenius), alors*

$$(A^t)^k \underset{k \to \infty}{\to} P.\mathbf{1}_n,$$

où $\mathbf{1}_n$ est le vecteur ligne de taille n ne comportant que des 1.

4. En fait, il y a deux vecteurs propres, un droit et un gauche, le gauche de A étant le droit de A^t et vice versa.

Preuve : Il s'agit d'une simple application de la méthode de la puissance, ou méthode de Jacobi. En effet, comme 1 est valeur propre de dimension 1, on peut décomposer A^t sur l'espace propre engendré par P d'une part et sur l'espace associé aux autres valeurs propres d'autre part (qui n'est pas forcément un espace propre). Ainsi, il existe une matrice de passage T inversible telle que

$$A^t = T. \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & Q & \\ 0 & & & \end{pmatrix} T^{-1},$$

où Q est une matrice de rayon spectral strictement inférieur à 1 (les valeurs propres de Q sont celles de A sauf 1). Le rayon spectral de Q implique que Q^k converge de manière géométrique vers 0. $(A^t)^k$ converge donc de manière géométrique vers :

$$\lim_{k\to\infty} T. \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & Q^k & \\ 0 & & & \end{pmatrix} T^{-1} = T. \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & \mathbf{0} & \\ 0 & & & \end{pmatrix} T^{-1}$$

Cette matrice, que nous appellerons A^t_∞, est de fait une projection, puisque $(A^t_\infty)^2 = A^t_\infty$. L'espace de projection est de dimension 1, car la matrice est de rang 1. Comme par passage à la limite, on a $A^t_\infty P = P$, la droite de projection est celle engendrée par P. A^t_∞ est également une matrice stochastique (car limite de matrices stochastiques). En particulier, si e_i est le vecteur de probabilité sûre en i, défini par $(e_i)_k = \delta^k_i$, alors

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

d'où $A^t_\infty = P.\mathbf{1}$. □

Le comportement asymptotique de la chaîne de Markov homogène associée est alors donné par le corollaire 1 :

Corollaire 1
Soit P_0 une distribution initiale de probabilité quelconque. La distribution des états à l'instant k $P_k = (A^t)^k P_0$ converge, quand k tend vers l'infini, vers P.

Preuve : Pour toute distribution de proabilité P_0, on a $(A^t)^k P_0 \underset{k\to\infty}{\to} (A^t_\infty)P_0 = P.\mathbf{1}.P_0 = P.$ □

Pour le lecteur désireux de se familiariser avec les applications de l'étude asypmtotique des chaînes de Markov, la section suivante est consacrée à une petite analyse des probabilités dans le jeu de Monopoly™.

4.4 Intermezzo : Le Monopoly^{TM} selon Markov

Sources : [Ste96, Col, Gau96]

FIG. 4.1 – *Plateau de Monopoly^{TM}*

À peu près tout le monde connaît le jeu du Monopoly^{TM}, jeu de plateau qui consiste à acquérir des propriétés, constituer des monopoles appelés lotissements (ensembles de propriétés de la même couleur), construire des maisons, puis des hôtels, et ultimement ruiner tous ses adversaires.
Il y a une certaine stratégie au Monopoly. Comme dans la vraie vie financière, tout est question de négociations, de prise de risque et de retour sur investissement. Quel est le rôle des chaînes de Markov dans le Monopoly ? Une des sources de revenus (et de ruine !), la principale en fin de jeu, est l'obligation à chaque tour de payer un loyer au possesseur (s'il existe) de la propriété où son pion s'arrête.

Une question à se poser est : quelles sont les chances de tomber sur une case donnée ? Si certaines cases sont plus probables que d'autres, on comprend bien qu'elles auront un intérêt stratégique accru. On peut voir l'évolution de la position d'un joueur au fil des lancers de dés comme une chaîne de Markov. Ian Stewart [Ste96] associe à chaque état-case 11 transitions possibles correspondant aux résultats possibles d'un lancer de dés, de 2 à 12. La probabilité de chaque transition est celle d'obtenir le résultat avec deux dés. Ian Stewart conclut que la matrice stochastique représentant la chaîne de Markov est circulante, et que la distribution de probabilité asymptotique est la distribution uniforme. En fait, si on regarde de plus près les règles, on s'aperçoit que tous les états ne sont pas équivalents, et que la distribution de probabilité limite n'est pas forcément équiprobable.

4.4.1 Bref rappel des règles et notations

Par convention, on considèrera qu'il y a 41 cases : cela va de la case *Départ* (numéro 0) à la case *Rue de la Paix* (numéro 39), la case *Prison* ayant le numéro 40, la case *Simple Visite* ayant le numéro 10. Le tableau 4.1 récapitule les différentes cases, avec le nom et la couleur de lotissement éventuelle.

Une partie commence sur la case *Départ*. À chaque tour, le joueur lance deux dés. Au bout de trois doubles consécutifs, le joueur va en prison. S'il tombe sur une case *Chance* ou bien *Caisse de Communauté*, il tire une carte dans la pile correspondante, et ce tirage est éventuellement suivi d'un effet immédiat au niveau de la position. Quand on est en prison, on peut en sortir gratuitement en faisant un double dans les trois tours qui suivent celui de l'emprisonnement, sinon on doit payer pour sortir. On peut sortir avant la fin des trois tours en payant.

N°	Nom	Groupe	N°	Nom	Groupe
0	Case Départ		21	Matignon	rouge
1	Belleville	brun	22	Carte Chance	
2	Caisse de Communauté		23	Malesherbes	rouge
3	Lecourbe	brun	24	Henri-Martin	rouge
4	Impôts		25	Gare du Nord	
5	Gare Montparnasse		26	Saint-Honoré	jaune
6	Vaugirard	bleu clair	27	Bourse	jaune
7	Carte Chance		28	Cie des Eaux	
8	Courcelles	bleu clair	29	La Fayette	jaune
9	République	bleu clair	30	Allez en Prison	
10	Simple visite		31	Breteuil	vert
11	La Villette	violet	32	Foch	vert
12	Cie Électricité		33	Caisse de Communauté	
13	Neuilly	violet	34	Capucines	vert
14	Paradis	violet	35	Gare Saint-Lazare	
15	Gare de Lyon		36	Carte Chance	
16	Mozart	orange	37	Champs-Élysées	bleu foncé
17	Caisse de Communauté		38	Taxe de Luxe	
18	Saint-Michel	orange	39	La Paix	bleu foncé
19	Pigalle	orange	40	Prison	
20	Parc Gratuit				

TAB. 4.1 – *Listes des cases du Monopoly^{TM}*

Voici la liste détaillée des cartes *Chance* : 1 envoie en prison, 1 envoie vers l'avenue Henri-Martin, 1 envoie vers boulevard de la Villette, 1 envoie vers la rue de la Paix, 1 envoie vers la gare de Lyon, 1 envoie sur la case Départ, 1 *Reculez de trois cases*. Il y a 9 autres cartes *Chance* qui n'ont aucune influence sur la position

Voici maintenant la liste détaillée des cartes *Caisse de Communauté* : 1 *Retournez à Belleville*, 1 envoie en prison, 1 envoie sur la case Départ, 1 possibilité de tirer une carte *Chance* (alternative avec une amende). Il y a 12 autres cartes *Caisse de Communauté* qui n'ont aucune influence sur la position.

4.4.2 Matrice des transitions

À cause des règles, tous les états n'ont pas les même transitions : ainsi, toute transition vers la case 30 (*Allez en Prison*) doit en fait être remplacée par une transition vers la case 40 (*Prison*). De même, il faut considérer pour toute transition vers les cases *Chance* ou *Caisse de Communauté*, les éventuelles redirections. Il y a aussi le problème des doubles : le processus stochastique utilise

une mémoire (nombre de tours en prison ou nombre de doubles consécutifs déjà faits) et n'est donc pas un vrai processus de Markov. Mais comme la mémoire est finie (3 lancers), on peut se ramener à un processus sans mémoire en considérant un espace à 123 états[5] : 120 états du type $(i,j)_{i\in[0,40],j\in[0,2]}$ représentant *être sur la case i en ayant déjà fait j doubles consécutifs*, et 3 états *prison* représentant les trois tours que l'on peut passer en prison. Il faut d'ailleurs noter que si l'on paie, comme on a souvent intérêt à le faire en début de partie, on ne passe qu'un tour en prison, et les transitions sont donc modifiées. Il faut donc considérer les transitions pour une stratégie *prison* et celle pour une stratégie *liberté*[6].

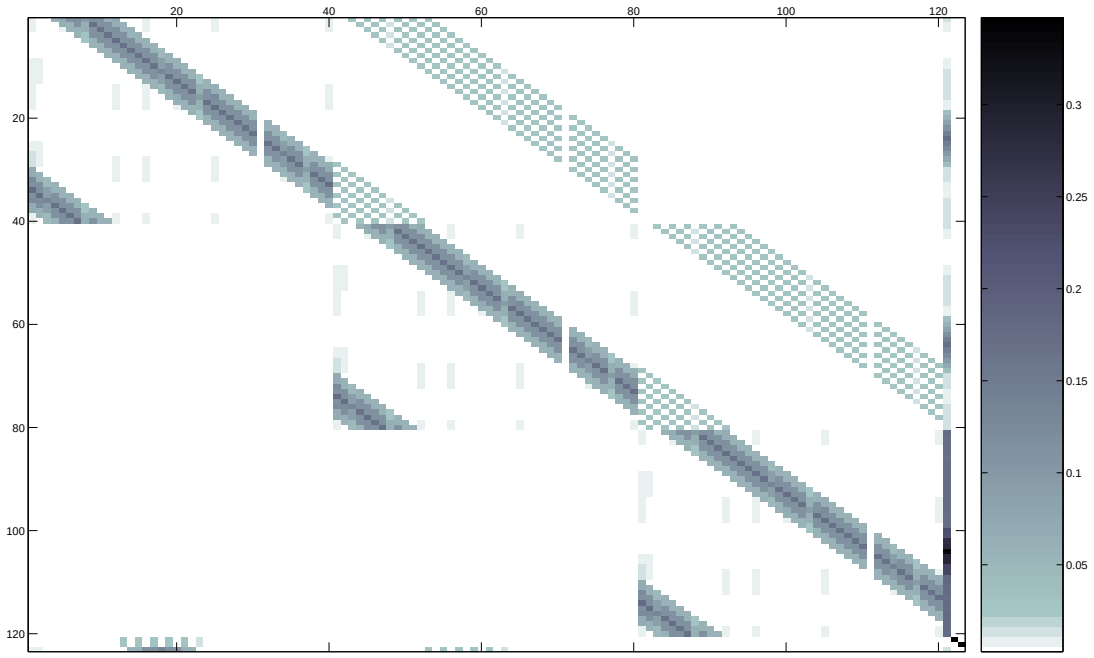


FIG. 4.2 – *Matrice des transitions (stratégie* prison*) dans l'espace*
(case,nombre de doubles) + (prison,nombre de tours)

On arrive ainsi à écrire, pour chacune des 2 stratégies considérées, une matrice stochastique

5. Une solution alternative, proposée par [Col], consiste à estimer pour chaque case la probabilité d'y être arrivé par 2 doubles consécutifs.

6. D'autres facteurs peuvent également altérer les probabilités de transitions :

- La carte *Tirez une Carte Chance ou Payez une Amende de…*, qui propose deux stratégies.
- Les cartes *Sortez de prison*, qui, si elles sont conservées par les joueurs, augmentent légèrement les probabilités de tirer une carte de déplacement.

L'effet de ces variations étant relativement faible, nous nous permettons de ne pas les prendre en compte ici.

décrivant la stratégie en question. La figure 4.2 représente ainsi de manière graphique la matrice correspondant à la stratégie *prison*.

4.4.3 Probabilités asymptotiques et conclusion

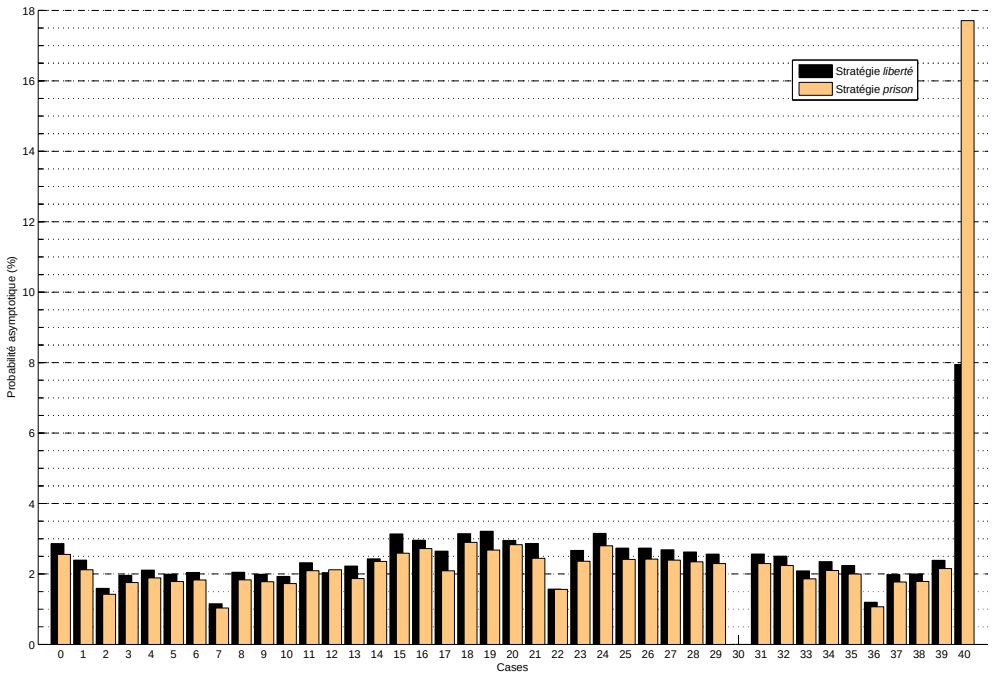


FIG. 4.3 – *Probabilités asymptotiques du jeu de Monopoly^{TM}*

Une fois la matrice des transitions A calculée, grâce au corollaire 1, on sait qu'il suffit d'itérer $P_n = A^t P_{n-1}$ (P_0 étant par exemple la distribution équiprobable) pour avoir une convergence vers la distribution asymptotique. Il n'y a alors plus qu'à revenir dans l'espace à $40+1$ cases pour avoir des résultats exploitables, lesquels sont représentés figure 4.3. Pour compléter cette étude, il faudrait maintenant prendre en compte les prix de vente et les loyers pour avoir un temps moyen de retour sur investissement, sans oublier de considérer le pouvoir d'achat, mais cela n'est plus du ressort des matrices de Markov[7]. Contentons-nous pour conclure de ces quelques remarques :

- La case *Allez en prison* a une probabilité nulle, puisqu'on n'y reste pas.
- De même, les cases *Chance* et *Caisse de Communauté* ont une probabilité assez faible, à cause des cartes de déplacement immédiat.

7. Pour avoir une idée des résultats que l'on obtient, voir [Col]. Attention, les résultats correspondent au jeu de monopoly international, qui comporte des cartes différentes de la version française.

- Les deuxième et troisième quarts du plateau ont globalement des probabilités plus grandes, à cause de la sortie de prison. Cela confère un intérêt certains aux propriétés qui s'y trouvent (les groupes orange et rouge en particulier).
- Paradoxalement, on a plus de chance d'atterrir sur la *Compagnie d'Électricité* en choisissant de rester en prison. Pourquoi ce résultat contre-intuitif ? Avec la stratégie *liberté*, la probabilité de tomber dessus en sortant de prison est de $1/36$. Avec la stratégie *prison*, cette probabilité devient $1/36 + \frac{5}{6}1/36 + \frac{25}{36}1/36\ldots$

4.5 Matrices (sous-)stochastiques : cas général

Au cours des chapitres suivants, nous aurons parfois affaire à des matrices présentant une périodicité, ou qui sont non irréductibles, ou encore sous-stochastiques[8], les *ou* n'étant pas forcément exclusifs. Nous nous proposons donc d'étudier la viabilité du corollaire 1 sous ces différentes hypothèses.

4.5.1 Matrices non irréductibles

Intéressons-nous tout d'abord au cas où $A = (a_{i,j})_{1\leq i,j\leq n}$ est stochastique, mais non irréductible. Cela veut dire que le graphe correspondant $G = (V, E)$ n'est pas fortement connexe. Considérons alors la décomposition en composantes fortement connexes de G : $G = ((C_1, \ldots, C_k), E)$. Chaque composante C_c a exactement une des deux propriétés suivantes :

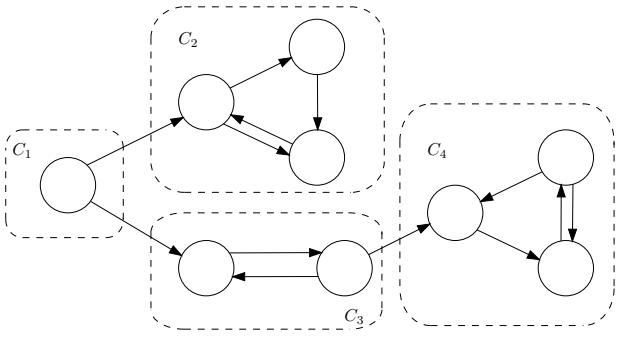


FIG. 4.4 – *Exemple de graphe non fortement connexe*

- Soit $\exists i \in C_c, j \notin C_c, a_{i,j} > 0$. La composante et ses états sont alors dits transitoires.
- Soit $\forall i \in C_c, j \notin C_c, a_{i,j} = 0$. La composante et ses états sont alors dits récurrents.

8. Dans ce cas, on ne peut *a priori* plus parler de chaîne de Markov associée.

Si l'on quotiente G selon ses composantes fortement connexes, le graphe réduit donne un ordre partiel sur les composantes (car il n'y a pas de circuit) dont les composantes récurrentes sont les maxima.

Par exemple, dans le graphe de la figure 4.4, il y a quatre composantes fortement connexes, C_1, C_2, C_3 et C_4. C_1 et C_3 sont transitoires, alors que C_2 et C_4 sont récurrentes.

Nous allons maintenant réordonner les états V comme suit : d'abord les k_t états transitoires (toutes composantes confondues), puis les k_1 états d'une première composante fortement connexe récurrente,..., jusqu'au k_d états de la dernière composante fortement connexe récurrente. Dans ce réarrangement des états, la matrice stochastique associée (que nous continuerons d'appeler A) s'écrit maintenant :

$$A = \begin{pmatrix} T & & E & & \\ & R_1 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ & \vdots & \ddots & \ddots & 0 \\ & 0 & \cdots & 0 & R_d \end{pmatrix},$$

où T est une matrice sous-stochastique, non stochastique, de taille k_t, E une matrice positive non nulle de taille $k_t \times \sum_{i=1}^{d} k_i$, et les R_i des matrices stochastiques irréductibles de taille k_i.

Théorème 3 *Soit A une matrice stochastique réduite selon ses composantes fortement connexes transitoires et récurrentes. A est de la forme*

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

où T est une matrice sous-stochastique, non stochastique, de taille k_t, E une matrice positive non nulle de taille $k_t \times \sum_{i=1}^{d} k_i$, et

$$R = \begin{pmatrix} R_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & R_d \end{pmatrix},$$

les R_i étant des matrices stochastiques irréductibles de taille k_i.

Si toutes les matrices R_i sont apériodiques, alors les puissances itérées de A convergent. Plus précisément, on a :

$$A^k \xrightarrow[k\to\infty]{} \begin{pmatrix} 0 & F \\ 0 & R_\infty \end{pmatrix}, \text{ avec}$$

$$R_\infty = \begin{pmatrix} R_{1\infty} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & R_{d\infty} \end{pmatrix},$$

$$R_{i\infty} = \lim_{k\to\infty} R_i^k = P_{i\infty}.\mathbf{1}_{k_i},$$

où $P_{i\infty}$ est le vecteur de probabilité de taille k_i vérifiant $R_i^t P_{i\infty} = P_{i\infty}$, et

$$F = (Id_{k_t} - T)^{-1} E R_\infty.$$

Corollaire 2
Pour toute distribution de probabilité P_0 sur V, $(A^t)^k P_0$ converge quand k tend vers l'infini, et le vecteur limite est une distribution de probabilité appartenant à l'espace à d dimensions engendré par le plongement canonique des $P_{i\infty}$ dans V.

Remarque 1
Quand tous les R_i ne sont pas apériodiques, il n'y a pas a priori *de convergence. La technique que nous allons voir en 4.5.2 permet quand même de s'assurer une convergence à moindre coût.*

Preuve : On constate tout d'abord que

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

Lemme 1

$$T^k \xrightarrow[k\to\infty]{} 0, \text{ et la convergence est géométrique.}$$

De plus, $(Id_{k_t} - T)$ est inversible, et son inverse vaut $\sum_{k=1}^{\infty} T^k$.

En effet, étudions de plus près la structure de T. Nous allons réordonner les états par composantes fortement connexes, en commençant par celles qui, dans le graphe quotient G/C, n'ont pas d'arête entrante (composantes ultra-transitives), et en triant par éloignement selon la distance aux composantes ultra-transitives. La matrice T est alors de la forme :

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

où les T_i sont des matrices irréductibles sous-stochastiques, non stochastiques (par convention, la matrice unidimensionnelle 0 est considérée comme irréductible). D'après 4.5.3, chaque matrice T_i de la diagonale principale a un rayon spectral $\rho(T_i)$ strictement inférieur à 1. Comme la structure de T est triangulaire, le rayon spectral de T est $\rho(T) = \max_{1\leq i\leq l} \rho(T_i) < 1$. Ceci assure que T^k converge géométriquement vers 0.

Comme 1 n'est pas valeur propre de T, $(Id_{k_t} - T)$ est inversible. Or, pour tout k, on a :

$$(Id_{k_t} - T)\sum_{j=0}^{k} T^j = Id_{k_t} - T^{k+1}, \text{ d'où}$$

$$\sum_{j=0}^{k} T^j = (Id_{k_t} - T)^{-1}(Id_{k_t} - T^{k+1}).$$

On en déduit que

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

ce qui achève la démonstration du lemme.

Revenons à notre démonstration. Il est maintenant acquis que $T^k \xrightarrow[k\to\infty]{} 0$. Avec l'hypothèse d'apériodicité des R_i, nous avons aussi $R^k \xrightarrow[k\to\infty]{} R_\infty$.

Il nous reste à prouver que

$$\sum_{i=0}^{k} T^i E R^{k-i} \xrightarrow[k\to\infty]{} (Id_{k_t} - T)^{-1} E R_\infty.$$

Or, on a

$$\sum_{i=0}^{k} T^i E R^{k-i} = \sum_{i=0}^{k} T^i E R_\infty + \sum_{i=0}^{k} T^i E (R^{k-i} - R_\infty).$$

Le premier terme converge vers $(Id_{k_t} - T)^{-1} E R_\infty$. Quant au deuxième, on a :

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

Le premier des deux termes de droite tend vers 0 à cause de la convergence géométrique de R^k ver R_∞, le deuxième à cause de la convergence (également géométrique) de $\sum_{i=0}^{k} T^k$. Ceci assure la convergence vers 0 du terme de gauche. C.Q.F.D.

□

4.5.2 Matrices périodiques

Si une matrice stochastique A est périodique, il n'y a *a priori* pas de convergence. On peut par exemple penser à la matrice circulante

$$C = (c_{i,j})_{1 \leq i,j \leq n}, \text{ avec} c_{i,j} = \begin{cases} 1 \text{ si } j \equiv (i+1)[n], \\ 0 \text{ sinon.} \end{cases}$$

Les différentes itérations de C décrivent une orbite de taille n correspondant aux racines n-ièmes de l'unité, et il n'y a en particulier pas convergence au sens classique, bien qu'il existe une convergence au sens de Césaro ($\frac{1}{k}\sum_{i=0}^{k} C^i$ converge).

La convergence au sens de Césaro pourrait nous permettre de retrouver la direction propre associée à la valeur propre positive maximale, mais ce n'est pas nécessaire : comme le montre le théorème 4, il est possible de se ramener à une convergence « classique ».

Théorème 4 *Soit A une matrice stochastique irréductible de taille n, éventuellement périodique. Soit P l'unique vecteur de probabilité tel que $A^t P = P$. Pour tout $\alpha \in]0,1[$, on a*

$$(\alpha A^t + (1-\alpha)Id)^k \xrightarrow[k\to\infty]{} P.\mathbf{1}_n$$

Corollaire 3
Soit P_0 un vecteur de probabilité quelconque. Si on pose $B = (\alpha A + (1-\alpha)Id)$, alors

$$(B^t)^k P_0 \xrightarrow[k\to\infty]{} P$$

Preuve : La matrice B définie par $B = (\alpha A + (1-\alpha)Id)$ est stochastique, irréductible, et apériodique, à cause de la présence de circuits de longueur 1. $(B^t)^k$ converge donc vers $Q.\mathbf{1}_n$, où Q est l'unique distribution de probabilité vérifiant $B^t Q = Q$. Comme $B^t P = (\alpha A^t + (1-\alpha)Id)P = \alpha P + (1-\alpha)P = P$, on a $P = Q$.

C.Q.F.D.

□

4.5.3 Matrices sous-stochastiques

Le cas des matrices strictement sous-stochastiques semble *a priori* simple à résoudre :

Théorème 5 *Soit A une matrice sous-stochastique, non stochastique, irréductible de taille n. Alors les puissances itérées de cette matrice tendent vers 0 :*

$$A^k \underset{k\to\infty}{\to} 0.$$

Preuve : A est inférieure et non égale à une matrice stochastique irréductible. D'après le (d) du théorème de Perron-Frobenius (voir page 151), son rayon spectral est strictement inférieur à 1, ce qui assure le résultat, à savoir, si on appelle ρ le rayon spectral, une convergence dominée par une suite géométrique de raison ρ. □

Remarque 2
Une matrice sous-stochastique mais non stochastique correspond à une chaîne de Markov mal définie, dans le sens où toutes les transitions possibles n'ont pas été données. Par analogie avec les automates, on parlera de chaîne de Markov incomplète[9]*. Dans ce cas, pour toute distribution* P_0 *de probabilité,* $(A^t)^k P_0 \underset{k\to\infty}{\to} 0$*, résultat peu satisfaisant en termes d'informations utiles. Ce problème est crucial dans le cadre des calculs de PageRank, et les solutions pour* « *compléter* » *une matrice sous-stochastique seront données plus en détails dans le chapitre 5.*

Remarque 3
Le théorème 5 peut en fait s'appliquer à toute matrice sous-stochastique A *inférieure et non égale à une matrice stochastique irréductible. Nous appellerons de telles matrices des matrices sous-irréductibles.*

Cependant, comme nous le verrons lors du chapitre 5, l'étude de la valeur propre maximale d'une matrice sous-irréductible ainsi que de l'espace propre associé est importante pour l'étude du PageRank, c'est pourquoi nous allons développer un peu.

Valeur propre maximale et espace propre associé d'une matrice positive[10]

Théorème 6 *Soit* A *une matrice positive*[10]*.*

Soient $(C_1, \ldots, C_k)$ *la décomposition en composantes fortement connexes du graphe* G *associé à* A*.*

On définit le rayon spectral $\rho(C_i)$ *d'une composante* C_i *comme étant le rayon spectral de* A_{C_i}*. On appellera composante pseudo-récurrente de* G *toute composante* C_i *vérifiant :*

- $\rho(C_i)$ *est maximal :* $\forall 1 \leq j \leq k$*,* $\rho(C_j) \leq \rho(C_i)$*.*
- *les composantes accessibles à partir de* C_i *ont un rayon spectral strictement inférieur :* $\forall C_j \subset \uparrow C_i$[11]*,* $\rho(C_j) = \rho(C_i) \Rightarrow C_j = C_i$*.*

Alors :

(a) Le rayon spectral de A *est égal au rayon spectral maximum des composantes fortement connexes de* G*.*

9. En effet, tout comme avec les automates, il est possible de compléter notre chaîne de Markov en rajoutant un état *poubelle* recevant le *défaut stochastique* des autres états, et pointant sûrement vers lui-même.

10. Bien que l'étude que nous allons faire nous intéresse avant tout du point de vue des matrices sous-irréductibles, elle est en effet valable pour toute matrice positive.

11. Pour $C \subset V$, le filtre de C, noté $\uparrow C$, est l'ensemble des pages de V accessibles à partir de C.

(b) Il existe une valeur propre maximale qui est positive (c'est la seule si les composantes pseudo-récurrentes sont apériodiques). La dimension de l'espace propre associé est alors égal au nombre d de composantes pseudo-récurrentes.

(c) Si C_i est une composante pseudo-récurrente de G, il existe un vecteur propre positif de support $\uparrow C_i$ associé à la valeur propre maximale.

Corollaire 4
S'il existe une seule composante pseudo-récurrente C_p, et si tous les sommets sont accessibles de cette composante ($\uparrow C_p = V$), alors les espaces propres associés aux valeurs propres maximales[12] *sont de dimension 1, et il existe un vecteur propre positif de support V associée à la valeur propre maximale positive. On dira alors que A est pseudo-irréductible.*

Preuve : La preuve est en fait très similaire à celle du théorème 3 pour les matrices stochastiques non-irréductibles, même s'il n'est plus possible d'avoir un résultat de convergence des puissances de A. Grâce à l'ordre partiel induit par le graphe quotient des composantes fortement connexes, il est possible de rendre A triangulaire par blocs selon les composantes fortement connexes : quitte à se placer dans la bonne permutation, on peut écrire

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

où les T_{ij} sont les matrices de transition entre composantes i et j.

Cette décomposition triangulaire assure que le rayon spectral de A est égal au rayon spectral maximum des différentes composantes C_i. En fait, en appliquant le théorème de Perron-Frobenius sur chacune des composantes fortement connexes, il apparaît qu'il existe une valeur propre $\lambda > 0$ égale au rayon spectral, et que sa multiplicité est égale au nombre de composantes de rayon maximal.

Si C_i est pseudo-récurrente, alors la multiplicité de λ dans $A^t_{\uparrow C_i}$ est 1. Il existe donc à homothétie près un unique vecteur propre x associé à λ dans $A^t_{\uparrow C_i}$. Comme $\uparrow C_i$ est stable par A^t, le plongement de x dans V est un vecteur propre de A^t. Par construction, on a également $A^t_{C_i} x_{C_i} = \lambda x_{C_i}$. D'après le théorème de Perron-Frobenius appliqué sur A_{C_i}, les composantes de x_{C_i} sont strictement positives à homothétie près. De proche en proche, l'égalité $A^t_{\uparrow C_i} x = \lambda x$ montre que x est strictement positif.

Pour chaque composante pseudo-récurrente de G, il existe donc un vecteur propre positif de support $\uparrow C_i$ associé à λ.

Il reste à montrer qu'il n'existe pas de vecteur propre associé à λ en dehors de l'espace engendré par les d vecteurs propres associés aux composantes pseudo-récurrentes. Il suffit pour

12. Le pluriel est employé ici pour les éventuelles périodicités. En cas d'apériodicité, il n'y a bien sûr qu'une seule valeur propre maximale.

cela de constater que toute composante de rayon maximal non pseudo-récurrente C_i génère une structure triangulaire dans l'espace associé à λ : s'il existe une composante pseudo-récurrente C_j avec $C_j \subset\uparrow C_i$, alors $A_{\uparrow C_i}$ contient (dans une base adaptée) un bloc de la forme $\begin{pmatrix} \lambda & \mu \\ 0 & \lambda \end{pmatrix}$, avec $\mu \neq 0$, ce qui montre qu'il n'est pas possible d'associer un vecteur propre de λ à C_i.

C.Q.F.D.

□

4.5.4 Cas général : conclusion

Nous venons de voir qu'on peut s'arranger pour trouver un algorithme de convergence même si la matrice n'est ni apériodique, ni irréductible. Notons simplement que dans ce dernier cas, il n'y a pas unicité. En revanche, si la matrice est sous-stochastique, mais non stochastique, le vecteur asymptotique sera nul partout sauf sur d'éventuelles composantes fortement connexes récurrentes à l'intérieur desquelles la matrice est stochastique. La complétion canonique, qui consiste à rajouter un état « poubelle », n'est pas satisfaisante car elle ne change pas fondamentalement le résultat : l'état « poubelle » récupère les probabilités perdues au niveau des composantes sous-stochastiques, mais les valeurs sur les états autres que l'état *poubelle* ne sont pas modifiées. Heureusement, les divers algorithmes de calcul de PageRank que nous allons étudier maintenant vont nous fournir d'autres méthodes de complétion pour trouver à coup sûr un vecteur ayant toutes les propriétés désirées, qui s'appellera vecteur de PageRank.

Chapitre 5

PageRank, une manière d'estimer l'importance des pages web

Lorsqu'un chercheur veut publier une communication dans une revue spécialisée, il doit passer par des comités de lecture qui évaluent la qualité de son travail selon des critères précis. Pour passer à la télévision ou dans un journal, il suffit d'avoir une bonne histoire à raconter.

Michel DE PRACONTAL*, l'imposture scientifique*

Surfer sur internet c'est comme pour le sexe : tout le monde se vante de faire plus qu'il ne fait. Mais pour le cas d'Internet, on se vante bien plus.

Tom FASULO

Omnes Viae ad Googlem ducent

CE CHAPITRE va essayer de poser les bases à la fois intuitives, axiomatiques et théoriques des méthodes de classement de type *PageRank*, mises en valeur par le célèbre moteur de recherche *Google* [Goo98]. Ce travail fastidieux de *survey* m'a été grandement facilité par Mohamed Bouklit, qui en débroussaillant, avec l'aide d'Alain Jean-Marie, le terrain avant moi [Bou01, BJM02], m'a permis d'avoir toutes les références nécessaires.

Il existe dejà un certain nombre de *surveys* à propos du PageRank (cf [BGS02, BGS03, LM04]), mais nous avons jugé ce chapitre nécessaire, car il présente l'ensemble des PageRanks sous un même point de vue, celui de l'interprétation stochastique, et met en évidence quelques résultats de convergence qu'on ne trouve pas ailleurs[1].

1. Tout au moins pas appliqués aux PageRanks.

5.1 Une aiguille dans un océan de foin…

On trouve (de) tout sur le web, à peu près tous les usagers du réseau sont d'accord là-dessus. Savoir si l'information que je recherche existe sur la toile n'est plus crucial. De nos jours, le problème est devenu : *Comment trouver la page que je recherche ?*

- En connaissant, d'une manière ou d'une autre, l'adresse en question. Par exemple, en lisant sur un paquet de farine l'adresse d'un site qui donne des recettes de cuisine, en recevant un mail d'un ami qui recommande d'aller visiter une adresse web, ou en ayant l'adresse dans ses *Bookmarks*…
- En naviguant (surfant) sur le web, à partir d'une adresse connue. Si par exemple je recherche une page qui parle des ratons laveurs, et si je connais un site de zoologie, partir de ce site peut être une bonne façon de trouver la page que je cherche.
- En utilisant un moteur de recherche. Celui-ci, en échange d'une *requête*, c'est-à-dire d'un ensemble de mots essayant de décrire plus ou moins précisément ce que je recherche, va nous donner une liste de pages susceptibles de répondre à cette requête. D'après [LG99], l'utilisation des moteurs de recherche concerne 85 % des internautes.

Requête	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

TAB. 5.1 – *Nombre de réponses fournies par Google et Yahoo à quelques requêtes (août 2004)*

Tout le problème des moteurs de recherche est d'arriver à renvoyer les pages que l'utilisateur recherche. Seulement, les réponses à une requête donnée se comptent souvent par centaines, voire par milliers[2], comme le montre le tableau 5.1. D'un autre côté, les utilisateurs se lassent vite, et on estime que 90 % des utilisateurs ne dépassent pas la première page de résultats.

Le but des moteurs est donc d'arriver à afficher dans les dix à vingt premières réponses les documents répondant le mieux à la question posée. Dans la pratique, aucune méthode de tri n'est parfaite, mais leur variété offre aux moteurs la possibilité de les combiner pour mieux affiner leurs résultats. Les principales méthodes de tri sont les suivantes :

Tri par pertinence Le tri par pertinence est la méthode de tri la plus ancienne et la plus utilisée. Elle est basée sur le nombre d'occurrences des termes de la recherche dans les pages, de leur proximité, de leur place dans le texte [Sal89, YLYL95]... Malheureusement, cette

2. Le problème est en fait moins crucial si on élimine les doublons — voir tableau 2.1 page 33 — mais il reste important.

méthode présente l'inconvénient d'être facile à détourner par des auteurs désireux de placer leurs pages en tête de liste. Pour cela, il suffit de surcharger la page de mots importants, soit dans l'en-tête, soit en lettres invisibles à l'intérieur du corps de la page.

Étude de l'URL On peut également donner de l'importance à une page en fonction de son URL. Par exemple, selon le contexte, les URLs appartenant au *Top Level Domain* `.com` pourraient avoir plus d'importance que les autres, de même que les URLs contenant la chaîne de caractère `home` [LKVT00]. Il a également été suggéré que les URLs de faible hauteur dans l'arbre-cluster étaient plus importantes que les autres [CGMP98].

Liens entrants Le comptage de citations consiste à attribuer aux pages une importance proportionnelle aux nombres de liens vers cette page connus. Cette méthode a largement été utilisée en scientométrie pour évaluer l'importance des publications [SP63].

PageRank Comme nous allons le voir, le PageRank est une sorte de généralisation récursive du comptage de citations.

5.2 Les deux axiomes du PageRank

Le PageRank, introduit par Brin *et al.* en 1998 [PBMW98], est la méthode de classement qui a fait la spécificité du moteur de recherche *Google* [Goo98]. Il s'agit en fait d'une adaptation au Web de diverses méthodes introduites par les scientomètres[3] depuis les années 1950. Deux méthodes scientométriques en particulier doivent être évoquées :

Le comptage de citations En 1963, Price publie *Little Science, Big Science* [SP63]. Dans ce livre, premier ouvrage majeur traitant de ce qui deviendra plus tard la scientométrie, il propose de mesurer la qualité de la production scientifique grâce, entre autres, à une technique très simple, le comptage de citations : une des manières de mesurer la qualité d'une publication est de dénombrer le nombre de fois où cette publication est citée.

Modélisation markovienne Les chaînes de Markov, décrites dans le chapitre 4, permettent non seulement de jouer au Monopoly™, mais aussi de modéliser l'évolution d'une population répartie entre plusieurs états à condition d'arriver à estimer les probabilités de transition

3. Cela fait maintenant quelques dizaines d'années qu'un nouveau domaine de la recherche est apparu qui se consacre à l'étude de la production intellectuelle humaine. Les principales branches de cette méta-science sont :

Bibliométrie définie en 1969 comme « l'application des mathématiques et des méthodes statistiques aux livres, articles et autres moyens de communication ».

Scientométrie on peut la considérer comme la bibliométrie spécialisée au domaine de l'Information Scientifique et Technique (IST). Toutefois, la scientométrie désigne d'une manière générale l'application de méthodes statistiques à des données quantitatives (économiques, humaines, bibliographiques) caractéristiques de l'état de la science.

Infométrie terme adopté en 1987 par la F.I.D. (International Federation of Documentation, IFD) pour désigner l'ensemble des activités métriques relatives à l'information, couvrant aussi bien la bibliométrie que la scientométrie.

entre états. Par exemple, Goffman propose en 1971 l'étude de l'évolution de la recherche dans différents sous-domaines de la logique à l'aide de chaînes de Markov [Gof71].

Voyons maintenant comment Brin *et al.* ont adapté ces concepts à l'estimation de l'importance des pages Web.

5.2.1 Une page importante est pointée par des pages importantes

Tranposée telle quelle aux graphes du Web, la méthode du comptage de citations revient à dire que l'importance d'une page est proportionnelle à son degré entrant, c'est-à-dire au nombre de pages qui la citent à travers un hyperlien :

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

où $w \to v$ signifie w *qui pointe sur* v.

Bien que cette mesure puisse effectivement être utilisée pour estimer l'importance des pages [CGMP98], elle se trouve en partie dénaturée par l'inexistence d'un contrôle de qualité. En effet, lorsqu'un chercheur veut publier une communication dans une revue spécialisée, il doit passer par des comités de lecture qui évaluent la qualité de son travail selon des critères précis. À cause de cela, le fait même d'être publié donne un minimum d'importance aux articles considérés, et on a une certaine garantie que les citations que reçoit un papier ne sont pas complètement fantaisistes. Dans le cas du Web, ce garde-fou n'existe pas : à cause du faible coût intellectuel et matériel d'une page Web, n'importe qui peut pointer vers n'importe quoi sans que cela ait forcément un véritable sens[4]. Par exemple, pourquoi ne pas créer une multitude de pages vides de sens, mais qui me citent par des hyperliens, pour augmenter à l'envi ma propre importance ?

Brin *et al.* proposent de contrer ce problème par une description récursive de l'importance : Qu'est-ce qu'une page importante ? C'est une page pointée par des pages importantes. Concrètement, si une page v est pointée par k pages $v_1, \ldots, v_k$, l'importance de v doit être définie par :

$$I(v) = f_v(I(v_1), \ldots, I(v_k)) \tag{5.1}$$

Il nous reste à définir f_v et résoudre l'équation (5.1).

5.2.2 Le surfeur aléatoire : le hasard fait bien les choses

Une image d'Épinal du Web est le principe de la navigation par hyperliens : pour trouver ce qu'il cherche, l'internaute va naviguer de page en page et de clic en clic jusqu'à l'arrivée à bon port. Bien sûr, ce n'est pas ce qui se passe en pratique, autant pour des raisons techniques (il n'est

4. Après tout, ne dit-on pas qu'*on trouve de tout sur le Web* ?

pas toujours possible de rejoindre une page p à partir d'une page q) que sociales (utilisation des moteurs de recherche, lassitude de l'internaute...)[5].

Brin *et al.* ont eu pour idée de modéliser le comportement de l'internaute cliqueur par une chaîne de Markov. Tout ce qu'il fallait, c'était trouver les probabilités de transitions d'une page à une autre. Une des manières les plus simples de voir les choses est de considérer qu'une fois sur une page donnée, l'internaute va cliquer de manière équiprobable sur un des liens contenu dans cette page :

$$p_{v,w} = \begin{cases} \dfrac{1}{d(v)} \text{ si } v \to w \\ 0 \text{ sinon} \end{cases}, \text{ où } d \text{ est le degré sortant.}$$

Ceci est la base du modèle du surfeur aléatoire. Pour Brin *et al.*, étant donné que les graphes du Web reflètent une architecture volontaire et réfléchie, les pages intéressantes doivent être structurellement facile d'accès, tout comme une ville est d'autant plus accessible par le réseau routier qu'elle est importante. Donc, comme le surfeur aléatoire se laisse guider par le réseau des hyperliens, statistiquement, il devrait tomber d'autant plus souvent sur une page que celle-ci est importante. D'où l'idée de définir l'importance d'une page Web par la probabilité asymptotique de se trouver sur cette page dans le modèle du surfeur aléatoire.

5.2.3 Cohérence des deux interprétations

Maintenant que nous avons défini deux façons de considérer l'importance d'une page, constatons qu'elles se recoupent et désignent le même phénomène : en effet, dans le modèle du surfeur aléatoire, il est possible d'estimer la probabilité asymptotique de présence sur une page v en fonction de celles des pages w qui pointent vers v :

$$P(v) = \sum_{w \to v} \frac{1}{d(w)} P(w) \tag{5.2}$$

On peut constater qu'on a bien une relation de transfert d'importance comme celle définie par l'équation (5.1), et que (5.2) obéit en plus à un principe de conservation : une page donnée transmet l'intégralité de son importance, celle-ci étant équitablement répartie entre les différentes pages pointées. La probabilité, vue comme une importance, se transmet donc à travers les hyperliens à la manière d'un flot.

5.3 Les modèles classiques

Bien qu'on parle souvent du PageRank au singulier, il existe en réalité une multitude de PageRank(s). Nous allons voir ici comment, à partir du modèle théorique du surfeur aléatoire que

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

nous venons de décrire, plusieurs variations ont été introduites afin de s'adapter à la réalité des graphes du Web. Ce travail de *survey* a déjà été effectué pour l'ensemble des PageRank(s) issus de processus rendus explicitement stochastiques [BGS02, BGS03, LM04], mais nous voulons ici prendre également en compte les modèles sous-stochastiques.

5.3.1 Cas idéal

Dans le cas où le graphe du Web $G = (V, E)$ que l'on veut étudier est apériodique et fortement connecté, les principes vus dans la section 5.2 s'appliquent directement : en effet, il s'agit de rechercher une distribution de probabilité sur V vérifiant :

$$\forall v \in V,\, P(v) = \sum_{w \to v} \frac{P(w)}{d(w)} \tag{5.3}$$

Ceci revient à trouver la distribution asymptotique de la chaîne de Markov homogène dont la matrice de transition est :

$$A = (a_{i,j})_{i,j \in V}, \text{ avec } a_{i,j} = \begin{cases} \dfrac{1}{d(i)} \text{ si } i \to j, \\ 0 \text{ sinon.} \end{cases} \tag{5.4}$$

Comme vu lors de la section 4.3, la suite de distribution

$$P_{n+1} = A^t P_n,$$

initiée par une distribution de probabilité quelconque P_0 [6], converge géométriquement vers l'unique distribution P vérifiant

$$P = A^t P,$$

c'est-à-dire vérifiant la relation (5.3).

Cette distribution de probabilité P est appelée PageRank.

Remarque 4
Comme précisé dans notre définition de graphe du Web, les liens d'une page vers elle-même ne sont pas comptés. D'après [PBMW98], cela permet de « fluidifier » le calcul du PageRank.

Remarque 5
Si le graphe n'est pas fortement connexe, d'après le théorème 3, il y a quand même convergence, mais ni unicité (la dimension de l'espace des solutions est égal au nombre de composantes fortement connexes récurrentes), ni garantie que le support de la solution soit égal à V (en particulier s'il existe des composantes transitoires). L'existence de périodicité(s) peut en revanche empêcher la convergence, mais le théorème 4 indique comment il est possible de contourner le problème.

6. Très souvent, on prend comme vecteur de probabilité initial une distribution de *zap Z* — voir section 5.3.3.

5.3.2 Renormalisation simple

La plupart du temps, un graphe du Web n'est pas fortement connexe. Il existe en particulier un nombre non négligeable de pages sans lien, qui sont soit des pages ne contenant effectivement aucun lien, soit tout simplement des pages connues, mais non indexées. Les lignes de A correspondant à ces pages sans lien ne contiennent donc que des 0, et A est donc strictement sous-stochastique. En conséquent, la suite des P_n va converger[7] vers un vecteur nul en dehors des éventuelles composantes fortement connexes récurrentes sur lesquelles A est stochastique[8]. Pour éviter ce problème, on pourrait envisager d'enlever récursivement toutes les pages sans liens jusqu'à obtenir une matrice stochastique, avec l'inconvénient de considérer un graphe plus petit que le graphe initial. Une autre approche, proposée par [PBMW98], consiste à renormaliser P_n à chaque itération :

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

Ce procédé itératif est une méthode de la puissance ([Ste94]), on sait donc qu'il va converger[7] vers un vecteur propre associé à la plus grande valeur propre de A. Deux cas sont alors à considérer :

- Si la matrice A est sous-irréductible, alors sa valeur propre maximale est strictement inférieure à 1. D'après le théorème 6[9], l'espace propre associé est de dimension d, où d est le nombre de composantes pseudo-récurrentes, et son support est celui engendré par l'union des filtres des composantes pseudo-récurrentes. Dans le cas particulier où A est pseudo-irréductible, l'espace propre est une droite, et il existe un vecteur propre associé strictement positif : grâce à la renormalisation, tout se passe alors comme dans le cas d'une matrice stochastique irréductible.
- Si la matrice A n'est pas sous-irréductible, G contient au moins une composante fortement connexe récurrente sur laquelle A est stochastique. La valeur propre maximale de A est donc 1, ce qui signifie que la renormalisation ne va rien changer au résultat initial : le vecteur propre sera une combinaison linéaire des vecteurs propres sur les différentes composantes fortement connexes sur lesquelles A est stochastique[10].

Processus stochastique équivalent

L'utilisation de matrices sous-stochastiques fait que l'on perd l'interprétation naturelle du surfeur aléatoire. Il est cependant parfois possible de compléter la matrice en une matrice sto-

7. Sous réserve d'apériodicité. Voir la section 4.5.2 page 74 pour les éventuelles périodicités.

8. Compte tenu de la façon dont le graphe est construit, toute composante fortement connexe récurrente non réduite à un élément induit un processus stochastique.

9. Voir page 75.

10. En d'autres termes, les composantes fortement connexes récurrentes.

chastique équivalente, c'est-à-dire de trouver une matrice stochastique B vérifiant :

- $A \leq B$
- si $A^t P = \lambda P$, avec λ maximal, alors $P = B^t P$

S'il existe un unique vecteur de probabilité P maximal pour A^t, la plus simple façon de définir une telle matrice B est de considérer le défaut stochastique de A : si A est sous-stochastique, on définit le défaut stochastique de A comme étant le vecteur $s = \mathbf{1}_n^t - A.\mathbf{1}_n^t$. La matrice

$$B = A + s.P^t$$

est bien une matrice stochastique[11] supérieure à A, et il est facile de voir que $B^t P$ est une distribution de probabilité homogène à P, donc égale à P.

La matrice B présente l'avantage de donner une interprétation stochastique au procédé de renormalisation simple : asymptotiquement, le surfeur aléatoire suit à chaque étape un lien suivant les probabilités données par A. Quand il ne sait pas quoi faire, il *zappe* vers une autre page selon la distribution de probabilité qui est vecteur propre maximal de A.

Par contre, dès que l'espace propre maximal a une dimension supérieure à 1, le problème est délicat, voire très délicat dans le cas de composantes pseudo-récurrentes dont les filtres ont une intersection non nulle. Nous limiterons par conséquent notre étude des processus stochastiques équivalents aux cas d'unicité du vecteur propre maximal.

5.3.3 Complétion stochastique

Afin de remplacer A par une matrice stochastique, et d'éviter à travers une simple renormalisation d'établir un processus stochastique équivalent non contrôlé, une idée naturelle est d'ajouter des transitions aux pages sans liens. Une des méthodes possibles consiste à modéliser l'emploi de la touche *Back*, et sera l'objet du chapitre 6. Une autre méthode, proposée[12] initialement par [PBMW98] (sous la forme de l'algorithme 2) et étudiée entre autres par [LM04], consiste à définir une distribution de probabilité *par défaut* Z sur V, et à s'en servir pour modéliser le comportement du surfeur aléatoire quand il arrive sur une page sans lien. Concrètement, on va remplacer dans A chaque ligne nulle (qui correspond donc à une page sans lien) par la ligne Z^t.

Ce procédé peut se généraliser pour compléter toute matrice sous-stochastique : La complétion par Z de A est alors la matrice stochastique

$$\bar{A} = A + s.Z^t.$$

11. Si A est sous-stochastique et D une distribution de probabilité sur V, par construction, $(A + s.D^t)$ est toujours une matrice stochastique.

12. Sans le savoir ? Voir remarque 6.

Algorithme 2: PageRank : modèle par complétion stochastique (d'après [PBMW98])

Données

- une matrice sous-stochastique A sans périodicité ;
- une distribution de *zap* Z recouvrante ;
- un réel ϵ.

Résultat

Un (le si A est sous-irréductible) vecteur propre de probabilité P de $\bar{A}^t$ associé à la valeur propre 1.

début

$P_0 = Z$

répéter

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

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

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

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

jusqu'à $\delta < \epsilon$

fin

Choix de Z et interprétation

Z représente le comportement du surfeur aléatoire lorsque G ne précise pas où il doit aller, c'est-à-dire tous les changements de page qui ne sont pas dûs à l'usage des hyperliens (adresse tapée manuellement, *Bookmarks*, requête dans un moteur de recherche…). Généralement, on choisit pour Z la distribution uniforme :

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

qui représente un *zap* n'importe où et au hasard sur le Web connu.

Il a également été proposé de « personnaliser » Z, notamment par [PBMW98]. Par exemple, il est possible de se restreindre aux pages d'accueil des sites. D'une part, cela évite de donner implicitement aux sites une mesure partielle d'importance proportionnelle au nombre de pages crawlées (voir section 7.5 page 136). D'autre part, cela obéit à une certaine intuition naturelle : quand on rompt la navigation par hyperliens pour *zapper* sur autre chose, il est probable qu'on va commencer par une page d'accueil. Cette intuition est confirmée par de nombreuses études qui démontrent l'existence de pages qui jouent le rôle de « hubs » pour les utilisateurs réels [CP95, Kle98, WM04, MFJR+04].

Une autre personnalisation envisageable est de prendre pour Z une distribution d'importance. Cela peut être une estimation précédente du PageRank, mais dans le cas de la complétion stochastique, ce n'est pas forcément un choix très heureux, puisqu'asymptotiquement, cela revient à

considérer un processus stochastique équivalent à une renormalisation simple (voir section 5.3.2), et n'apporte donc rien de mieux. On préférera donc souvent une autre estimation d'importance (par exemple le degré entrant).

Irréductibilité de $\bar{A}$

On dira d'une distribution de probabilité Z qu'elle est recouvrante si son support χ_Z vérifie $\uparrow\chi_Z = V$. C'est une condition naturelle à imposer à toute distribution de *zap*, puisqu'elle garantit que toutes les pages connues sont potentiellement accessibles après un *zap*. La distribution uniforme est évidemment recouvrante. Il en est de même de la distribution sur les pages d'accueil si toutes les pages connues d'un site sont accessibles à partir de la page d'accueil. C'est aussi le cas de la distribution par importance si, et seulement si, toutes les pages de V ont un degré entrant non nul.

Théorème 7 *Soit Z une distribution de probabilité recouvrante, et A une matrice strictement sous-stochastique. La complétion par Z de A est irréductible si, et seulement si, A est sous-irréductible.*

Preuve : Si A est sous-irréductible, alors de toute page v de V, il est possible d'accéder à une page présentant un défaut stochastique $w \in \chi_s$. En effet, soit B une matrice stochastique irréductible telle que $A < B$, et un chemin reliant $v \in V$ à $w \in \chi_s$ dans le graphe induit par B. Soit ce chemin existe dans G, et on a ce qu'on veut, soit il n'existe pas, ce qui implique qu'au moins une des pages w' du chemin présente un défaut stochastique, et donc $w' \in \chi_s$.

$\bar{A}$ est donc irréductible, puisque que n'importe quel couple de pages est relié dans le graphe induit par au moins un chemin passant par χ_s.

Réciproquement, si A n'est pas sous-irréductible, il existe au moins une composante fortement connexe stochastique strictement inférieure à V. Cette composante se retrouve inchangée dans $\bar{A}$, ce qui prouve que $\bar{A}$ n'est pas irréductible.

C.Q.F.D. □

Conclusion

La complétion stochastique permet d'avoir une convergence vers une unique distribution de probabilité de support V dès que A est sous-irréductible, alors que dans le cas de la renormalisation simple, il faut que A soit pseudo-irréductible, ce qui est plus restrictif. En revanche, s'il existe des composantes fortement connexes stochastiques (et donc récurrentes), la complétion ne change pas fondamentalement le résultat.

5.3.4 Source de rang : facteur *zap*

Afin de régler le problème de l'irréductibilité, Brin *et al.* [PBMW98] proposent une autre incorporation de la distribution de *zap* Z. Ils remplacent en effet A par $(A + \alpha.(Z.\mathbf{1})^t)$, et cherchent

à résoudre :

$$P = c(A^t + \alpha.Z.\mathbf{1}).P \tag{5.5}$$

Si $\chi_Z = V$, on a alors la garantie d'opérer sur une matrice positive irréductible et apériodique, puisque le graphe sous-jacent est une clique[13]. Le théorème de Perron-Frobenius assure donc la convergence du processus itératif vers le vecteur propre associé à la valeur propre maximale. Par contre, sauf si $\alpha.Z.\mathbf{1} \leq Z.s^t$, la nouvelle matrice n'est pas stochastique, ni même sous-stochastique, ce qui rend l'interprétation en terme de surfeur aléatoire délicate. Afin de pouvoir plus facilement donner une interprétation au processus itératif, Brin et Page normalisent la matrice en prenant la moyenne pondérée par d de A et de $(Z.\mathbf{1})^t$:

$$A \rightarrow \widehat{A} = d.A + (1-d).(Z.\mathbf{1})^t$$

La nouvelle matrice obtenue $\widehat{A}$ possède la (sous-)stochasticité de A, et elle est irréductible et apériodique. Si A est stochastique, $\widehat{A}$ correspond au cas idéal de la section 5.3.1[14]. Le vecteur propre maximal, que nous appellerons PageRank avec facteur *zap*, est alors obtenu par l'algorithme 3. Ce vecteur représente la dernière version académique du PageRank présentée par Brin et Page avant que les méthodes de classement de Google ne deviennent un secret industriel. C'est à lui qu'on fait généralement référence lorsque l'on parle de PageRank.

Remarque 6
À titre d'anecdote, signalons dans l'article fondateur du PageRank ([PBMW98]) une légère confusion : l'algorithme 2 est proposé pour résoudre (5.5)*. Même s'il est précisé que l'introduction de μ peut avoir un léger impact sur l'influence de Z*[15]*, le qualificatif « léger » peut relever de l'euphémisme : dans* (5.5)*, Z permet d'avoir une matrice apériodique irréductible. On a donc la garantie d'un unique vecteur propre strictement positif. En revanche, dans l'algorithme 2, on travaille implicitement sur la complétion par Z de A, et nous venons de voir que l'influence de Z est quasi-nulle dès que A n'est pas sous-irréductible*[16] *(théorème 7). Heureusement, la confusion a disparu dans les articles suivants, notamment avec la normalisation du facteur* zap *[BP98]. Encore que… Langville et Meyer, dans [LM04], proposent pour rendre la matrice irréductible une méthode de type complétion stochastique (voir section 5.6.4) et tombent six ans plus tard dans la même illusion d'irréductibilité, puisque comme nous l'avons vu dans la section 5.3.3, il ne suffit pas forcément de rajouter les clics manquants pour rendre une matrice irréductible.*

13. Si Z est simplement recouvrante, l'irréductibilité est toujours garantie, puisqu'on peut aller de n'importe quelle page vers n'importe quelle autre page *via* une page de χ_Z. En revanche, on n'a plus forcément apériodicité (il existe des contre-exemples simples, par exemple un arbre-branche complété sur sa racine), même si empiriquement, le problème ne se pose pas pour les graphes du Web.

14. Les cas où A est strictement sous-stochastique seront étudiés en détail dans la section 5.4

15. « The use of [μ] may have a small impact on the influence of [Z]. », *op. cit.*

16. Rappelons qu'il suffit pour cela de deux pages qui ne pointent que l'une sur l'autre

Algorithme 3: PageRank : modèle par ajout d'un facteur *zap* (d'après [BP98])

Données

- une matrice stochastique A ;
- une distribution de *zap* Z recouvrante ;
- un coefficient de *zap* $d \in]0,1[$;
- un réel ϵ.

Résultat

Le vecteur propre de probabilité P de $\widehat{A}^t$ associé à la valeur propre maximale.

début

$P_0 = Z$

répéter

$P_{n+1} = d.A^t P_n + (1-d).Z$

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

jusqu'à $\delta < \epsilon$

fin

Interprétation

Lorsque A est stochastique, la double interprétation de $\widehat{A}$ est assez simple :

Transfert d'importance : avec le facteur *zap*, les pages ne transmettent qu'une fraction d de leur importance. En contrepartie, chaque page p se voit attribuer un PageRank Minimal d'Insertion (PRMI) égal à $(1-d).Z(p)$.

Surfeur aléatoire : à chaque étape du processus stochastique décrit par $\widehat{A}$, le surfeur aléatoire va cliquer au hasard sur un des liens sortants, avec une probabilité d, ou *zapper* avec une probabilité $(1-d)$ quelque part sur le graphe en selon la distribution Z.

5.3.5 Choix de d

Avant d'aller plus loin, il convient de parler du choix du paramètre d et des raisons qui ont poussé à faire ce choix. Pour commencer, précisons une réalité empirique universelle et inaltérable : d vaut $0,85$, à $0,05$ près. Depuis les débuts du PageRank, $0,85$ a en effet toujours été la valeur de référence, et à ma connaissance, les calculs pratiques de PageRank suivant le modèle de source de rang vu lors de la section 5.3.4 utilisent toujours un d compris entre $0,8$ et $0,9$.

Compromis convergence/altération du graphe

D'après le théorème 8, si A est stochastique, ce que l'on peut supposer quite à effectuer une complétion, les valeurs propres de $\widehat{A}$ autres que 1 sont inférieures à d en valeur absolue[17]. Cela garantit aux algorithmes 3 et 4 une convergence géométrique de raison au plus égale à d. On a donc intérêt à choisir d le plus petit possible... sauf que plus d est petit, plus l'influence du *zap*, qui est une composante extérieure au graphe intrinsèque du Web, est grande. Un d petit altère, voire dénature le graphe sous-jacent. Choisir le plus grand d garantissant une convergence raisonnable semble donc un bon compromis. Or, les limitations techniques font que le nombre d'itérations réalisables par un moteur de recherche comme *Google* est de l'ordre de la centaine[18]. $d = 0,85$ offre une précision de 10^{-8} au bout de 114 itérations, 10^{-11} au bout de 156 itérations, et semble donc être heuristiquement le compromis recherché. En effet, comme nous le verrons dans la section 5.5 page 99, 10^{-8} correspond au seuil de différenciation d'un graphe du Web d'un million de pages, alors que 10^{-11} est le seuil de différenciation pour un milliard de pages.

Théorème 8 *Soit A une matrice stochastique. Si x est un vecteur propre de $\widehat{A}^t$ associé à $\lambda \neq 1$, alors x est un vecteur propre de A^t et $\widehat{A}^t x = dA^t x$. En particulier, $|\lambda| \leq d$.*

Preuve : Comme **1** est vecteur propre gauche de $\widehat{A}^t$ associé à 1, on a

$$\begin{aligned} \mathbf{1}\widehat{A}^t x &= \mathbf{1}.x \\ &= \lambda \mathbf{1}.x \end{aligned}$$

Comme $\lambda \neq 1$, $\mathbf{1}.x = 0$, d'où

$$\begin{aligned} \widehat{A}^t x &= \lambda x \\ &= (d.A^t + (1-d).(Z.\mathbf{1}))x \\ &= dA^t x \end{aligned}$$

□

Remarque 7
Dans [KHMG03b] se trouve une preuve du fait que toute valeur propre autre que 1 (qui est simple pour $\widehat{A}$) est inférieure à d. Dans [LM04], il est montré en plus que les valeurs propres secondaires de $\widehat{A}$ sont égales à d fois celles de A (les multiplicités de 1 étant comptées comme secondaires), et

17. Si A possède plus d'une composante fortement connexe récurrente, d est une valeur propre.

18. Il faut en effet recalculer le PageRank périodiquement, et avec plusieurs milliards de pages à traiter, chaque itération prend un temps non négligeable.

les auteurs affirment que leur preuve est plus compacte que celle de [KHMG03b]. Le théorème 8 montre en plus que les vecteurs propres secondaires de $\widehat{A}^t$ sont ceux de A^t, et nous affirmons que notre preuve est plus compacte que celle de [LM04]. Il ne reste plus qu'à trouver un théorème plus précis que le théorème 8, avec une preuve plus compacte…

Améliorer le modèle du surfeur aléatoire

L'interprétation du facteur *zap* en terme de surfeur aléatoire, avec à chaque étape une probabilité d de suivre un lien, fait que la longueur des chemins suivis entre deux *zap* suit une loi géométrique de raison d. En particulier, la longueur moyenne d'un chemin entre deux *zap* vaut

$$\sum_{k=0}^{\infty} kd^k(1-d) = \sum_{k=1}^{\infty} d^k = \frac{d}{1-d}$$

Pour $d = 0,85$, cela donne une longueur moyenne entre deux *zaps* successifs d'environ $5,67$. On peut interpréter cela comme le nombre moyen de liens que notre surfeur aléatoire va suivre avant de se lasser et de *zapper* ailleurs. À titre de comparaison, différentes études donnent, suivant l'époque et la méthode employée, des nombres variant entre 3 et 10 [CP95, WM04]. Dans [MFJR$^+$04], il apparaît que la taille moyenne des *trails*[19] est de… $5,6$! Pourtant, on trouve dans ce même article que l'usage des liens de navigations représente seulement $42,5\%$ des moyens d'accès aux pages. De plus, aucune des études citées n'a mis en évidence une répartition selon une loi géométrique.

On conclura donc que le choix pour d de $0,85$ peut s'interpréter comme une manière de modéliser le comportement des surfeurs, mais qu'il ne faut pas forcément considérer cette modélisation comme réaliste.

Conclusion

Nous venons de voir des raisons justifiant le choix $d = 0,85$. J'aimerais conclure par une anecdote concernant l'étude du PageRank à l'intérieur du site de l'INRIA (cf annexe B). Afin de calculer le PageRank local, j'ai testé différentes valeurs de d et constaté expérimentalement les effets obtenus.

- Quand d est trop petit, le classement par PageRank se rapproche de plus en plus du classement par degré entrant. En effet, prendre d petit revient, nous l'avons vu, à réduire la longueur du chemin moyen parcouru par le surfeur aléatoire entre deux *zaps*. Intuitivement, on réduit ainsi la portée de la vision du surfeur. À $d = 0,7$, il ne « voit » qu'à $2,33$ pages, et pour $d = 0,5$, la portée du surf est d'une page. On se rapproche donc de la version non-récursive du PageRank, le comptage de liens entrants (pondéré par Z). Plus

19. Un *trail* (séquence) commence à chaque fois qu'un utilisateur tape une URL à la main, utilise ses *Bookmarks* ou tout autre liste pré-établie pour accéder à une URL.

formellement, il est facile de constater que la distribution stationnaire est égale, à l'ordre 1 en d, à $Z + d.A^t.Z$. Si Z est la distribution uniforme (ce qui est le cas par défaut, et en particulier pour l'étude du PageRank de l'INRIA), quand $d \to 0$, avec $d \neq 0$, le classement induit par la distribution stationnaire est égal au classement par degré entrant.

- Quand d est trop grand, $\widehat{A}$ tend vers A, et le vecteur propre maximal se rapproche d'un vecteur propre maximal de A. En particulier, le PageRank se met à se concentrer sur les composantes fortement connexes récurrentes et autres pièges à PageRank (voir la section 7.5 sur le rôle modérateur de d dans l'auto-amplification du PageRank d'une composante). Plus que le problème de la convergence (la taille du site de l'INRIA autorise suffisamment d'itérations pour toujours obtenir un point fixe) se pose le problème dit des « puits de rang » : le PageRank est « absorbé » par un sous-ensemble de pages, et ne donne donc pas forcément la distribution que l'on souhaiterait.

Ces constatations faites, j'ai essayé de minimiser l'importance d'un couple de pages peu pertinentes, mais possédant un fort degré entrant et situées dans un fort « puit de rang ». Cette minimisation (manuelle) a été atteinte pour… $d = 0,9$. De mon expérience personnelle, il apparaît donc que le choix de d est surtout motivé par des raisons structurelles, même si le choix initial de Brin et Page semble avoir été motivé par la modélisation du surfeur aléatoire [PBMW98, BP98].

5.4 Source de rang et matrices sous-stochastiques

Dans le modèle avec source de rang vu lors de la section 5.3.4, nous avons vu que si l'ajout d'un facteur *zap* associé à une distribution recouvrante Z garantit l'irréductibilité de $\widehat{A}$, la stochasticité de $\widehat{A}$ reste celle de A. Lorsque A est sous-stochastique, il faut donc s'adapter, et nous allons voir dans cette section les principales solutions envisageables.

5.4.1 Modèle hybride : facteur *zap* et renormalisation

$\widehat{A}$ est pseudo-irréductible (puisqu'elle est irréductible). Une première méthode envisageable pour obtenir un point fixe est donc d'appliquer la méthode de la renormalisation simple (cf section 5.3.2) à la matrice $\widehat{A}$.

L'interprétation en terme de surfeur aléatoire est la même que dans le cas où A est stochastique, à ceci près qu'il faut compléter le défaut stochastique de $\widehat{A}$ par un zap selon la loi définie par le vecteur propre maximal de probabilité associé à $\widehat{A}$.

5.4.2 Complétion et source de rang : μ-compensation

La complétion stochastique étant un moyen relativement simple d'assimiler toute matrice sous-stochastique à une matrice stochastique, il est intéressant d'hybrider la méthode de complétion stochastique avec une méthode de type *zap* ou ajout d'une page virtuelle (voir section 5.6).

On évite ainsi de devoir renormaliser à chaque itération, et on a une plus grande cohérence en terme d'interprétation stochastique. De plus, si on choisit une distribution de complétion égale à la distribution de *zap* Z, on obtient un algorithme très simple (l'algorithme 4) : l'algorithme de μ-compensation. L'interprétation est tout aussi simple : à chaque étape du processus stochastique décrit par $\widehat{\bar{A}}$, le surfeur aléatoire va cliquer au hasard sur un des liens sortants (s'il en existe), avec une probabilité d. Dans tous les autres cas, il va zapper selon Z.

Algorithme 4: PageRank : μ-compensation (d'après [KHMG03a])

Données

- une matrice (sous-)stochastique A ;
- une distribution de *zap* et de complétion Z recouvrante ;
- un coefficient de *zap* $d \in]0,1[$;
- un réel ϵ.

Résultat

Le vecteur propre de probabilité P de $\widehat{\bar{A}}^t$ associé à la valeur propre maximale.

début

$P_0 = Z$

répéter

$P_{n+1} = d.A^t P_n$

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

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

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

jusqu'à $\delta < \epsilon$

fin

5.4.3 PageRank non-compensé

L'algorithme de PageRank non-compensé consiste à calculer $P_{n+1} = dA^t P_n + (1-d)Z$, exactement comme dans l'algorithme 3, sans se préoccuper de la renormalisation. On obtient un vecteur strictement positif, qui peut servir à définir un PageRank, comme le montre le théorème 9 et l'interprétation qui s'en suit.

Théorème 9 *Soit A une matrice sous-stochastique, d un facteur de* zap, $0 < d < 1$, *et Z une distribution de probabilité recouvrante pour le graphe induit par A.*

La suite $P_{n+1} = dA^t P_n + (1-d)Z$ converge géométriquement, avec une raison inférieure ou égale à d, vers un unique point fixe P, quelque soit le vecteur initial P_0. P est un vecteur strictement positif, et pour toute complétion $\bar{A}$ de A, si $\bar{P}$ est la distribution de probabilité associée à $\widehat{\bar{A}}$, on a $P \leq \bar{P}$, avec inégalité totalement stricte sauf si A est stochastique (i.e. $A = \bar{A}$).

Plus encore, si le graphe induit possède deux composantes fortement connexes dont les filtres sont disjoints, alors on a

$$P = \sup\left\{X > 0 / \forall \bar{A} \textit{ complétion de } A,\ X \leq \bar{P}\right\}$$

Preuve : Comme $\forall X \in \mathbb{R}^n$, $\|A^t X\|_1 \leq \|X\|_1$, l'application $X \to dA^t X + (1-d)Z$ est d-lipschitzienne. Elle possède donc un point fixe unique vers lequel toute suite $X_{n+1} = dA^t X_n + (1-d)Z$ converge géométriquement, avec une raison inférieure ou égale à d[20].

Ce point fixe P vérifie $P = dA^t P + (1-d)Z$, et vaut donc :

$$P = (1-d)\sum_{k=0}^{\infty}(dA^t)^k Z \tag{5.6}$$

En particulier, il est strictement positif, puisque comme Z est recouvrante, pour tout w dans V, il existe v dans χ_Z, k dans $\mathbb{N}$ tel que $\left((dA)^k\right)_{v,w} > 0$, et donc

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

Considérons maintenant une complétion $\bar{A}$ de A et le PageRank $\bar{P}$ associé. La différence entre $\bar{P}$ et P vaut :

$$\begin{aligned}
\bar{P} - P &= d\bar{A}^t\bar{P} - dA^t P \\
&= dA^t(\bar{P} - P) + d(\bar{A} - A)^t\bar{P} \\
&= d\left(\sum_{k=0}^{\infty}(dA^t)^k\right)(\bar{A} - A)^t\bar{P}
\end{aligned}$$

Comme $\bar{P}$ est strictement positif, $(\bar{A} - A)^t\bar{P}$ est strictement positif sur le support de complétion et nul en dehors.

On peut en conclure que la différence $\bar{P} - P$ est positive. Plus précisément, cette différence est strictement positive sur le filtre du support de complétion et nulle en dehors. Si il existe deux complétions à filtres disjoints, alors en considérant ces deux complétions on déduit donc que

$$P = \sup\left\{X > 0 / \forall \bar{A} \text{ complétion de } A,\ X \leq \bar{P}\right\}$$

C.Q.F.D.

□

20. Notons au passage que dans le cas où A est stochastique, nous avons là une preuve très simple de la convergence de raison d, mais le théorème 8 donne quand même plus d'informations…

Interprétation du modèle non-compensé

Quand A est sous-stochastique, le modèle compensé conserve l'interprétation de transfert d'importance du modèle avec *zap* : il existe un PageRank Minimum d'Insertion (PRMI) de valeur $(1-d)Z$, et chaque page redistribue une part d de son PageRank, à condition qu'elle le puisse (degré sortant non nul).

Dans les modèles compensés, le PRMI est augmenté par complétion ou renormalisation de manière à ce que le flot transitant sur l'ensemble des pages reste une distribution de probabilité, en dépit des pertes dues aux pages sans lien.

Le vecteur P obtenu par l'algorithme non-compensé peut s'interpréter comme la part de PageRank qui ne dépend pas du flot de complétion choisi pour conserver une distribution de probabilité.

5.4.4 Comparaison : algorithmes μ-compensé ou non-compensé ?

Il existe un lien très fort entre l'algorithme μ-compensé et l'algorithme non-compensé : ils donnent le même résultat, comme le montre le théorème 10.

Théorème 10 *Soit A une matrice sous-stochastique, Z une distribution recouvrante. Si $\bar{P}$ est le PageRank obtenu par μ-compensation (cf algorithme 4), et P le PageRank non-compensé, point fixe de l'application $X \to dA^tX + (1-d)Z$, alors P est homogène à $\bar{P}$.*

Preuve : Par passage à la limite, il est facile de voir que $\bar{P}$ vérifie

$$\bar{P} = dA^t\bar{P} + \mu Z, \text{ où } \mu = 1 - \left\|dA^t\bar{P}\right\|_1$$

On en déduit

$$\bar{P} = \mu \sum_{k=0}^{\infty} (dA^t)^k Z \tag{5.7}$$

Les équations 5.6 et 5.7 nous donne $(1-d)\bar{P} = \mu P$. □

Convergence

Les tests que nous avons pu effectuer montrent que avec le choix de Z comme distribution initiale, il n'y a aucune différence significative entre la convergence de l'algorithme de μ-compensation et celui non-compensé. La convergence est dans les deux cas très rapide lors des premières itérations, et se stabilise ensuite vers une convergence de raison d (cf boucle principale de la figure 5.1).

Vitesse des itérations

La μ-compensation doit calculer le paramètre μ à chaque itération, alors que la non-compensation n'en a pas besoin. Est-ce que cela a une grande influence sur les performances ?

- Si la matrice A ne tient pas en mémoire, le facteur limitant dans le calcul d'une itération est la multiplication de A^t par P_n. Le temps utilisé pour compenser est alors négligeable, et la vitesse des itérations n'a alors aucune influence dans le choix entre les deux algorithmes.
- En revanche, si A, ou même simplement la matrice d'adjacence, tient en mémoire, le calcul et l'incorporation de μ prend une durée comparable à celle du calcul de A^tP. En fait, tout calcul de norme alourdit de manière non-négligeable le calcul d'une itération, et même le calcul du paramètre de convergence δ diminue considérablement les performances. On aboutit ainsi à l'algorithme SpeedRank (algorithme 5), qui calcule le PageRank non-compensé en estimant grossièrement le nombre d'itérations nécessaires à une bonne convergence. Nos expériences ont mis en évidence un gain de vitesse de plus de 300% sur les itérations entre SpeedRank et l'algorithme 4, ce qui compense plus que largement la légère sur-estimation du nombre d'itérations nécessaires pour converger.

En terme de performances, l'algorithme μ-compensé est donc à proscrire si l'on travaille sur des « petits » graphes. On s'étonnera au passage que Kamvar *et al.* utilisent la μ-compensation dans leur algorithme BlockRank [KHMG03a], qui est justement basé sur la décomposition du PageRank sur des petits graphes. Pour des spécialistes de l'optimisation du calcul de PageRank (cf [KHMG03b]), il est curieux de passer à côté d'un gain de vitesse de 300%...

Effeuillage-Remplumage

En pratique, les algorithmes de PageRank sont rarement appliqués sur le graphe $G = (V, E)$ tout entier. On utilise très souvent une technique dite d'« effeuillage-remplumage » : le PageRank est d'abord calculé sur le graphe (R, E_R), où R est l'ensemble des sommets de V possédant au moins un lien sortant. (R, E_R) est appelé graphe effeuillé, ou encore rafle. Une fois qu'une bonne convergence est atteinte, on procède au « remplumage » : on se replace sur le graphe G et on effectue quelques itérations de PageRank avec le PageRank sur R comme estimation initiale.

Le problème est que le PageRank sur le graphe effeuillé n'est pas forcément une bonne estimation du PageRank sur G, comme le montre la figure 5.1 : à cause du remplumage, le facteur de convergence δ est quasiment réinitialisé. Si on veut de nouveau atteindre la condition de convergence ($\delta < \epsilon$), il faut effectuer presqu'autant d'itérations qu'en partant de la distribution Z.

La méthode d'effeuillage-remplumage présente quand même un intérêt si on limite le nombre d'itérations de la phase de remplumage : comme la boucle principale se fait sur la rafle, les itérations sont plus rapides. Quant au vecteur final, bien que ce ne soit pas un vecteur stationnaire, il est à mi-chemin entre le PageRank sur la rafle et celui sur G, et le fait que ce soit ce vecteur qui est utilisé en pratique semble indiquer que le classement que l'on obtient est digne d'intérêt.

Algorithme 5: SpeedRank : calcul rapide du PageRank dans le cas où la matrice d'adjacence tient en mémoire vive

Données

- La matrice d'adjacence M d'un graphe $G = ((R \cup S), E)$ quelconque, R étant l'ensemble des sommets de degré sortant non nul ;
- une distribution de probabilité Z recouvrante ;
- un coefficient de *zap* $d \in]0, 1[$;
- un réel ϵ.

Résultat

Avec une précision d'au moins ϵ, un vecteur P homogène au PageRank μ-compensé sur G.

début

$P = Z$

$$C : v \to \begin{cases} \frac{d}{d(v)} \text{ si } v \in R, \\ 0 \text{ sinon.} \end{cases}$$

$Y = (1 - d)Z$

pour i *variant de* 1 *à* $\frac{\ln(\epsilon)}{\ln(d)}$

faire

$P = M^t(C.\times\ P) + Y$ ($.\times$: produit terme à terme)

fin

fin

En quoi la description de la méthode d'effeuillage-remplumage nous aide-t-elle à comparer les algorithmes μ-compensé et non-compensé ? Nous avons démontré que les deux algorithmes convergeaient vers le même vecteur, à homotéthie près. Nous avons également remarqué que les vitesses de convergence étaient quasi-identiques. En revanche, rien ne prouve que les deux convergences suivent le même chemin. Expérimentalement, il semble au contraire que ce ne soit pas le cas : en limitant le remplumage à 4 itérations, nous avons observé que le classement μ-compensé n'était pas identique au classement non-compensé.

Pour estimer cette différence, nous avons calculé le chevauchement à 1% : on considère les pages situées dans le premier centième du classement μ-compensé, et on regarde quelle fraction de ces pages appartient également au premier centième du classement non-compensé. Le chevauchement à 1% s'est avéré être une très bonne méthode pour estimer la différence entre deux PageRank(s) (voir section 6.4.2). Sur les échantillons que nous avons examinés, le chevauchement à 1% entre algorithmes μ-compensé et non-compensé valait $96,8\%$.

S'il n'existe aucune différence entre la μ-compensation et la non-compensation au niveau du comportement asymptotique, l'arrêt de l'algorithme avant convergence génère donc des diffé-

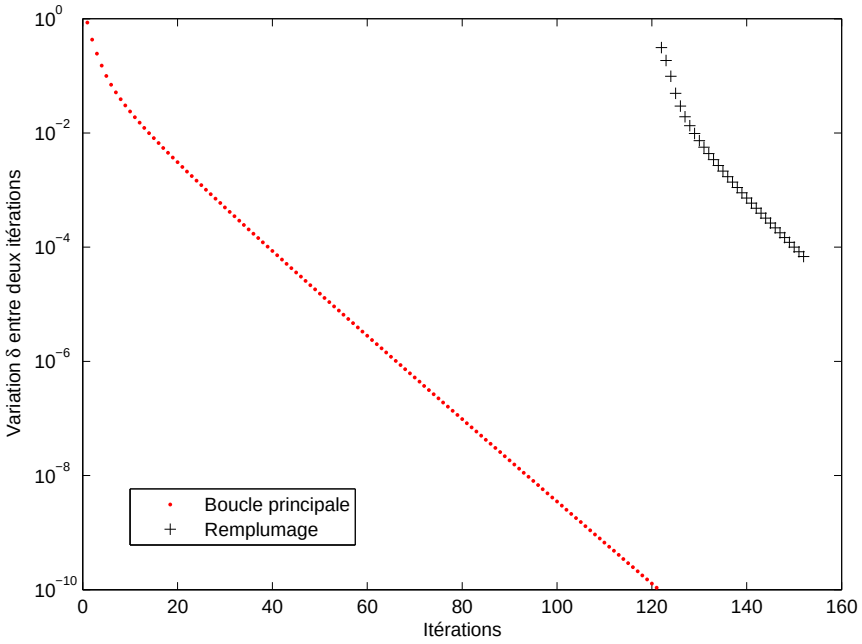


FIG. 5.1 – *Convergence des PageRank μ-compensé et non-compensé ; problème du remplumage*

rences au niveau du classement. Outre les considérations techniques que nous venons de voir, le choix d'un algorithme à la place de l'autre peut également être influencé par les quelques pour cents de différence entre les deux classements, mais une telle étude qualitative sort du sujet de ce mémoire.

5.5 Convergence en norme 1 et convergence du classement

Dans tous les algorithmes de PageRank que nous avons présentés, comme dans tous ceux que nous allons présenter, nous utilisons comme critère de convergence la convergence en norme 1 d'une suite P_n de vecteurs positifs. Ainsi, lorsque qu'une source de rang associée à un facteur *zap* d est utilisée et que le critère de convergence ϵ est atteint, nous savons que l'erreur par rapport au vecteur limite est d'au plus $\frac{\epsilon}{1-d}$. Pourtant, seul le classement induit par P nous intéresse *a priori*, puisque l'intérêt principal du PageRank est de fournir un ordre d'importance sur les pages Web considérées[21].

21. Dans la réalité, les choses sont légèrement différentes. Le classement renvoyé pour une requête donnée est vraisemblablement le résultat de la confrontation de plusieurs estimations d'importances, la pertinence et le PageRank étant les principales d'entre elles. Connaître le PageRank quantitatif des pages peut alors avoir un intérêt.

5.5.1 Distance de Kendall normalisée

Une première solution est de comparer à chaque itération les classements induits, et d'arrêter lorsqu'il n'y a plus de changement. On peut également définir une distance sur les classements et remplacer la convergence en norme 1 par une convergence sur les classements. Une distance assez classique sur les classements est la distance de la différence symétrique, ou distance de Kendall[22] : si σ_1 et σ_2 sont deux classements, présenté sous la forme de permutations, alors la distance de Kendall entre ces deux permutations est le nombre minimum d'inversion de deux éléments conjoints nécessaires pour passer de l'une à l'autre. On peut montrer que cette distance est invariante par translation et que la distance d'une permutation σ à l'identité est :

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

La distance entre deux permutations σ_1 et σ_2 est donc $dist(\sigma_1, \sigma_2) = dist(\sigma_1 \circ \sigma_2^{-1}, Id)$.

Comme la plus grande distance possible $dist_{\max}$ entre deux permutations de taille n est celle entre deux classements inversés, à savoir $\frac{n(n-1)}{2}$, on pourra si l'on veut un critère de convergence indépendant de la taille du classement considérer la distance de kendall normalisée par $dist_{\max}$.

Les applications pratiques de l'étude de la convergence selon la distance de kendall sont encore à l'étude, nous nous conterons donc dans ce mémoire de cette courte introduction théorique.

5.5.2 Densité de PageRank

Par un simple raisonnement d'ordre de grandeur, il est possible d'établir un lien entre ϵ et la convergence du classement. Le point de départ est l'étude du rapport entre le classement d'une page et son PageRank. La figure 5.2 représente ce lien pour deux modèles de PageRank que nous allons étudier plus en détail : le PageRank μ-compensé standard avec *zap* uniforme sur V, ainsi que le PageRank μ-compensé avec technique d'effeuillage-remplumage et *zap* uniforme sur R. Le facteur de *zap* d vaut évidemment $0,85$.

La régularité des courbes[23] nous incite à considérer la densité mésoscopique de pages à un PageRank donné : on cherche à savoir quel est le nombre dN de pages dont le PageRank est compris entre p et $p + dp$. On se place pour cela à l'échelle mésoscopique, c'est-à-dire que l'on suppose $dp \ll p$ et $dN \gg 1$. Expérimentalement, nous avons constaté que l'hypothèse mésoscopique était tout à fait réaliste sur des graphes de plus d'un million de sommets. Nous avons également observé qu'il existait, pour chaque modèle de PageRank, une fonction ρ, relativement indépendante du graphe du Web considéré[24], telle que, si n est le nombre de pages du graphe,

22. Merci à François Durand et à son rapport de maîtrise[Dur03] pour m'avoir fait connaître la distance de Kendall.

23. Pour chacun des deux PageRanks étudiés ici, nous n'avons représenté qu'une seule courbe, mais expérimentalement, les autres échantillons étudiés génèrent des courbes extrêmement similaires.

24. Dans le cas du PageRank avec effeuillage-remplumage, ceci est valable pour une proportion de pages sans lien donnée. Empiriquement, cette constante est souvent un invariant de crawl.

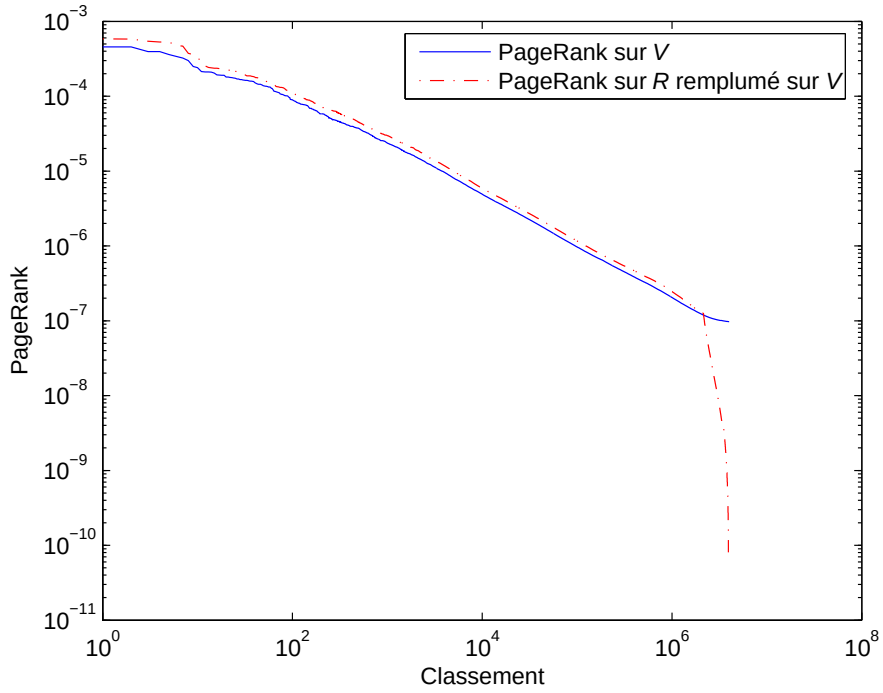


FIG. 5.2 – *Lien entre le classement d'une page et son PageRank*

$$\frac{dN}{dp} \approx n^2 \rho(np) \tag{5.8}$$

ρ est la densité mésoscopique normalisée (indépendante de la taille n du graphe) typique du modèle de PageRank considéré. La figure 5.3 montre des mesures expérimentales de ρ correspondant aux deux modèles étudiés ici. Pour le PageRank sur V avec *zap* uniforme, on constate que ρ est d'autant plus fort que le PageRank est faible, avec une zone extrêmement dense autour du PageRank Minimum d'Insertion : un grand nombre de pages n'ont presque pas d'autre source de PageRank que le *zap*.

Pour le PageRank sur R remplumé sur V, la nullité du PageRank en dehors de R provoque une discontinuité de la densité mésoscopique : au-dessus du PageRank Minimum d'Insertion, la densité présente un profil semblable à celui rencontré dans le cas du PageRank avec *zap* uniforme, avec un maximum plus faible. En dessous du PRMI, le profil de densité correspond aux pages sans lien — qui ne reçoivent donc pas le PRMI — dont le PageRank reçu est inférieur au PRMI. La comparaison des profils de densité des deux modèles nous incite à penser que l'utilisation d'une source de *zap* uniforme de support R au lieu de V permet d'étaler le PageRank des pages sans lien de faible PageRank au lieu de le concentrer autour du PRMI.

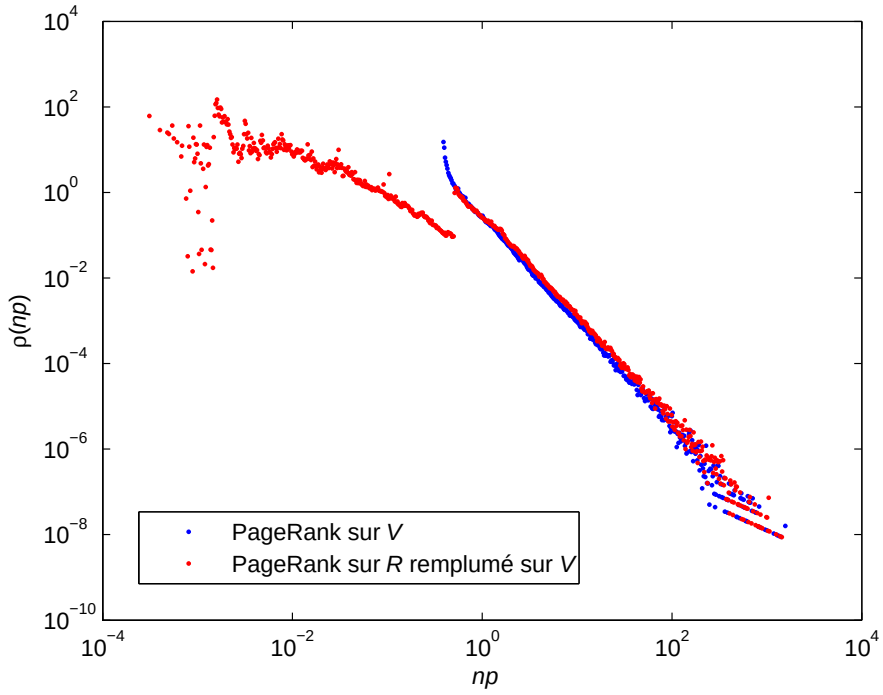


FIG. 5.3 – *Densité mésoscopique normalisée de pages en fonction du PageRank*

5.5.3 Seuil de différenciation

La densité nous fournit un seuil minimum de différenciation naturel : pour différencier les pages dont le PageRank se situe autour de p, il est nécessaire d'avoir pour chaque page une précision de PageRank d'au moins $\frac{1}{n^2\rho(np)}$ (à un facteur 2 près). Ce seuil de différenciation correspond à un cas idéal où les PageRanks des pages seraient régulièrement réparties, et où la différence de PageRank entre 2 pages consécutives serait $\frac{1}{n^2\rho(np)}$. Il nous fournit un ordre de grandeur de la précision à atteindre pour espérer avoir le bon classement de PageRank.

Il reste à relier la précision sur une page à la précision ϵ sur l'ensemble des pages. Expérimentalement, nous avons constaté que pour l'immense majorité des pages de PageRank p, l'erreur était inférieure à $p\epsilon$, mais qu'il existait quelques pages qui dépassaient cette erreur d'un ordre de grandeur.

Si l'on ne tient pas compte de ces pages atypiques, on obtient, si l'on veut se placer sous le seuil de différenciation, la relation

$$p\epsilon < \frac{1}{n^2\rho(np)}, \text{ c'est-à-dire}$$

$$\epsilon < \frac{1}{n(np)\rho(np)}$$

Tout comme pour la densité mésoscopique normalisée, la quantité $\frac{1}{X\rho(X)}$ ne dépend pas de n, mais juste du modèle de PageRank. Nous l'appellerons seuil de différenciation normalisé. La figure 5.4 montre ce que vaut ce seuil de différenciation pour les deux PageRanks considérés ici.

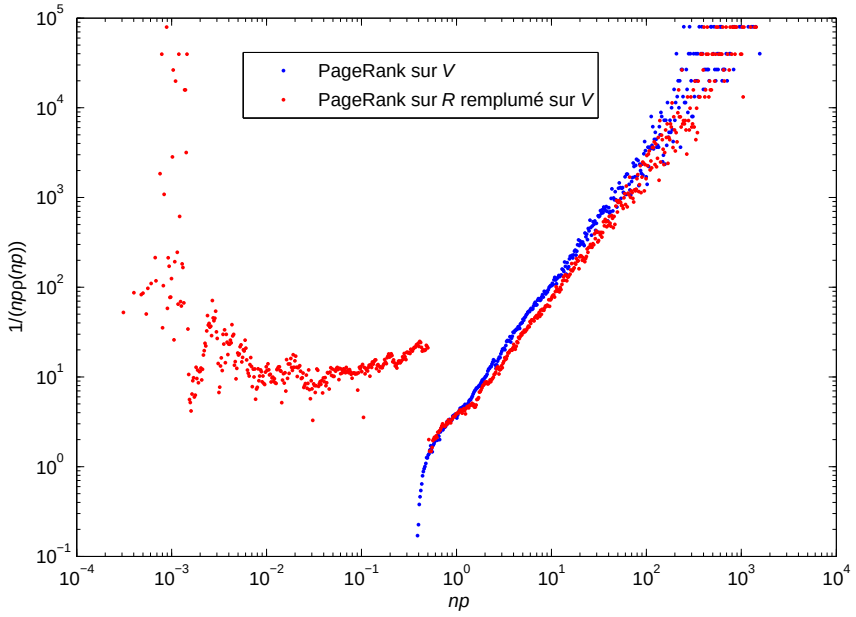


FIG. 5.4 – *Seuil de différenciation normalisé*

Interprétation

Pour un PageRank donné, plus le seuil de différenciation est petit, plus il sera difficile de différencier les pages. On constate, comme l'intuition le laissait prévoir, que la différenciation la plus difficile est toujours située au niveau du PageRank Minimum d'Insertion. Avec un *zap* uniforme sur V, elle vaut environ 10^{-1}, soit une valeur préconisée de ϵ inférieure à $\frac{1}{10n}$. Si le *zap* est uniforme sur R, le seuil minimum pour ϵ est simplement de $\frac{1}{n}$.

Nous suggérons donc, comme choix pratique de ϵ dans les algorithmes de PageRank, de prendre $\epsilon = \frac{1}{10n}$ si le *zap* est uniforme sur R, $\epsilon = \frac{1}{100n}$ s'il est uniforme sur V (nous rajoutons un facteur 10 de sécurité, soit une quinzaine d'itérations pour $d = 0,85$). Restreindre le *zap* sur les pages sans lien permet donc une différenciation plus rapide, au prix d'une dépréciation des pages sans lien de faible PageRank par rapport aux pages avec lien(s) de faible PageRank[25].

Remarque 8

L'étude du seuil de différenciation que nous venons de réaliser donne une explication à la convergence rapide du classement des pages de fort PageRank généralement observée. En effet, on ob-

25. Pour les pages sans lien de fort PageRank, le fait de recevoir ou non un PRMI a relativement peu d'importance.

serve sur la figure 5.4 que le seuil de différenciation est beaucoup plus facile à atteindre pour les pages de fort PageRank. Ainsi, pour les pages dont le PageRank est au moins 100 *fois plus grand que le PageRank moyen, le seuil normalisé vaut* 1000*, soit selon le modèle entre* 45 *et* 60 *itérations de gagnées par rapport à une différenciation sur l'ensemble des pages.*

Remarque 9
Nous avons signalé l'existence de quelques pages pour lesquelles l'erreur de PageRank était bien supérieure à $p\epsilon$*. À cause de ces pages atypiques, nous pouvons être à peu près certain que même avec le facteur* 10 *de sécurité que nous avons choisi, le classement ne sera pas totalement stabilisé.* Avoir un classement stabilisé sur toutes les pages demande beaucoup plus de précision que pour avoir un classement stabilisé sur la plupart des pages. *Cette constatation, confirmée par un projet d'étudiants de l'université de l'Indiana [KL], met en évidence la nécessité de considérer une distance sur les classements (la distance de Kendall) si l'on veut travailler sur une convergence du classement.*

5.6 Modèles avec page virtuelle

L'ajout d'un facteur *zap* est parfois appelé méthode d'irréductibilité maximale, dans le sens où si $\chi_Z = V$, alors le graphe sous-jacent devient une clique. Il est souvent reproché à cette méthode d'être trop intrusive et de dénaturer la structure du graphe du Web. La méthode de complétion, en rajoutant $|\chi_Z| \,.\, |\chi_s|$ liens fictifs, peut aussi être considérée comme intrusive.

Une alternative est d'employer des méthodes dites d'irréductibilité minimale[26], c'est-à-dire d'ajouter une page virtuelle qui va jouer le rôle de « hub ». Ce type de méthodes est entre autres employé par [APC03, Tom03].

5.6.1 Page virtuelle de *zap*

Le principe de la page virtuelle de *zap* est simple : rajouter une $(n+1)^{\text{ème}}$ page, pointée et pointant vers toutes les autres, et contrôlée par d et Z.

Formellement, si A est une matrice stochastique (éventuellement complétée), on considère la matrice

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

Le vecteur asymptotique de probabilité correspondant est obtenu par l'algorithme 6.

26. Cf [Tom03].

Algorithme 6: PageRank : modèle avec page virtuelle

Données

- une matrice stochastique A ;
- une distribution de *zap* Z recouvrante ;
- un coefficient de *zap* $d \in]0,1[$;
- un réel ϵ.

Résultat

Le vecteur propre de probabilité (P, v) de $\tilde{A}^t$ associé à la valeur propre maximale.

début

Choisir un vecteur positif de taille n P_0 et un réel positif v_0 tel que

répéter

$P_{n+1} = d.A^t P_n + v_n.Z$

$v_{n+1} = (1-d)\left\|P_n\right\|_1$

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

jusqu'à $\delta < \epsilon$

fin

5.6.2 De l'utilité de la page virtuelle

Le théorème 11 (voir page 106) montre qu'*a priori*, dès que $d > 0,5$, l'utilisation d'une page virtuelle de *zap* ne change strictement rien par rapport au rajout d'un facteur *zap*, aussi bien au niveau du vecteur asymptotique[27] que de la convergence (les valeurs propres autres que 1 sont inférieures à $\max(d, 1-d)$ en valeur absolue).

Pourquoi donc employer une page virtuelle, puisque les concepts d'irréductibilité maximale et minimale débouchent sur exactement la même chose ? En fait, la page virtuelle peut présenter un avantage pratique à certains problèmes techniques rencontrés lors des calculs de PageRank. Il peut être par exemple intéressant d'avoir tout le PageRank de *zap* concentré sur une page unique au lieu d'être immédiatement redistribué selon Z.

Ainsi, dans *Adaptive on-line page importance computation* [APC03], le PageRank se calcule par un système d'écoulement de *cash* qui réalise très approximativement une itération de A. Par un principe d'ergodicité, l'historique du cash augmente pour chaque page de manière proportionnelle à son PageRank. L'avantage de cette méthode est d'avoir une estimation précise du PageRank à moindre coût. En particulier, il suffit d'un octet par page pour gérer l'écoulement du *cash*. En revanche, il est assez délicat de répartir $(1-d)$ fois un nombre stocké sur un octet à travers toutes les pages de V. C'est là qu'intervient la page virtuelle. En lui donnant une capacité de *cash* plus importante, la gestion du *zap* devient plus facile. Cela autorise de plus une plus grande adaptativité de la redistribution Z, que l'on peut faire évoluer dynamiquement selon

27. En effet, à un poids de $(1-d)$ sur la page virtuelle près, les vecteurs propres maximaux sont identiques.

certains besoins [APC03].

Théorème 11 *Soit A une matrice stochastique. Comme $\tilde{A}$ est stochastique, irréductible et apériodique, on sait que 1 est une valeur propre singulière et dominante. Plus précisément, si $(\lambda, (P, v))$ vérifie $\tilde{A}^t(P, v) = \lambda(P, v)$, alors on est dans l'un des 3 cas suivants :*

- *Soit $\lambda = 1$, et (P, v) est alors homogène à $(\widehat{P}, (1-d))$, où $\widehat{P}$ est la distribution de probabilité telle que $\widehat{P} = \widehat{A}^t\widehat{P}$.*
- *Soit $\lambda = d - 1$.*
- *Soit λ ne vaut ni 1, ni $d-1$. P est alors un vecteur propre de A^t, v est nul, et on a $\tilde{A}^t(P, 0) = (dA^tP, 0)$. En particulier, $|\lambda| \leq d$.*

Preuve : Il est facile de vérifier que $(\widehat{P}, (1-d))$ est vecteur propre de $\tilde{A}^t$ associé à 1 :

$$\begin{aligned}
\tilde{A}^t \begin{pmatrix} \widehat{P} \\ 1-d \end{pmatrix} &= \begin{pmatrix} dA^t & Z \\ (1-d)\mathbf{1} & 0 \end{pmatrix} \begin{pmatrix} \widehat{P} \\ 1-d \end{pmatrix} \\
&= \begin{pmatrix} d.A^t\widehat{P} + (1-d).Z \\ (1-d)\left\|\widehat{P}\right\|_1 \end{pmatrix} \\
&= \begin{pmatrix} \widehat{P} \\ 1-d \end{pmatrix}
\end{aligned}$$

Si $\lambda = 1$, (P, v) est donc bien homogène à $(\widehat{P}, (1-d))$.

Si $\lambda \neq 1$, alors nous savons (cf preuve du théorème 8) que $(\mathbf{1}_{n+1}. \begin{pmatrix} P \\ v \end{pmatrix}) = 0$, d'où

$$\begin{aligned}
\tilde{A}^t \begin{pmatrix} P \\ v \end{pmatrix} &= \begin{pmatrix} dA^t & Z \\ (1-d)\mathbf{1} & 0 \end{pmatrix} \begin{pmatrix} P \\ v \end{pmatrix} \\
&= \begin{pmatrix} dA^tP + vZ \\ (1-d)\mathbf{1}.P \end{pmatrix} \\
&= \begin{pmatrix} dA^tP + vZ \\ (d-1).v \end{pmatrix} \\
&= \begin{pmatrix} \lambda P \\ \lambda v \end{pmatrix}
\end{aligned}$$

2 cas sont alors à considérer :

- Soit $\lambda = (d-1)$. On est vraisemblablement en présence d'une valeur propre introduite par l'ajout de la page virtuelle. Nous ne chercherons pas à approfondir ce cas.

- Soit $\lambda \neq (d-1)$. Puisque $\lambda v = (d-1)v$, on a $v = 0$ et $\lambda P = dA^t P$.

C.Q.F.D.

□

Remarque 10
La preuve du théorème 11 est dans le même esprit que pour le théorème 8. Nous voulons en particulier être plus compact que la preuve donnée dans [LM04], et nous affirmons que notre théorème, en plus d'être plus complet, est juste. En effet, selon [LM04], la valeur propre étrangère introduite par l'ajout de la page virtuelle est 0[28]*. Nous affirmons dans le théorème 11 qu'elle vaut en fait* $(d-1)$*. Une confirmation rapide par l'exemple est donnée par l'étude des valeurs propres de*

$$\tilde{A} = \begin{pmatrix} 0,85 & 0,15 \\ 1 & 0 \end{pmatrix}, \text{ qui sont } 1 \text{ et } -0,15.$$

5.6.3 Page virtuelle de complétion

Une variation intéressante du modèle de la page virtuelle permet de supprimer le facteur *zap* et d'avoir quand même une matrice irréductible : mettre les liens vers la page virtuelle au même niveau que les liens réels : quand on se trouve sur une page v, on considère qu'il existe $d(v)+1$ liens sortants (avec la page virtuelle) et que le choix est équiprobable. La page virtuelle redistribue quand à elle son PageRank selon une distribution recouvrante Z.

Concrètement, la matrice basique des transitions est modifiée : on considère la nouvelle matrice

$$\underline{A} = (\underline{a}_{i,j})_{i,j \in V}, \text{ avec } \underline{a}_{i,j} = \begin{cases} \dfrac{1}{d(i)+1} \text{ si } i \to j, \\ 0 \text{ sinon.} \end{cases} \tag{5.9}$$

Par construction, $\underline{A}$ est sous-irréductible, puisque son défaut de stochasticité T défini par $T(i) = \frac{1}{d(i)+1}$ a V pour support. La matrice de transition sur les n pages réelles plus la page virtuelle est alors :

$$\tilde{\underline{A}} = \begin{pmatrix} \underline{A} & T \\ Z^t & 0 \end{pmatrix}$$

La matrice obtenue est stochastique, irréductible et apériodique, la convergence est donc acquise. Nous avons également les résultats suivants (applications et adaptation des théorèmes 6, 7 et 11) :

- dans le pire des cas, la raison de la convergence est $\frac{d_{\max}}{d_{\max}+1}$

28. Erratum : depuis la version du 20 octobre 2004 (après soumission du présent mémoire aux rapporteurs), l'article de Langville et Meyer trouve aussi $d-1$ comme valeur introduite, et la preuve est beaucoup plus légère.

- il est équivalent de considérer $\underline{\tilde{A}}$ et $\underline{\overline{A}}$, complétion par Z de $\underline{A}$. On peut donc tout aussi bien utiliser l'algorithme 2, en remplaçant A par $\underline{A}$.

Algorithme 7: Algorithme hybride : page virtuelle de complétion et μ-compensation

---

Données

- une matrice d'adjacence M d'un graphe quelconque ;
- une distribution de probabilité Z recouvrante ;
- un coefficient de *zap* $d \in]0,1[$;
- un réel ϵ.

Résultat

Le vecteur propre de probabilité P de $\underline{\widehat{\tilde{A}}}^t$ associé à la valeur propre maximale.

début

$P_0 = Z$

$T = 1./(M\mathbf{1}_n^t + \mathbf{1}_n^t)$ ($1./$: inverse terme à terme)

répéter

$P_{n+1} = d.M^t(T.\times P_n)$ ($.\times$: produit terme à terme)

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

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

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

jusqu'à $\delta < \epsilon$

fin

Il faut noter que le fait de supprimer le facteur d n'est pas forcément un avantage : dans ce modèle il n'est pas possible de contrôler la convergence ou le comportement du surfeur associé. Si l'on veut garder l'idée du lien virtuel qui vaut un lien réel, on aura intérêt à considérer un modèle hybride avec source de rang, par exemple basé sur l'algorithme 4 (PageRank μ-compensé). Ce nouveau modèle correspondra à l'ajout d'une source de rang sur une complétion de $\underline{A}$ (la matrice implicite est alors $\underline{\widehat{\tilde{A}}}$). On obtient ainsi l'algorithme 7.

5.6.4 Erratum

Pour en finir avec les pages virtuelles, signalons que dans [LM04], un modèle de complétion stochastique par page virtuelle incorporant le facteur *zap* est proposé. Malgré tous mes efforts, je n'ai hélas pas réussi à voir comment la matrice de transition associée,

$$\begin{pmatrix} dA & (1-d)s \\ (1-d)Z^t & d \end{pmatrix}, \text{ où } s \text{ est le défaut stochastique de } A \text{ (pas de } \underline{A}),$$

peut être stochastique si on ne supprime pas d, alors que les auteurs affirment qu'elle l'est. Comme il est dit explicitement qu'on obtient une irréductibilité grâce à cette matrice et que dans le processus stochastique équivalent, on ne *zappe* qu'à partir des pages sans liens[29], je me permets d'émettre quelques doutes sur la rigueur de ce modèle.

5.7 Gestion des ressources physiques

Avant de clore ce chapitre, je voudrais mettre en avant quelques considérations techniques fondamentales. Le lecteur aura peut-être remarqué qu'aucun des algorithmes présenté ne fait jamais intervenir explicitement les matrices dont on calcule le vecteur propre maximale ($\bar{A}$, $\widehat{A}$, $\widehat{\bar{A}}$ et $\tilde{A}$). C'est un phénomène général : si on veut que toutes les constantes de l'algorithme puissent tenir en mémoire vive[30], il est nécessaire de réfléchir un minimum. En effet, il faut se souvenir que A est une matrice creuse contenant $n.\overline{d}$ éléments non nuls, où $\overline{d}$ est le degré moyen. $\overline{d}$ étant relativement constant (entre 7 et 11 selon la prise en compte des pages non visitées et l'éventuel filtrage des liens), cela fait approximativement une taille linéaire en n, ce qui est à la fois beaucoup compte tenu des tailles des graphes considérés, et un minimum si on veut utiliser la structure du Web. Les matrices implicites générées par complétion et usage du facteur *zap* sont loin d'être creuses, et leur taille peut être en n^2. On voit ici l'intérêt d'user d'astuces de réécriture dans les algorithmes de PageRank afin de ne jamais faire intervenir de matrice moins creuse que A.

Personnellement, j'effectue mes expériences de PageRank avec simplement la matrice d'adjacence, stockée sous forme de matrice logique creuse (*logical sparse matrix*) et quelques vecteurs de taille n : $D : i \to d(i)$, Z,s,T... Les algorithmes 5 et 7 donnent des exemples du traitement effectif des algorithmes. Il est ainsi possible de traiter jusqu'à 8 millions de sommets sur un PC domestique avec 1 Go de mémoire vive, en conservant toutes les constantes en mémoire et en réalisant un minimum d'opérations[31]. Pour aller plus loin, il faut utiliser des techniques de compression transparente du graphe[32].

29. Ce qui ne garantit en général pas l'irréductibilité, cf remarque 6.

30. Il est possible, voire nécessaire pour les très grands graphes, d'effectuer des calculs de PageRank sans charger toutes les constantes en mémoire vive et avec des accès disques optimisés [APC03], mais chercher à minimiser la taille des constantes reste très important quand on veut faire un algorithme de PageRank.

31. Cela peut paraître peu quand on sait qu'un Go de mémoire vive permet de stocker le PageRank de 125 millions de pages en double flottant. Il faut cependant toujours se rappeler du gain colossal que l'on obtient quand la matrice d'adjacence est en mémoire vive.

32. Voir par exemple le projet Webgraph [BV].

Chapitre 6

BackRank

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

Back to the Future

La vraie nouveauté naît toujours dans le retour aux sources.

Edgar MORIN*, Amour, poésie, sagesse*

NOUS venons de voir les principes théoriques généraux qui régissent les différents algorithmes de PageRank. En particulier, il est apparu que le problème de la complétion stochastique et celui de la diffusion du PageRank peuvent, voire doivent, être traités indépendamment. Dans ce chapitre, qui est le prolongement d'une série d'articles co-écrits avec Mohamed Bouklit [BM03, MB04], nous allons étudier une manière d'effectuer la complétion stochastique tout en perfectionnant le modèle du surfeur aléatoire : prendre en compte la possibilité qu'a l'utilisateur de pouvoir à tout instant de sa navigation revenir en arrière grâce à la touche *Back* de son navigateur[1]. Comme nous allons le voir, l'algorithme de PageRank qui résulte de cette modélisation présente de nombreux avantages par rapport aux PageRanks classiques.

6.1 De l'importance de la touche *Back*

En 1995, Catledge et Pitkow publient une étude dans laquelle il apparaît que la touche *Back* représente 41% de l'ensemble des actions de navigation recueillies (52% des actions étant l'usage d'hyperliens) [CP95]. Une autre étude, par Tauscher et Greenberg, datant de 1997, rend compte

1. Je présente par avance mes plus plates excuses aux défenseurs de la langue française, mais j'avoue préférer l'anglicisme *Back* à *page précédente*.

de 50% d'hyperliens et de 30% de *Back* [TG97]. Citons enfin une étude plus récente (2004) de Milic *et al.* [MFJR$^+$04], dont les principaux résultats sont portés dans le tableau 6.1

Hyperliens	$42,5\%$	*Bookmarks*	$2,9\%$
Touche *Back*	$22,7\%$	URL écrite à la main	$1,6\%$
Nouvelle session	$11,2\%$	Page de démarrage	$1,1\%$
Autres méthodes[a]	$10,7\%$	Touche *Actualiser*	$0,5\%$
Formulaires	$6,6\%$	Touche *Forward*	$0,2\%$

TAB. 6.1 – *Étude statistique des modes de navigation des surfeurs réels (d'après [MFJR$^+$04])*

[a] Chargements dynamiques, redirections automatiques, complétion automatique d'adresse…

De toutes ces études, il semble émerger un certain recul de l'emploi de la touche *Back* entre 1995 et 2004, mais selon [CM01], il faut tenir compte des évolutions survenues pendant cet intervalle[2], ainsi que de changements dans les protocoles expérimentaux utilisés.

L'autre information intéressante que nous donnent ces chiffres est le fait que dans tous les cas, on constate que l'emploi de la touche *Back* arrive très largement en deuxième position, derrière l'utilisation des hyperliens qui reste le mode de navigation privilégié. Il faut en particulier retenir que selon toutes les études, pour 2 clics sur un hyperlien, il y a au moins 1 utilisation de la touche *Back*. Pourtant, l'utilisation de la touche *Back* n'a pas d'équivalent dans les PageRank(s) « classiques », alors que des modes de navigation quantitativement moins importants comme les URLs tapées à la main et les *Bookmarks* peuvent être pris en compte dans les algorithmes de calcul à travers le vecteur de *zap*.

Même si l'on ne sait pas vraiment à ce jour si un modèle plus proche du surfeur réel donne forcément un meilleur PageRank, l'importance de la touche *Back* dans le comportement des internautes réels nous est apparu comme une motivation suffisante pour étudier les possibilités de l'incorporer dans un nouveau modèle de PageRank.

6.2 Travaux antérieurs et contemporains

Kleinberg, dans l'algorithme HITS [Kle98], utilise aussi la matrice des liens entrants, et travaille donc sur un modèle où remonter les hyperliens est implicitement pris en compte. Néanmoins, le but de l'algorithme HITS n'est pas de modéliser la touche *Back* mais de voir le graphe du Web comme une superposition de « hubs » et d'« autorités », les premiers pointant vers les secondes.

2. À l'échelle du Web, si 9 millions de pages représentent une brindille, 9 années représentent une éternité. Dans cet intervalle, l'ergonomie des navigateurs a évoluée (*Bookmarks*, complétion automatique des URLs, consultation de l'historique…), de même que le comportement des internautes a été modifié par l'omni-présence des moteurs de recherche. Ainsi, plutôt que de revenir en arrière, certains préfèrent relancer dans leur moteur de recherche la requête qui a permis d'accéder à la page d'où ils viennent.

D'un autre côté, Fagin *et al.*, dans [FKK+00] proposent une étude très complète de l'influence de l'introduction de retours arrières dans une marche aléatoire, et proposent une modélisation stochastique du surfeur aléatoire tenant compte de la touche *Back*. Dans ce modèle une chaîne de Markov initiale *classique* est considérée, à laquelle on adjoint un historique de capacité infinie des pages visitées. À chaque page v est associée une probabilité $\alpha(i)$ de revenir en arrière, à condition que l'historique soit non vide. Les principaux résultats obtenus sont :

- Suivant les probabilités de retour arrière, le processus peut être transitoire (asymptotiquement, la page de départ est « oubliée » et fini par ne plus influer sur la distribution de probabilité) ou ergodique (les retours arrières ramènent le surfeur infiniment souvent sur la page de départ), avec une transition de phase entre les deux.
- Dans le cas particulier où la probabilité de retour α ne dépend pas de la page, si $\alpha < 0,5$, le processus converge vers la même distribution que la chaîne de Markov *classique* (sans retour possible).
- Les autres cas sont caractérisés, différents types de convergence sont considérés, et les valeurs limites (si elles existent) sont données.

Le principal inconvénient du modèle de Fagin *et al.* est le coût prohibitif[3] du calcul des distributions asymptotiques.

Dans [Syd04], Sydow propose de réduire la complexité en limitant la taille de l'historique. Il introduit ainsi un modèle où seule la dernière page visitée est conservée en mémoire (on ne peut pas utiliser deux fois de suite la touche *Back*), et où la probabilité de *Back* est uniforme[4]. L'algorithme résultant présente des performances comparables à celles d'un PageRank standard, en termes de vitesse de convergence et de besoin en ressources. Le classement obtenu par la distribution de probabilité asymptotique reste similaire à celui d'un PageRank standard, avec des différences significatives au niveau du classement des pages dominantes.

Par rapport à l'algorithme de Sydow, notre prise en compte de la touche *Back* présente l'avantage de supprimer de nombreux problèmes liés aux PageRanks classiques (complétion stochastique, effeuillage et remplumage, vitesse de convergence).

6.3 Modélisation de la touche *Back*

Adoptant une approche similaire à [Syd04][5], nous choisissons d'introduire la possibilité de revenir en arrière avec un historique limité. Potentiellement, pour ramener un processus stochastique à mémoire m à une chaîne de Markov sans mémoire, on peut être amené à considérer l'ensemble des chemins de longueur m dans G. Dans le cas où $m = 1$, qui est celui que nous

3. Avec les graphes du Web, tout algorithme dont la complexité est supérieure à n, ou à la rigueur $n \log n$, est considéré comme prohibitif.

4. Comme l'historique est limité, la distribution limite n'est pas forcément égale à la distribution du modèle classique, même pour $\alpha < 0,5$.

5. À moins que ce ne soit Sydow qui ait adopté une approche similaire à la notre, les recherches ayant été menées indépendamment, et chacun ayant découvert les travaux de l'autre lors de la treizième conférence *WWW*.

allons étudier, l'espace de travail canonique est l'ensemble E des hyperliens. Nous allons introduire deux modèles intuitifs de touche *Back* pour le cas où $m = 1$, et montrer que pour l'un d'entre eux, il est possible de réduire l'espace de travail de E à V. Nous verrons également que ce dernier cas est compatible avec l'emploi d'un facteur *zap*.

6.3.1 Modèle réversible[6]

Dans ce modèle, nous supposons qu'à chaque étape, l'utilisateur peut soit choisir un lien sortant de la page où il se trouve, soit appuyer sur la touche *Back*, ce qui va le ramener à l'étape précédente. La touche *Back* est considérée comme un hyperlien comme les autres. En particulier, la probabilité d'utiliser la touche *Back* est la même que celle de cliquer sur un lien donné, s'il en existe au moins 1, et vaut 1 en l'absence de lien sortant. Nous pensons que cette approche présente deux avantages par rapport au choix d'une probabilité de retour uniforme fait dans [Syd04] :

- Intuitivement, il semble logique que l'usage de la touche *Back* dépende du choix disponible sur la page considérée, c'est-à-dire du nombre de liens sortants. Ceci est partiellement confirmé par les comportements observé dans [MFJR+04].
- Tout comme la page virtuelle de complétion introduite dans la section 5.6.3, la touche *Back* que nous venons de modéliser assure une échappatoire même dans les pages sans liens, et crée une forme de complétion stochastique.

Précisons enfin que le terme *réversible* est dû au fait que deux emplois consécutifs de la touche *Back* s'annulent.

Formellement, le processus stochastique que nous venons d'introduire peut se décrire ainsi : pour chaque $v \in V$, nous définissons $P_n(v)$ comme la probabilité de présence en v à l'instant n. Nous introduisons également le terme $P_n^{rb}(w, v)$, pour $w, v \in V$, probabilité d'être passé de w à v entre les instants $n-1$ et n. $P_n^{rb}(w, v)$ est non nul dès que (w, v) ou (v, w) est dans E, et il existe une relation très simple entre P_n et P_n^{rb} :

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

où $w \leftrightarrow v$ signifie w *pointé par ou pointant vers* v. On remarque que l'usage de la touche *Back* implique de travailler sur le graphe non-orienté induit par G.

De la même façon, il est possible d'écrire une équation décrivant les termes $P_{n+1}^{rb}(w, v)$: si $(w, v) \in E$, pour aller de w à v entre les instants n et $n+1$, on peut soit suivre le lien qui relie w à v (il faut pour cela être en w à l'instant n et choisir parmi les $d(w)+1$ possibilités), soit utiliser la touche *Back* (il faut alors être allé de v à w entre les instants $n-1$ et n, et choisir parmi les $d(w)+1$ possibilités). Si $(w, v) \notin E$, mais $(v, w) \in E$, alors la transition de w à v ne peut être

6. Le concept de réversibilité a une signification bien précise en modélisation markovienne, mais nous l'employons ici simplement pour décrire un certain comportement de l'utilisateur, et il conviendra donc ne pas se méprendre sur le sens de « réversible ».

effectuée que *via* l'emploi de la touche *Back*. Il n'y a pas de transition dans les autres cas. Nous pouvons donc écrire :

$$P^{rb}_{n+1}(w,v) = \begin{cases} \dfrac{1}{d(w)+1}(P_n(w) + P^{rb}_n(v,w)) \text{ si } (w,v) \in E, \\ \dfrac{P^{rb}_n(v,w)}{d(w)+1} \text{ si } (v,w) \in E \text{ et } (w,v) \notin E, \\ 0 \text{ sinon.} \end{cases} \tag{6.2}$$

Grâce aux équations (6.1) et (6.2), il est possible de réaliser un processus itératif de calcul de P_n, que l'on pourra par exemple initier par la distribution

$$P^{rb}_0(w,v) = \begin{cases} \dfrac{1}{|E|} \text{ si } (w,v) \in E, \\ 0 \text{ sinon.} \end{cases}$$

Si $G' = (V, E')$ est le graphe non-orienté induit par G, il semble donc nécessaire d'utiliser $|V| + |E'|$ variables, contre $|V|$ pour le PageRank standard. En injectant (6.1) dans (6.2), il est possible de réduire le nombre de variables à $|E'|$, mais cela reste très important. Ceci nous a amené à considérer le modèle du *Back* irréversible[7].

6.3.2 Modèle irréversible

Dans ce nouveau modèle, nous supposons qu'il n'est pas possible d'utiliser la touche *Back* deux fois d'affilée : celle-ci est grisée après utilisation, et il faut d'abord utiliser au moins une fois un lien réel avant de pouvoir en faire usage à nouveau. Bien que plus complexe en apparence, ce modèle présente des avantages certains sur le modèle réversible :

- Il est plus proche du comportement de la touche *Back* des navigateurs réels, qui se désactive effectivement lorsque l'historique est épuisé. L'usage d'une touche *Back* après épuisement de l'historique dans le modèle réversible se rapproche plus de l'usage de la touche *Forward* dans les navigateurs réels. Or, si l'on en croit [MFJR+04] (cf tableau 6.1), le rôle de la touche *Forward* reste relativement anecdotique.
- Il est compatible avec l'introduction effective d'un facteur *zap* (cf 6.3.3).
- Le calcul de la distribution asymptotique, même avec un facteur *zap*, ne nécessite pas plus de ressources que dans le cas d'un PageRank classique(cf 6.4.1).
- L'emploi de la touche *Back* introduit forcément une sorte d'« effet de serre » au niveau des pages sans liens, qui peut s'avérer gênant (effet similaire à celui des composantes fortement

7. Il est peut-être possible de réduire le nombre de variables à $|V|$, comme nous allons le faire avec le modèle irréversible, mais nous n'avons pour l'instant pas formalisé cette réduction, et laissons donc le modèle de *Back* réversible à un stade purement descriptif.

connexes récurrentes décrites dans le chapitre précédent). Le modèle irréversible réduit cet effet par rapport au modèle réversible. Considérons l'exemple de la figure 6.1 : une page 1 possède 2 liens, l'un vers une page sans lien 2, l'autre vers une page d'échappement d'où tout le graphe est accessible. Si le surfeur aléatoire se retrouve en 2, il va forcément revenir en 1 par la touche *Back*. Retourner à nouveau en 2 va alors créer un début d'effet de serre, or le retour en 2 se fait avec une probabilité $\frac{2}{3}$ dans le modèle réversible (2 cas favorables sur 3), contre $\frac{1}{2}$ dans le cas irréversible : la probabilité de rester « coincé » en 2 est plus faible dans le modèle irréversible.

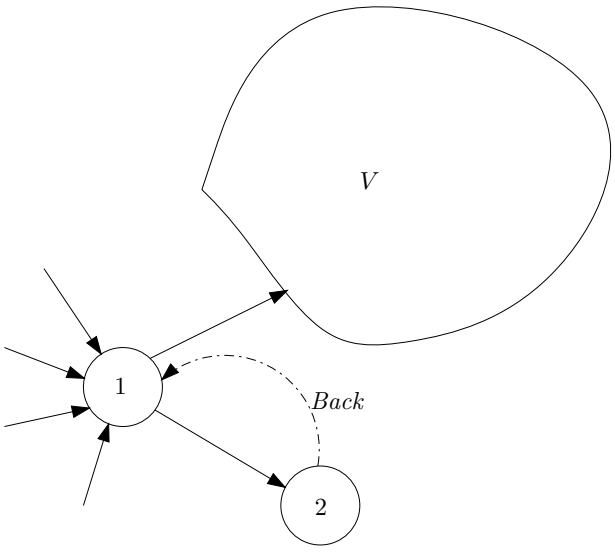


FIG. 6.1 – *Touche* Back *et effet de serre : rôle de l'(ir)réversibilité*

Afin de calculer l'évolution dans un tel modèle, nous allons introduire :

$P_n^{hl}(w,v)$ **:** probabilité d'aller de la page w à la page v entre les instants $n-1$ et n à l'aide d'hyperlien.

$P_n^{ib}(v)$ **:** probabilité d'être arrivé sur la page v à l'instant n par l'emploi de la touche *Back*.

Il est possible de décrire $P_{n+1}^{hl}(w,v)$ en fonction de la situation à l'instant n : pour suivre le lien qui va de w à v, soit on est arrivé à w en suivant un lien réel, et il faut choisir parmi $d(w)+1$ possibilités, soit on est arrivé en w par la touche *Back*, ce qui a grisé cette dernière et limite les possibilités à $d(w)$. Nous avons ainsi, pour $(w,v) \in E$,

$$P_{n+1}^{hl}(w,v) = \frac{\sum_{u \to w} P_n^{hl}(u,w)}{d(w)+1} + \frac{P_n^{ib}(w)}{d(w)} \tag{6.3}$$

On remarquera que comme $w \to v$, il n'y a aucune ambiguïté à effectuer une division par $d(w)$. On notera également que le sommet d'arrivée v n'intervient en aucune façon dans l'expression de $P_{n+1}^{hl}(w,v)$. Si R est l'ensemble des pages possédant au moins un lien, et S l'ensemble

des pages sans lien, on parlera donc à partir de maintenant de $P_n^{hl}(w)$, avec $w \in R$, plutôt que de $P_{n+1}^{hl}(w, v)$, avec $(w, v) \in E$.

De même, pour être arrivé sur une page v grâce à la touche *Back*, il faut précédemment être allé de v vers une page w pointée par v grâce à un hyperlien, puis avoir choisi la touche *Back* parmi les $d(w) + 1$ possibilités. En particulier, on ne peut pas être arrivé sur une page de S par la touche *Back*, et $P_n^{ib}(v)$ est nul pour $v \in S$. Pour $v \in R$, nous pouvons écrire :

$$P_{n+1}^{ib}(v) = \sum_{w \leftarrow v} \frac{P_n^{hl}(v)}{d(w) + 1}, \tag{6.4}$$

où $w \leftarrow v$ signifie w *pointé par* v.

Si on définit la *Back*-attractivité d'un sommet v appartenant à R par

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

il est alors possible de réécrire les équations (6.4) et (6.3) comme suit :

$$P_{n+1}^{ib}(v) = a(v) P_n^{hl}(v) \tag{6.5}$$

$$P_{n+1}^{hl}(v) = \frac{1}{d(v) + 1} \sum_{w \to v} P_n^{hl}(w) + \frac{P_n^{ib}(v)}{d(v)} \tag{6.6}$$

En fusionnant (6.5) et (6.6), on obtient l'équation (6.7)

$$P_{n+1}^{hl}(v) = \frac{1}{d(v) + 1} \sum_{w \to v} P_n^{hl}(w) + \frac{a(v)}{d(v)} P_{n-1}^{hl}(v) \tag{6.7}$$

Et le PageRank dans tout ça ? La probabilité de présence en une page v est la somme de la probabilité d'y être arrivé par la touche *Back* et de celle d'y être arrivé en suivant un lien. En posant, par convention, $P_n^{hl}(v) = 0$ pour $v \in S$, on a :

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

6.3.3 Incorporation du facteur *zap*

L'ajout de la touche *Back* irréversible nous garantit un processus stochastique quel que soit le graphe initial, à la seule condition que le support de la distribution initiale ne contienne aucune page sans lien. Le problème des impasses à courte distance, comme la page 2 de la figure

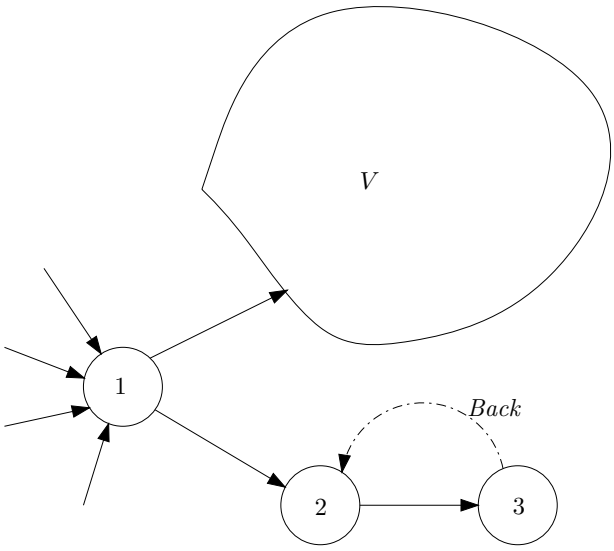


FIG. 6.2 – *Composante fortement connexe récurrente créée par la touche* Back

6.1, est résolu, mais les problèmes d'irréductibilité restent présents. Pire encore, la touche *Back* transforme les impasses à moyenne et longue distance en puits de rang (cf figure 6.2).

Il est donc nécessaire d'introduire du *zap* dans le processus. Nous avons choisi un *zap* classique, avec un facteur d et une distribution Z. Nous imposons que le fait de « zapper » désactive la touche *Back* : après un *zap*, il n'est pas possible de revenir en arrière. Cela présente l'avantage pratique de ne pas avoir à considérer toutes les transitions implicitement générées par le *zap*, à savoir $(V \times \chi_Z) \subset E$. On peut aussi interpréter ce choix au niveau de la modélisation du surfeur : selon le tableau 6.1, beaucoup de *zaps* réels correspondent en fait au démarrage d'une nouvelle session, et donc à un historique remis à 0.

Maintenant que le cadre est défini, il est possible de récrire $P_n^{hl}(v)$, pour $v \in R$:

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

De la même façon, nous pouvons récrire la probabilité $P_n^{ib}(v)$ de se trouver à un instant n sur une page v avec un historique nul. En posant, par convention, $P_n^{hl}(v) = 0$ pour $v \in S$, on a :

$$P_{n+1}^{ib}(v) = d.a(v)P_n^{hl}(v) + (1-d)Z(v) \tag{6.9}$$

Et là, un problème se pose : que se passe-t-il si $\chi_Z \cap S \neq \emptyset$? Nous allons avoir une probabilité non nulle d'être dans une page de S sans retour en arrière possible. Dans cette situation, avec une probabilité $(1-d)$, notre surfeur va « zapper » ailleurs, et avec une probabilité d... il ne saura pas quoi faire !

Nous envisageons deux méthodes pour contourner ce problème :

- Choisir Z tel que $\chi_Z \cap S \neq \emptyset$. Puisque la seule condition que l'on demande à une distribution de *zap* est d'être recouvrante, c'est tout à fait possible. Une distribution sur les pages d'accueil conviendra, à la condition qu'il n'existe pas de page d'accueil d'un site réduit à une page sans lien. On peut aussi choisir de prendre la distribution uniforme sur R définie par

$$Z(v) = \begin{cases} \dfrac{1}{|R|} \text{ si } v \in R, \\ 0 \text{ sinon.} \end{cases}$$

- Compléter le processus stochastique à la volée. On connaît en effet la probabilité de ne pas savoir quoi faire entre les instants n et $n+1$: $d \sum_{v \in S} P_n^{ib}(v)$. Il suffit alors de répartir cette probabilité, par exemple en *zap* selon Z. L'équation (6.9) devient alors :

$$P_{n+1}^{ib}(v) = d.a(v)P_n^{hl}(v) + \left[(1-d) + d\sum_{v \in S} P_n^{ib}(v)\right] Z(v) \tag{6.10}$$

6.4 Algorithme pratique : BackRank

Nous venons de définir un processus stochastique, qui grâce au facteur *zap* est irréductible et apériodique. Le théorème de Perron-Frobenius s'applique donc[8] et nous permet d'affirmer que des itérations successives des équations (6.9) et (6.8) vont converger vers un point fixe unique (à renormalisation près). Les conditions initiales peuvent par exemple être une distribution selon Z avec un historique nul ($P_0^{ib} = Z$ et $P_0^{hl} = 0$).

6.4.1 Optimisation

Nous supposerons ici que nous avons choisi Z tel que $Z(v) = 0$ si $v \in S$.
D'après (6.9) et (6.8), $P_{n+1}^{hl}(v)$ peut se définir récursivement, pour $v \in R$ par :

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

L'équation (6.11) est une récurrence à deux termes, dont on sait qu'elle converge vers un point fixe. Comme seul le point fixe nous intéresse, on peut remplacer P_{n-1}^{hl} par P_n^{hl} en gardant

8. On notera au passage que nous n'avons pas besoin d'écrire explicitement la matrice des transitions associée, qui est une matrice carrée de taille $|V| + |E|$. Il nous suffit de savoir que cette matrice existe et que c'est elle qui régit implicitement notre processus.

une convergence vers le même point fixe (méthode de Gauss-Seidel). On obtient ainsi l'équation (6.12).

$$P^{hl}_{n+1}(v) = \frac{d}{d(v)+1}\sum_{w\to v} P^{hl}_n(w) + \frac{d}{d(v)}\left(d.a(v)P^{hl}_n(v) + (1-d)Z(v)\right) \tag{6.12}$$

On remarquera au passage que les itérations se font sur les sommets de degré sortant non nul : il y a un effeuillage implicite, semblable à ce qui se pratique pour les PageRanks standards (cf section 5.4.4).

Après que le vecteur P^{hl}_n a convergé vers un vecteur P^{hl}, il reste à effectuer le « remplumage » pour obtenir la distribution asymptotique de présence P :

$$\begin{aligned} P(v) &= \sum_{w\to v} P^{hl}(w) + P^{ib}(v) \\ &= \sum_{w\to v} P^{hl}(w) + d.a(v).P^{hl}(v) + (1-d)Z(v) \end{aligned} \tag{6.13}$$

Nous avons maintenant tous les éléments constitutifs de l'algorithme ***BackRank*** (algorithme 8 page 125). On remarquera au passage qu'il n'y a pas besoin d'effectuer de renormalisation, puisqu'il y a convergence vers un point fixe.

6.4.2 Résultats

Une fois un algorithme complètement défini, il faut le soumettre à l'épreuve du feu. Afin d'effectuer des comparaisons, il nous fallait un baril de lessive X, et c'est le PageRank μ-compensé défini dans l'algorithme 4 qui a été choisi. C'est en effet le PageRank le plus simple garantissant un contrôle stochastique total, mis à part le PageRank non-compensé bien sûr[9]. La technique d'effeuillage-remplumage a été rajoutée par souci d'équité par rapport à BackRank, qui la réalise implicitement, et aussi afin d'être plus réaliste (c'est la méthode qui semble être employée en pratique pour les calculs *off-line* sur de très gros graphes, et BackRank est conçu pour supporter de très gros graphes.). Pour les deux algorithmes, nous avons pris $d = 0,85$ (sauf indication contraire) et Z est la distribution uniforme sur la rafle R.

Le test a porté sur un crawl de 8 millions d'URLs qui, pour des raisons techniques[10] est scindé en deux échantillons de 4 millions d'URLs.

Convergence

Un des premiers critères de viabilité d'un algorithme de type PageRank est sa vitesse de convergence. La figure 6.3 compare, sur une échelle semi-logarithmique, la valeur du paramètre

9. Nous avons gardé la compensation pour rendre les comparaisons plus simples, puisqu'ainsi les deux concurrents fournissent une distribution de probabilité.

10. Les algorithmes tournent sous Matlab.

de convergence δ au bout de n itérations. Pour BackRank, la condition $\delta < 10^{-10}$ est atteinte au bout de 87 itérations dans les deux échantillons, alors que pour PageRank, il faut entre 126 et 127 itérations. Nous avons mesuré la raison de la décroissance géométrique observée à 10^{-10}, et trouvé $0,844$ pour le PageRank standard (la convergence est toujours inférieure à d, mais tend asymptotiquement vers d) contre $0,809$ pour BackRank. Nous supposons que cet écart, qui explique les performances de BackRank, est du au fait qu'à la différence de PageRank, BackRank utilise implicitement une méthode de Gauss-Seidel partielle. Or, il a été montré que l'utilisation d'une méthode de type Gauss-Seidel améliore considérablement la convergence d'un PageRank [ANTT01].

Nous nous devons de préciser que les algorithmes utilisés sont, aux jeux de réécriture près[11], exactement ceux décrits dans cet ouvrage. En particulier, les méthodes de la puissance employées sont vraiment des méthodes de la puissance. Pour une étude plus complète des performances de convergence de BackRank, il faudrait étudier comment il se comporte soumis aux nombreuses méthodes d'accélération de la convergence qui existent [ANTT01, Hav99, KHMG03b].

Pour conclure cette étude de la convergence, précisons que sur les échantillons considérés, une itération de BackRank prend en moyenne 2 secondes contre 3 secondes pour PageRank. Cette différence vient du fait que toutes les constantes de calcul, y compris la matrice d'adjacence, tiennent en mémoire vive. La μ-compensation a donc un coût non négligeable dans une itération. En fait, même par rapport à un PageRank non-compensé avec estimation de δ, BackRank ne présente qu'un faible surcoût, de l'ordre de 5%.

Le remplumage

La finalisation du calcul du PageRank, que nous appelons remplumage, est une phase assez peu décrite. Selon [PBMW98], après convergence du PageRank sur le graphe restreint aux sommets de degré sortant non nul, les pages sans lien sont rajoutées au processus, *sans affecter les choses de manière significative*[12]. Mais afin d'attribuer à ces nouvelles pages une importance, il faut bien réaliser au moins une itération de PageRank, ce qui implique de modifier la plupart des constantes. Dans notre essai, nous avons choisi d'effectuer quatre itérations sur le graphe complet après convergence sur le graphe effeuillé. Outre des itérations plus lentes (6 secondes, sachant qu'environ 50% des pages des échantillons étaient des pages sans lien), on constate, comme dans la section 5.4.4 page 97, la réinitialisation du paramètre δ (cf figure 6.3) : au bout de quatre itérations, on est au même niveau qu'après huit itérations de la boucle principale, c'est-à-dire loin d'un état stationnaire…

Pour BackRank, le remplumage est beaucoup moins problématique, puisqu'il se limite à l'application de (6.13) *une seule et unique fois*. Inutile donc de démarrer une nouvelle série d'itérations. Précisons quand même pour être honnête que puisque δ se rapporte à P^{hl} et non P, les variations sur P peuvent être plus grandes que δ. Expérimentalement, il semble effectivement

11. L'algorithme 4 a été réécrit suivant le modèle de l'algorithme 6.
12. « without affecting things significantly », *op. cit.*

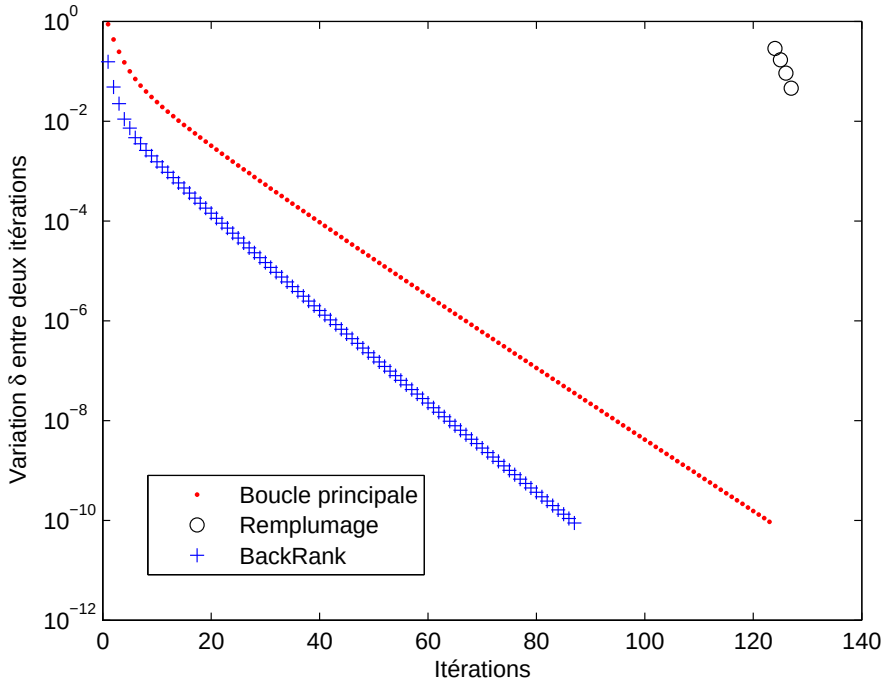


FIG. 6.3 – *Convergences comparées de BackRank et d'un PageRank standard*

que lorsque δ est aux alentours de 10^{-10}, les variations au niveau de P sont de l'ordre de 10^{-9} (légèrement inférieures en fait). Ce résultat reste malgré tout plus qu'acceptable, surtout au regard des variations de 10^{-1} générées par le remplumage du PageRank.

Classement

Les performances techniques ne sont que la moitié des critères d'évaluation d'un algorithme de type PageRank. Il faut aussi tester la pertinence du classement par importance induit par la distribution de probabilité obtenue. Une première approche consiste à comparer quantitativement les résultats renvoyés par BackRank et ceux renvoyés par le PageRank de référence. La figure 6.4 présente une telle comparaison, en donnant le pourcentage de pages communes aux n premières pages renvoyées respectivement par BackRank et PageRank. On observe des fluctuations importantes sur les toutes premières pages. Au fur et à mesure que le nombre de pages considérées augmente, le chevauchement se stabilise, pour arriver à une valeur relativement stable de $0,845$ lorsque le nombre de pages considérées s'approche de 1% de la taille de l'échantillon (40000 pages). Ceci tend à prouver que BackRank offre des résultats assez proches de ceux renvoyés par PageRank, avec quelques spécificités.

Ce chevauchement à 1% stabilisé valant $0,845$ pour les deux échantillons nous a intrigués.

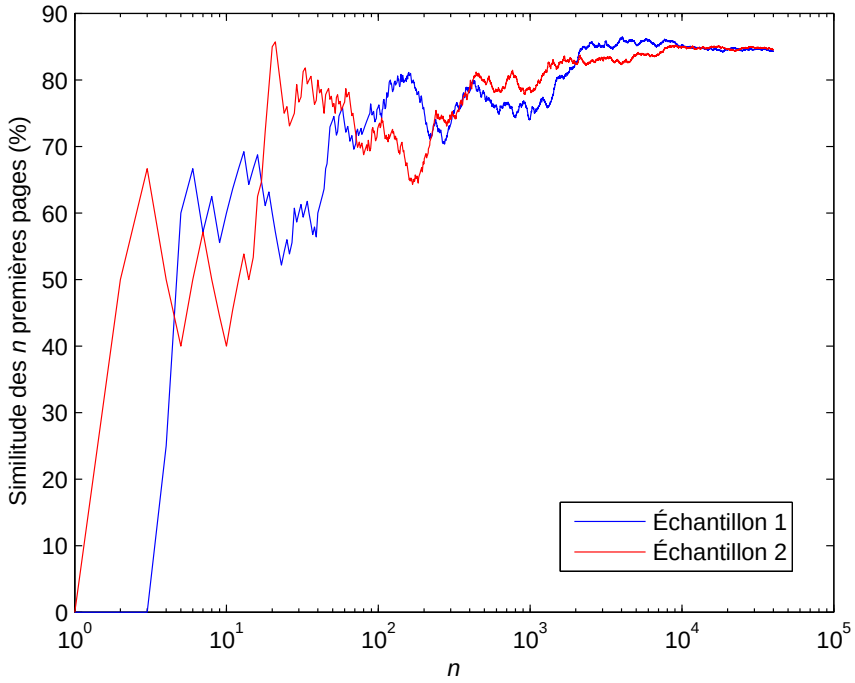


FIG. 6.4 – *Mesures de chevauchement entre les n premières pages de BackRank et de PageRank*

En étudiant le phénomène de plus près, nous avons remarqué que le taux de chevauchement à 1% est une variable qui ne dépend que de d (sur nos échantillons en tout cas). La figure 6.5 montre ce lien entre d et chevauchement à 1% pour $0 \leq d \leq 1$. On remarquera en particulier que :

- Le chevauchement est une fonction décroissante de d. On retrouve l'idée, évoquée dans la section 5.3.5, du rôle lisseur du facteur *zap*.
- En particulier, le chevauchement tend vers 1 lorsque d tend vers 0, ce qui signifie que BackRank tend, tout comme PageRank, vers la distribution par degré entrant (cf 5.3.5).
- Le chevauchement tend vers 50% lorsque d tend vers 1, ce qui semble indiquer que BackRank est intrinsèquement à moitié différent (ou à moitié semblable ?) du PageRank standard. Il ne faut cependant pas oublier que lorsque $d = 1$, le classement est sclérosé par les puits de rang et n'a plus forcément de sens. Vérification sur quelques URLs faite, ce chevauchement de 50% veut en effet seulement dire que BackRank ne se fait pas piéger sur exactement les même puits que PageRank.

Nous sommes conscients de ne donner ici quelques indices de la qualité du classement par BackRank. En toute rigueur, une validation complète nécessiterait d'incorporer BackRank dans un moteur de recherche et d'effectuer une série de tests de satisfaction sur une population témoin. À défaut, nous nous contenterons d'utiliser le lecteur comme population témoin, qui pourra comparer dans les listes 1, 2, 3 et 4 les 10 premières pages renvoyées par BackRank et PageRank sur

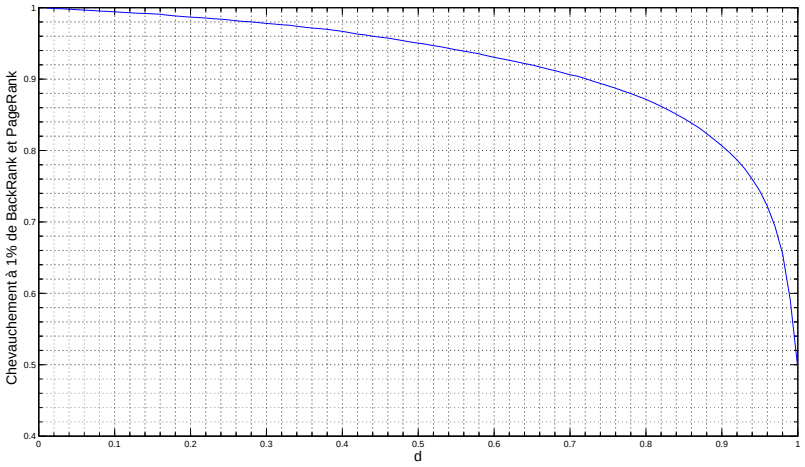


FIG. 6.5 – *Influence du facteur* zap *dans le chevauchement à* 1%

les deux échantillons considérés. On remarquera par exemple que la page principale du CNRS est classé par BackRank, mais pas par PageRank (en fait, elle est $16^{\text{ème}}$), alors que pour l'Éducation Nationale, c'est l'inverse (page classée également $16^{\text{ème}}$, mais par BackRank cette fois).

Algorithme 8: BackRank : modèle de PageRank avec touche *Back* irréversible

Données

- une matrice d'adjacence M d'un graphe quelconque $G = (V, E)$, avec $V = (R \cup S)$, où R est l'ensemble des pages avec lien, S celui des pages sans lien ;
- une distribution de *zap* Z sur R recouvrante ;
- un coefficient de *zap* $d \in]0, 1[$;
- un réel ϵ.

Résultat

Le vecteur de PageRank P du modèle *Back* irréversible avec *zap*.

début

$D = M\mathbf{1}_n^t$
$T = 1./(D + \mathbf{1}_n^t)$
$A = M.T$
$T = T_R$
$D = D_R$
$C = 1./D$
$B = d.C.\times A_R$
$Y = d(1-d)C.\times Z$
$N = M_R$
$R_0 = dC.\times Z$
répéter
  $R_{n+1} = T.\times (N^t R_n) + B.\times R_n$
  $R_{n+1} = dR_{n+1}$
  $R_{n+1} = R_{n+1} + Y$
  $\delta = \|R_{n+1} - R_n\|_1$
jusqu'à $\delta < \epsilon$
$R_{n+1} \rightarrow R_{n+1}$ sur V (nul sur S)
$P = M^t R_{n+1} + dA.\times R_{n+1} + (1-d)Z$

fin

Liste 1 Échantillon 1 : les 10 pages les plus importantes selon BackRank

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

Liste 2 Échantillon 1 : les 10 pages les plus importantes selon PageRank

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

Liste 3 Échantillon 2 : les 10 pages les plus importantes selon BackRank

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

Liste 4 Échantillon 2 : les 10 pages les plus importantes selon PageRank

```
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
```

Chapitre 7

Décomposition fine du PageRank

Je composerai jusqu'à la décomposition
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

L'OBJET de ce chapitre, qui est une extension de [MV03a] et de [MV04], est d'étudier comment le PageRank se comporte en regard de la structure de site présentée lors de la partie I, section 3.3. Nous allons montrer qu'il existe une décomposition naturelle du PageRank en deux terme, le PageRank interne et le PageRank externe. Cette décomposition permet de mieux comprendre les rôles joués par les liens internes et externes. Une première application est un algorithme d'estimation du PageRank local à l'intérieur d'un site. Nous allons également montrer quelques résultats quantitatifs sur les possibilités offertes à un site d'augmenter son propre PageRank.

Plus précisément, la section 7.1 présente sommairement les différentes contributions existantes quant à l'utilisation de la structure en sites du Web pour calculer un PageRank. La section 7.2 précise quelles seront les hypothèses et conventions utilisées. Les notions de PageRank interne et externe seront introduites lors de la section 7.3, et appliquées à un algorithme théorique décomposé de PageRank dans la section 7.4. Enfin, après avoir profité dans la section 7.5 de

l'étude des variations locales du PageRank pour introduire le facteur *zap* dans notre modélisation, nous donnerons dans la section 7.6 des algorithmes applicables aux graphes réels.

7.1 Travaux antérieurs et contemporains

Sur l'ensemble des études publiées sur le PageRank, seules quelques unes essaient de prendre avantage de la structure de site. [KHMG03a] donne une méthode pour calculer rapidement une bonne approximation du PageRank à l'aide d'une partition en sites.

Bianchini *et al.* décomposent le PageRank en liens internes, liens entrants, liens sortants et fuites [BGS03, BGS02]. Cette décomposition permet entre autres d'apporter une certaine compréhension de la manière dont un site peut changer son propre PageRank, tout en donnant des résultats de stabilité du PageRank face à des changements dans la structure en liens internes d'un site.

Enfin, Arasu *et al.* ont montré qu'un calcul de PageRank sur le graphe quotienté selon les serveurs convergeait plus vite que sur les pages, et encore plus vite en prenant en compte la multiplicité des liens (cf [ANTT01]).

Par rapport à ces travaux, notre approche consiste à utiliser, comme dans [BGS03, BGS02], une décomposition exacte en flots du PageRank, avec des hypothèses plus flexibles sur la distribution de *zap*. De cette décomposition, nous déduisons un algorithme semi-distribué exact de calcul de PageRank, que nous hybridons avec l'algorithme proposé dans [KHMG03a] afin d'obtenir un algorithme semi-distribué rapide et avec peu d'approximations.

7.2 Hypothèses

Nous avons vu qu'une définition structurelle d'un site pouvait être un ensemble de pages étroitement reliées entre elles par des hyperliens. La structure en blocs de la matrice d'adjacence (cf figures 3.5 et 3.6, pages 47 et 49) nous permet d'espérer de nombreuses méthodes pour décomposer un graphe du Web en sites, et la section 3.3 page 40 nous en donne une. Pour la suite de ce chapitre, nous supposerons donc que notre graphe du Web est muni d'une partition qui permet de le décomposer en sites, notée $\mathcal{S} = (S_1, \ldots, S_k)$, avec $k > 1$.

Dans un premier temps, nous nous placerons dans le cas idéal où le graphe du Web $G = (V, E)$ considéré est fortement connexe et apériodique. Nous supposerons également l'absence de lien d'une page vers elle-même.

Comme il a été vu lors de la section 5.3.1, si l'on considère la matrice A définie par

$$A = (a_{v,w})_{v,w \in V}, \text{ avec } a_{v,w} = \begin{cases} \dfrac{1}{d(v)} \text{ si v}\rightarrow\text{w,} \\ 0 \text{ sinon.} \end{cases},$$

on sait qu'il existe une unique distribution de probabilité, notée P, vérifiant

$$P = A^t P \tag{7.1}$$

Nous allons chercher à mettre en évidence les liens entre P et $\mathcal{S}$.

7.3 PageRank interne, PageRank externe

7.3.1 Notations

Pour v dans V, nous désignons par $S(v)$ l'élément de $\mathcal{S}$ tel que $v \in S(v)$. Nous définissons également $\delta_{\mathcal{S}} : V \times V \to \{0, 1\}$ comme suit :

$$\delta_{\mathcal{S}}(v, w) = \begin{cases} 1 \text{ si } S(v) = S(w), \\ 0 \text{ sinon.} \end{cases}$$

Nous appellerons $A_{\mathcal{S}}$ la restriction de A aux éléments internes aux composantes de $\mathcal{S}$:

$$A_{\mathcal{S}} = (a_{v,w}\delta_{\mathcal{S}}(v, w))_{v,w \in V}$$

Nous avons également besoin de définir le degré interne d_i (resp. degré externe d_e) d'un sommet v comme son degré sortant dans le graphe induit par $S(v)$ (resp. induit par $\{v\} \cup (V \setminus S(v))$).

Nous sommes maintenant en mesure de définir les notions de PageRank interne et externe, et de les relier au PageRank P défini par l'équation (7.1).

- Le PageRank entrant interne P_{ei} (resp. PageRank entrant externe P_{ee}) d'une page v de V est la probabilité, lorsque le surfeur aléatoire est en régime stationnaire, de venir en v à partir d'une page de $S(v)$ (resp. à partir d'une page de $V \setminus S(v)$). De cette définition sont déduites les équations (7.2) et (7.3) :

$$P_{ei} = A_{\mathcal{S}}^t P \tag{7.2}$$
$$P_{ee} = (A - A_{\mathcal{S}})^t P = P - P_{ei} \tag{7.3}$$

- Le PageRank sortant interne P_{si} (resp. PageRank sortant externe P_{se}) d'une page v de V est la probabilité, lorsque le surfeur aléatoire est en régime stationnaire, d'aller de v à une page de $S(v)$ (resp. à une page de $V \setminus S(v)$). Les équations (7.4) et (7.5) formalisent cette définition :

$$P_{si} = (A_{\mathcal{S}}.\mathbf{1}_n^t).\times P \tag{7.4}$$
$$P_{se} = ((A - A_{\mathcal{S}}).\mathbf{1}_n^t).\times P = P - P_{si} \tag{7.5}$$

7.3.2 Lois de conservation

On peut définir un PageRank (éventuellement interne, sortant,...) sur un site S comme étant la somme des PageRanks de ses pages : $P(S) = \sum_{v \in S} P(v)$. Cette convention étant prise, nous pouvons énoncer les lois de conservation interne et externe d'un site :

Théorème 12 *Soit S un site. Les PageRanks entrant externe et sortant externe de S sont égaux :*

$$P_{ee}(S) = P_{se}(S) \text{ (loi de conservation externe)} \tag{7.6}$$

Il en est de même des PageRanks entrant interne et sortant interne :

$$P_{ei}(S) = P_{si}(S) \text{ (loi de conservation interne)} \tag{7.7}$$

Preuve :

Commençons par prouver la loi de conservation interne (équation (7.7)) :

$$\begin{aligned} P_{ei}(S) &= \sum_{v \in S} P_{ei}(v) = \sum_{v \in S} \sum_{w \to v,\, w \in S} \frac{P(w)}{d(w)} && (7.8)\\ &= \sum_{w \in S} \sum_{v \leftarrow w,\, v \in S} \frac{P(w)}{d(w)} = \sum_{w \in S} P_{si}(w) && (7.9)\\ &= P_{si}(S) && (7.10) \end{aligned}$$

Ensuite, les équations (7.3) et (7.5) nous permettent d'écrire :

$$P = P_{ee} + P_{ei} = P_{se} + P_{si} \tag{7.11}$$

L'équation (7.11) associée à la loi de conservation interne (7.7) nous donne la loi de conservation externe :

$$P_{se}(S) = P_{ee}(S) + P_{ei}(S) - P_{si}(S) = P_{ee}(S)$$

□

La loi de conservation externe (7.6) nous montre qu'un site restitue, à travers le PageRank sortant externe, exactement le PageRank qu'il reçoit (le PageRank entrant externe). Comme disait Lavoisier, *rien ne se perd, rien ne se crée, tout se transforme*. Si l'on considère le flot de PageRank sur le graphe quotient $G/\mathcal{S}$, il y a donc conservation du flot (cf figure 7.1). Cette remarque est à la base du calcul décomposé du PageRank.

Remarque 11

Une autre façon, peut-être plus simple, de prouver le théorème 12 aurait été de considérer directement le PageRank en tant que flot stationnaire. Il est alors évident que le flot sur tout sous-ensemble S de V est également stationnaire. Nous avons préféré l'approche matricielle car c'est

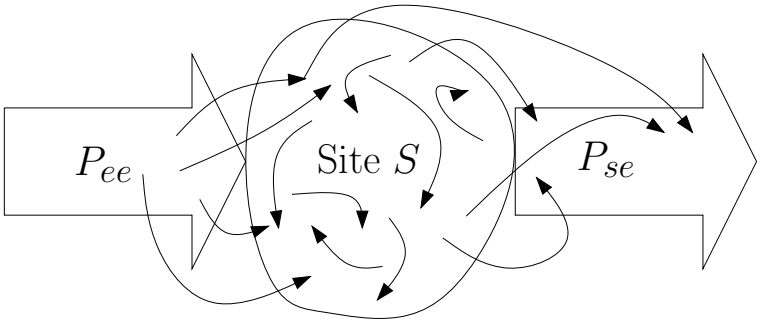


FIG. 7.1 – *Loi de conservation du PageRank externe :* $P_{ee}(S) = P_{se}(S)$

celle que nous continuerons d'utiliser par la suite, même si nous essaierons toujours d'interpréter les résultats en terme de flot quand cela sera possible.

7.4 Décomposition du calcul du PageRank

7.4.1 Relation entre PageRank externe et PageRank

À partir des équation (7.2) et (7.3), nous pouvons écrire que $A_{\mathcal{S}}^t.P = P - P_{ee}$, et donc que $P_{ee} = (Id - A_{\mathcal{S}}^t)P$, où Id est la matrice identité.

Lemme 2 *La matrice* $(Id - A_{\mathcal{S}}^t)$ *est inversible.*

Preuve : Il nous suffit de montrer que $A_{\mathcal{S}}$ est sous-irréductible. Cela prouvera que son rayon spectral est strictement inférieur à 1 (théorème 5, remarque 3), et donc que $(Id - A_{\mathcal{S}}^t)$ est inversible.

Raisonnons par l'absurde : si $A_{\mathcal{S}}$ n'est pas sous-irréductible, il existe au moins une composante fortement connexe stochastique dans le graphe des transitions associé à $A_{\mathcal{S}}$. Cette composante est forcément interne à un site puisqu'il n'y a aucun lien externe. Elle existe donc aussi dans A, qui ne peut alors être irréductible que si la composante est V tout entier, ce qui est impossible (nous avons fait l'hypothèse que A était irréductible et que $\mathcal{S}$ contenait au moins deux sites).

□

Le lemme 2 nous permet alors d'exprimer P comme une fonction du PageRank entrant externe P_{ee} :

$$P = (Id - A_{\mathcal{S}}^t)^{-1} P_{ee} \tag{7.12}$$

Pour calculer le PageRank d'un site S, il suffit donc de connaître sa structure interne, à travers la matrice A_S, et le PageRank entrant externe qu'il reçoit des autres.

Remarque 12
La matrice $(Id - A_{\mathrm{S}}^t)^{-1} = \sum_{k=0}^{\infty}(A_{\mathrm{S}}^t)^k$, qui est tout comme A_{S} une matrice diagonale par blocs, peut s'interpréter comme la matrice de transition de tous les chemins internes possibles. En effet, pour $v, w \in V$, $(A_{\mathrm{S}})^k_{v,w}$ représente la probabilité d'aller de v à w par un chemin de longueur k qui ne suit que des liens internes (en particulier, $(A_{\mathrm{S}})^k_{v,w} = 0$ si $S(v) \neq S(w)$.)

7.4.2 Matrice de transition du PageRank externe

Nous voulons formaliser l'intuition d'une propagation du PageRank de site à site donnée par la loi de conservation du PageRank externe (cf figure 7.1), et trouver une description des relations entre les différentes composantes de P_{ee}. En combinant les équation (7.3) et (7.12), nous obtenons :

$$P_{ee} = (A - A_{\mathrm{S}})^t P = (A - A_{\mathrm{S}})^t (Id - A_{\mathrm{S}}^t)^{-1} P_{ee} \tag{7.13}$$

Nous pouvons ainsi définir la matrice de transition du PageRank externe :

$$A_e^t = (A - A_{\mathrm{S}})^t (Id - A_{\mathrm{S}}^t)^{-1}$$

Cette matrice possède quelques propriétés très intéressantes :

Lemme 3 *La matrice A_e est stochastique.*

Preuve :
A_e est de toute évidence positive, il nous suffit donc de montrer que la somme de chaque colonne de A_e^t vaut 1. Pour cela, nous commençons par réécrire A_e^t :

$$\begin{aligned} A_e^t &= \sum_{k=0}^{\infty} \left(A^t (A_{\mathrm{S}}^t)^k - (A_{\mathrm{S}}^t)^{k+1} \right) \\ &= A^t + \sum_{k=1}^{\infty} \left(A^t (A_{\mathrm{S}}^t)^k - (A_{\mathrm{S}}^t)^k \right) \\ &= A^t + A^t M - M, \text{ avec } M = \sum_{k=1}^{\infty} (A_{\mathrm{S}}^t)^k \end{aligned}$$

Considérons la somme s_w de la colonne de $A^t M$ associée à la page w :

$$s_w = \sum_{u \in V} \sum_{v \in V} A^t_{u,v} M_{v,w} = \sum_{v \in V} \left(\sum_{u \in V} A^t_{u,v} \right) M_{v,w} = \sum_{v \in V} M_{v,w}$$

Ainsi, la somme de chaque colonne de $A^tM - M$ est nulle, ce qui montre que A_e est stochastique, puisque la somme de chaque colonne w vaut $\sum_{v \in V} A^t_{v,w} = 1$. □

Lemme 4 *Soit V_{int} l'ensemble des pages sans lien entrant externe, et V_{ext} celui des pages possédant au moins un lien entrant externe. Si l'on réordonne les pages selon (V_{int}, V_{ext}), alors A_e peut s'écrire*

$$A_e = \begin{pmatrix} 0 & T \\ 0 & \tilde{A}_e \end{pmatrix},$$

où $\tilde{A}_e$ est une matrice stochastique irréductible.

Preuve : Les colonnes de $(A - A_{\mathcal{S}})$ correspondant à des pages de V_{int} sont nulles. Il en est donc de même de celles de A_e, ce qui montre que A_e peut s'écrire sous la forme

$$A_e = \begin{pmatrix} 0 & T \\ 0 & \tilde{A}_e \end{pmatrix}$$

$\tilde{A}_e$ est stochastique, puisque A_e l'est. Il reste à montrer qu'elle est irréductible. Considérons deux sommets v et w de V_{ext}, et un chemin $\mathcal{C} = v, \ldots, w$ qui mène de v à w dans G. Soit $i_0, i_1, \ldots, i_{k-1}, i_k$ la suite de sommets obtenue en ne conservant dans $\mathcal{C}$ que les sommets de V_{ext} ($i_0 = v$ et $i_k = w$). Alors, $i_0, i_1, \ldots, i_{k-1}, i_k$ est un chemin dans le graphe induit par $\tilde{A}_e$. En effet, entre i_{l-1} et i_l, il existe un sous-chemin de $\mathcal{C}$ constitué d'un chemin interne à $S(i_{l-1})$, puis d'un saut externe menant à i_l. D'après la définition de A_e, nous avons donc

$$\tilde{A}_{e(i_{l-1},i_l)} = A_{e(i_{l-1},i_l)} > 0$$

$i_0, i_1, \ldots, i_{k-1}, i_k$ est donc bien un chemin dans le graphe induit par $\tilde{A}_e$, ce qui montre que $\tilde{A}_e$ est irréductible.

C.Q.F.D.

□

A_e possède donc un PageRank unique, qui est nul sur V_{int} [1] et est égal au PageRank de $\tilde{A}_e$ sur V_{ext}. Sous réserve d'apériodicité, il peut être calculé de manière itérative. Seuls les coefficients de $\tilde{A}_e$ sont nécessaires pour calculer ce PageRank. Bien que nous n'ayons pas fait de recherches poussées d'estimation de la taille de V_{ext}, les quelques analyses que nous avons pu effectuer, autant sur des crawls que sur des logs de serveurs (en particulier ceux de l'INRIA) semblent indiquer que l'on peut espérer $|V_{ext}| \leq 0.1\,|V|$.

1. Ce résultat est naturel : une page qui n'a pas de lien entrant externe ne peut pas recevoir du PageRank entrant externe.

7.4.3 Calcul décomposé théorique du PageRank

À partir de (7.12) et de (7.13), nous pouvons établir une méthode théorique semi-distribuée de calcul du PageRank.

- Chaque site S calcule, à partir de son bloc A_S de la matrice des transitions internes A_{S}, son bloc $(Id - A_S^t)^{-1}$ de la matrice $(Id - A_{\mathrm{S}}^t)^{-1}$.
- Les différentes lignes de $\tilde{A}_e$ peuvent alors être reconstituées et centralisées.
- Le PageRank externe P'_e associé à A_e est alors calculé (il faut ici faire une hypothèse d'apériodicité sur $\tilde{A}_e$).
- Chaque site S obtient son propre PageRank $P'(v)$, $v \in S$, en appliquant $P'_e(v)$, $v \in S$, à sa matrice $(Id - A_S^t)^{-1}$.

Lemme 5 *Le vecteur P' ainsi obtenu est homogène au PageRank P associé à G.*

Preuve : Comme A est irréductible, il nous suffit de montrer que $A^t P'$ est égal à P :

$$\begin{aligned} A^t P' &= A^t (Id - A_{\mathrm{S}}^t)^{-1} P'_e \\ &= (A^t - A_{\mathrm{S}}^t)(Id - A_{\mathrm{S}}^t)^{-1} P'_e + A_{\mathrm{S}}^t (Id - A_{\mathrm{S}}^t)^{-1} P'_e \\ &= A_e^t P'_e + \left((Id - A_{\mathrm{S}}^t)^{-1} - (Id - A_{\mathrm{S}}^t)(Id - A_{\mathrm{S}}^t)^{-1} \right) P'_e \\ &= P'_e + \left((Id - A_{\mathrm{S}}^t)^{-1} - Id \right) P'_e \\ &= P'_e + P' - P'_e = P' \end{aligned}$$

C.Q.F.D.

□

7.5 Intermezzo : modifier son propre PageRank

Avant d'attaquer la pièce centrale de ce chapitre, l'algorithme FlowRank, nous voulons montrer que notre décomposition du PageRank permet d'expliquer dans quelle mesure un site peut modifier son propre PageRank, ce qui sera l'occasion d'introduire en douceur le facteur *zap* dans notre modèle.

Les résultats que nous allons présenter prennent du sens si l'on accepte l'idée qu'un site modifie très difficilement son PageRank externe, alors qu'il en est tout autrement du PageRank interne. De fait, les échanges de PageRank entre sites sont étroitement surveillés par Google, qui n'hésite pas à sanctionner les sites qui échangent des liens dans le seul but d'augmenter leur PageRank externe. De telles usines à PageRank, baptisées *farm links* ou *pouponnières*, se voient généralement gratifiées d'un PageRank nul, et se retrouvent donc classées derrière toutes les autres pages[2].

2. Notons que cette politique a été l'objet de nombreux procès opposant Google à des sociétés de référencement.

7.5.1 Coefficient d'amplification

Considérons un site $S \in \mathcal{S}$, son PageRank $P(S)$ et son PageRank entrant externe $P_{ee}(S)$. Nous définissons le coefficient d'amplification α de S comme le rapport entre PageRank et PageRank entrant externe :

$$\alpha(S) = \frac{P(S)}{P_{ee}(S)}$$

Puisque $P = (Id - A_{\mathcal{S}}^t)^{-1} P_{ee}$, $\alpha(S)$ dépend seulement de la structure de S et de la distribution du PageRank externe sur S[3].

La seule connaissance de S nous donne une estimation de $\alpha(S)$:

Lemme 6 *Un encadrement du coefficient d'amplification $\alpha(S)$ est*

$$\frac{1}{1-\omega} \leq \alpha(S) \leq \frac{1}{1-\Omega} \tag{7.14}$$

avec $\omega = \min_{v \in S} \frac{d_i(v)}{d(v)}$ *and* $\Omega = \max_{v \in S} \frac{d_i(v)}{d(v)}$.

Preuve : Si l'on considère l'espace vectoriel associé à S, pour tout vecteur élémentaire e_v, $v \in S$, nous avons $\|A_S(e_v)\|_1 = \frac{d_i(v)}{d(v)}$, et donc $\omega \|X\|_1 \leq \|A_{\mathcal{S}} X\|_1 \leq \Omega \|X\|_1$ pour tout vecteur $X > 0$ défini sur S.

On en déduit la première inégalité de (7.14) :

$$P(S) = \sum_{v \in S} P(v) = \left\| \sum_{k \in \mathbb{N}} (A_S^t)^k (P_{ee}) \right\|_1 \geq \sum_{k=0}^{\infty} \omega^k \|P_{ee}\|_1 = \frac{1}{1-\omega} P_{ee}(S)$$

ainsi que la seconde :

$$P(S) = \sum_{v \in S} P(v) = \left\| \sum_{k \in \mathbb{N}} (A_S^t)^k (P_{ee}) \right\|_1 \leq \sum_{k=0}^{\infty} \Omega^k \|P_{ee}\|_1 = \frac{1}{1-\Omega} P_{ee}(S)$$

□

La conséquence de (7.14) est que dès que $\Omega = 1$, rien n'empêche un site d'amplifier arbitrairement un PageRank. Dans le cas limite où $\omega = 1$ (site sans lien sortant externe, par exemple un site commercial n'ayant pas envie que l'internaute aille voir ailleurs), on a une amplification infinie et un phénomène de court-circuit. Nous retrouvons le phénomène bien connu du puits de rang (cf [PBMW98]), vu cette fois du point de vue de l'amplification : un ensemble de pages

En dépit de soupçons quant à l'impartialité de Google quand il s'agit de définir une pouponnière, la société Google n'a jamais été reconnue coupable : un moteur de recherche classe ses résultats comme il l'entend.

3. Notons que cette distribution peut dans une certaine mesure être influencée par des modifications de la structure de S. Mais comme nous l'avons vu, des variations trop importantes peuvent être le signe d'une pouponnière.

sans lien sortant va absorber et accumuler tout le PageRank externe reçu jusqu'à épuisement du flot.

Heureusement, la sur-amplification est contrôlée par le facteur *zap*.

7.5.2 Introduction du facteur *zap*

Nous allons à partir de maintenant quitter le cadre idéal où G est fortement connexe et apériodique pour considérer un graphe du Web G quelconque. Il nous faut donc en particulier choisir quel modèle de PageRank adopter, et notre choix s'est naturellement porté sur le PageRank non-compensé avec facteur *zap*. Grâce au théorème 10, nous savons en effet que c'est un modèle strictement équivalent au PageRank μ-compensé généralement utilisé, à la différence près qu'il permet de travailler avec un flot de *zap* constant.

P est donc maintenant l'unique vecteur vérifiant $P = dA^tP + (1-d)Z$, Z étant une distribution recouvrante et d le facteur *zap*.

Nous avons également besoin d'étiqueter le flot de *zap*. Nous pourrions le répartir en flot externe et interne, selon que le *zap* nous fasse sortir du site où l'on est ou non, mais nous avons jugé plus judicieux de considérer le flot de *zap* comme un flot externe dans sa totalité, et de séparer le flot externe en flot externe de clic et flot externe de *zap*. Nous allons continuer à réserver les termes de PageRank externe entrant et PageRank externe sortant au flot externe de clic, et nous allons par analogie avec l'électricité définir deux nouveaux flots de PageRank liés au *zap* : le PageRank induit, noté P_{ind}, qui est la probabilité[4] de venir sur une page par *zap*, et le PageRank dissipé, noté P_{dis}, qui est la probabilité[4] de quitter une page par *zap*.

Nous avons maintenant un bestiaire constitué de six PageRanks, ou plutôt de six flots, qui sont récapitulés dans le tableau 7.2 (rappel : s est le défaut stochastique, définit par $s = \mathbf{1}^t - A.\mathbf{1}^t$).

flot	entrant	sortant
interne	$P_{ei} = dA_S^tP$	$P_{si} = d(A_S\mathbf{1}^t).\times P$
externe (clic)	$P_{ee} = d(A - A_S)^tP$	$P_{se} = d((A - A_S)\mathbf{1}^t).\times P$
externe (*zap*)	$P_{ind} = (1-d)Z$	$P_{dis} = (1-d)P + ds.\times P$

FIG. 7.2 – *Les six flots de PageRank dans le modèle non-compensé*

On remarquera au passage que $P = P_{ei} + P_{ee} + P_{ind} = P_{si} + P_{si} + P_{dis}$.

Les lois de conservation interne et externe sont toujours valables. Nous ne chercherons pas cette fois à le prouver par un calcul matriciel, et nous nous contenterons de les justifier par le fait que l'on est présence d'un flot stationnaire. En particulier, la loi de conservation externe sur un site S s'écrit maintenant :

$$P_{ee}(S) + P_{ind}(S) = P_{se}(S) + P_{dis}(S) \tag{7.15}$$

4. Même si le modèle non-compensé fait que l'on ne devrait plus parler de probabilité, nous nous permettrons sporadiquement de continuer à utiliser ce terme, même si il est plus correct de parler de flot.

L'équation nous donne un résultat intéressant : si un site S possède un PageRank supérieur à $Z(S)$, son PageRank sortant externe est inférieur à son PageRank entrant externe, avec égalité si, et seulement si, $P(S) = Z(S)$ et $s(S) = 0$. Cela signifie qu'un site S ne peut espérer avoir un PageRank supérieur au PageRank par défaut $Z(S)$ qu'à la condition de donner moins que ce qu'il reçoit.

7.5.3 *Zap* et coefficient d'amplification

Avec l'introduction du facteur *zap*, l'équation (7.12) devient maintenant

$$P = (Id - dA_{\mathrm{S}}^t)^{-1}(P_{ee} + P_{ind})$$

L'encadrement vu lors de la section 7.5.1 est toujours valable en remplaçant A par dA et P_{ee} par le PageRank entrant externe total $P_{ee} + P_{ind}$, et en posant par convention $\frac{d_i(v)}{d(v)} = 0$ si $d(v) = 0$. On obtient ainsi le lemme 7.

Lemme 7 *Le facteur d'amplification α' défini par $\alpha'(S) = \frac{P(S)}{P_{ie}(S)+P_{ind}(S)}$ vérifie*

$$\frac{1}{1 - d\omega} \leq \alpha'(S) \leq \frac{1}{1 - d\Omega}. \tag{7.16}$$

Preuve : On procède exactement comme pour la preuve du lemme 6. Si l'on considère un site fixé S, et si l'on restreint P_{ee} et P_{dis} à leurs valeurs sur S, la première inégalité s'obtient en écrivant

$$\begin{aligned} P(S) &= \sum_{v \in S} P(v) = \left\| \sum_{k \in \mathbb{N}} (dA_S^t)^k (P_{ee} + P_{ind}) \right\|_1 \\ &\geq \sum_{k=0}^{\infty} (d\omega)^k (\|P_{ee}\|_1 + \|P_{ind}\|_1) \\ &\geq \frac{1}{1 - d\omega} \left(P_{ee}(S) + P_{ind}(S)\right) \end{aligned}$$

et la deuxième de manière similaire :

$$\begin{aligned} P(S) &= \sum_{v\in S} P(v) = \left\| \sum_{k\in\mathbb{N}} (dA_S^t)^k (P_{ee} + P_{ind}) \right\|_1 \\ &\leq \sum_{k=0}^{\infty} (d\Omega)^k (\|P_{ee}\|_1 + \|P_{ind}\|_1) \\ &\leq \frac{1}{1-d\Omega} (P_{ee}(S) + P_{ind}(S)) \end{aligned}$$

□

Valeur numérique

Pour un site réel, il est tout à fait possible d'avoir $\omega = \Omega = 0$ (site dépourvu de lien interne), ou au contraire $\omega = \Omega = 1$ (site sans lien externe, et dont toutes les pages possèdent au moins un lien). Ainsi, le coefficient d'amplification peut varier entre 1 (le site ne tire aucun profit du PageRank qu'il reçoit) et $\frac{1}{1-d}$ (utilisation maximale du PageRank reçu). Comme d est une constante universelle qui vaut $0,85$, on en conclut qu'à PageRank entrant externe total fixé, le PageRank d'un site peut selon sa structure varier avec un facteur $\frac{20}{3}$. Par exemple un site très mal structuré peut en se restructurant avoir un nouveau PageRank égal à environ 666%[5] fois l'ancien PageRank.

Robustesse du PageRank

Bianchini *et al.* [BGS02, BGS03] montrent que l'effet que peut produire un site sur le Web est controlé par le PageRank de ce site. Plus précisément, si l'on considère un graphe dynamique entre deux instants t et $t+1$, ils ont prouvé que :

$$\sum_{v\in V} |P_t(v) - P_{t+1}(v)| \leq \frac{2d}{1-d} \sum_{s\in S} P_t(s) \tag{7.17}$$

Ce résultat peut également se déduire du lemme 7 : si un site S change entre t et $t+1$, la plus grande variation relative possible est celle où l'on passe de $\alpha'(S) = 1$ à $\alpha'(S) = \frac{1}{1-d}$. Cette modification de la structure du site, qui correspond à la création d'un puits de rang, ne pouvant se faire qu'au détriment du PageRank externe, et donc du PageRank entrant externe, on a alors forcément $P_{eet} \geq P_{eet+1}$, d'où une variation d'au plus $\frac{d}{1-d}P(S)$. Comme Bianchini *et al.* travaillent ici dans un modèle compensé (la somme des PageRanks est constante), une variation de $\frac{d}{1-d}P(S)$ dans S génère la même variation hors de S, ce qui nous donne l'inégalité (7.17).

5. Ce joli chiffre est une preuve supplémentaire de la nécessité d'avoir $0,85$ comme valeur de d.

7.5.4 Amplification d'une page donnée

Pour un site, l'intérêt du PageRank est avant tout d'être visible par les internautes. En particulier, l'administrateur d'un site sera vraisemblablement moins intéressé par un PageRank élevé sur tout son site que par un PageRank très élevé sur quelques pages, voire sur une page. Mieux vaut donc concentrer son PageRank sur une page d'accueil généraliste plutôt que de le répartir entre plusieurs pages spécialisées. Nous allons donc considérer le problème suivant : considérons un site S de $n+1$ pages alimenté par un PageRank entrant externe P_{ee}. Comment maximiser le PageRank d'une page donnée $v_0 \in S$?

La réponse n'est pas difficile une fois que l'on a remarqué que la structure optimale est celle où toutes les pages de S autre que v_0 pointent vers v_0 (et seulement v_0) et v_0 pointe sur au moins une autre page de S. v_0 récupère ainsi, aux dissipations près, tout le PageRank des autres pages, et récupère son propre PageRank à un facteur d^2 près, ce qui est le maximum possible dans un graphe où, rappelons-le, les liens d'une page vers elle-même ne sont pas pris en compte. On a alors

$$P(v_0) = \frac{P_{ee}(v_0)}{1-d^2} + \frac{Z(v_0)}{1+d} + d \sum_{\substack{v \in S \\ v \neq v_0}} \left(\frac{P_{ee}(v)}{1-d^2} + \frac{Z(v)}{1+d} \right) \tag{7.18}$$

Dans le cas particulier où Z est la distribution uniforme, on obtient ainsi

$$P(v_0) \leq \frac{P_{ee}(S)}{1-d^2} + \frac{1+nd}{(1+d)\,|V|}, \tag{7.19}$$

avec égalité si, et seulement si, tout le PageRank entrant externe est concentré sur v_0, c'est-à-dire $P_{ee}(S) = P_{ee}(v_0)$.

On déduit de tout cela quelques stratégies pour améliorer le PageRank d'une page v_0, qui recoupent les recommandations que l'on peut trouver sur de nombreux sites destinés à l'amélioration de son PageRank :

- Demander aux administrateurs d'autres sites de toujours pointer, dans la mesure du possible, sur la page principale plutôt que sur une page spécifique. Éventuellement, mettre des scripts qui redirigent vers la page d'accueil tout accès d'une page extérieure au site[6].
- Toujours penser à mettre des liens de retour vers la page d'accueil, et limiter autant que possible la profondeur du site (et donc la dissipation). Dans le cas particulier des sites à frames, mettre des balises `<noframe>` à l'intérieur desquelles la structure en étoile est explicitement représentée.

Concluons enfin par quelques remarques valables si Z est la distribution uniforme :

- Avec la stratégie optimale, un site formé simplement de deux pages qui pointent l'une sur l'autre possède un PageRank qui est au moins égal au PageRank moyen, même si le PageRank entrant externe est nul : $P(v_0) \geq \frac{1}{|V|}$.

6. Cette recommandation est à prendre avec précaution : la façon dont Google considère les redirections n'est pas très claire. De plus, cela peut rendre la navigation moins ergonomique pour l'internaute.

- Si $1 \ll n \leq |V|$ (par exemple un site qui génère dynamiquement des pages pointant sur v_0, comme un site de consultation d'une base donnée avec des liens autres que des formulaires), le rapport $\frac{P(v_0)}{P_{moyen}}$ vaut approximativement $\frac{d}{1+d}n$. Il est donc possible d'augmenter linéairement son PageRank sur v_0. Dans la réalité, ceci est valable à condition que les robots de Google prennent la peine d'explorer toutes les pages[7], et que le but de toutes ces pages ne soit pas uniquement d'augmenter son PageRank[8].

7.6 Cas réel : FlowRank et BlowRank

L'objectif de cette section est d'adapter le calcul théorique vu lors de la section 7.4.3 aux situations réelles.

7.6.1 Relations à l'équilibre avec le facteur *zap*

Maintenant que nous avons eu le temps de nous familiariser avec les flots induits et dissipés, nous pouvons réécrire les équations vues au cours de la section 7.4.

Avec l'introduction du facteur *zap*, l'équation (7.12), qui décrit le lien entre PageRank entrant externe et PageRank, devient comme nous l'avons vu

$$P = (Id - dA_{\mathrm{S}}^t)^{-1}(P_{ee} + (1-d)Z) \tag{7.20}$$

La relation d'équilibre du PageRank externe, équivalent de l'équation (7.13), s'obtient quant à elle en fusionnant (7.20) avec la définition de P_{ee}. On obtient ainsi :

$$\begin{aligned} P_{ee} &= dA_e^t P_{ee} + Z_e, \text{ avec} \\ A_e^t &= (A - A_{\mathrm{S}})^t (Id - dA_{\mathrm{S}}^t)^{-1} \text{ et} \\ Z_e &= d(1-d)A_e^t Z \end{aligned} \tag{7.21}$$

Lemme 8 *Le rayon spectral de A_e est inférieur à* 1.

Preuve :

La preuve est similaire à celle du lemme 3 : nous montrons en effet que A_e est sous-stochastique (au sens large). Nous allons pour cela montrer que le rayon spectral de dA_e est inférieur à d. Comme A_e est positive, il suffit, d'après le théorème de Perron-Frobenius, de montrer que pour tout vecteur X positif, $\|dA_e^t X\|_1 \leq d \|X\|_1$. Pour cela, nous commençons par réécrire dA_e^t :

7. À mon grand regret, Google n'a pas encore fini d'explorer la *page qui pointe vers toutes les pages* (cf section 1.4 page 21), ce qui explique que cette dernière n'ait pas encore un PageRank maximal…

8. Dans le cas contraire, attention à la sanction.

$$\begin{aligned} dA_e^t &= \sum_{k=0}^{\infty} \left(dA^t (dA_{\mathrm{S}}^t)^k - (dA_{\mathrm{S}}^t)^{k+1} \right) \\ &= dA^t + \sum_{k=1}^{\infty} \left(dA^t (dA_{\mathrm{S}}^t)^k - (dA_{\mathrm{S}}^t)^k \right) \\ &= dA^t + dA^t M - M, \text{ avec } M = \sum_{k=1}^{\infty} (dA_{\mathrm{S}}^t)^k \end{aligned}$$

Considérons maintenant un vecteur X positif. Comme $dA_e^t X$, $dA^t X$, $dA^t MX$ et MX sont tous des vecteurs positifs, on a

$$0 \leq \left\| dA_e^t X \right\|_1 = \left\| dA^t X \right\|_1 + \left\| dA^t MX \right\|_1 - \left\| MX \right\|_1$$

Le rayon spectral de A^t étant inférieur à 1, on a $\left\| dA^t MX \right\|_1 \leq \left\| MX \right\|_1$, d'où

$$\left\| dA_e^t X \right\|_1 \leq \left\| dA^t X \right\|_1 \leq d \left\| X \right\|_1$$

C.Q.F.D. □

Remarque 13
L'inégalité $\left\| dA^t MX \right\|_1 \leq \left\| MX \right\|_1$ utilisée dans la preuve du lemme 8 est très grossière. Dans la pratique, on peut donc légitimement s'attendre à ce que le rayon spectral de A_e soit plus petit que 1, et donc que l'équation 7.21 permette de trouver P_{ee} avec une raison de convergence plus petite que d. Les résultats présentés par Arasu et al. *(cf [ANTT01]) vont dans ce sens et nous autorisent à espérer une convergence très rapide en pratique.*

Application : estimation du PageRank d'un site

Pour l'administrateur d'un site S, pouvoir estimer l'importance de ses pages sans faire appel à une aide extérieure ni chaluter le Web indexable peut être d'un intérêt certain. Par exemple, cette importance pourrait être incorporée à un moteur de recherche interne. Or, d'après (7.20), il suffit pour cela d'arriver à estimer le PageRank entrant externe sur S.

En effet, si on définit la fonction $\mathrm{SpeedRank}(M, X)$, inspirée de l'algorithme 5 page 98, comme une fonction qui renvoie, pour M positive de rayon spectral strictement inférieur à $\frac{1}{d}$ et X positif, le vecteur P vérifiant $P = dM^t P + X$, alors

$$P_S = \mathrm{SpeedRank}\left(A_S, \left((P_{ee})_S + (1-d)\, Z_S \right) \right)$$

Pour estimer ce PageRank entrant externe, deux approches locales sont possibles :

- Nous avons vu que le PageRank est censé, dans une certaine mesure, essayer de représenter le comportement des internautes réels (cf section 5.2.2 page 82). On peut alors prendre,

comme estimation du PageRank externe entrant, le nombre de clics réels de l'extérieur du site vers une page du site, mesuré à partir de l'analyse des logs du serveur Web.

- Grâce, ou à cause, du facteur d, le classement par degré entrant est une approximation du classement par PageRank (cf section 5.3.5 page 90). En comptant le nombre de références externes associé à chaque page, obtenu là aussi grâce à une analyse des logs du serveur Web, on obtient donc une autre estimation de P_{ee}.

Une fois que l'on a une estimation de $(P_{ee})_S$, il faut l'équilibrer par rapport à Z_S. Une manière, parmi beaucoup d'autres, de réaliser cet équilibrage est de faire une moyenne pondérée des deux vecteurs. On pourra par exemple renvoyer comme estimation normalisée du PageRank sur S

$$P_S = \mathrm{SpeedRank}\left(A_S, \left(d\frac{(P_{ee})_S}{\|(P_{ee})_S\|_1} + (1-d)\frac{Z_S}{\|Z_S\|_1}\right)\right)$$

On remarquera au passage que la source de rang est alors normalisée à 1 quelque soit l'envergure du site S considéré. Comme il s'agit juste d'estimer l'importance relative des pages à l'intérieur de S, cela ne pose aucun problème.

7.6.2 Une première approche : FlowRank

Nous avons maintenant toutes les données en main pour construire l'algorithme FlowRank. Constatons tout d'abord que grâce au facteur *zap*, P_{ee} est complètement défini par l'équation (7.21), alors que l'équation (7.13) garantissait seulement d'obtenir un vecteur homogène. Afin d'éviter d'inverser explicitement les matrices $(Id - dA_S)$, pour $S \in \mathcal{S}$, nous allons avoir recours à la fonction $\mathrm{SpeedRank}$ définie précédemment. Grâce à cette fonction, toutes les valeurs que nous avons besoin de connaître peuvent être calculées :

- $(A - A_{\mathcal{S}})^t\,\mathrm{SpeedRank}(A_{\mathcal{S}}, e_v)$, où e_v est le vecteur valant 1 sur v, 0 ailleurs, donne la colonne de A_e^t associée à v. Ce calcul peut se limiter, en entrée comme en sortie, aux pages de V_{ext}. On obtient ainsi la matrice $V_{ext} \times V_{ext}$ que nous avons précédemment appelée $\tilde{A_e}$, et que nous continuerons par abus d'écriture à appeler A_e.. Notons également que ce calcul peut s'effectuer localement au sein de chaque site $S(v) \in \mathcal{S}$.
- Z_e vaut quant à lui $d(1-d)(A - A_{\mathcal{S}})^t\,\mathrm{SpeedRank}(A_{\mathcal{S}}, Z)$. Ce calcul peut lui aussi être réparti sur l'ensemble des sites. Notons que Z_e est nul en dehors de V_{ext}, nous pouvons donc nous contenter de considérer $\tilde{Z_e}$, restriction de Z_e à V_{ext}.
- Une fois calculé $\tilde{A_e}$ et $\tilde{Z_e}$, il est possible de calculer $\tilde{P_{ee}}$, restriction de P_{ee} à V_{ext} : $\tilde{P_{ee}} = \mathrm{SpeedRank}(\tilde{A_e}, \tilde{Z_e})$.
- Le PageRank non-compensé est alors donné par $P = \mathrm{SpeedRank}(A_{\mathcal{S}}, (P_{ee} + (1-d)Z))$. Encore une fois, cette étape peut se faire à l'échelle d'un site.

Toutes ces opérations sont résumées dans l'algorithme 9 page 147.
L'algorithme FlowRank possède de nombreux avantages :

- En fractionnant les calculs de SpeedRank au niveau des sites, on a la possibilité de stocker la matrice en mémoire vive, ce qui permet un calcul extrêmement rapide, d'autant plus rapide que SpeedRank ne fait aucun calcul de norme.
- Mis à part le calcul de P_{ee}, qui est global, tous les autres calculs peuvent être effectués de manière décentralisée à l'échelle des sites. Il est ainsi possible de paralléliser une grande partie des calculs.
- Pour prendre un compte la mise à jour d'un site donné, il n'est pas nécessaire de relancer l'intégralité de l'algorithme (contrairement au cas d'un algorithme totalement centralisé). Il suffit de relancer les calculs de SpeedRank au niveau du site en question, et de réactualisater P_{ee}. Utiliser les précédentes valeurs comme valeurs initiales pour les SpeedRank peut rendre l'opération très rapide.

Mais il a aussi des inconvénients. Ainsi, il faut effectuer $2k + 1 + |V_{ext}|$ calculs de SpeedRank. Même si SpeedRank, comme son nom l'indique, est très rapide, ce nombre reste élevé. De même, le calcul de P_{ee} est un SpeedRank sur une matrice $|V_{ext}| \times |V_{ext}|$. Si cette matrice ne tient pas en mémoire vive, on perd une grande partie de l'intérêt pratique du calcul décomposé. Pour résoudre ces problèmes, nous allons nous inspirer d'un algorithme concurrent du nôtre, l'algorithme *BlockRank*.

7.6.3 Algorithme distribué de Kamvar *et al.* : BlockRank

Parallèlement à nos travaux sur la structure en blocs des graphes du Web [MV02, MV03b] et les applications possibles au PageRank [MV03a, MV04], Kamvar *et al.* ont fait des recherches très similaires. Dans [KHMG03a], ils utilisent une décomposition en sites pour proposer un algorithme semi-distribué de calcul d'une estimation du PageRank : BlockRank. Cet algorithme est basé sur le calcul d'un PageRank local, qui avec nos notations est le PageRank sur $A_{\mathbb{S}}$, et d'un PageRank de site, basé sur une matrice sous-stochastique de BlockRank, notée B, définie sur le graphe quotient $G/\mathbb{S}$. L'estimation du PageRank d'une page v est alors donnée par le produit du PageRank local de v par le PageRank de $S(v)$, également appelé BlockRank[9]. Bien que FlowRank et BlockRank se ressemblent au premier coup d'œil, il existe d'importantes différences que nous voulons souligner :

- Aux erreurs introduites par ϵ près, FlowRank donne le PageRank non-compensé[10], et non une approximation. Ainsi, le problème de sous-estimation du PageRank des pages principales rencontré avec l'algorithme BlockRank [KHMG03a] ne se pose pas. Il y a évidemment un prix à payer au niveau de la complexité de l'algorithme.

9. Pour les détails complets, voir [KHMG03a].

10. Rappelons encore une fois que du point de vue théorique, il n'y a strictement aucune différence entre le PageRank non-compensé et le PageRank μ-compensé traditionnellement utilisé.

- La clé de voûte de FlowRank est le PageRank non-compensé, et l'algorithme SpeedRank qui en découle, alors que BlockRank reste sur l'idée du PageRank μ-compensé, qui est trois fois plus lent sur les « petits » graphes.
- Si le PageRank local dans un site S vaut, à normalisation près, $\mathrm{SpeedRank}(A_S, Z_S)$, la matrice globale de FlowRank, $\tilde{A}_e$, et celle de BlockRank, B, n'ont quasiment aucun point commun. Par exemple, B possède des transitions d'un site vers lui-même. On constatera aussi qu'alors que FlowRank utilise un *zap* externe Z_e adapté à G, le PageRank sur B utilise un *zap* uniforme.

7.6.4 Algorithme hybride : BlowRank

Le principal avantage de BlockRank sur FlowRank est ce passage de $|V_{ext}|$ à $k = |\mathcal{S}|$ rendu possible par les approximations. Ce gain se fait tout autant pour le nombre de calculs locaux que pour la taille de la matrice globale. Nous sommes donc tentés de réduire $\tilde{A}_e$ à une matrice $k \times k$. Pour cela nous devons réduire l'information de flot externe à un scalaire par site S, en remplaçant par exemple $(P_{ee})_S$ par $P_{ee}(S)$. Il nous faut alors définir la façon dont le PageRank entrant externe est injecté à l'intérieur de chaque site. Nous supposerons donc que chaque site S est muni d'une distribution de probabilité qui estime la distribution du PageRank entrant externe. Cette distribution, que nous noterons D_S, pourra être la distribution uniforme sur S, ou plus finement une distribution sur les pages d'entrée du site, qui sont les plus susceptibles d'être pointées.

Nous pouvons maintenant considérer l'algorithme hybride BlowRank (algorithme 10 page 148), qui diffère de FlowRank par une réorganisation imposée du flot externe à l'entrée de chaque site : tout se passe comme si à l'entrée de chaque site, le PageRank entrant externe (par clic) était collecté et redistribué selon D_S.

On obtient ainsi un algorithme qui ne nécessite que $2.k$ appels locaux à $\mathrm{SpeedRank}$, plus un appel global sur une matrice $k \times k$, ce qui le place en terme de performances au même niveau que BlockRank, alors que les approximations faites sont moindres, ce qui permet d'obtenir un PageRank moins perturbé par rapport au modèle global.

Algorithme 9: FlowRank : calcul décomposé du PageRank non-compensé

Données

- un graphe du web $G = (V, E)$;
- une partition en sites de G, $\mathcal{S} = (S_1, \ldots, S_k)$;
- une distribution de probabilité Z recouvrante ;
- un coefficient de *zap* $d \in]0, 1[$;
- un réel ϵ ;
- une fonction $\mathrm{SpeedRank}$, inspirée de l'algorithme 5, telle que $P = \mathrm{SpeedRank}(M, X)$ vérifie $P = dM^tP + X$, avec une précision ϵ.

Résultat

Le vecteur de PageRank non-compensé associé à (G, Z, d).

début

$A_e = 0_{|V_{ext}|\times|V_{ext}|}$
pour *Chaque site* S *de* $\mathcal{S}$ **faire**
Calculer A_S, sous-matrice $|S| \times |S|$ de $A_{\mathcal{S}}$
Calculer $\bar{A}_S$, sous-matrice $|S| \times |V_{ext}|$ de $(A - A_{\mathcal{S}})$
$Z_{S_e} = d(1-d)\bar{A}_S^t\,\mathrm{SpeedRank}(A_S, Z_S)$
pour *Chaque page* v *de* $S \cap V_{ext}$ **faire**
$(A_e^t)_{*,v} = \bar{A}_S^t\,\mathrm{SpeedRank}(A_S, e_v)$
fin
fin
$Z_e = \sum_{S\in\mathcal{S}} Z_{S_e}$
$P_{ee} = \mathrm{SpeedRank}(A_e, Z_e)$
pour *Chaque site* S *de* $\mathcal{S}$ **faire**
$P_S = \mathrm{SpeedRank}(A_S, (P_{ee})_S + (1-d)Z_S)$
fin

fin

Algorithme 10: BlowRank : approximation à la BlockRank de l'algorithme FlowRank

Données

- un graphe du web $G = (V, E)$;
- une partition en sites de G, $\mathcal{S} = (S_1, \ldots, S_k)$, chaque site S étant associé à une distribution de probabilité D_S ;
- une distribution de probabilité Z recouvrante ;
- un coefficient de *zap* $d \in]0, 1[$;
- un réel ϵ ;
- une fonction $\mathrm{SpeedRank}$, inspirée de l'algorithme 5, telle que $P = \mathrm{SpeedRank}(M, X)$ vérifie $P = dM^tP + X$, avec une précision ϵ.

Résultat

Une estimation du vecteur de PageRank non-compensé associé à (G, Z, d).

début

$B_e = 0_{k\times k}$

pour *Chaque site* S *de* $\mathcal{S}$ **faire**

$A_S = (A_{v,w})$, pour v et w dans S

$(\bar{A}_S)_{v,T} = \sum_{\substack{w \leftarrow v \\ w \in T}} A_{v,w}$, pour $v \in S$ et $T \neq S$

$Z_{S_i} = \mathrm{SpeedRank}(A_S, (1-d)Z_S)$

$Z_{S_e} = d\bar{A}_S^t Z_{S_i}$

$B_S = \mathrm{SpeedRank}(A_S, D_S)$

$(B_e^t)_{*,S} = \bar{A}_S^t B_S$

fin

$Z_e = \sum_{S\in\mathcal{S}} Z_{S_e}$

$P_{ee} = \mathrm{SpeedRank}(B_e, Z_e)$

pour *Chaque site* S *de* $\mathcal{S}$ **faire**

$P_S = P_{ee}(S)B_S + Z_{S_i}$

fin

fin

Conclusion

> *Internet est le rendez-vous des chercheurs, mais aussi celui de tous les cinglés, de tous les voyeurs et de tous les ragots de la terre.*
>
> *Alain* FINKIELKRAUT

Au bout de ces quelques années de thèse, il m'apparaît que ma compréhension des graphes du Web, des PageRanks et des mécanismes qui les régissent a bien augmentée depuis mes timides débuts où, suivant une piste que m'avait tracée Daniel Krob, je cherchais à détecter des *points chauds* du Web ; compréhension que je me suis efforcé de faire partager au lecteur au long de cet ouvrage. Cette conclusion récapitule le travail déjà fait, mais aussi celui qui reste à faire.

—

Le premier chapitre a présenté cette réalisation humaine qu'est le Web. Définir ce dernier a généré plus de problèmes que de solutions, et face aux multiples facettes de la toile électronique, nous avons dû limiter notre champ d'étude à ce que nous avons appelé le Web indexable.

Nous avons ensuite considéré les *crawleurs*, ces chalutiers du Web qui parcourent sans cesse le Web indexable afin d'alimenter des bases de données, et avons étudié les tailles des plus grandes de ces bases, celles des moteurs de recherche. Nous avons en particulier mis en évidence les précautions qu'il était nécessaire de prendre face aux chiffres annoncés par les moteurs de recherche commerciaux, qui semblent parfois fantaisistes.

Nous nous sommes alors attardés sur quelques aspects de la structure de graphe induite par les hyperliens : la structure en nœud papillon, et les mauvaises interprétations qui en sont parfois faites ; la structure en sites, qui est intimement liée à l'arbre de décomposition des URLs. Nous avons laissé de côté quelques aspects structurels abondamment étudiés par ailleurs, comme la répartition des degrés entrants et sortants [AJB99, BA99, ABJ00] ou encore la structure en petit monde des graphes du Web [Ada99, Kle99, PLH01]. Ces aspects, bien qu'intéressants, ne rentrent pas dans le cadre de cette thèse.

—

La deuxième partie est rentrée dans le vif du sujet, les méthodes de classement de type PageRank. Nous avons commencé au cours du chapitre 4 par effectuer quelques rappels sur les chaînes

de Markov, leur représentation à l'aide de matrices stochastiques, et l'application du théorème de Perron-Frobenius aux matrices stochastiques pour trouver des distributions stationnaires. Nous avons également étudié quelques résultats de convergence pour des matrices sous-stochastiques. Tous les résultats présentés au cours de ce chapitre on très certainement été maintes fois présentés dans la littérature, mais nous avons choisi de les redémontrer afin d'introduire un formalisme abondamment utilisé par la suite, par exemple pour décrire les matrices sous-stochastiques. La roue a certainement été réinventée plus d'une fois au cours de ce chapitre, mais nous avons ainsi eu l'assurance que la taille des dites roues était la bonne.

Au cours du chapitre 5, nous avons défini les principes généraux qui régissent le PageRank, posé la problématique et répertorié les algorithmes les plus classiques. Nous avons voulu nous démarquer des autres états de l'art qui existent sur le sujet [Bou01, BJM02, BGS02, BGS03, LM04] par un éclairage différent, avec notamment une unification de la double interprétation — stochastique et en termes de flots — associée aux différents algorithmes. Certains problèmes soulevés par ce chapitre auraient mérité une étude plus approfondie, que j'espère avoir prochainement l'occasion de réaliser. Il en est ainsi du lien entre la convergence du classement d'un PageRank et celui de sa distribution, qui est traité dans la section 5.5.

Les deux derniers chapitres ont ensuite été l'occasion de présenter de nouveaux algorithmes de PageRank. Le lecteur aura ainsi découvert l'algorithme *BackRank*, qui modélise la possibilité pour un surfeur de revenir en arrière au cours de sa navigation. Nous avons montré que cet algorithme n'était pas plus compliqué à mettre en place qu'un algorithme de PageRank classique, tout en présentant de nombreux avantages pour ce qui est de la convergence et de la gestion des pages sans lien. Enfin, après avoir abandonné l'interprétation stochastique du PageRank pour une décomposition en flots, nous avons introduits deux algorithmes de calcul de PageRank basés sur la structure fortement clusterisée des graphes du Web : un algorithme exact, *FlowRank* ; un algorithme approché, inspiré de l'algorithme *BlockRank* proposé dans [KHMG03a] : *BlowRank*. La décomposition sur laquelle se basent ces algorithmes ouvre la voie à un large éventail de méthodes d'estimation semi-distribuées et locales du PageRank.

L'étude de ces algorithmes est loin d'être terminée, il reste encore à les tester sur de très grands graphes et, pourquoi pas, à les incorporer à un moteur de recherche réel. À la date où j'écris ces lignes, je réalise avec Mohamed Bouklit des expériences sur les graphes disponibles sur le site du projet *WebGraph* [BV], dont le plus grand fait 118 millions de sommets. Les résultats, très prometteurs, vont dans le même sens que ceux présentés dans ce mémoire.

Annexe A : Théorème de Perron-Frobenius

En 1907, Oskar Perron (1880-1975) a publié une théorie des matrices strictement positives, que Georg Ferdinand Frobenius (1849-1917) a étendu en 1908, 1909 et 1912 au cas des matrices positives au sens large[1]. Le théorème de Perron-Frobenius, qui résume cette théorie, est en quelque sorte la pièce centrale de la plupart des algorithmes de convergence des matrices stochastiques, et en particulier des algorithmes de PageRank. Nous avons donc cru intéressant de mettre une démonstration en appendice, car en plus de garantir la convergence des algorithmes, le concept de flot est inhérent à la preuve (lemme de propagation de la stricte inégalité).

Soit $A = (a_{i,j})_{1 \leq i,j \leq n}$ une matrice carrée positive de taille n, irréductible. Alors, il existe une valeur propre de A, notée r, strictement positive, qui a les propriétés suivantes :

(a) Il existe un vecteur propre gauche et un vecteur propre droit associés à r strictement positifs.

(b) pour toute valeur propre λ de A, $|\lambda| \leq r$.

(c) L'espace propre associé à r est de dimension 1.

(d) Pour toute matrice B positive inférieure à A, et pour toute valeur propre β de B, $|\beta| \leq r$, avec égalité seulement si $A = B$.

(e) si $\frac{1}{r}A$ a la cyclicité d, alors les valeurs de A de module r sont exactement les $r.\omega^j, j = 1, \ldots, d$, avec $\omega = e^{\frac{2\pi i}{d}}$, et les espaces propres associés sont de dimension 1.

Preuve : Pour démontrer le théorème de Perron-Frobenius, nous aurons besoin du lemme suivant, que nous appellerons lemme de propagation de la stricte inégalité.

Lemme 9 *Si A est une matrice positive irréductible de taille n, et si x et y sont deux vecteurs positifs tels que $x \leq y$, avec au moins une composante pour laquelle il y a inégalité stricte, alors $\sum_{k=0}^{n-1} A^k x < \sum_{k=0}^{n-1} A^k y$.*

En effet, soit i une composante pour laquelle il y a inégalité stricte. Comme la matrice A est irréductible, pour chaque composante j, il existe $k \in [0, n-1] \cap \mathbb{N}$ tel que $(A^k)_{j,i} > 0$. On en déduit que $A^k x \leq A^k y$, avec inégalité stricte en j. En utilisant l'opérateur $\sum_{k=0}^{n-1} A^k$, on s'assure que chaque composante « bénificiera » par propagation de la stricte inégalité présente en i. En d'autres termes, $\sum_{k=0}^{n-1} A^k x < \sum_{k=0}^{n-1} A^k y$.

1. Helmut Wielandt - 1910-2001 - expose en 1950 une approche plus simple du problème, qui est celle utilisée de nos jours.

Grâce à ce lemme, nous pouvons maintenant aborder la démonstration proprement dite.

(a) Considérons l'ensemble Γ des vecteurs positifs de $\mathbb{R}^n$ de norme 1 (on prendra la norme 1 : si $x = (x_i)_{1 \leq i \leq n}$, $\|x\|_1 = \sum_{i=1}^{n} |x_i|$).
Pour chaque γ de Γ, on pose

$$\mu_\gamma = \sup\{x \in \mathbb{R} / x\gamma \leq (A\gamma)\}$$

.
À l'évidence, μ_γ est un réel positif, puisque minoré par 0 et majoré par $n \|A\gamma\|_\infty$. Considérons maintenant $r = \sup_{\gamma \in \Gamma} \mu_\gamma$. r est un réel (fini) strictement positif, car on a l'encadrement :

$$0 < \frac{\min_{1 \leq i \leq n} \sum_{1 \leq j \leq n} a_{i,j}}{n} \leq r \leq n \|A\|_\infty$$

Pour montrer que r est une valeur propre, considérons une suite $(\gamma_n)_{n \in \mathbb{N}}$ d'éléments de Γ telle que μ_{γ_n} converge vers r. Comme Γ est compact, il est possible (théorème de Bolzano-Weierstrass) d'extraire une suite $(\gamma_{\phi(n)})_{n \in \mathbb{N}}$ qui converge vers un vecteur $\gamma_\infty \in \Gamma$.
γ_∞ est un vecteur propre (à droite) de A, associé à la valeur r. En effet, nous avons $r\gamma_\infty \leq A\gamma_\infty$. Si il n'y a pas égalité, alors d'après le lemme 9,

$$r((\sum_{k=0}^{n-1} A^k)\gamma_\infty) < A((\sum_{k=0}^{n-1} A^k)\gamma_\infty)$$

, ce qui contredit[2] le caractère maximal de r.
On a donc $r\gamma_\infty = A\gamma_\infty$, ce qui démontre que γ_∞ est un vecteur propre (droit) de A, associé à r. Le caractère strictement positif de γ_∞ est assuré par le lemme de propagation de la stricte inégalité.
En raisonnant avec A', on trouve une valeur propre s associée à un vecteur propre gauche de A strictement positif. Pour prouver (a), il nous suffit de montrer que $r = s$. Quitte à interchanger A et A', supposons $r \leq s$. Il suffit alors de prendre $x \neq 0$ tel que $Ax = sx$[3] et de constater que $s|x| = |Ax| \leq A|x|$, ce qui impose $s \leq r$, d'où l'égalité.

(b) La preuve est la même que pour montrer que $r = s$. Si $\lambda x = Ax$, avec $x \neq 0$, alors $|\lambda| |x| = |\lambda x| = |Ax| \leq A|x|$, d'où $|\lambda| \leq r$.

(c) Le fait que r soit une valeur propre simple se démontre aussi grâce au lemme de propagation de la stricte inégalité. En effet, si l'espace propre est de dimension supérieure à 2, il existe un vecteur propre $\gamma_\diamond$ non homogène à γ_∞. Le vecteur $\gamma_\diamond + (\max_{1 \leq i \leq n}(-\frac{\gamma_{\diamond i}}{\gamma_{\infty i}}))\gamma_\infty$ est également un vecteur propre associé à r. Par construction, il est non nul, positif, et une de ses

2. Quitte à normaliser $(\sum_{k=0}^{n-1} A^k)\gamma_\infty \ldots$
3. s est valeur propre de A, il existe donc un vecteur propre droit associé.

composantes au moins est nulle. Par propagation de la stricte inégalité, $n-1$ itérations de A vont rendre cette composante strictement positive, ce qui est contradictoire et démontre que r est une racine simple.

(d) Même preuve que (b) : si $\beta x = Bx$, avec $x \neq 0$, alors $|\beta|\,|x| = |\beta x| = |Bx| \leq B\,|x| \leq A\,|x|$, d'où $|\beta| \leq r$.
Cas d'égalité : $|\beta| = r$ impose $|\beta|\,|x| = r\,|x| = B\,|x| = A\,|x|$. $|x|$ est donc strictement positif, puisque homogène à γ_∞. $(A-B)$ est une matrice positive qui vérifie $(A-B)\,|x| = 0$, donc $(A-B) = 0$.

(e) Quitte à considérer $\frac{1}{r}A$, on peut supposer que $r = 1$. Considérons la relation d'équivalence sur les composantes suivante : deux composantes i et j sont équivalentes si il existe une composante k telle que $a_{k,i}$ et $a_{k,j}$ soient strictement positifs, ou une composante l telle que $a_{i,l}$ et $a_{j,l}$ soient strictement positifs. Alors, la cyclicité de A est égale au nombre de classe d'équivalence dans les composantes. En effet, la cyclicité de A correspond à la cyclicité sur le graphe sous-jacent (les sommets correspondant aux composantes et les arêtes aux coefficients $a_{i,j}$ non nuls). Comme le graphe est fortement connexe (puisque la matrice est irréductible), cette cyclicité se trouve en mettant dans la même classe d'équivalence les successeurs et les antécédents de chaque composantes.
Soient $I_0,\ldots,I_{d-1}$ les classes d'équivalence pour la relation antécédent/successeur, rangées dans l'ordre de succession.

- Si λ est une racine $d-$ième de l'unité, alors λ est valeur propre de A, et un vecteur propre associé est $x = (x_i)_{1\leq i\leq n}$, avec

$$x_i = \lambda^{-k}\gamma_{\infty i}, \text{ si } i \in I_k$$

  En effet, pour i dans I_k, on a

$$\begin{aligned}(Ax)_i &= \sum_{j=1}^{n} a_{i,j}x_j = \sum_{j=1}^{n} a_{i,j}\lambda^{-k+1}\gamma_{\infty j} \\ &= \lambda^{-k+1}\sum_{j=1}^{n} a_{i,j}\gamma_{\infty j} = \lambda\lambda^{-k}\gamma_{\infty i} = \lambda x_i\end{aligned}$$

- Soit λ une valeur propre de module 1, et x un vecteur propre associé. On écrit alors :

$$|x| = |Ax| \leq A\,|x|$$

  En fait, il y a forcément égalité, sinon par le lemme de propagation de la stricte inégalité, 1 ne serait plus valeur propre maximale. Notons au passage que $|x|$ est homogène à γ_∞. Pour des nombres complexes, la valeur absolue d'une somme n'est égale à la somme des valeurs absolues que si tous les éléments (non nuls) ont même phase. Comme les coefficients de A sont positifs, cela implique que tous les antécédents d'une

composante donnée de x_i par A ont même phase. Or, par construction, les successeurs ont aussi même phase. On peut en déduire que toutes les composantes d'une même classe d'équivalence ont la même phase. De plus, si $\phi(k)$ est la phase de la composante I_k, alors $\phi(k-1)/\phi(k) = \lambda$. Par cyclicité, et en utilisant la convention d'écriture $I_d = I_0$, on a :

$$1 = \frac{I_0}{I_d} = \prod_{k=1}^{d} \frac{I_{k-1}}{I_k} = \prod_{k=1}^{d} \lambda = \lambda^d$$

λ est donc une racine $d-$ième de l'unité.
Remarquons enfin que compte tenu de tout ce qui a été dit, le vecteur x est colinéaire au vecteur y défini par

$$y_i = \lambda^{-k}\gamma_{\infty i}, \text{ si } i \in I_k.$$

L'espace propre associé à λ est donc de dimension 1. C.Q.F.D.

□

Annexe B : Petite étude du PageRank sur le site de l'INRIA

À partir d'une capture du graphe du site `http://www.inria.fr`, nous avons appliqué un algorithme de type PageRank pour voir quelles étaient les pages qui obtenaient le plus fort classement. Plusieurs observations intéressantes sont apparues :

- La justification de l'ablation, dans le calcul du PageRank, des feuilles du graphes (entendre par feuille nœud de degré sortant nul).
- Il est apparu vital d'enlever les liens d'une page vers elle-même, car cela provoque un phénomène de résonance. Sur le graphe de l'INRIA, avant de procéder à cette modification, le PageRank était largement dominé par

  `http://www.inria.fr/DR:/multimedia/Bsv-fra.html`,

  qui cumule le rôle de page autoréférencée et de racine d'un puits.
- Le choix de l'importance du « click aléatoire » s'avère primordial : s'il est trop petit, les puits ou quasi-puits vont absorber tout le PageRank. S'il est trop grand, l'aspect itératif du PageRank va disparaître, et le classement sera à peu près un classement selon le degré entrant. Dans le cas de l'INRIA, un click aléatoire de 10% à chaque itération paraît un bon compromis.

Résultats Il semble assez intéressant d'analyser les dix premières URLs renvoyées par notre algorithme de PageRank (voir tableau B.1). On constate que, tout en étant bien sûr corrélé au classement des degrés entrant, il s'en démarque significativement (comparer les tableaux B.1 et B.2).

Au niveau de la pertinence, les pages renvoyées par notre PageRank apparaissent bien trouvées dans leur ensemble (page d'accueil en première place, pages de type « index » ou « plan »), à l'exception notable de deux pages :

`http://www.inria.fr/rapportsactivite/RA94/RA94.kw.html` et
`http://www.inria.fr/rapportsactivite/RA94/RA94.pers.html`.

URL (http ://www.inria.fr/...)	PR local	PR Google	$\mathcal{D}e$
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

TAB. B.1 – *Les dix premières URLs de www.inria.fr d'après un PageRank local. Comparaison avec Google.*

URL (http ://www.inria.fr/...)	$\mathcal{D}e$
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

TAB. B.2 – *Les dix URLs possédant le plus fort degré entrant*

Après vérification, et comme on pouvait s'y attendre, ces deux pages s'avèrent être les deux principaux nœuds d'un quasi-puits, à savoir `rapportsactivite/RA94/`. Ces deux pages, présentant à la fois un fort degré entrant et étant dans un quasi-puits, paraissent très difficiles à écarter simplement à l'aide d'un PageRank local.

Comparaison avec Google Google attribue un classement de 9/10 à la page d'accueil de l'INRIA et de 8/10 aux autres dix premières pages du PageRank local, à l'exception de

`http://www.inria.fr/rapportsactivite/RA94/RA94.kw.html` et
`http://www.inria.fr/rapportsactivite/RA94/RA94.pers.html`,

qui se voient attribuer la note de 6/10. Deux principales remarques :

- Les deux pages `RA94` avaient un PageRank local quasi-égal aux autres pages, exception faite de la page d'accueil. Le PageRank global de Google a réussi à les isoler. On peut avancer comme explication l'existence probable de nombreux liens de pages extérieures vers les pages de type « index », alors qu'il est fort probable qu'il existe très peu de pages extérieures pointant vers `RA94`.
- La note de 6/10 attribuée à

  `http://www.inria.fr/rapportsactivite/RA94/RA94.kw.html` et
  `http://www.inria.fr/rapportsactivite/RA94/RA94.pers.html`.

  reste élevée, certainement plus élevée que ce que l'on voudrait. Beaucoup de pages d'accueil de sites considérés comme plus intéressants n'ont pas cette note.

Annexe C : Persistence et diffusion des fichiers dans les réseaux pair-à-pair

L'impossibilité de finir de télécharger des fichiers est une situation fréquente dans les réseaux pair-à-pair de partage de fichiers. Si parfois ce problème semble intrinsèquement lié à la nature du réseau, il semble que même des réseaux *intelligents* vis-à-vis du téléchargement peuvent aboutir à une situation où toutes les parties du fichier sont disponibles sauf une.

Nous proposons un modèle simple et évolutif de partage de fichiers qui peut s'appliquer à tous les protocoles de téléchargement par blocs, comme ou BitTorrent. Les simulations montrent que le cas du *fichier manquant* peut se produire même pour des fichiers populaires, et donnent quelques pistes théoriques.

Ces nouveaux résultats permettent une autre approche des problèmes de téléchargements dans les réseaux de partage de fichiers, et de nouvelles stratégies sont proposées.

C.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[SMK+01, ZKJ01, RFH+01, HJS+03] ;
- publication and search of shared content[KRR01, CS02] ;
- 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[FP03] 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 C.3 shows interesting results coming from simulations of the model. In section C.4 we present some stability results for simulations of C.3. This highlights a frequent issue in real file-sharing networks, called *missing block* issue. Section C.5 compares the different upload strategies and gives characteristics of safe states, where the data can potentially survive forever...

C.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 :

- 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

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.

maximum number of connections and stay stuck to these maxima. As we show in section C.4, this leads to an interesting result, called "torpor" where upload can sometimes severely injure a torrent.

C.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.

C.3.1 Strategies

The state of a torrent at a given moment can be represented by a logical $T = (t_{p,k})_{\substack{1 \leq p \leq P \\ 1 \leq k \leq K}}$ matrix where :

$$t_{p,k} = \begin{cases} 1 \text{ if user } p \text{ possesses the block } k \\ 0 \text{ otherwise} \end{cases}$$

Anytime a block upload is finished, according to our assumptions, the uploader must choose another couple $(\text{peer}, \text{block})$ 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 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.

2. We remark that a decomposition strategy that uses random in both choices is not equivalent to the globally random strategy.

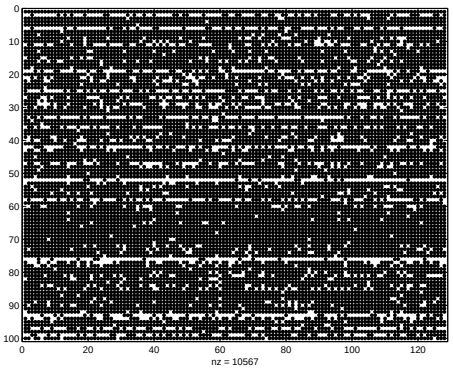


FIG. C.3 – *Stable state in a globally random upload strategy*

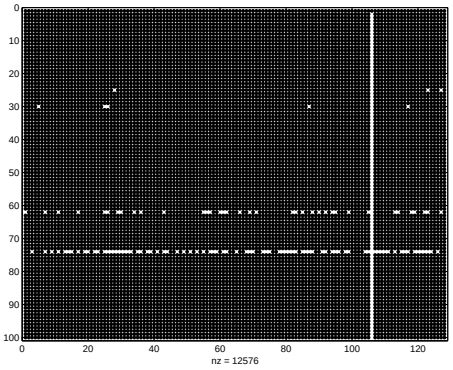


FIG. C.4 – *"torpor" state in a globally random upload strategy*

C.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 C.3 shows the matrix T in a typical stable state and figure C.4 shows a typical torpor state (lines represent peers ; line 0 stands for the seed).

3. Actually, for most people using ADSL, this ratio is 4 or 8. Later, we will give results for such ratios.

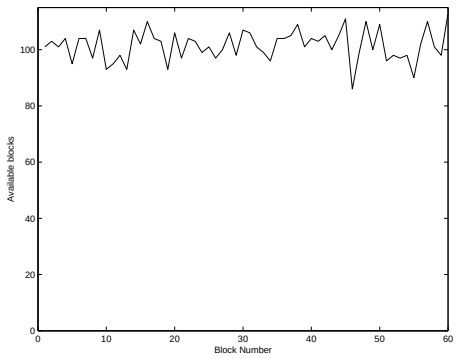


FIG. C.5 – *Population of each block in a stable state*

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.

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 C.5.

C.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.

C.4.1 *Torpor* apparition

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 C.5). 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.

C.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 :

$$P \geq S + S(K-1)(1+\frac{1}{r}) \tag{C.22}$$

Démonstration: 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 $P - P' \leq S(\frac{K-1}{r} + 1)$. These two inequalities lead to (C.22).

□

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 (C.22) 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.

C.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 C.22 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.

C.5 Efficiency of upload strategies

We now want to compare the different strategies seen in C.3.1 on other domains than stability, such as average download speed, original diffusion, density, and robustness to *very greedy peers*.

C.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 $\bar{D}_{max} = \frac{S+P}{P}U$. 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 $\min(\frac{S.K}{P}\bar{D}_{max}, \bar{D}_{max})$.

C.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 $P.K$ 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 $Q.K$ 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 :

Théorème 13 *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 $k \ln(k)$.*

Démonstration: 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

$$\mathbb{E}(T_k) = \sum_{\omega} T_k(\omega)$$

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

$$\begin{aligned}\mathbb{E}(T_k) = &\sum_{i_1...i_{k-1}} \Big((k+i_1+i_2+...+i_{k-1}).\\ &(1/k)^{i_1}\frac{k-1}{k}...(1-1/k)^{i_{k-1}}\frac{1}{k}\Big)\\ &= \frac{k-1}{k}...\frac{1}{k}\sum_{i_1...i_{k-2}}(1/k)^{i_1}...(1-2/k)^{i_{k-2}}.\\ &\Big(\sum_{i_{k-1}}(k+i_1+i_2+...+i_{k-1})(1-1/k)^{i_{k-1}}\Big)\\ &= \frac{(k-1)!}{k^{k-1}}\sum_{i_1...i_{k-2}}(1/k)^{i_1}...(1-2/k)^{i_{k-2}}.\\ &\Big((k-1+i_1+i_2+...+i_{k-2})\frac{1}{1-(1-1/k)}\\ &+\sum_{i_{k-1}}(1+i_{k-1})(1-1/k)^{i_{k-1}}\Big)\end{aligned}$$

Notice that

$$\sum_{i\in\mathbb{N}}(i+1)X^i = \frac{d}{dX}(\frac{1}{1-X})(X) = (\frac{1}{1-x})^2$$

it also true in $\mathbb{R}$ for $X = x$ with $|x| < 1$. Thus

$$\begin{aligned}\mathbb{E}(T_k) &= 1 + \frac{1}{1-1/k} + ...\frac{1}{1-(1-1/k)}\\ &= k(1/k + 1/(k-1) + ... + 1/1)\\ &= k(\ln(k) + \gamma + \epsilon(k))\end{aligned}$$

with $\epsilon(k)$ tending towards 0 and γ is the Euler constant.

□

C.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

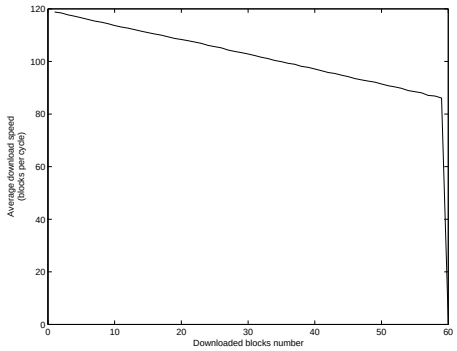


FIG. C.6 – *Average download speed*
in the download progress in a positive peer discrimination strategy

density, so the two parameters are basically proportional (see figure C.5). 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.

C.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 C.6 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 C.6), 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 :

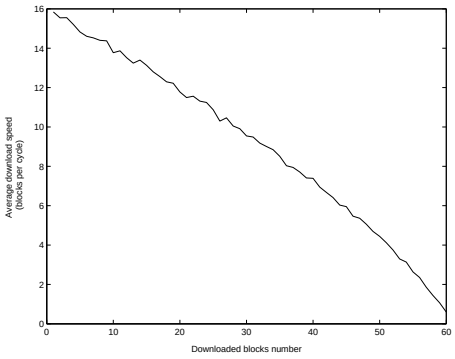


FIG. C.7 – *Average download speed in the download progress : GRS-stable case*

$$\frac{f(k)}{K-k} \text{ increases} \tag{C.23}$$

Démonstration: 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 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 $k \rightarrow \frac{f(k)}{K-k}$ 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 (C.23) for $k = K - 1$. Contrary, in GRS, the download speed (in a safe state) is basically an affine function of k (see figure C.7) that fulfill (C.23). 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 C.7 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 C.8 for a typical download distribution in GRS) to gain robustness. Especially when that does not harm the global download speed, as we have seen.

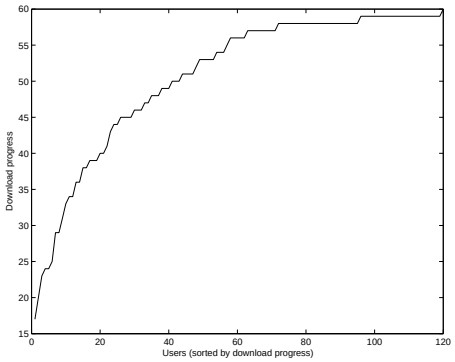


FIG. C.8 – *Repartition of download progresses among users in GRS*

C.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 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.

C.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. 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.

Bibliographie

[ABJ00] R. ALBERT, A. BARABASI, et H. JEONG. Scale-free characteristics of random networks : the topology of the world-wide web, 2000. Cité page(s) 149

[Ada99] L. A. ADAMIC. The small world web. In *Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries*, pages 443–452. Springer-Verlag, 1999. ISBN 3-540-66558-7. Cité page(s) 44, 149

[AJB99] R. ALBERT, H. JEONG, et A.-L. BARABASI. Diameter of the world wide web. *Nature*, vol. 401, pages 130–131, 1999. Cité page(s) 149

[ANTT01] A. ARASU, J. NOVAK, A. TOMKINS, et J. TOMLIN. Pagerank computation and the structure of the web : Experiments and algorithms, 2001. `citeseer.ist.psu.edu/arasu02pagerank.html` Cité page(s) 36, 121, 130, 143

[APC03] S. ABITEBOUL, M. PREDA, et G. COBENA. Adaptive on-line page importance computation. In *Proc. 12th International World Wide Web Conference*, pages 280–290. 2003. Cité page(s) 26, 28, 36, 104, 105, 106, 109

[BA99] A.-L. BARABASI et R. ALBERT. Emergence of scaling in random networks. *Science*, vol. 286, pages 509–512, 1999. Cité page(s) 149

[BB98] K. BHARAT et 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*, pages 379–388. Elsevier Science Publishers B. V., 1998. Cité page(s) 20, 26, 27, 33, 38, 39

[BBM03] T. BENNOUAS, M. BOUKLIT, et F. DE MONTGOLFIER. Un modèle gravitationnel du web. In *Actes de* ALGOTEL03 *5ème Rencontres Francophones sur les aspects Algorithmiques des Télécommunication*. 2003. `http://dept-info.labri.u-bordeaux.fr/algotel03/CameraReady/51.ps`. Cité page(s) 41

[Ber00] M. K. BERGMAN. The deep web : Surfacing hidden value. White Paper, 2000. `http://www.brightplanet.com/pdf/deepwebwhitepaper.pdf` Cité page(s) 19, 26, 27, 39

[BGS02] M. BIANCHINI, M. GORI, et F. SCARSELLI. Pagerank : A circuital analysis, 2002. `citeseer.nj.nec.com/bianchini02pagerank.html` Cité page(s) 79, 84, 130, 140, 150

[BGS03] M. BIANCHINI, M. GORI, et F. SCARSELLI. Inside pagerank. In *ACM Transactions on Internet Technology*. 2003. Cité page(s) 79, 84, 130, 140, 150

[BJM02] M. BOUKLIT et A. JEAN-MARIE. Une analyse de pagerank, une mesure de popularité des pages web. In *Actes ALGOTEL'02*. May 2002. Cité page(s) 79, 150

[BKM+00] A. BRODER, R. KUMAR, F. MAGHOUL, P. RAGHAVAN, S. RAJAGOPALAN, R. STATA, A. TOMKINS, et J. WIENER. Graph structure in the web. In *Proc. 9th International World Wide Web Conference*, pages 309–320. 2000. `http://www9.org/w9cdrom/160/160.html` Cité page(s) 21, 28, 36, 39

[BM03] M. BOUKLIT et F. MATHIEU. Effet de la touche back dans un modèle de surfeur aléatoire : application à pagerank. In *Journées Francophones de la Toile 2003*. 2003. `http://www.antsearch.univ-tours.fr/jft2003/` Cité page(s) 11, 13, 111

[BMC94] T. BERNERS, L. MASINTER, et M. M. CAHILL. Rfc-1738 : Uniform ressource locators (url), 1994. `http://www.ietf.org/rfc/rfc1738.txt` Cité page(s) 20, 45

[BMPW98] S. BRIN, R. MOTWANI, L. PAGE, et T. WINOGRAD. What can you do with a Web in your Pocket? *Data Engineering Bulletin*, vol. 21, n^o 2, pages 37–47, 1998. `citeseer.nj.nec.com/brin98what.html` Cité page(s) 44

[Bou01] M. BOUKLIT. Quelques méthodes de classement des moteurs de recherche basée sur la structure du graphe du web. Rapport tech., Université Montpellier II, 2001. `http://www.lirmm.fr/~bouklit/memoire.pdf` Cité page(s) 79, 150

[BP98] S. BRIN et L. PAGE. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, vol. 30, n^o 1–7, pages 107–117, 1998. `citeseer.ist.psu.edu/brin98anatomy.html` Cité page(s) 89, 90, 93

[Bra97] D. BRAKE. Lost in cyberspace. *New Scientist*, June 1997. Cité page(s) 39

[Bri02] M. BRINKMEIER. Communities in graphs, 2002. `citeseer.nj.nec.com/brinkmeier02communities.html` Cité page(s) 45

[BV] P. BOLDI et S. VIGNA. Projet webgraph. `http://vigna.dsi.unimi.it/papers.php#BoVWFI` Cité page(s) 109, 150

[CF02] C. COOPER et A. FRIEZE. Crawling on web graphs. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 419–427. ACM Press, 2002. ISBN 1-58113-495-9. Cité page(s) 36

[CGMP98] J. CHO, H. GARCÍA-MOLINA, et L. PAGE. Efficient crawling through URL ordering. *Computer Networks and ISDN Systems*, vol. 30, n^o 1–7, pages 161–172, 1998. `citeseer.ist.psu.edu/article/cho98efficient.html` Cité page(s) 27, 81, 82

[CM01] A. COCKBURN et B. MCKENZIE. What do web users do? an empirical analysis of web use. *International Journal of Human Computer Studies*, vol. 54, pages 903–922, 2001. Cité page(s) 83, 112

[Coh03] B. COHEN. Incentives build robustness in bittorrent, 2003. citeseer.nj.nec.com/cohen03incentives.html Cité page(s) 159, 160

[Col] T. COLLINS. Probabilities in the game of monopoly. http://www.tkcs-collins.com/truman/monopoly/monopoly.shtml Cité page(s) 66, 68, 69

[CP95] L. D. CATLEDGE et J. E. PITKOW. Characterizing browsing strategies in the world wide web. *Computer Networks and ISDN Systems*, vol. 27, n^o 6, pages 1065–1073, 1995. Cité page(s) 83, 87, 92, 111

[Cra] P. CRAVEN. Google's pagerank explained. http://www.webworkshop.net/pagerank.html Cité page(s) 27

[CS02] E. COHEN et S. SHENKER. Replication strategies in unstructured peer-to-peer networks, 2002. In The ACM SIGCOMM'02 Conference, August 2002. citeseer.nj.nec.com/cohen02replication.html Cité page(s) 159

[Dah00] M. DAHN. Counting angels on a pinhead : Critically interpreting web size estimates. *Online*, vol. 24, n^o 1, pages 35–40, 2000. http://www.infotoday.com/online/OL2000/dahn1.html Cité page(s) 20, 26

[DKM+01] S. DILL, S. R. KUMAR, K. S. MCCURLEY, S. RAJAGOPALAN, D. SIVAKUMAR, et A. TOMKINS. Self-similarity in the web. In *The VLDB Journal*, pages 69–78. 2001. citeseer.ist.psu.edu/dill01selfsimilarity.html Cité page(s) 39, 40

[Dur03] F. DURAND. Profils de vote : expérience et modélisation. Rapport tech., Laboratoire d'Économétrie de l'École Polytechnique, 2003. Cité page(s) 100

[EMT04] N. EIRON, K. S. MCCURLEY, et J. A. TOMLIN. Ranking the web frontier. In *Proc. 13th International World Wide Web Conference*, pages 309–318. May 2004. Cité page(s) 26, 38

[FCE95] Q. FENG, R. F. COHEN, et P. EADES. How to draw a planar clustered graph. *Journal of the ACM*, vol. 959, pages 21–31, 1995. Cité page(s) 45

[FGM+99] R. FIELDING, J. GETTYS, J. MOGUL, H. FRYSTYK, L. MASINTER, P. LEACH, et T. BERNERS-LEE. Hypertext transfer protocol – HTTP/1.1, 1999. citeseer.ist.psu.edu/fielding97hypertext.html Cité page(s) 22, 25

[FKK+00] R. FAGIN, A. R. KARLIN, J. KLEINBERG, P. RAGHAVAN, S. RAJAGOPALAN, R. RUBINFELD, M. SUDAN, et A. TOMKINS. Random walks with "back buttons" (extended abstract). pages 484–493. 2000. citeseer.nj.nec.com/fagin00random.html Cité page(s) 113

[FP03] F. L. FESSANT et S. PATARIN. Mldonkey, a multi-network peer-to-peer file-sharing program, April 2003. www.inria.fr/rrrt/rr-4797.html Cité page(s) 159

[Gau96] P. GAUCHER. Le monopoly pour les nuls, 1996. http://www.pps.jussieu.fr/~gaucher/monopoly/monopoly.html Cité page(s) 66

[Gen04] GENITRIX. Une histoire de réseaux, 2004. http://tempo2.phpnet.org/genitrix/dossiers/dossiers.php?id_dossier=30 Cité page(s) 17

[GL02] J.-L. GUILLAUME et M. LATAPY. Le graphe du web. *Tangente*, vol. Hors-Série, n^o 12, pages 12–15, 2002. Cité page(s) 36

[GLM02] J.-L. GUILLAUME, M. LATAPY, et F. MATHIEU. Tout le web accessible en quelques clics, 2002. http://www.liafa.jussieu.fr/~fmathieu/arbre.php Cité page(s) 20, 21

[GLV02] J. GUILLAUME, M. LATAPY, et L. VIENNOT. Efficient and simple encodings for the web graph. *In Proceedings of the 11-th international conference on the World Wide Web*, 2002. citeseer.nj.nec.com/guillaume02efficient.html Cité page(s) 41

[Gof71] W. GOFFMAN. A mathematical method for analyzing the growth of a scientific discipline. *J. ACM*, vol. 18, n^o 2, pages 173–185, 1971. ISSN 0004-5411. Cité page(s) 82

[Goo98] GOOGLE. http ://www.google.com/, 1998. Cité page(s) 79, 81

[Hav99] T. HAVELIWALA. Efficient computation of PageRank. Rapport tech., Computer Science Department, Stanford University, 1999. Cité page(s) 121

[HHMN99] M. R. HENZINGER, A. HEYDON, M. MITZENMACHER, et M. NAJORK. Measuring index quality using random walks on the Web. *Computer Networks (Amsterdam, Netherlands : 1999)*, vol. 31, n^o 11–16, pages 1291–1303, 1999. citeseer.ist.psu.edu/henzinger99measuring.html Cité page(s) 20, 27, 28, 34, 44

[HJS+03] N. HARVEY, M. JONES, S. SAROIU, M. THEIMER, et A. WOLMAN. Skipnet : A scalable overlay network with practical locality properties, 2003. citeseer.nj.nec.com/harvey03skipnet.html Cité page(s) 159

[KHMG03a] S. KAMVAR, T. HAVELIWALA, C. MANNING, et G. GOLUB. Exploiting the block structure of the web for computing pagerank, 2003. citeseer.ist.psu.edu/kamvar03exploiting.html Cité page(s) 36, 41, 44, 94, 97, 130, 145, 150

[KHMG03b] S. KAMVAR, T. HAVELIWALA, C. MANNING, et G. GOLUB. Extrapolation methods for accelerating pagerank computations. In *Proceedings of the Twelfth International World Wide Web Conference*. 2003. citeseer.nj.nec.com/kamvar03extrapolation.html Cité page(s) 25, 91, 92, 97, 121

[KL] A. KHALIL et Y. LIU. Experiments with pagerank computation. http://www.informatics.indiana.edu/fil/Class/b659/Projects/S04-g2/main.htm Cité page(s) 104

[Kle98] J. KLEINBERG. Authoritative Sources in a Hyperlinked Environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages

668–677. San Francisco, California, 25–27 janvier 1998. Cité page(s) 41, 51, 87, 112

[Kle99] J. KLEINBERG. The small-world phenomenon : An algorithmic perspective, 1999. Cité page(s) 149

[KRR01] J. KANGASHARJU, J. ROBERTS, et K. ROSS. Object replication strategies in content distribution networks, 2001.
citeseer.nj.nec.com/kangasharju01object.html Cité page(s) 159

[Lev01] S. LEVANT. http ://hipercom.inria.fr/soleil/, 2001. Cité page(s) 48

[LG98] S. LAWRENCE et C. L. GILES. Searching the world wide web. *Science*, vol. 280, pages 98–100, 1998. Cité page(s) 26, 27, 33

[LG99] S. LAWRENCE et C. L. GILES. Accessibility of information on the web. *Nature*, vol. 400, n^o 6740, pages 107–109, 1999. Cité page(s) 26, 27, 33, 39, 80

[LKVT00] W.-S. LI, O. KOLAK, Q. VU, et H. TAKANO. Defining logical domains in a web site. In *Proceedings of the eleventh ACM on Hypertext and hypermedia*, pages 123–132. ACM Press, 2000. ISBN 1-58113-227-1. Cité page(s) 44, 48, 51, 81

[LM04] A. N. LANGVILLE et C. D. MEYER. Deeper inside pagerank. Rapport tech., NCSU Center for Res. Sci Comp.,, 2004. Cité page(s) 36, 79, 84, 86, 89, 91, 92, 107, 108, 150

[Mat01] F. MATHIEU. Structure supposée du graphe du web. Rapport tech., Première journée Graphes Dynamiques et Graphes du Web, December 2001.
http://www.liafa.jussieu.fr/~latapy/gdgw.html Cité page(s) 11, 13, 36

[MB04] F. MATHIEU et 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*, pages 370–371. ACM Press, 2004. ISBN 1-58113-912-8. Cité page(s) 11, 13, 111

[MFJR+04] N. MILIC-FRAYLING, R. JONES, K. RODDEN, G. SMYTH, A. BLACKWELL, et R. SOMMERER. Smartback : supporting users in back navigation. In *Proceedings of the 13th international conference on World Wide Web*, pages 63–71. ACM Press, 2004. ISBN 1-58113-844-X. Cité page(s) 83, 87, 92, 112, 114, 115

[MR04] F. MATHIEU et J. REYNIER. File sharing in p2p : Missing block paradigm and upload strategies. Rapport Tech. RR-5193, INRIA, May 2004.
http://www.inria.fr/rrrt/rr-5193.html Cité page(s) 12, 13

[Mur00] B. H. MURRAY. Sizing the internet. A Cyveillance White Paper, July 2000. Cité page(s) 26

[MV02] F. MATHIEU et L. VIENNOT. Structure intrinsèque du web. Rapport Tech. RR-4663, INRIA, décembre 2002.
www.inria.fr/rrrt/rr-4663.html Cité page(s) 11, 13, 145

[MV03a] F. MATHIEU et L. VIENNOT. Aspects locaux de l'importance globale des pages web. In *5es rencontres francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL'2003)*. 2003. Cité page(s) 11, 13, 129, 145

[MV03b] F. MATHIEU et L. VIENNOT. Local structure in the web. In *12-th international conference on the Worl Wide Web*. May 2003.
`http://www2003.org/cdrom/papers/poster/p102/p102-mathieu.htm` Cité page(s) 11, 13, 145

[MV04] F. MATHIEU et L. VIENNOT. Local aspects of the global ranking of web pages. Rapport Tech. RR-5192, INRIA, May 2004.
`http://www.inria.fr/rrrt/rr-5192.html` Cité page(s) 11, 13, 41, 129, 145

[Net04] NETCRAFT. http ://www.netcraft.com/, 2004. Cité page(s) 43

[OLB+02] E. O'NEILL, B. LAVOIE, R. BENNETT, A. DYER, et S. WORTHINGTON. Web characterization project, 2002.
`http://wcp.oclc.org` Cité page(s) 42, 44

[PBMW98] L. PAGE, S. BRIN, R. MOTWANI, et T. WINOGRAD. The PageRank Citation Ranking : Bringing Order to the Web. Rapport tech., Computer Science Department, Stanford University, 1998.
`http://google.stanford.edu/~backrub/pageranksub.ps` Cité page(s) 81, 84, 85, 86, 87, 88, 89, 93, 121, 137

[PLH01] A. R. PUNIYANI, R. M. LUKOSE, et B. A. HUBERMAN. Intentional walks on scale free small worlds. Rapport tech., Dept of Physics, Stanford University, 2001. Cité page(s) 149

[RFH+01] S. RATNASAMY, P. FRANCIS, M. HANDLEY, R. KARP, et S. SHENKER. A scalable content addressable network. In *Proceedings of ACM SIGCOMM 2001*. 2001.
`citeseer.nj.nec.com/ratnasamy01scalable.html` Cité page(s) 159

[RGM03] S. RAGHAVAN et H. GARCIA-MOLINA. Representing web graphs, 2003.
`citeseer.nj.nec.com/article/raghavan03representing.html` Cité page(s) 36

[RSWW01] K. RANDALL, R. STATA, R. WICKREMESINGHE, et J. WIENER. The link database : Fast access to graphs of the web. *Research Report 175, Compaq Systems Research Center, Palo Alto, CA*, 2001.
`citeseer.nj.nec.com/randall01link.html` Cité page(s) 41

[Sal89] G. SALTON. Automatic text processing. Massachusetts, 1989. Cité page(s) 80

[SC96] L. SALOFF-COSTE. Lectures on finite Markov chains. In *Lecture Notes on Probability Theory and Statistics*, edité par G. G. E. GINÉ et L. SALOFF-COSTE, n° 1665 in LNM, pages 301–413. Springer Verlag, 1996. Cité page(s) 61

[Sea] SEARCHENGINESHOWDOWN. http ://www.searchengineshowdown.com/. Cité page(s) 29, 32, 38

[SGG02] S. SAROIU, P. K. GUMMADI, et S. D. GRIBBLE. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and*

Networking 2002 (MMCN '02). San Jose, CA, USA, January 2002. http://www.cs.washington.edu/homes/tzoompy/publications/mmcn/2002/abstract.html Cité page(s) 170

[She88] O. B. SHEYNIN. A a. markov's work on probability. *Archive for History of Exact Science*, vol. 39, pages 337–377, 1988. Cité page(s) 61

[SMK+01] I. STOICA, R. MORRIS, D. KARGER, M. F. KAASHOEK, et H. BALAKRISHNAN. Chord : A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 ACM SIGCOMM conference*, pages 149–160. 2001. citeseer.nj.nec.com/stoica01chord.html Cité page(s) 159

[SP63] D. J. DE SOLLA PRICE. *Little Science, Big Science*. Columbia University Press, New York, 1963. Cité page(s) 81

[SP01] C. SHERMAN et G. PRICE. *The Invisible Web : Uncovering Information Sources Search Engines Can't See*. Independent Publishers Group, 2001. Cité page(s) 20

[Ste94] W. J. STEWART. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994. Cité page(s) 85

[Ste96] I. STEWART. Monopoly revisited. *Scientific American*, vol. 175, pages 116–119, October 1996. Cité page(s) 66

[Sul00] D. SULLIVAN. Invisible web gets deeper, August 2000. http://searchenginewatch.com/sereport/article.php/2162871 Cité page(s) 19

[Syd04] M. SYDOW. Random surfer with back step. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 352–353. ACM Press, 2004. ISBN 1-58113-912-8. Cité page(s) 113, 114

[TG97] L. TAUSCHER et 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, n° 1, pages 97–137, July 1997. Cité page(s) 83, 112

[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. Cité page(s) 104

[Who04] WHOIS. http ://www.whois.net/, 2004. Cité page(s) 43

[WM04] L. WANG et C. MEINEL. Behaviour recovery and complicated pattern definition in web usage mining. In *ICWE*, pages 531–544. LNCS, 2004. Cité page(s) 83, 87, 92

[YLYL95] B. YUWONO, S. L. LAM, J. H. YING, et D. L. LEE. A world wide web resource discovery system. In *Proc. 4th International World Wide Web Conference*. December 1995. Cité page(s) 80

[ZKJ01] B. Y. ZHAO, J. D. KUBIATOWICZ, et A. D. JOSEPH. Tapestry : An infrastructure for fault-tolerant wide-area location and routing. Rapport Tech. UCB/CSD-01-1141, UC Berkeley, avril 2001. citeseer.nj.nec.com/zhao01tapestry.html Cité page(s) 159

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