Imperial College of Science, Technology and Medicine
(University of London)
Department of Computing

PageRank: Three Distributed Algorithms

Douglas de Jager
<douglas.de-jager@imperial.ac.uk>

September, 2004

Submitted in partial fulfillment of the requirements of the MSc Degree in Computing Science of the University of London and the Diploma of the Imperial College of Science, Technology and Medicine

Abstract

This paper shows for the first time that there are multiple PageRank definitions, and that more is required to justify PageRank than is offered in the literature ([1], [2] and [5]). Adopting a formal approach, this paper provides a justification of PageRank. It also shows that the irreducibility restriction on the PageRank transition matrix [5] is unnecessary. This is important because it gives the personalisation vector more intuitive force.

Noting the difficulties in calculating PageRank centrally, the paper then shows, via the Chazan-Miranker theorem [10], that distributed calculation algorithms are possible; and, it presents three such novel algorithms. The first of these takes documents / pages as being of principal interest; the second and third, in an attempt to reduce communication overheads, focus attention on nodes/ webservers. Empirical test results are shown. These confirm reduced message numbers for algorithms 2 and 3. They also show that more investigation is required into suitable ϵ-thresholds within asynchronous environs.

Acknowledgments

I would like to thank my supervisor, Dr Jeremy Bradley, for encouraging my theoretical meanderings, and for reining in my interest in distributed search indices. This thesis would have been markedly poorer without your enthusiasm and your support.

I would like to thank the *lab rats*: Ricky, Saar, Jolo, Lukester, NicNac, JP, Lawrence, Dan. *It has been emotional*: coffee, coffee, coffee, emacs, pillows in the library, last orders,...

To my dearest Olivia, my love: thank you for your patient understanding. I am blowing you a kiss across the oceans.

To my brother, *my favoritest brother in all the world*: a wassail!

And, to parents: I dedicate this thesis to you. I do not often enough say, 'Thank you', but without you none of this would have been possible. You have been so encouraging of my dreams. Thank you so very much.

Contents

Chapter 1

Introduction

PageRank is an assignment of import to pages based on the hyperlink structure of the Web. It is fundamental to current conceptions of Web search [1].

The intuitive explanation of PageRank centres on the Random Surfer Model [2]. Suppose there exists a websurfer, a surfer who randomly follows weblinks from page to page, or, due to some randomly induced state of boredom, skips to some other page. PageRank may be described in terms of this surfer. The PageRank of a page, say p, is the probability of the surfer being at this page after some suitably large period of time - or, equivalently, it is the normalised time our surfer would spend at this page were she to surf the (unchanging) Web into infinity.

That this probability is well-defined (uniquely fixed) is, of course, not trivially clear. However, using Markov processes to underpin the Random Surfer Model, we can show that the probability is indeed well-defined. In Markov terminology, we may say that there exists a unique stationary distribution of probabilities corresponding to each of the pages in the web graph.

One may argue that a Markov underpinning is unnecessary: that Linear Algebra results suffice to show that the stationary distribution exists and is unique - in terms, perhaps, of the dominant eigenvector of the (Markov transition) matrix. There is reason to be sceptical, though. It is one thing to show that such a vector exists and is unique. It is quite another to show that this vector is the vector we require - namely a vector of intuitively justifiable PageRanks. What we require is not just some unique, computable vector, but a vector whose entries convey a sense of webpage import.[1]

Having established well-definedness of PageRank (in an intuitively justifiable way), our next concern is calculation. By the Ergodic Theorem for Markov chains [2], we know that the PageRank vector [3] is the limit vector [4] of $r_{n+1} = r_n \times M$, where M is the transition matrix. This forms the basis of the centralised model for calculation of PageRank [5]. A centralised crawler (set of crawlers) is used to retrieve the hyperlink structure of the Web - to put together a web-graph. From this, a transition matrix is constructed[6]. Then, with a transition matrix in place, iteration commences,

[1] *NB* Whilst Linear Algebra is not being endorsed as a means of intuitive justification, this is not to deny its potential usefulness. In fact, there is every reason to suppose PageRank research, along Linear Algebraic lines, could prove very useful - with respect to computation acceleration [3] or, perhaps, to spam-prevention.

[2] This theorem forms the basis of the Markov, well-definedness proof of PageRank.

[3] The vector which has as entries the PageRanks of the pages on the Web.

[4] $lim_{n\to\infty} r_n$

[5] This is clearly the case in [1]; just as it is when employing the accelerated techniques of [4, 5]

[6] The size of this square matrix being the size the Web - or the size of crawled Web.

only terminating when $\|r_{n+1} - r_n\|_1$ falls below some threshold value [7].

It needs be noted: whilst PageRank has so far been presented here as unambiguous, the principal authors offer different explicit formulations [1, 2, 5] (without acknowledging this fact). Also, two of the formulations ([1], [2]) do not accord with our Markov reasonings: they do not yield probability distributions[8], and, if they do converge uniquely (something not shown by the authors), it is unclear that the limits have any useful relation to the stationary distributions that we require. These are issues that will be explored in this paper. It will be shown that under the PageRank umbrella there are two different types – the offerings in the literature being special cases of these. It will also be shown that the transition matrices need not be irreducible, as stated in the literature [5]. Yet, despite all this, it well be seen that there is such a thing as the centralised calculation model, and that this model is as described above.

The centralised model is clearly ill-suited to the peer-to-peer domain [13, 14, 15, 16]. Firstly, centralised crawling and iteration is a heinous contravention of peer-to-peer philosophy. But, even were this fact to be ignored, anonymity guarentees in [13], for example, make the centralised model almost entirely irrelevant. Given the growing demand for peer-to-peer, it seems reasonable, then, to question whether an alternative model is achievable - that is, whether distributed PageRanking is possible.

However, it is not just the peer-to-peer domain that ought to motivate such enquiry. There are two points worth noting within an Web setting that ought to make us, at the very least, question the centralised model. The Web is not small - if the current Google crawl figures are correct, then there are in excess of 4.2 Billion pages. Also, the numbers are growing. Accordingly, scale is a concern for the centralised model: both in terms of memory and processing power - a lot of information needs to be stored after both the crawling and calculation phases; and, matrix calculations, even when employing sparse matrix formulations, are extremely processor-demanding.

Furthermore, crawling takes time. And, this poses a problem for PageRank accuracy. Webpages come and go. They change. As the only way for the centralised model to keep abreast of variation is to continually recrawl the Web [9], this means that PageRank figures within the centralised model can be substantially out of date [10].

These points make louder the call for distributed PageRanking. If distributed set-ups are achievable, then these could quite plausibly allay concerns over scale. Also, if webservers could be made responsible for their own pages' PageRanks, then this could equally plausibly undo the need for crawling - perhaps thereby giving more accurate PageRanks.

By Chazan and Miranker's work on chaotic relaxation [10], together with a Markov theorem about the existence of stationary distribution, we can show formally that distributed PageRanking algorithms are possible. In this paper we consider three such novel algorithms.

The first of these is a paragon of the fully asynchronous - each page's PageRank is updated at its host node independently of updates of all other pages' PageRanks. The algorithm proceeds as follows: PageRank update messages are initially sent from each page to outbound links; on receipt of a message, a page's PageRank is suitably updated; subsequently, if PageRank change is significant, then this change is propagated on to outbound links. The second and third algorithms are a response to perhaps the most obvious of the first algorithm's concerns: communication overheads. In spirit, these algorithms lie somewhere between the first and the centralised: asynchronous, but still involving matrices. Both send collections of update information – update information relating to multiple rather than individual pages. The difference between the algorithms is given by how

[7] $\|v\|_1 := \sum_{i=1}^{n} |v_i|$, where v has n entries.

[8] If the web-graph comprises more than one page.

[9] Changes are not somehow reported to the centralised system. The whole system needs to be recrawled repeatedly.

[10] Google took over five weeks to update the PageRank of http://www.doc.ic.ac.uk/~sm1603 - July, 2004.

eager they are to send and receive update messages.

All three algorithms are put through empirical testing, and the results are presented. These are encouraging. They confirm an appreciable reduction in messages for algorithms two and three. But, they do introduce questions over algorithm termination – whether any claims can be made over the suitability of certain ϵ values[11].

The rest of this paper is organised as follows: Chapter 2 provides a theoretical grounding for PageRank. It presents inconsistencies in PageRank definition; and, focusing on the principal formulation [1], it offers a formal justification. Chapter 3 turns attention to distributed PageRank algorithms. A formal justification is again provided, as are three novel algorithms. Chapter 4 presents empirical test results for the algorithms. Chapter 5 presents conclusions and describes potential, future, research directions.

Note: page and document will be used interchangeably in this paper, as will node and webserver. This is because, whilst the Web and peer-to-domains are different, the notion of PageRank is thought to transcend this difference.

[11] ϵ, broadly speaking, gives the difference in PageRank between successive iteration.

Chapter 2

PageRank

There are two explanations of PageRank in the literature.

In [2], Page and Brin present PageRank as a development of the following idea: an outlink from a webpage w to a webpage w^* evidences w^*'s import. More specifically, the importance propagated on by the former to the latter is proportional to the importance of w and inversely proportional to the number of outlinks from w. Superficially this appears to lend itself toward bootstrapping (defining PageRank in terms of PageRank - w^* is important because w is, and w is important because...). Page and Brin assure us that this is not the case, though: they present an iterative algorithm for calculating all PageRanks.

Alongside this explanation, Page and Brin provide *another intuitive basis*, namely the Random Surfer Model. Their presentation of the model is somewhat brief, but they do discuss some of the model's Markov Theoretical underbelly: *random walks on graphs*; a boredom factor E; and, a *standing probability distribution*.

In [4, 5], Kamvar and Haveliwala offer these same two explanations of PageRank. Quite unlike Page and Brin, though, their treatment of the first is rather brief whilst their treatment of the second quite detailed. They state the Ergodic Theorem for Markov chains; and, noting certain properties of the transition matrix, they explain how the theorem assures PageRank to be well-defined.

This bias is not just a matter of whim. Under close scrutiny it is clear: not just is the second explanation "a more rigorous analysis" than the first, but there is, in fact, little in the first explanation to assure us that what we want is the algorithm's limit. It is one thing talking about the relation between a page and its inlinks; it is quite another talking about a whole system of pages and the workings of an algorithm. Furthermore, there is little in the explanation to warrant inclusion of the PageRank personalisation vector. This leaves us with just one defensible explanation and a quandary. Though unacknowledged in the literature, the precise definitions provided by Page and Brin and Kamvar and Haveliwala are different – both at first glance and under close scrutiny. Actually, the definitions given by Page and Brin in [1, 2] differ also. The quandary, then: Page and Brin do not do enough to convince us, in their second explanation, that a Markov framework is directly applicable to their definitions; nor do they give any hint at how best to proceed in the absence of such a framework.

In the remains of Chapter 2, we shall be seeking to resolve this matter - to provide a defensible justification of Page and Brin's formulations.

2.1 Disparate Definitions

The first three definitions below are the definitions given in [1, 2, 5]. The fourth is peculiar to this paper.

Definition 1:
Let r_k be a row vector with n entries, where n is the size of the Web-graph.
Let p_n be a row vector with n entries, with $\|p_n\| = n$ and all p_n entries are ≥ 0.
Let $c \in \mathbf{R}$ be such that $0 < c < 1$.
Let d_i be the number of outlinks from a webpage i.
Let A be an $n \times n$ square matrix such that

$$A_{ij} = \begin{cases} 1/d & \text{if there is an outlink from webpage i to j} \\ 0 & \text{otherwise} \end{cases}$$

Then,
PageRank vector $\rho_1 = lim_{k\to\infty} r_k$ where $r_{k+1} = cr_kA + (1-c)p_n$.

Definition 2:
Let r_k be as above.
Let p_1 be a row vector with n entries, with $\|p_1\| = 1$, and all p_1 entries are ≥ 0.
Let c be as above.
Let A be as above.

Then,
PageRank vector $\rho_2 = lim_{k\to\infty} r_k$ where $r_{k+1} = r_kA + c(1-c)p_1$.

Definition 3:
Let r_k be as above.
Let $\|r_0\| = 1$.
Let p_1 be as above.
Let c be as above.
Let A be as above.
Let E be an $n \times n$ square matrix such that $E = \mathbf{1}p_1$ (i.e. $E_{ij} = p_{1i}$).
Let d be a row vector with n entries such that

$$d_i = \begin{cases} 0 & \text{if there are outlinks from webpage i} \\ 1/n & \text{otherwise} \end{cases}$$

Let $D = dp_1$ (i.e. $D_{ij} = d_i p_{1j}$)

Then,
PageRank vector $\rho_3 = lim_{k\to\infty} r_k$ where $r_{k+1} = r_k(c(A+D) + (1-c)E)$.

Definition 4:
Let r_k be as above.
Let p_1 be as above.
Let c be as above.
Let A be as above.

Then,
PageRank vector $\rho_4 = lim_{k\to\infty} r_k$ where $r_{k+1} = r_k A + (1-c)p_1$.

These four definitions are distinct from one another[1].

Consider the symmetric webgraph comprising two pages, each of which has an outlink to the other. Suppose that p_{1i} is constant for all i, and similarly for p_n – i.e. the boredom factor is uniform. Let $c = 0.5$ Then, the corresponding PageRanks are:

	ρ_1	ρ_2	ρ_3	ρ_4
Page 1	$\frac{1}{4}$	1	$\frac{1}{2}$	$\frac{1}{2}$
Page 2	$\frac{1}{4}$	1	$\frac{1}{2}$	$\frac{1}{2}$

This gives us that definitions 1 and 2 are distinct.

Let us now suppose that the situation is as above, except the first page has only a reflexive outlink to itself, and the second page has no outlinks.[2] Restricting attention to definitions 3 and 4, the corresponding PageRanks are:

	ρ_3	ρ_4
Page 1	$\frac{2}{3}$	$\frac{1}{2}$
Page 2	$\frac{1}{3}$	$\frac{1}{4}$

Accordingly, definitions 3 and 4 and distinct also.

[1]For i=1,..4, Defintion i is distinct if there exists some webgraph w for which ρ_i is distinct. Whether this is a fundamental difference or a superficial one is something that will be considered in due course.

[2]We shall consider later whether this ought to be deemed a valid webgraph.

2.2 Foundational Requirement

As provided by Kamvar and Haveliwala in [5, 4], the second explanation of Definition 3 places heavy demands on the reader's prior knowledge of Markov Theory. Without this knowldege, it is unclear that Definition 4 is sound. With this knowledge, the reader is left somewhat confused about Definitions 1 and 2. In the previous section we learnt that $\|\rho_1\|$ and $\|\rho_2\|$ need not equal one [3] – which is to say that the vectors need not give probability distributions. This runs contrary to the Markov framework. It introduces potential concerns over unique convergence; and it introduces grave doubt over the relevance of the limit vectors, if these exist.

In the next section we shall be surveying the requisite background theory. In the section after we will be building on this; we shall be trying to make more rigorous our concept of PageRank; and, having done so, we shall revisit definitions 1 through 4.

[3] For the example webgraph, $\|\rho_1\| = 2$ and $\|\rho_2\| = c$

2.3 Markov Theory

The aim in this section is to review the definitions and theorems which underlie Stationary Distributions.[4] Of particular interest are theorems 2.9 and 2.20. 2.9 is a standard result which was rediscovered whilst trying to justify PageRank – it was rediscovered and proved independently of any literature. 2.20 has a novel proof.

2.3.1 Markov Processess

Definition 2.1 *Let $(X_1, X_2, ...)$ be some random[5] process – that is to say, a sequence which gives the evolution in discrete time[6] of a random quantity (X_n being the quantity at time n). Let $S = \{s_1, ..., s_k\}$ be the corresponding finite state space – that is to say, the finite set of values which the random quantity can take. Let P be a $k \times k$ square matrix with elements $\{P_{i,j} : i, j \in \{1, ..., k\}\}$. We say $(X_1, X_2, ...)$ is a* ***(homogeneous[7]) Markov process with transition matrix*** *P, if for all n, all i,j $\in$ {1,...,k} and all $i_0, ..., i_{n-1} \in$ {1,...,k}*

$$\begin{aligned} \boldsymbol{P}(X_{n+1} = s_j | X_0 = s_{i1}, ..., X_{n-1} = s_{in-1}, X_n = s_{in}) &= \boldsymbol{P}(X_{n+1} = s_j | X_n = s_i) \\ &= P_{ij} \end{aligned}$$

From this we have immediately that a Markov process is a random process exhibiting two properties:

- *Memoryless (Markov) Property*
  The probability distribution of X_n given $(X_0, ..., Xn)$ depends only on X_n
- *(Time) Homogeneity Property*
  The probability distribution of X_n given that $X_n = s$, say, is independent of n.

In addition, we have immediately that transition matrices satisfy:

- $P_{ij} \geq 0$, $\forall i, j \in \{1, ..., k\}$
- $\sum_{j=1}^{k} P_{ij} = 1$, $\forall i \in \{1, ..., k\}$

[4] We follow the treatment in [12].
[5] Stochastic.
[6] We shall not concern outselves with continuous-time Markov processes.
[7] The term homogeneous is often dropped.

Definition 2.2 *Let $(X_0, X_1, ...)$ be a Markov process with finite state space $\{s_1, ..., s_k\}$ and transition matrix P. The* ***initial distribution*** *is defined as $\mu^{(0)} = (\boldsymbol{P}(X_0 = s_1),...,\boldsymbol{P}(X_0 = s_k))$ – it gives us how the Markov chain starts. We denote the distributions of the Markov chain at time n by $\mu^{(n)} = (\boldsymbol{P}(X_n = s_1),...,\boldsymbol{P}(X_n = s_k))$.*

From definition 2.1 we can derive the following:

Theorem 2.3 *Let $(X_0, X_1, ...)$ be a Markov process with finite state space $\{s_1, ..., s_k\}$, initial distribution $\mu^{(0)}$ and transition matrix P. Then, $\mu^{(n)} = \mu^{(0)}P^n$, where $\mu^{(n)}$ denotes the probability distribution (row vector) of the Markov process at time n.*

Proof:

We proceed by induction.
Base case (n=1)
$\forall j \in \{1, ..., k\}$

$$\begin{aligned}
\mu_j^{(1)} &= \mathbf{P}(X_1 = s_j) = \sum_{i=1}^{k} \mathbf{P}(X_0 = s_i, X_1 = s_j) \\
&= \sum_{i=1}^{k} \mathbf{P}(X_0 = s_i)\mathbf{P}(X_1 = s_j | X_0 = s_i) \\
&= \sum_{i=1}^{k} \mu_i^{(0)} P_{ij} = (\mu^{(0)}P)_j
\end{aligned}$$

where $(\mu^{(0)}P)_j$ is the j^{th} element of row vector $\mu^{(0)}P$.
Accordingly, $\mu^{(1)} = \mu^{(0)}P$.
This completes the base case proof.

Inductive step:
Suppose $n = m$, and suppose $\mu^{(m)} = \mu^{(0)}P^m$
For $n = m + 1$,

$$\begin{aligned}
\mu_j^{(m+1)} &= \mathbf{P}(X_{m+1} = s_j) = \sum_{i=1}^{k} \mathbf{P}(X_m = s_i, X_{m+1} = s_j) \\
&= \sum_{i=1}^{k} \mathbf{P}(X_m = s_i)\mathbf{P}(X_{m+1} = s_j | X_m = s_i) \\
&= \sum_{i=1}^{k} \mu_i^{(m)} P_{ij} = (\mu^{(m)}P)_j
\end{aligned}$$

Accordingly, $\mu^{(m+1)} = \mu^{(m)}P$.
But, $\mu^{(m)} = \mu^{(0)}P^m$, by induction hypothesis.
So, $\mu^{(m+1)} = \mu^{(m)}P = \mu^{(0)}P^mP = \mu^{(0)}P^{m+1}$.
This completes the inductive step, and the proof.

□

2.3.2 Irreducibility and Aperiodicity

Definition 2.4 *Let s_i and s_j be states. We say s_i **communicates** with s_j, and write $s_i \to s_j$, if $\exists n \in \boldsymbol{N}$ such that $\boldsymbol{P}(X_{m+n} = s_j | X_m = s_i) > 0$.*

This is to say, there is a positive probability of reaching s_j when starting from s_i.

Definition 2.5 *Let s_i and s_j be states. We say s_i **intercommunicates** with s_j, and write $s_i \leftrightarrow s_j$, if $s_i \to s_j$ and $s_j \to s_i$.*

Definition 2.6 *Let $(X_0, X_1...)$ be a Markov process with state space $S = \{s_1...s_k\}$ and transition matrix P. We say this chain is **irreducible** if $\forall s_i, s_j \in S$ we have $s_i \leftrightarrow s_j$. Or, equivalently, it is irreducible if $\forall s_i, s_j \in S$ $\exists n \in \boldsymbol{N}$ such that $(P^n)_{ij} > 0$. Otherwise we say the chain is **reducible**.*

Definition 2.7 *Let $(X_0, X_1...)$ be a Markov process with state space $S = \{s_1...s_k\}$. Let $gcd\{s_1, s_2, ...\}$ denote the greatest common divisor of $s_1, s_2, ...$ The **period** $d(s_i)$ of a state $s_i \in S$ is defined as $d(s_i) = gcd\{n \geq 1 : (P^n)_{ii} > 0\}$. If $d(s_i) = 1$, we say the state $d(s_i)$ is **aperiodic**.*

Definition 2.8 *We term a Markov process **aperiodic** if all its states are aperiodic. Otherwise, we term it **periodic**.*

Theorem 2.9 *Let $(X_0, X_1...)$ be an irreducible Markov chain with state space $S = \{s_1...s_k\}$ and transition matrix P. Let $P_{aa} > 0$ for some $a \in \{1, ..., k\}$. Then, $(X_0, X_1...)$ is aperiodic.*

Proof:

Let $i \in \{1, ..., k\}$.
As $(X_0, X_1...)$ is irreducible, $\exists r \in \mathbf{N}\backslash\{0\}$ such that $\mathbf{P}(X_r = a | X_0 = i) > 0$
Also, $\exists s \in \mathbf{N}\backslash\{0\}$ such that $\mathbf{P}(X_s = i | X_0 = a) > 0$
As $P_{aa} > 0$, we have that $\mathbf{P}(X_{n+1} = i | X_n = a) > 0$, $\forall n \in \mathbf{N}$.
And so,

$$\begin{aligned} \mathbf{P}(X_{r+1+s} = i | X_0 = i) &\geq \mathbf{P}(X_r = a, X_{r+1} = a, X_{r+1+s} = i | X_0 = i) \\ &= \mathbf{P}(X_r = a | X_0 = i)\mathbf{P}(X_{r+1} = a | X_r = a)\mathbf{P}(X_{r+1+s} = i | X_{r+1} = a) \\ &> 0 \end{aligned}$$

Furthermore,

$$\begin{aligned} \mathbf{P}(X_{r+2+s} = i | X_0 = i) &\geq \mathbf{P}(X_r = a, X_{r+1} = a, X_{r+1} = a, X_{r+2+s} = i | X_0 = i) \\ &= \mathbf{P}(X_r = a | X_0 = i)\mathbf{P}(X_{r+1} = a | X_r = a)\mathbf{P}(X_{r+2} = a | X_{r+1} = a) \\ &\quad \mathbf{P}(X_{r+2+s} = i | X_{r+2} = a) \\ &> 0 \end{aligned}$$

Now, the greatest common divisor of consecutive naturals is one.
Thus, s_i is aperiodic.
Thus, $(X_0, X_1...)$ is aperiodic.

□

Theorem 2.10 *Let $(X_0, X_1...)$ be an aperiodic Markov chain with state space $S = \{s_1...s_k\}$ and transition matrix P. Then, $\exists$ finite $N \in \mathbf{N}$ for which $(P^n)_{ii} > 0$, $\forall n \geq N$, $\forall i \in \{1,...,k\}$.*

Proof:

The proof presupposes a theorem from number theory[17]:

Let $A = \{a_1, a_2, ...\}$ be a set of naturals such that

1. *gcd(A) = 1;*
2. *A is closed under addition – i.e. if $a, a' \in A$, then $a + a' \in A$*

Then, $\exists$ finite $N \in \mathbf{N}$ such that $n \in A$, $\forall n \geq N$

Let $A_i = \{n \geq 1 : (P^n)_{ii} > 0\}$.
As $(X_0, X_1...)$ is aperiodic, $\gcd(A_i) = 1$.
Now, if $a, a' \in A_i$, $\mathbf{P}(X_a = s_i | X_0 = s_i) > 0$ and $\mathbf{P}(X_{a'} = s_i | X_0 = s_i) > 0$.
From the second of these, and time homogeneity, we have that $\mathbf{P}(X_{a'+a} = s_i | X_a = s_i) > 0$.
These inequalities give us the following:

$$\begin{aligned} \mathbf{P}(X_{a'+a} = s_i | X_0 = s_i) &\geq \mathbf{P}(X_a = s_i, X_{a'+a} = s_i | X_0 = s_i) \\ &= \mathbf{P}(X_a = s_i | X_0 = s_i)\mathbf{P}(X_{a'+a} = s_i | X_a = s_i) \\ &> 0. \end{aligned}$$

This implies that $a + a' \in A_i$.
This implies that A_i is closed under addition.
By the presupposed theorem, then, $\exists$ finite $N_i \in \mathbf{N}$ such that $(P^n)_{ii} > 0$, $\forall n \geq N$.
The theorem now follows with $N = max\{N_1, ..., N_k\}$.

Corollary 2.11 *Let $(X_0, X_1...)$ be an irreducible and aperiodic Markov chain with finite state space $S = \{s_1...s_k\}$ and transition matrix P. Then, $\exists$ finite $N \in \mathbf{N}$ for which $(P^n)_{ij} > 0$, $\forall n \geq N$ and $\forall i, j \in \{1,...,k\}$.*

Proof:
By aperiodicity, we may use theorem 2.9.
From this, $\exists$ finite $M \in \mathbf{N}$ for which $(P^n)_{ii} > 0$, $\forall n \geq N$, $\forall i \in \{1,...,k\}$.
Choose two states $s_i, sj \in S$.
By irreducibility, $\exists m_{ij}$ such that $(P^{m_{ij}})_{ij} > 0$.
Let $N_{ij} = M + m_{ij}$
Now, $\forall n \geq N_{ij}$,

$$\begin{aligned} \mathbf{P}(X_n = s_j | X_0 = s_i) &\geq \mathbf{P}(X_{n-m_{ij}} = s_i, X_m = s_j | X_0 = s_i) \\ &= \mathbf{P}(X_{n-m_{ij}} = s_i | X_0 = s_i)\mathbf{P}(X_n = s_j | X_{n-m_{ij}} = s_i) \\ &> 0. \end{aligned}$$

This is because:

$$\begin{aligned} \mathbf{P}(X_{n-m_{ij}} = s_i | X_0 = s_i) > 0 &: n - m_{ij} \geq N \\ \mathbf{P}(X_n = s_j | X_{n-m_{ij}} = s_i) > 0 &: \textit{given choice of } m_{ij} \end{aligned}$$

The corollary follows with $N = max\{N_{11}, N_{12}, ..., M_{1,k}, M_{2,1}, ..., M_{kk}\}$.

2.3.3 Stationary Distributions

Definition 2.12 *Let $(X_0, X_1...)$ be a Markov chain with finite state space $S = \{s_1...s_k\}$ and transition matrix P. We term a row vector $\pi(\pi_1, ..., \pi_k)$ a* ***stationary distribution*** *for the Markov chain, if it satisfies*

1. $\pi \geq 0, \forall i \in \{1, ..., k\}$
2. $\sum_{i=1}^{k} \pi_i = 1$
3. $\pi P = \pi$ – *i.e.* $\forall j \in \{1, ..., k\}, \sum_{i=1}^{k} \pi_i P_{ij} = \pi_j$

In this sub-section we prove the existence of stationary distributions. We prove also that irreducible, aperiodic Markov Chains converge uniquely to stationary distributions.

Definition 2.13 *Let $(X_0, X_1...)$ be a Markov chain with finite state space $S = \{s_1...s_k\}$ and transition matrix P. Let $X_0 = s_i$. We define the* ***hitting time*** *T_{ij} to be*

$$T_{ij} = \begin{cases} min\{n \geq 1 : X_n - s_j\} & \text{if the chain visits } s_j \\ \infty & \text{otherwise} \end{cases}$$

Definition 2.14 *Let $(X_0, X_1...)$ be a Markov chain with finite state space $S = \{s_1...s_k\}$ and transition matrix P. Let $X_0 = s_i$. We define the* ***mean hitting time*** *τ_{ij} to be $\boldsymbol{E}[T_{ij}]$ – where $\boldsymbol{E}[T_{ij}] = \sum_{k=1}^{\infty} k \ \boldsymbol{P}(X = k)$, or, equivalently, $\boldsymbol{E}[T_{ij}] = \sum_{k=1}^{\infty} \boldsymbol{P}(X \geq k)$. In the case of $i = j$, we term τ_{ii} the* ***the mean return time*** *for s_i.*

Lemma 2.15 *Let $(X_0, X_1...)$ be a Markov chain with finite state space $S = \{s_1...s_k\}$ and transition matrix P. Then, $\forall s_i, s_j \in S$, if $X_0 = s_i$, $\boldsymbol{P}(T_ij < \infty) = 1$. Also, the mean hitting time is finite.*

Proof
We prove these claims in turn.
By corollary 2.10, $\exists M < \infty$ such that $(P^M)_{ij} > 0$, $\forall i, j \in \{1, ..., k\}$.
Choose such an M.
Let $\alpha = min\{(P^M)_{ij} : i, j \in \{1, ..., k\}\}$.
Let $a, b \in \{1, ..., k\}$.
Let $X_0 = s_a$
Now, $\mathbf{P}(T_{ab} > M) \leq \mathbf{P}(X_M \neq s_b), for \begin{cases} \text{if } T_{ab} > M \text{then certainly} X_M \neq s_b; and \\ \text{if } T_{ab} < M \text{then it may be the case that} X_M \neq s_b. \end{cases}$
And, $\mathbf{P}(X_M \neq s_b) \leq 1 - \alpha$, as $\mathbf{P}(X_M = s_b) > \alpha$.
This gives $\mathbf{P}(T_{ab} > M) \leq 1 - \alpha$.
Further, $(T_{ab} \leq 2M | T_{ab} > M) > \alpha$, by definition of α, M and time homogeneity.
This gives $\mathbf{P}(T_{ab} > 2M | T_{ab} > M) \leq 1 - \alpha$,
So,

$$\begin{aligned} \mathbf{P}(T_{ab} > 2M) &= \mathbf{P}(T_{ab} > M)(T_{ab} > 2M | T_{ab} > M) \\ &\leq (1 - \alpha)(1 - \alpha) \\ &= (1 - \alpha)^2 \end{aligned}$$

By induction we have that, $\forall n \in \mathbf{N}\backslash\{0\}$.

$$\begin{aligned}\mathbf{P}(T_{ab} > nM) &= \mathbf{P}(T_{ab} > M)(T_{ab} > 2M | T_{ab} > M)...(T_{ab} > nM | T_{ab} > M) \\ &\leq (1-\alpha)^n \\ &\to 0 \text{ as n } \to \infty\end{aligned}$$

Hence, $\mathbf{P}(T_{ab} = \infty) = 0$, and the first part of the theorem is proved.

To prove the second claim, we proceed as follows.

$$\begin{aligned}\mathbf{E}[T_{ab}] &= \sum_{k=1}^{\infty} \mathbf{P}(T_{ab} \geq k) \\ &= \sum_{k=0}^{\infty} \mathbf{P}(T_{ab} > k) \\ &= \sum_{l=0}^{\infty} \sum_{k=lM}^{(l+1)M-1} \mathbf{P}(T_{ab} > k) \\ &\leq \sum_{l=0}^{\infty} \sum_{k=lM}^{(l+1)M-1} \mathbf{P}(T_{ab} > lM) \\ &= M \sum_{l=0}^{\infty} \mathbf{P}(T_{ab} > lM) \\ &\leq M \sum_{l=0}^{\infty} (1-\alpha)^l \text{ , by the first claim} \\ &= M \frac{1}{1-(1-\alpha)} \\ &= \frac{M}{\alpha} \\ &< \infty\end{aligned}$$

Theorem 2.16 *Let $(X_0, X_1...)$ be a Markov chain with finite state space $S = \{s_1...s_k\}$ and transition matrix P. Then, $(X_0, X_1...)$ has at least one stationary distribution.*

Proof:

Let $X_0 = s_1$

$\forall i \in \{1, ..., k\}$, let $\rho_i = \sum_{n=0}^{\infty} \mathbf{P}(X_n = s_i, T_{11} > n)$.

(So ρ is the expected number of visits to i expected before time T_{11}.)

We note:

$$\begin{aligned}\rho_i &= \sum_{n=0}^{\infty} \mathbf{P}(X_n = s_i, T_{11} > n) \\ &\leq \sum_{n=0}^{\infty} \mathbf{P}(T_{11} > n) \\ &= \sum_{n=1}^{\infty} \mathbf{P}(T_{11} \geq n)\end{aligned}$$

$$
\begin{aligned}
&= \mathbf{E}[T_{11}] \\
&= \tau_{11} \\
&< \infty, \text{ by the preceding lemma.}
\end{aligned}
$$

So, $\forall i \in \{1, ..., k\}$, ρ_i is finite.
With this, we propose as a stationary distribution $\pi = (\pi_1, ..., \pi_k) = (\frac{\rho_1}{\tau_{11}}, ..., \frac{\rho_k}{\tau_{kk}})$
We prove that the stationary distribution conditions 1, 2 and 3 are satisfied.
Condition 1:

Clearly, $\pi_i \geq 0, \forall i \in \{1, ..., k\}$

Condition 2:

We note the following:

$$
\begin{aligned}
\tau_{11} &= \mathbf{E}[T_{11}] \\
&= \sum_{n=1}^{\infty} \mathbf{P}(T_{11} \geq n) \\
&= \sum_{n=0}^{\infty} \mathbf{P}(T_{11} > n) \\
&= \sum_{n=0}^{\infty} \sum_{i=1}^{k} \mathbf{P}(X_n = s_i, T_{11} > n) \\
&= \sum_{i=1}^{k} \sum_{n=0}^{\infty} \mathbf{P}(X_n = s_i, T_{11} > n) \\
&= \sum_{i=1}^{k} \rho_i
\end{aligned}
$$

From this it follows that $\sum_{i=1}^{k} \pi_i = \sum_{i=1}^{k} \frac{\rho_i}{\tau_{11}} = 1$.

Condition 2:

We consider two cases.

Case 1: $j \neq 1$

$$
\begin{aligned}
\pi_j &= \frac{\rho_j}{\tau_{11}} \\
&= \frac{1}{\tau_{11}} \sum_{n=0}^{\infty} \mathbf{P}(X_n = s_j, T_{11} > n). \\
&= \frac{1}{\tau_{11}} \sum_{n=1}^{\infty} \mathbf{P}(X_n = s_j, T_{11} > n), \text{ as } j \neq 1. \\
&= \frac{1}{\tau_{11}} \sum_{n=1}^{\infty} \mathbf{P}(X_n = s_j, T_{11} > n-1), \text{ as } j \neq 1. \\
&= \frac{1}{\tau_{11}} \sum_{n=1}^{\infty} \sum_{i=1}^{k} \mathbf{P}(X_{n-1} = s_i, X_n = s_j, T_{11} > n-1). \\
&= \frac{1}{\tau_{11}} \sum_{n=1}^{\infty} \sum_{i=1}^{k} \mathbf{P}(X_{n-1} = s_i, T_{11} > n-1) \mathbf{P}(X_n = s_j | X_{n-1} = s_i)
\end{aligned}
$$

$$
\begin{aligned}
& \text{(As } (T_{11} > n-1) \text{ is independent of } X_n.) \\
= & \frac{1}{\tau_{11}} \sum_{n=1}^{\infty} \sum_{i=1}^{k} P_{ij} \mathbf{P}(X_{n-1} = s_i, T_{11} > n-1) \\
= & \frac{1}{\tau_{11}} \sum_{i=1}^{k} P_{ij} \sum_{n=1}^{\infty} \mathbf{P}(X_{n-1} = s_i, T_{11} > n-1) \\
= & \frac{1}{\tau_{11}} \sum_{i=1}^{k} P_{ij} \sum_{n=0}^{\infty} \mathbf{P}(X_n = s_i, T_{11} > n) \\
= & \frac{\sum_{i=1}^{k} \rho_i P_i j}{\tau_{11}} \\
= & \sum_{i=1}^{k} \pi_i P_{ij}
\end{aligned}
$$

Case 2: $j = 1$

$$
\begin{aligned}
\rho_1 &= 1, \text{ by definition of } \rho_i. \\
&= \mathbf{P}(T_{11} < \infty) \\
&= \sum_{n=1}^{\infty} \mathbf{P}(T_{11} = n) \\
&= \sum_{n=1}^{\infty} \sum_{i=1}^{k} \mathbf{P}(X_{n-1} = s_i, T_{11} = n) \\
&= \sum_{n=1}^{\infty} \sum_{i=1}^{k} \mathbf{P}(X_{n-1} = s_i, T_{11} > n-1) \mathbf{P}(X_n > s_1 | X_{n-1} = s_i) \\
&= \sum_{n=1}^{\infty} \sum_{i=1}^{k} P_{i1} \mathbf{P}(X_{n-1} = s_i, T_{11} > n-1) \\
&= \sum_{i=1}^{k} P_{i1} \sum_{n=1}^{\infty} \mathbf{P}(X_{n-1} = s_i, T_{11} > n-1) \\
&= \sum_{i=1}^{k} P_{i1} \sum_{n=0}^{\infty} \mathbf{P}(X_n = s_i, T_{11} > n) \\
&= \sum_{i=1}^{k} \rho_i P_{i1}
\end{aligned}
$$

From this, we have $\pi_1 = \frac{\rho_1}{\tau_{11}} = \sum_{i=1}^{k} \frac{\rho_i P_{i1}}{\tau_{11}} = \sum_{i=1}^{k} \pi_i P_{i1}$.

Condition 3 follows by cases 1 and 2.

Definition 2.17 *Let $v^{(1)} = (v_1^{(1)}, ..., v_k^{(1)})$ and $v^{(2)} = (v_1^{(2)}), ..., v_k^{(1)})$ be probability distributions on state space $S = \{s_1, ..., s_k\}$. Then,* ***total variation distance*** *between $v^{(1)}$ and $v^{(1)}$ is defined as $d_{TV}(v^{(1)}, v^{(2)}) = \frac{1}{2} \sum_{i=1}^{k} |v_i^{(1)} - v_i^{(2)}|$.*

Definition 2.18 *Let $v^{(1)} = (v_1^{(1)}, ..., v_k^{(1)})$ and $v^{(2)} = (v_1^{(2)}), ..., v_k^{(1)})$ be probability distributions on state space $S = \{s_1, ..., s_k\}$. Then, we say $v^{(n)}$* ***converges to v in total variation as*** *$v \to \infty$, and write $v^n \overset{TV}{\to} v$, if $lim_{n\to\infty} d_{TV}(v^{(n)}, v) = 0$.*

Definitions 2.17 and 2.18 allow us to make sense of vector convergence.

Theorem 2.19 *Let $(X_0, X_1...)$ be an irreducible, aperiodic Markov chain having finite state space $S = \{s_1...s_k\}$ and transition matrix P. Let $\mu^{(0)}$ be the initial distribution. Then, $\forall\pi$ such that π is a stationary distribution for P, $\mu^{(n)} \overset{TV}{\to} \pi$.*

Proof:
The proof is bipartite, and proceeds by way of a *coupling* argument.
Let $\mu^{(0)}$ be the initial distribution of $(X_0, X_1...)$.
Let $(X_0', X_1'...)$ be a second Markov chain, with the same state space and transition matrix.
Let $(X_0', X_1'...)$ have initial distribution π.
As π is a stationary distribution, this gives us that $\forall n \in\mathbf{N}$, X_n' has distribution π.
The aim now is to show: $\exists n \in\mathbf{N}$ such that $X_n = X_n'$ – i.e. the chains meet.
Let the first meeting time T be defined as $T = min\{n : X_n = X_n'\}$.
By corollary 2.11, $\exists M < \infty$ such that $(P^M)_{ij} > 0$, $\forall i, j \in \{1, ..., k\}$.
Fix M and set $\alpha = min\{(P^M)_{ij} : i, j \in \{1, ..., k\}\}$
We now proceed by induction to show that $\forall l \in\mathbf{N}\backslash\{0\}$, $\mathbf{P}(T > lM) \leq (1 - \alpha^2)^l$.
Base case: l=1

$$\begin{aligned}
\mathbf{P}(T \leq M) &\geq \mathbf{P}(X_M = X_M') \\
&\geq \mathbf{P}(X_M = s_1, X_M' = s_1) \\
&= \mathbf{P}(X_M = s_1)\mathbf{P}(X_M' = s_1) \\
&= (\sum_{i=1}^{k} \mathbf{P}(X_0 = s_i, X_M = s_1))(\sum_{i=1}^{k} \mathbf{P}(X_0' = s_i, X_M' = s_1)) \\
&= (\sum_{i=1}^{k} \mathbf{P}(X_0 = s_i)(X_M = s_1 | X_0 = x_i))(\sum_{i=1}^{k} \mathbf{P}(X_0' = s_i)(X_M' = s_1 | X_0' = x_i)) \\
&\geq (\alpha \sum_{i=1}^{k} \mathbf{P}(X_0 = s_i))(\alpha \sum_{i=1}^{k} \mathbf{P}(X_0' = s_i))
\end{aligned}$$

Hence, $\mathbf{P}(T > M) \leq (1 - \alpha^2)$, and the base case is proved.

Inductive step
Suppose the claim holds for $l = k \in\mathbf{N}\backslash\{0\}$.
By time homogeneity, $\mathbf{P}(X_{(k+1)M} = X'_{(k+1)M} | T > kM = s_1) > \alpha^2$.
From this, $\mathbf{P}(X_{(k+1)M} \neq X'_{(k+1)M} | T > kM) \leq (1 - \alpha^2)$.
And so,

$$\begin{aligned}
\mathbf{P}(T > (k+1)M) &= \mathbf{P}(T > kM)\mathbf{P}(T > (k+1)M | T > kM) \\
&\leq (1 - \alpha)^k \mathbf{P}(T > (k+1)M | T > kM) \\
&\leq (1 - \alpha)^k \mathbf{P}(X_{(k+1)M} \neq X'_{(k+1)M} | T > kM) \\
&\leq (1 - \alpha)^{k+1}
\end{aligned}$$

So, the claim follows by induction, and $lim_{n\to\infty}\ \mathbf{P}(T > n) = 0$.
Hence, the two chains meet with probability 1.
The next step in the proof is to construct a third Markov chain $(X"_0, X"_1...)$.
Let $(X"_0, X"_1...)$ have the same state space and transition matrix.
Let $X"_0 = X_0$, with initial distribution $\mu^{(0)}$,and $\forall n \in$, let $\mathbf{N}\backslash\{0\}$, $X"_n = \begin{cases} X_n + 1 & \text{if } X"_n \neq X_n \\ X'_n + 1 & \text{if } X"_n = X_n \end{cases}$
(This third chain is Markov because of the memoryless property)
As $X"_0$ has initial distribution $\mu^{(0)}$, $X"_n$ has initial distribution $\mu^{(n)}$, $\forall n \in\mathbf{N}$.
So, $\forall i \in$ {1,...,k}, $\forall n \in\mathbf{N}$,

$$\begin{aligned} \mu_i^{(n)} - \pi_i &= \mathbf{P}(X"_n = s_i) - \mathbf{P}(X'_n = s_i) \\ &\leq \mathbf{P}(X"_n = s_i, X'_n \neq s_i) \\ &\leq \mathbf{P}(X"_n \neq X'_n) \\ &= \mathbf{P}(T > n) \\ &\quad \to 0 \text{ as } n \to \infty \text{ by the first part of the proof.} \end{aligned}$$

By changing the roles of $X"_n$ and X_n), we also have that $pi_i - \mu_i^{(n)} \leq \mathbf{P}(T > n) \to 0$ as $n \to \infty$.
Hence, $lim_{n\to\infty}|\mu_i^{(n)} - \pi_i| = 0$
From which $lim_{n\to\infty} d_{TV}(\mu^{(n)}, \pi) = lim_{n\to\infty} \frac{1}{2}\sum_{i=1}^{k} |\mu_i^{(n)} - \pi_i| = 0$.

Theorem 2.20 *Let $(X_0, X_1...)$ be an irreducible, aperiodic Markov chain. $(X_0, X_1...)$ has exactly one stationary distribution.*

Proof:
Let π and π' be any two stationary distributions.
Let $\mu^{(n)}$ be the chain's distribution at time n.
Now,

$$\begin{aligned} d_TV(\pi, \pi') &= lim_{n\to\infty} d_TV(\pi, \pi') \\ &= lim_{n\to\infty} \frac{1}{2}\sum_{i=1}^{k} |\pi_i - \pi'_i| \\ &= lim_{n\to\infty} \frac{1}{2}\sum_{i=1}^{k} |\pi_i - \mu_i^{(n)} + \mu_i^{(n)} - \pi'_i| \\ &\leq lim_{n\to\infty} \frac{1}{2}\sum_{i=1}^{k} |\pi_i - \mu_i^{(n)}| + |\mu_i^{(n)} - \pi'_i| \\ &= lim_{n\to\infty} \frac{1}{2}\sum_{i=1}^{k} |\pi_i - \mu_i^{(n)}| + lim_{n\to\infty} \frac{1}{2}\sum_{i=1}^{k} |\mu_i^{(n)} - \pi'_i| \\ &= lim_{n\to\infty} d_{TV}(\mu^{(n)}, \pi) + lim_{n\to\infty} d_{TV}(\mu^{(n)}, \pi') \\ &= 0 \end{aligned}$$

Hence, $\pi = \pi'$.

2.3.4 Summary

This section presented two key ideas that will be drawn on in section 2.4: the probability distribution of an irreducible, aperiodic Markov chain converges to its unique stationary distribution (equilibrium); and $P_{ii} \neq 0$ is a sufficient condition for an irreducible Markov chain's being aperiodic. The first of these, a version of the Ergodic Theorem for Markov Chains, follows from [12] (though this paper presents an alternative proof to theorem 2.20). The second of the ideas was one conjured up and proved independently of any literature - it was born of the investigation into Markov applicability to PageRank.

2.4 Random Surfer Model

This section opens with the key intuition underlying PageRank; it derives yet another PageRank definition from this intuition, and it shows this definition to be well-defined.

With this definition as guide, attention is then turned to the definitions in section 2.1. Definition 4 is shown to be correct; as is definition 3. And, Definitions 1 and 2 are put into into context – they are shown to be fundamentally of the same type as definition 4. Whilst considering these definitions, a potential justification is provided for Page and Brin's cryptic call to remove "dangling links" and cul de sac[8] Webpages in [2]. Also, Kamvar and Haveliwala's irreducibility restriction on PageRank transition matrices [5] is shown to be unnecessary – it is shown that, contrary to the claims in the literature, personalisation vectors can take on zero entries. This irreducibility relaxation has at heart the novel theorem 2.26.

2.4.1 Intuition

Suppose some Web-browser, with no back button and no text box for entering URLs, has, instead, a built-in *randomised search engine*, by which any of the Web's pages is randomly retrieved at the click of a button. Suppose also that some rather peculiar Websurfer makes use of this browser. This surfer randomly follows hyperlink from page to page, or, when overcome with sporadic bouts of boredom, she clicks on the browser's search-engine button. The forms the basis of the key PageRank intuition: the Random Surfer gives an unbiased representation of how we proceed on the Web - the randomised search engine accounting for our *favorites lists* and the urls we commit to memory. The idea is that if we define PageRank to be the equilibrium probability of this surfer being at a certain page, then this is a good measure of the webpage's import.

This is by no means the definitive word on webpage import. The Random Surfer Model has no place for a back button, and this is open to question. Most likely, Page and Brin excluded it because it contravenes the memoryless property of Markov chains, but one needs wonder: could not some other theoretical underpinning have been provided – or could the Markov underpinning not have been adapted? Unfortunately, this and other questions (about content-based optimisation[18], for example) are beyond the scope of this paper. Instead, we shall take the Random Surfer Model as a given, and turn attention to formal definition.

2.4.2 Definition 5

We first provide a formal definition of a Webgraph. Using this, we then define the random following of hyperlinks and the random retrieval of webpages via the randomised search engine. We then put these together to define the Random Surfer's activities. As culmination, we define PageRank 5.

Definition 2.21 *Let W be a set of webpages. Let $H \subseteq W \times W$ be a set of hyperlinks on W. Then, if $\forall w \in W, \exists w' \in W$ such that $w, w' \in W$, then $G = (W, H)$, is a* ***Webgraph****.*

Definition 2.22 *Let $G = (W, H)$ be a Webgraph. We term $(X_0, X_1, ...)$ a* ***Random-Hyperlink-Click*** *chain on G if $(X_0, X_1, ...)$ is a Markov chain with state space W and transition matrix A,*

[8] Webpage with no outgoing links.

where

$$A_{ij} = \begin{cases} 1/d_i & \text{if } i,j \in H, \text{ and } d_i \text{ is the number of distinct } k \text{ such that } (i,k) \in H \\ 0 & \text{otherwise} \end{cases}$$

Note: clearly, for every Webgraph G, there is a unique Random-Hyperlink-Click matrix A.

Definition 2.23 *Let $G = (W, H)$ be a Webgraph. We term function p a (G)**Randomised-Search-Engine** function if $p : W \to [0, 1]$ and $p(W) := \sum_{w \in W} p(w) = 1$ – i.e. e maps W to a probability distribution such that $\forall w \in W, p(w) \neq 0$.*

Definition 2.24 *Let $G = (W, H)$ be a Webgraph. Let p be a (G)Randomised-Search-Engine function. Then, $(X_0, X_1, ...)$ is a (G,p) **Random Surfer** Markov chain if it has state space W and transition matrix M, where $M_{ij} = cA_{ij} + (1 - c)p(j)$.*

Definition 2.25 *Let $G = (W, H)$ be a Webgraph. Let p be a Randomised-Search-Engine function on G. Let $(X_0, X_1, ...)$ be a (G,p) Random Surfer Markov chain. Then, we define the corresponding (G,p) **PageRank** vector ρ_5 to be the stationary distribution of $(X_0, X_1, ...)$.*

We recall that Markov chains converge to their unique stationary distributions. So an alternative formulation of the above is: $\rho_5 = lim_{n\to\infty} r_n$, where $r_{n+1} = r_n M$. Or, equivalently: $\rho_5 = lim_{n\to\infty} r_0 M^n$, for initial distribution r_0 (by theorem 2.3).

2.4.3 Well-Defined

To show that, given G and p, ρ_5 is well-defined, we first note the following:

1. A is a transition matrix;
2. M is a transition matrix;

Proof of 1:
We note: $n \times n$ matrix A is a transition matrix if $\forall i, \sum_{j=1}^{n} Aij = 1$.
By definition, $A_{ij} = \begin{cases} 1/d_i & \text{if } i,j \in H, \text{ and } d_i \text{ is the number of distinct k such that } (i,k) \in H \\ 0 & \text{otherwise} \end{cases}$
From this, $\forall i, \sum_{j=1}^{n} Aij = \frac{d_i}{d_i} = 1$; so, A is indeed a transition matrix.

Proof of 2:
As above, the $n \times n$ matrix M is a transition matrix if $\forall i, \sum_{j=1}^{n} M_{ij} = 1$.
By definition, $M_{ij} = cM_{ij} + (1 - c)p(j)$, and $\sum_{j=0}^{n} = 1$
From this, and proof 1, we have that

$$\begin{aligned} \sum_{j=1}^{n} M_{ij} &= \sum_{j=1}^{n} cA_{ij} + (1-c)p(j) \\ &= c\sum_{j=1}^{n} A_{ij} + (1-c)\sum_{j=1}^{n} p(j) \end{aligned}$$

$$
\begin{aligned}
&= c + (1-c) \\
&= 1
\end{aligned}
$$

So, M is a transition matrix also.

We also note the following interesting *new* theorem:

Theorem 2.26 *Let $G = (W, H)$ be a Webgraph. Let p be a (G)Randomised-Search-Engine function. Let $(X_0, X_1, ...)$ be a (G,p) Random Surfer Markov chain. Then, the PageRank of a page $w \in W$, namely $\rho_5(w)$, is zero if and only if there exists no $w' \in W$ such that $w' \to w$ and $p(w') > 0$.*

Proof:
$\Rightarrow$
We first show that $\forall k \in \mathbf{N}\backslash\{0\}$, $\forall w \in W$, $r_k w \geq (1-c)p(w)$
Suppose $w \in W$ and the size of W is n
Then,

$$
\begin{aligned}
\forall k \in \mathbf{N}, r_{k+1}(w) &= \sum_{w' \in W}^{n} r_k(w) M_{w'w} \\
&= \sum_{w' \in W}^{n} r_k(w)((cA_{w'w} + (1-c)p(w)), \text{ by definition of M} \\
&= \sum_{w' \in W}^{n} r_k(w) cA_{w'w} + \sum_{w' \in W}^{n} r_k(w')(1-c)p(w) \\
&= \sum_{w' \in W}^{n} r_k(w) cA_{w'w} + (1-c)p(w), \text{ as } \|r_k\|_1 = 1 \\
&\geq (1-c)p(w)
\end{aligned}
$$

Now, let $w* \in W$ be such that for some $w_{>0} \in W\backslash\{\text{w*}\}$, $w_{>0} \to w*$ and $p(w_{>0}) > 0$.
(The case where $w* = w_{>0}$ follows trivially from the proof above.)
By definition of $w_{>0} \to w*$, $\exists N \in \mathbf{N}\backslash\{0\}$ such that $M^N_{w_{>0}w*} > 0$.
This gives us

$$
\begin{aligned}
\forall k > N, r_{k+1}(w*) &= \sum_{w \in W}^{n} r_k(w) M_{ww*} \\
&= \sum_{w \in W}^{n} r_l(w) M^N_{ww*}, \text{ where } l = k - N, \text{ by theorem 2.3} \\
&\geq r_l(w_{>0}) M^N_{ww*} \\
&> 0
\end{aligned}
$$

Hence, $\forall w \in W$, if $\exists w' \in W$ such that $w' \to w$ and $p(w') > 0$, then $\rho_5(w) > 0$.

$\Leftarrow$
Let $w* \in W$.
Let $X = \{w \in W : w \text{ does } not \text{ communicate with } w*\}$
i.e. $X = \{w \in W : \forall n, A^n_{ww*} = 0\}$(by definition
Suppose that $\forall w \in W \backslash X$, p(w) = 0.
Now, by simple matrix algebra, it can be shown that:

$$\begin{aligned} \forall n \in \mathbf{N}, r_{n+1} &= r_n M \\ &= r_n(cA + E) \\ &= r_n cA + p \end{aligned}$$

Furthermore,

$$\begin{aligned} r_1 &= r_0 cA + (1-c)p \\ r_2 &= r_0 c^2 A^2 + pc(1-c)A + p(1-c) \\ r_3 &= r_0 c^3 A^3 + pc^2(1-c)A^2 + pc(1-c)A + p(1-c) \\ r_n &= r_0 c^n A^n + (1-c) * \sum_{i=0}^{n-1} pc^i A^i \end{aligned}$$

Now, consider $r_n(w*)$
Certainly $r_0 c^n A^n \to 0$ as $n \to \infty$, because $c < 0$ and $\|r_0 A^n\|_1 = \|r_n\|_1 = 1$.
So, the $w*^{th}$ entry of $r_0 c^n A^n$ also tends to 0 and n tends to infinity.
Also, the $w*^{th}$ entry of pA^k is $\sum_{w \in W} p(w) A^k_{ww*} = 0$.
This is because, by supposition, $p(w) = 0$ when $\exists k$ such that $A^k_{ww*} \neq 0$.
So the $w*^{th}$ entry of $(1-c) * \sum_{i=0}^{n-1} pc^i A^i$ also tends to 0 and n tends to infinity.
And, from this $\rho_5(w*) = 0$.

Let $Y = \{w \in W : \forall w' \in W$ such that $w' \to w, p(w') = 0\}$
Let M' be the equal to M when all rows in M with indices in Y are removed.
Now, M' would not be a transition matrix only if some column entry $M_{ww'} \neq 0$ was removed.
But, a row w remains only if there is some w" such that $p(w") \neq 0$ and $w" \to w$
And a column w' is removed only if there is no w" such that $p(w") \neq 0$ and $w" \to w$.
By transitivity of communication, it follows that M' is a transition matrix.
Given this, the aim is to show that Markov chains with M' are irreducible and aperiodic.
By theorems 2.19, and 2.20, it will then follow that $\rho_5(w)$ is well-defined for $w \in W \backslash Y$.
From which, by theorem 2.26, we will have that $\rho_5(w)$ is well defined on the whole of W.
So, let us consider:
Let $w, w' \in W \backslash Y$.
By definition of Y, $\exists w* \in W \backslash Y$ such that $e(w*) \neq 0$ and $w* \to w'$ (2)
Since $p(w*) \neq 0$, we have, by definition of M, that $w \to w*$.
By transitivity of communication, $w \to w'$.
This implies that Markov chains for M' are irreducible.
Also, by (2), $M'_{w*w*} \neq 0$.
By theorem 2.3, this implies that Markov chains with M' are aperiodic.
The result follows: ρ_5 is well-defined.

2.4.4 Definition 4

With ρ_5 well-defined, the aim now is to show that, given G and e, $\rho_4 = \rho_5$.
We first prove: if $r_{n+1} := r_n cA + (1-c)e$ and $r_0 = 0$, then $\forall n \in \mathbf{N}$, $\|r_n\|_1 = 1$
We prove this by induction.
Base case:
This is trivially true.
Inductive step:
Suppose the claim holds for $n = k$
We that that $\|r_k A\|_1 = 1$ – as A is a transition matrix.
This gives us that $\forall c \in (0,1)$, $\|r_k cA\|_1 = c$.
Now, $\|e\| = 1$ – by definition
So,$\forall c \in (0,1)$, $r_{k+1} = \|r_k cA + (1-c)p\|_1 = 1$.
The claim follows by induction.

With this, we get the following

$$\begin{aligned} r_n cA + (1-c)p &= r_n cA + \frac{(r_n)}{\|r_n\|_1}(1-c)E, \text{where } E_{ij} = p_j \\ &= r_n cA + (r_n)(1-c)E, \text{ by the inductive proof} \\ &= r_n(cA + (1-c)E) \\ &= r_n M, \end{aligned}$$

We recall: $\rho_4 = lim_{n\to\infty} r_n$ where $r_{n+1} = r_n cA + (1-c)p$. (1)
And so, $\rho_4 = lim_{n\to\infty} r_n$ $r_{n+1} = r_n M$.
Hence, $\rho_4 = \rho_5$, as required.

2.4.5 Cul de Sac Pages

The astute reader will have noted: 1 is true if and only if every webpage has an outlink (possibly a reflexive outlink – i.e. to itself). This accords with defintion 2.21 for Webgraphs. The resultant is: $\rho_5 = \rho_4$ if and only if every webpage has an outlink. This has important implications:

Suppose there is some cul de sac webpage (as there indeed is).
Then, ρ_4's A is no longer a transition matrix – $\exists i$ such that $\sum_j^n Aij = 0$.
From which: with an initial distribution r_0, r_n may not be a probability distribution.[9]
Indeed, the whole Markov framework collapses.
What is worse: it is difficult to see how even an indirect Markov justification could be provided.

It is most plausibly for this reason that Page and Brin write in [2]: "We simply remove [the cul de sac pages and corresponding dangling links]." Their supposed justification: "[Cul de sac pages] effect the model because it is not clear where their weight should be distibuted" looks suspiciously like a thinly veiled attempt to disguise an *ad hoc* amendment to their definition - an attempt to get their definition to fit the theory. A detailed investigation is unfortunately beyond the scope of

[9] Reconsider the example in section 2.1.

this paper. But, we ought to note the implications.

Page and Brin's idea is that we remove cul de sac pages and dangling links, thereby constructing a complete Webgaph. We then calculate ρ_4 (which is well-defined). Having done so, we re-introduce the cul de sac pages, and propagate to these the already calculated PageRanks of their inlinks. There are three issues to consider here. If we proceed in this fashion then the sum of the PageRanks will not add up to one. This is not of too great a concern though, because what we are really after is *relative* PageRank between webpages – not some absolute measure the importance of a webpage. We do not need to know equilibrium probability of a random surfer's being a particular page. What we require is ability to distinguish between the import of two webpages – equilibrium probabilities are a means to this end. A more difficult issue to wrestle with is that ρ_4 is theoretically sound for only those webpages which have outlinks. If we recall the first intuitive justification provided in section 2.2, then arguably we have a strong enough theory-intuition cocktail to drown out objections to ρ_4. Again, this is unfortunately beyond the scope of this paper. The third concern is practicality. Suppose we have a set of webpages $\{a, b, c, d, e, ..., z\}$ such that a links only to b, and b links only to c, and so on, until z, which is a cul de sac page. Now, if we were to remove all cul de sac pages from this set, then this would recursively result in our removing all pages – for once z is removed, y becomes a cul de sac page, etc.. This gives reason to doubt the practicality of removing cul de sac pages, but it does not rule out the possibility.

2.4.6 Definition 3

If there are no cul de sac pages, then $\rho_3 = \rho_5$ – as ρ_3's matrix $D = \mathbf{0}$. Also, if there are cul de sac pages, then $P + D$ is a transition matrix – by the proof of ρ_5 and the definition of D. From this we can show, in a manner analogous to the remainder of ρ_5's proof, both that A is a transition matrix and that any Markov chain with transition matrix A is aperiodic and irreducible. Accordingly, ρ_3 is theoretically well-defined with or without cul de sac pages.

But, this is not to say immediately that ρ_3 is an improvement on ρ_5. There is still the small matter of matrix D to address. The literature [5] provides little motivation of D other than *the theory demands it*, and this certainly does not suffice. Few would, however, argue with the following: one is more likely to skip to some random page on the web, perhaps a favorite, if faced with a cul de sac page. And, this idea does seem readily extensible to a strong, intuitive defense of D.

2.4.7 Definitions 1 and 2

As seen in section 2.1, ρ_1 and ρ_2 do not necessarily give probability distributions. Accordingly, Markov theory cannot be used directly as a theoretical justification. This paper proposes, however, that an indirect justification can be provided. If we recall that it is *relative* and not absolute PageRank that is requisite, then all we need do to justify ρ_1 and ρ_2 is to prove the following:

- if ρ_1 and ρ_2 converge, then they converge to scalar multiples of ρ_5;
- ρ_1 and ρ_2 both converge.

Preamble
Let Q be some $n \times n$ matrix with $Q_{ij} \geq 0$, $\forall ij$.
Let Q_{i-} denote the row vector equivalent to row i of matrix Q.

Let $\|Q_{i-}\|_1 = \alpha \in \mathbf{R}$, $\forall i$.
Let v be any vector in $\mathbf{R}^n$.
By definition, $vQ = v_1Q_{1-} + v_2Q_{2-} + ...v_n * Q_{n-}$
This gives us, by the triangle inequality:

$$\begin{aligned} \|vQ\|_1 &\leq |v_1|\|Q_{1-}\|_1 + |v_2|\|Q_{2-}\|_1 + ...|v_n|\|Q_{n-}\|_1 \\ &= \alpha(|v_1| + ... + |v_n|) \\ &= \alpha\|v\|_1 \end{aligned}$$

Now, let λ be any eigenvalue of Q.
Let $v^{(\lambda)}$ be the corresponding unit left eigenvector – $\|v^{(\lambda)}\|_1 = 1$.
Then, by the preceding,
$|\lambda| = |\lambda|\|v^{(\lambda)}\|_1 = \|v^{(\lambda)}\lambda\|_1 = \|v^{(\lambda)}Q\|_1 \leq \alpha\|v^{(\lambda)}\|_1 = \alpha$

Proof of Claim 1
By definition of stationary distributions, $\rho_4 = \rho_4 cA + (1-c)p$
This implies that $(I - cA)\rho_4 = (1-c)p$.
Let λ_{max} be the maximum eigenvalue (spectral radius) of cA.
Then, by the preamble, $\lambda_{max} \leq c < 1$.
From which, 1 is not an eigenvalue of cA.
So, $(I - cA)^{-1}$ exists.
And, accordingly, $\rho_4 = (1-c)p(I - cA)^{-1}$.
Let $r*_{n+1} = r_n cA + (1-c)sp$, where $s \in \mathbf{R}$.
Suppose $\rho_6 = lim_{n\to\infty} r*_n$ exists.
Then, certainly, $\rho_6 = \rho_6 cA + (1-c)sp$.
This implies that $(I - cA)\rho_6 = (1-c)sp$.
And, this implies that $\rho_6 = (1-c)sp(I - cA)^{-1}$.
Which is to say that, if ρ_6 exists, then it is a scalar multiple, s, of $\rho_4 = \rho_5$.
Hence, if ρ_1 and ρ_2 exist, then they are scalar multiples of $\rho_4 = \rho_5$.

Proof of claim 2:
We first show a matrix analogue of the geometric series result: $\sum_{i=0}^{\infty}(cA)^i = (I - cA)^{-1}$
Let (1) $S_n = \sum_{i=0}^{N}(cA)^i = I + (cA) + (cA)^2 + ... + (cA)^n$
Let (2) $(cA)S_n = (cA) + (cA)^2 + (cA)^3 + ... + (cA)^n + (cA)^{n+1}$.
Subtract (2) from (1) to get (3): $(I - cA)S_n = I - (cA)^{n+1}$.
Rearrangement gives: (4) $S_n = (I - cA^{n+1})(I - cA)^{-1}$.
We note: $(cA)^n$ clearly tends to $\mathbf{0}$ as $n \to \infty$, as $\forall ij, A_{ij}$ is bounded.
Hence, $lim_{n\to\infty} S_n = (I - cA)^{-1}$
We now turn attention to the crux of the matter.
Again, let $r *_n +1 = r_n cA + (1-c)sp$, where $s \in \mathbf{R}$.
Similarly to the proof of 2.26, it can be shown that $r*_n = r *_0 c^n A^n + (1-c)s\sum_{i=0}^{n-1} pc^i A^i$.
In proof 2.26, we also saw that $lim_{n\to\infty} r *_0 c^n A^n = 0$.
And, by the above geometric series analogue, $lim_{n\to\infty} \sum_{i=0}^{n-1} pc^i A^i = p(I - cA)^{-1}$
This gives us that $\rho_6 = lim_{n\to\infty} r*_n = (1-c)p(I - cA)^{-1}$ exists.
Hence, ρ_1 and ρ_2 exist also.

2.4.8 Irreducibility and the Personalisation Vector

The astute reader will have also noticed that definition 1 through 4 permit zero entries in their respective personalisation vectors, p_1 or p_n[10]. This is contrary to the literature. In [5], it is made explicit that the PageRank transition matrix needs be irreducible, and that the personalisation vector ensures this. This is most plausibly so, only if the personalisation vector entries are necessarily greater than zero[11]. But, this restriction rests uneasily with our intuitions: it does seem reasonable to suggest that there are some pages which some given surfer, s, would never visit unless s found a chain of hyperlinks to these. This cannot be permitted by the irreducibility requirement. Thankfully, this requirement was shown here to be too strong. Personalisation vectors can take on zero entries.

2.4.9 Summary

In this chapter we have seen that, if we assume there to be no cul de sac pages, then there is only one notion of PageRank – the one norm of the personalisation vector is of no imporance. Definitional differences only arise in the treatment of cul de sac pages. There are two possibilities: follow [5] and introduce a compensating matrix D; or follow [1] – remove cul de sac pages, calculate PageRank, and then, once this is done, propagate PageRank on to cul de sac pages.

[10] They are termed personalisation vectors because they are personalisable to a surfer's favorites and/or memorised urls

[11] To permit zero entries so long as they do not impinge on irreducibility would be to seriously fall foul of an *ad hoc* charge.

Chapter 3

Distributed Algorithms

Chapter 2 gave us two distinct species of PageRank calculation algorithm – one which incorporates a cul-de-sac compensating matrix, and another which proceeds by removing cul de sacs. The first does not lend itself to distribution: the cul-de-sac compensating matrix involves division by the number of pages in the system, and it is difficult to see how this figure could easily be arrived at within a distributed setting. In this chapter we shall consider what it means to distribute the the second method PageRank calculation; we shall review a Linear Algebraic theorem which permits calculation distribution; and we shall consider three methods of distribution.

3.1 Chaotic Relaxation

Consider the equation $X_{s+1} = X_s B + D$, where B is an $n \times n$ square matrix and D and X_s, for all s, are suitable vectors. Let us explicate what it means to distribute this calculation in a way which does not demand synchronous updating of the entries of X_s. Let $u : \mathbf{N} \rightarrow \mathbf{N}$ define an update function, such that, for each natural s, the component to be updated at step s is given by $u(s)$. Let $d : \mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$ define a delay function, such that, for each step s the value of the m^{th} component is the value d(s,m) steps back. Then, asynchronous updating, or *Chaotic Relaxation,* is defined by:

$$x_i^{s+1} = \begin{cases} \sum_{m=1}^{n} b_{ij} x_j^{s-d(s,j)} + d_i & if \quad i = u(s) \\ x_i^s & if \quad i \neq u(s) \end{cases}$$

Now, arguably, asynchronous updating is interesting in its own right. Our principal concern, however, is the limit of the original formulation, and the hope is that chaotic relaxation could yield such. Let the update and delay functions satisfy the following conditions:

1. The update function u maps to $1 \leq k \leq n$ infinitely often.
2. The delay function is bounded – i.e. $\exists D \in \mathbf{N}$ such that $\forall s, j \in \mathbf{N}$, $0 \leq d(s,j) < s \leq D$.

Then, from [10], we get the following exciting theorem:

Theorem 3.1 *Let x^i is defined as above, and let conditions 1 and 2 be satisfied. The Chazan-Miranker theorem states that if $\exists v$ such that $v_k \geq 0, \forall k$, and if $\exists \alpha$ such that both $0 \leq \alpha < 1$ and $|B|v \leq \alpha v$, then x^i converges to x as i tends to ∞, where $x = Bx + d$.*

Proof

We first note the following:

Let $y_i^{s+1} = \begin{cases} \sum_{m=1}^{n} b_{ij} y_j^{s-d(s,j)} & if \quad i = u(s) \\ y_i^s & if \quad i \neq u(s) \end{cases}$.

Let $x = Bx + d$.

Let $x^i = y^i + x$.

Then, $x_i^{s+1} = \begin{cases} \sum_{m=1}^{n} b_{ij} x_j^{s-d(s,j} + d_i & if \quad i = u(s) \\ x_i^s & if \quad i \neq u(s) \end{cases}$.

So, without loss of generality, we may assume d=$\mathbf{0}$.

Accordingly, let $x_i^{s+1} = \begin{cases} \sum_{j=1}^{n} b_{ij} x_j^{s-d(s,j)} & if \quad i = u(s) \\ x_i^s & if \quad i \neq u(s) \end{cases}$.

Suppose that $\exists v$ such that $v_k \geq 0, \forall k$, and $\exists \alpha$ such that $0 \leq \alpha < 1$ and $|B|v \leq \alpha v$.

We aim to show: $x^i \to 0$ as $i \to \infty$.

Step 1: Show that if $|z| \leq cx$, for some constant c, then $|Bz| \leq \alpha cx$.

Suppose $|z| \leq cx$

Then, $|Bz| = |B||z| \leq |B|cx \leq \alpha cx$.

Step 2: Show that if $|x^{s^*+1}|, ..., |x^{s^*+D}| \leq cx$, for some constant c and for delay bound D, then $\forall s \geq s^* + D$, $|x^s| \leq cx$.

We proceed by induction

Suppose $|x^{s^*+1}|, ..., |x^{s^*+D}| \leq cx$, for some constant c.

Let S be some natural such that $S \geq s^* + D$.

Suppose $|x^s| \leq cx$, $\forall s$ such that $S \geq s \geq s^* + D$.

Consider $|X^{S+1}|$.

By definition, $\forall i \neq u(s+1)$, $|x_i^{S+1}| = |x_i^S| \leq cx_i$.

Also, $\forall i = u(s+1)$ $|x_i^{S+1}| = \sum_{j=1}^{n} b_{ij} x_j^{S-d(S,j)} \leq (Bcx)_i \leq \alpha cx_i$ (by supposition and step 1).

Hence, $|X^{S+1}| \leq cx$.

The result follows by induction.

Step 3: Show that $|x^s| \to 0$ as s$\to \infty$.

Suppose $|x^{s^*+1}|, ..., |x^{s^*+D}| \leq cx$, for some constant c.

Then, by step 2, $|x^s| \leq cx$, $\forall s \geq s^*$. (1)

Let s_i be a natural such that $i = u(s_i)$ and $s_i > s^* + D$.

Then, $\forall s > s_i$, $|x_i^s| = \sum_{j=1}^{n} b_{ij} x_j^{S-d(S,j)} \leq (Bcx)_i \leq \alpha cx_i$

Let s_{n_1} be a natural such that $\forall j \in \{1, .., n\}$, $\exists s_j = u(s_j)$ and $s_n > s_j > s^* + D$.

Such an s_{n_1} exists by condition 1.

Then, $|x^{s_{n_1}}| \leq \alpha cx$, by (1).

From which, by argument as in step 2, $|x^s| \leq \alpha cx$, $\forall s$ such that $s_{n_1} + D \geq s \geq s_{n_1} + 1$.

Repeating this argument, we can find s_{n_k} such that $s_{n_k} > s_{n_2} > s_{n_1}$ and

$|x^s| \leq \alpha^k cx$, $\forall s (s_{n_k} + D \geq s \geq s_{n_k} + 1)$.

Now, clearly, as $k \to \infty$, $\alpha^k cx \to 0$.

Hence, as $s \to \infty$, $|x^s| \to 0$.

The theorem follows.

3.2 Suitability of PageRank

In the definitions in section 2.1, the matrix A is a Markov transition matrix.
So, by theorem 2.16, there exists positive vector v such that v = vA.
Hence, for c, which by definition is such that $0 < c < 1$, we have that $cv = vcA$.
Hence, theorem 3.1 may be applied to the PageRank equation[1].

3.3 3 Algorithms

3.3.1 Inspiration

In [8], Sankaralingam et al, offer for consideration the algorithm quoted below. The three novel algorithms to be presented in this section were born of a detailed investigation into this offering.

/ initialize */*
Set all pageranks to an initial value;

At time = 0;
Concurrently on all peers:

for all documents in this peer {
    compute newrank based on inlinks;
    relerr = abs(oldrank - newrank)/newrank;
    if (relerr > epsilon) {
        a) send pagerank update message to out-links on other peers
        b) weights of out-links in same peer updated;
    }
}
/ first pass is done */*
/ every peer listens for pagerank update messages */*

while pagerank update message received {
    recompute newrank based on message received {
    relerr = abs(oldrank - newrank)/newrank;
    if (relerror > epsilon) {
        a) send pagerank update message to out-links on other peers
        b) weights of out-links in same peer updated;
    }
}

[1]Note: Some care needs to be taken with the update-delay conditions.

3.3.2 A Closer Look

The algorithm suffers a number of apparent vagaries and ambiguities.

Every effort is made in [8] to extol the virtues of this "asynchronous" algorithm. The paper offers some rather peculiar synchronising assumptions in its testing (about how messages are sent and received), but the suggestion is that this is merely to give some sense to the results. The algorithm is purportedly "asynchronous", and, as an improvement on the centralised model, it is written that the algorithm can cope with dynamic document entry into and document removal from peer-to-peer systems. Yet, the fourth line of the algorithm reads: *Concurrently on all peers.* Clearly, as given, the algorithm's initial PageRank propagation is not asynchronous. And, this, surely, is a mistake. Line four seems an unwanted line which probably permeated from the testing presuppositions into the algorithm proper.

Lines seven and fourteen seem also to be open to question: *relerr = abs ((oldrank - newrank)/ newrank).* Let us suppose that Sankaralingam et al were right to use percentage change as a propagation test condition. Then, surely these lines ought to have read *relerr = abs((oldrank - newrank)/oldrank)* – Sankaralingam et al could not have meant to divide by newrank. But, this propagation test is not, as implied in [8] a distributed analogue of Page and Brin's formulation. By focussing on percentage change, the algorithm makes update propagation biased toward lower ranked pages – given a PageRank update message, a recipient, r, is more likely to meet a percentage change threshold if r has lower PageRank. Most likely, lines seven and fourteen ought to have read: *relerr = abs(oldrank - newrank).*

This assumes, though, that we know what what is picked out by the term *oldrank*. There are two possibilities. The most immediate of these, and the one probably intended by Sankaralingam et al[2], is: PageRank after the preceding update. This *oldrank* interpretation struggles. Suppose some document, d, receives update messages, and updates its PageRank accordingly. It is quite possible that these updates be very small, but, nevertheless, numerous. This would allow d's PageRank to increase substantially (because of the number of updates), but never be propagated onto d's outlinks (because the updates are each too small). This runs counter to the centralised PageRank calculation model where all substantial PageRank changes are propagated. The second *oldrank* interpretation, and the one advocated here, is an answer to this difficulty, it counts *oldrank* as being: PageRank after the preceding propagation.

Update message is another term in need of explication. Let us recapitulate the equation which is the heart of the algorithm: $P(a) = (1-c) + c\sum_{(inlinks\ to\ a)} P(i)/deg(i)$, where $P(i)$ is the PageRank of a document, $deg(i)$ is the number of outlinks from document i, and c is some real between 0 and 1, exclusive. Given this, the obvious messaging interpretation is of a document sending to its outlinks $\frac{PageRank}{number\ of\ outlinks}$. There are two principle difficulties associated with this interpretation. It demands that documents keep a record of all their inlinks. For if a document were to receives $\frac{PageRank}{number\ of\ outlinks}$ message, what it would need do is to substitute this value for the previous value sent by the same inlink. This is quite a memory overhead. Also, it lends itself to PageRanking error. For, unless a strict ordering is placed on messaging from any given document (something made more difficult if updating and messaging is done by multiple threads), it is quite possible that an earlier update message replace a later update at some given document. The interpretation proposed here is for documents to send to one another $(\frac{PageRank}{number\ of\ outlinks} - \frac{Previously\ propagated\ PageRank}{number\ of\ outlinks\ at\ previous\ propagation})$. This circumvents both difficulties.

[2]They write of using *successive* PageRanks to decide propagation.

The remaining two discussion points concern *(a)* and *(b)* . We consider first *(b) weights of out-links in same peer updated*. There are at least four different ways in which local outlinks may be updated. Suppose document d receives an update message from some foreign document. Suppose d' and d" are two documents local to d, both of which are outlinked by d. Then, one method of update would be to simply increment the PageRanks of d' and d" by $c\times$*(update message received from d)*. This is the least recommended of the methods. Suppose d' outlinks to d". Then, despite the PageRank of d' incrementing because of inlink d, this PageRank increase would not be propagated onto d as required. An answer to this problem case would be to have d to send *local* "update messages" to both d' and d". On receipt of these, d' and d" would update their own PageRanks. d' would then calculate the resultant change in its PageRanks. If the absolute change in PageRank were to exceed epsilon, then d' would send its own *local* "update message" to d". This message would trigger a further update in the PageRank of d". In essence, the idea would be to set-up a local messaging analogue of the asynchronous messaging between foreign linked documents.

The other two possibilities take issue with idea of asynchronous local updates. The first is a method of synchronous updates in a fashion which is describable as a matrix calculation. The idea is that if d receives a foreign update message and the propagation test is passed, then *all* documents local to d, d', for example, concurrently propagate to their local outlinks the weighted change from their previously propagated PageRank. This idea rests less easily with the algorithm as given: ideally this method ought to take the PageRanks of all local documents into account when testing whether local propagation ought to take place – using just the PageRank of d to test for propagation could also result in a significant but not propagated change in the PageRank of d'. Ideally the propagation criterion ought to be $(\|R - oldR\|_1 > epsilon)$, where R is the vector comprising the PageRanks of all local documents and $oldR$ is the vector comprising the *oldranks* of all local documents. The final method notes that this matrix method could run into difficulty if few foreign messages are sent to documents local to d, but each of these updates is large. In such a case, it could be that the resultant PageRanks are not the equilibrium PageRanks. What is needed is further local propagation. This is what the final method gives us. Rather just a single *iteration* on receipt of a foreign update message, the final method asks that after the first local update, we rerun the propagation test. If the test is passed, we repropagate. We continue in this vein until the propagation is failure in the propagation test.

Let us turn attention now to *(a) send pagerank update message to out-links on other peers.* This line dictates that it is only documents which receive update messages from remote documents which may send update message to remote hosts. This means that if d", in the earlier example, has only one outlink, and this outlink is at a remote host, then d' would never be able to propagate on its PageRank, no matter how large its PageRank. This cannot be right. Correction hereof requires fairly significant structural change to the algorithm.

3.3.3 Algorithm 1

Algorithm 1 is the first of the novel algorithm which attempt to address the concerns raised above. It employs the local asynchronous local update approach. It also abstracts away from the peer to peer setting of the earlier algorithm.

for all pages at this node{

/ Initialise and propagate */*

```
    PageRank = 1 - c;
    If there are outlinks {
        weightedPageRank is the function PageRank/(Number of outlinks);
        for all the page's outlinks
            propagate(outlink, c×weightedPageRank);
        propagatedPageRank = PageRank;
        propagatedWeightedPageRank = weightedPageRank
    }

    /* Update and propagate */
    loop{
        If newly removed head of update queue is such that (destination==PageName){
            PageRank = PageRank + update;
            If there are outlinks and abs(PageRank - reportedPageRank) > epsilon {
                for all the page's outlinks
                    propagate(outlink, c×(weightedPageRank - propagatedWeightedPageRank));
                propagatedPageRank = PageRank;
                propagatedWeightedPageRank = weightedPageRank;
            }
        }
    }

}



By propagate(outlink, update), we mean:
    if outlink is local
        add (outlink, update) pair to local update queue
    else
        send (outlink, update) message to appropriate foreign host.

When a host receives a (outlink, update) message,
    the (outlink, update) pair is added to the update queue.
```

3.3.4 Communication Overheads

Algorithms 2 and 3 are responses to the anticipated communication overheads of algorithm 1. Both send collections of page updates rather than individual page updates. Also, they adopt, respectively, the two synchronous local update methods described earlier. To get a sense of how this happens, we consider the following.

Suppose the following gives the centralised matrix caculation of a system's PageRanks: $R_{k+1} = R_k cA_{n\times n} + (1-c)P$. Suppose, for simplicity, that there are just two nodes in this system. Without loss of generality, we may assume that pages 1 through d are at the first node, and that pages $d+1$ though n are at the second node. Let $A = \begin{pmatrix} E_{d\times d} & F_{(n-d)\times d} \\ G_{d\times(n-d)} & H_{(n-d)\times(n-d)} \end{pmatrix}$. Let R_k^1 and R_k^2 be the restrictions of R_k to the pages at node 1 and 2, respectively. Let P^1 and P^2 be de-

fined similarly. Then, by simple matrix manipulation, we can reduce the centralised equaltion to $R^1_{k+1} = R^1_k cE + R^2_k cG + (1-c)P^1$ and $R^2_{k+1} = R^2_k cH + R^1_k cF + (1-c)P^2$. Noting that node 1 has access to E and F (because each page i has access to Aij, for all j) and node 2 similarly has access to G and H, then if nodes 1 and 2 can exchange with each other $R^1_k F$ and $R^2_k G$, then these twin equations can be calculated, respectively, at nodes 1 and 2. Generalising this two-node example, and introducing asynchronous messaging between nodes, gives a foundation for algorithms 2 and 3.

3.3.5 Algorithm 2

Let k be the number of local pages.

/ Initialise and propagate*/*
For each local page, PageRank =1-c.
If there are foreign nodes which host pages outlinked by some local page {
To each such foreign node n {
send an array of pairs ($(update_1, page_1)$,...$(update_f, page_f)$), where
$\{page_1 ... page_f\}$ comprises all outlinked pages hosted by n;
$update_i = \sum_{j=1}^{k} c \times$ weightedPageRank of local page j

$$j\text{'s } weightedPageRank = \begin{cases} \frac{PageRank}{deg(i)} & \text{if } j \text{ outlinks to another page,} \\ & \text{local or foreign} \\ 0 & \text{otherwise} \end{cases} ;$$

deg(j) is the number of pages, local or foreign, outlinked from j
}
For each local page, propagatedPageRank = PageRank;
For each local page, propagatedWeightedPageRank = weightedPageRank;
}

/ Update and Propagate */*
Let X be a k row vector where each entry X_i corresponds to a unique local page.
Set each $X_i = 0$
Let A denote the $k \times k$ matrix, where

$$A_{ij} = \begin{cases} 1/deg(i) & \text{if there is an outlink from local page } i \text{ to local page } j \\ 0 & \text{otherwise} \end{cases}$$

Let P be a k row vector, where each entry P_i is the PageRank of a unique local page.
Let X and P match one another with respect to index correspondence with local pages.
loop {
If there is an array of pairs ($(update_1, page_1)$,...) in the update queue {
Remove the head, h, of the update queue;
For each i, $X_i = X_i + update^*_i$, where

$$update_i^* = \begin{cases} update_g \text{ in } h & \text{if the page corresponding to } X_i \text{ is given by} \\ & \text{some } page_g \text{ in } h \\ 0 & \text{otherwise} \end{cases}$$

$P = P \times A + X.$

If there are foreign nodes hosting pages outlinked by some local page {
Let pP be a k row vector of the propagatedPageRanks of local pages.
Let wP be a k row vector of the weightedPageRanks of local pages.
Let pwP be a k row vector of the propagatedWeightedPageRanks of local pages.
Let pP, wP and pwP match P's index correspondence with local pages.
If $\|P - pP\|_1 < epsilon$ *{*
To each foreign node n hosting outlinked pages {
send an array of pairs $((update_1, page_1), \ldots (update_f, page_f))$*, where*
$\{page_1 \ldots page_f\}$ *comprises all outlinked pages hosted by n;*
$update_i = \sum_{j=1}^{k} c \times (wP_j - pwP_j)$.
}
}
pP = P;
pwP = wP;
}
}
}

3.3.6 Algorithm 3

Algorithm 3 is the result of substituting in algorithm 2

$$P = P \times A + X$$

with

do {
precedingP = P;
$P = precedingP \times A + X.$
} *while* $(\|P - precedingP\|_1 < epsilon)$

3.3.7 Novelty: Shi et al

It is noted that *after* algorithms 2 and 3 were put together, a paper [9] was discovered, by Shi et al, which presents two distributed algorithms involving matrix calculations at each node. These algorithms appear to throw into doubt the novelty of two of the algorithms presented here. Close inspection, however, shows this not to be the case.

In their section 3, Shi et al give the calculation done by each node as

$$R_{k+1} = cER_k + (1-c) \sum_{(inlinked\ node\ i)} G^{(i)} R_l^{(i)} + (1-c)P^1,$$

where $0 \leq l \leq k$, and $G^{(i)}$ and $R_l^{(i)}$ are the rank vector and G matrix of node i. Now, an obvious thing to note is that Shi et al. have switched the order of the matrix-vector multiplication in their equation. This would be only a superficial difference if E and G were the transposes of the matrices given algorithms 2 and 3. But, they are not. This is clear from the definitions of E and G, which they provide in the same section (they use the letters A and B). It is clear also from the appendix, where, in the proof of lemma 1, they claim the $max_i \sum_j cE_{ij} < 1$. This can only be so if they actually meant E and G to be as they are given in algorithms 2 and 3. This is certainly a mistake on their part.

Furthermore, if we note that Shi et al multiply the external PageRank contribution, namely $\sum_{\textit{(inlinked node i)}} G^{(i)} R_l^{(i)}$, by $(1 - c)$ and not c, it is clear that they are presenting something wholly different. Quite what they are presenting is unclear: their aim is to provide a distributed PageRank algorithm, but with no justification for multiplying by $(1-c)$, it appears they have made a mistake.

And, it is not just this coefficient which they fail to justify. The whole framework is somewhat shaky. Consider, for example, the convergence proof which they provide. Shi et al do not mention chaotic relaxation. Instead they present an *infinite sequence* proof which proceeds by contradiction. Supposedly, their algorithm is increasing in terms of PageRank, because assumption otherwise leads to *contradictory* sets of infinite sequences. Quite how the arrive at their infinite sequences, though, is unclear, and it is clearly wrong. Were their claim to be true, then, contrary to their suggestion, PageRank would *not* be independent of the initial PageRank assignment.

All in all, the feeling is that they have presented different (most likely, incorrect) algorithms, which they have failed to justify.

Chapter 4

Testing

In this chapter we shall be considering Doogle, the implementation framework used to test the three earlier algorithms. Doogle comprises: a Distributed PageRank Calculator, which is configurable so as to calculate PageRank according to algorithm 1, 2 or 3; a Centralised PageRank Calculator, for comparison purposes; and a Crawler. for configuration. The test results obtained from Doogle are also presented in this chapter.

4.1 Experimental Set-Up

The initial hope for Doogle was that it be extensible to a fully distributed search engine implementation. Given this, it was thought quite likely that Doogle would have a peer-to-peer setting, and that the programming language chosen would be such as to provide the requisite portability and celerity – C, most likely.

As the report took shape, however, it quickly became apparent that testing PageRank calculation algorithms would be the primary concern. In light hereof, a decision was made to set Doogle in the Web. The reason was transparency: it was thought that a Web setting would permit greatest abstraction from message routing algorithms, page movement, and node availability. For similar reasons, it was decided that coding would be in Java. It was thought that the object-oriented paradigm would lend itself to greatest clarity; it was thought that the Java Thread-Monitor method of implementation would be easily understood; and it was supposed that Java's Remote Method Invocation would provide the clearest route toward *exactly once* messaging semantics.

4.1.1 Pseudo-Webservers

One of the earliest, and most significant, of the test difficulties was finding a suitable test set. It was initially hoped that we would be given access to a set of Webservers, and that these servers would realistically host a closed[1] set of webpages. Unfortunately, this proved not to be the case. This left us with two configuration tasks: finding a suitable set of webpages; and mapping these realistically to a set of machines serving as pseudo-Webservers. Figure 4.1 gives Doogle's answer

[1] A set of webpages W is closed if none of its pages have links which transcend W.

the first of these tasks.

The Doogle crawler operates in a breadth-first fashion. Given a user-specified root url, the crawler retrieves the corresponding webpage and extracts url links from it. These links are added to a To-be-Crawled queue. Having finished link extraction, the crawler saves the retrieved page to disk and adds the crawled url to a Been-Crawled queue. The crawler then removes the head of the To-be-Crawled queue, checks to see whether the page corresponding to this url has been crawled, and, then, if it has not been not, the retrieval-extraction-saving cycle is repeated. (Note: the crawler is able to test whether two distinct urls pick out out the same page.) This process continues until a user-specified number of webpages has been crawled, or the To-be-Crawled queue becomes empty.

The crawler has two peculiarities, which accord with the task at hand. Once the actual crawling finishes, the crawler filters out links from the saved pages which transcend the set of saved pages – i.e. link to pages which have not been saved. The reason for this is that we require a closed set of pages which can be regarded as a microcosm of the Web. Also, if the crawler finds that two or more urls pick out the same webpage (e.g. http://www.ic.ac.uk and http://www.imperial.ac.uk), then it fixes the first of these as *the* url, and overwrites all the variation urls with this particular one. This second peculiarity is a requirement of the first.

There are two further points to note. Crawlers are traditionally notorious memory consumers. With a view to limiting memory-demands, the Doogle crawler only adds links to the to-be-crawled queue if they have not yet been crawled and they do not appear appear in the To-be-Crawled queue. Secondly, the Doogle crawler only retrieves html pages. This accords with much of the testing in the literature. The main reason, though, is: the parser packages[2], which lie at the heart of the Doogle crawler extractor, struggle with non-html pages.

When the Doogle crawler was put into action, different roots were experimented with. Hoping to find test sets which would exhibit as much variation in link topology as possible, three were chosen:

- http://www.google.com/dirhp;
- http://www.ic.ac.uk; and
- http://www.doc.ic.ac.uk/~cj603.

Crawls were limited to 1000 pages[3]. The following table hints at the topological differences[4].

	Cul de Sac Pages	Links	Intra-Webserver Linkage
http://www.google.com/dirhp	20	7,903	50.7%
http://www.ic.ac.uk	170	13,188	52.4%
http://www.doc.ic.ac.uk/~cj603	174	11,174	50.5%

With three webpage sets in place, the aim was now to upload webpages to test machines in a realistic manner. Let $W : WebageURL \rightarrow HostName$ give the mapping from Webpage URLs to Hosts[5]. Let $H : HostNames \leftrightarrow K$ give the bijective mapping from the set of hostnames of webpages in the saved sets to a subset of the naturals, K. Let $M : HostNames \leftrightarrow L$ give the

[2]Parser packages from sourceforge.net.

[3]This was because distributed PageRank calculation for 1000, with $\epsilon = 0.0001$, took up to nineteen hours.

[4]NB: By intra-webserver, we mean intra-pseudo-webserver – i.e. after the mapping of Webserver to test host.

[5]Java URL class can be used.

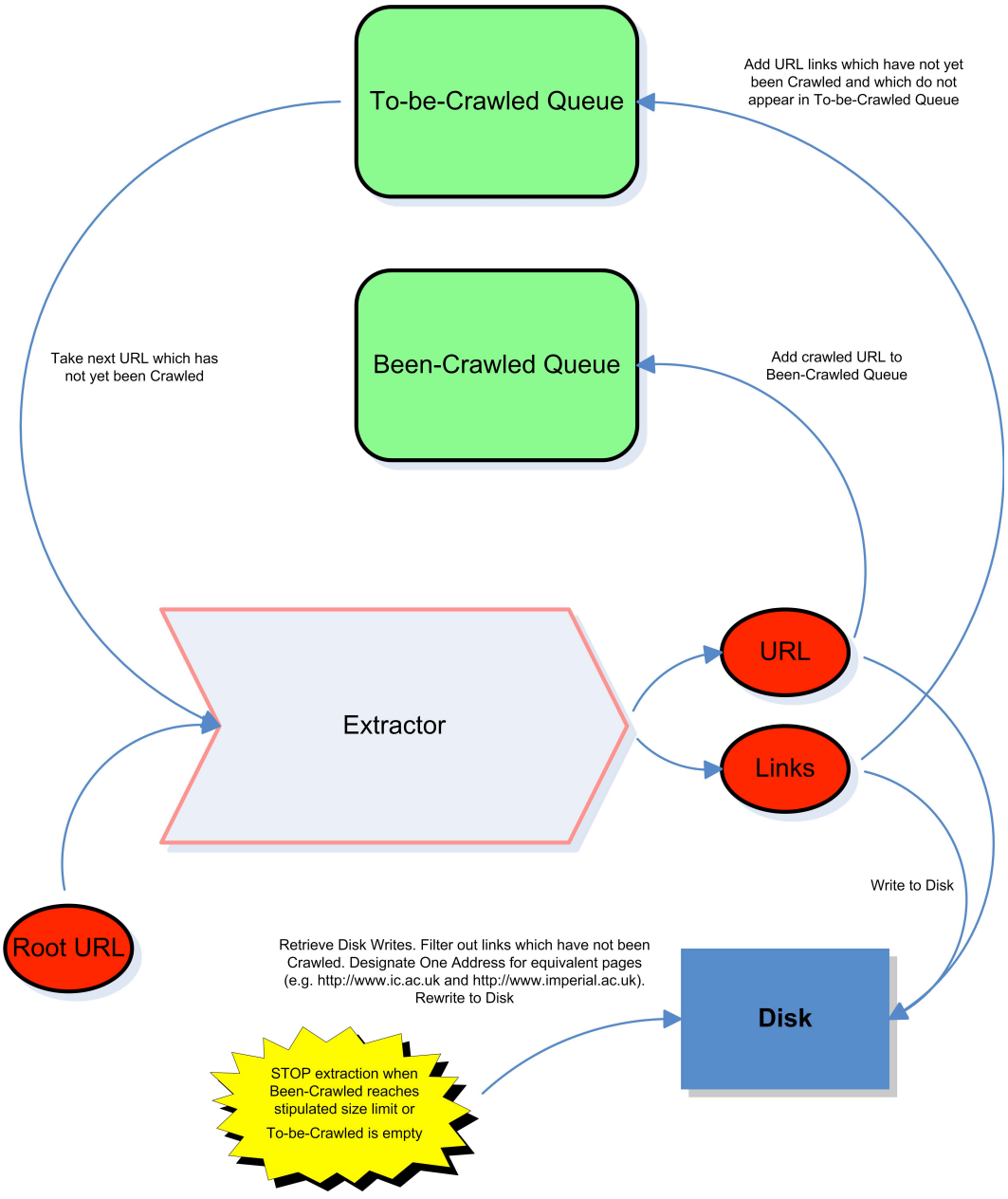

Figure 4.1: Breadth-First Crawler

bijective mapping from the set of machine names of used for the testing to a subset of the naturals, K. Then, if w is a webpageURL, a *realistic* mapping of webpage to test machine is given the rule $M^{-1}(H(W(w))\%|K|)$, where $|K|$ gives the number of elements in K.

4.1.2 Distributed PageRank Calculator

Figure 4.2 gives a course-grained depiction of Doogle's distributed PageRank Calculator – a course-grained depiction of the active and passive constituents of a distributed calculator process. It shows potentially many RMI Servers receiving Update Information from a different Calculator Process. This information is added to a Received-Message Queue. A user-defined number of Updater threads fetches this information from the Received-Message Queue (in the case of algorithms 2 and 3, the number of threads is restricted to one); the threads update the PageRank information of the appropriate pages; and, then, if PageRank change is sufficient, they fetch update information from the appropriate pages and add this to the Sending Message or Received Message queue (in the case of algorithms 2 and 3, addition is only to the sending message queue). Which queue is added to depends, respectively, on whether the update destination is a different process or the same one. A user-defined number of RMI Clients fetches update information from the Sending Message queue and sends it to the appropriate Calculator process.

Figure 4.2 also gives a flavour of some of the distributed calculator's intricacies – some of the attempts at maximising class cohesion and minimising class coupling.

- Updater access to pages is managed by a static Pages object, which has as its principle attribute a (PageName, Page) hashtable. This object also manages page entry into and exit from the system. In the case of algorithms 2 and 3, collection of Update information and addition is managed by a Propogator object – the collection of update information in algorithms 2 and 3 is not trivial.
- No page has direct access to its PagRank information – pages each have PageRank objects with suitable accessor and mutator methods.
- RMI Clients, given a webpage or host url, find correct destination addresses by way of a static ServerTable object – this too has a hashtable at its core.
- Processes are set up by way of factory and configuration classes: there are multiple configuration files to deal with, from webpage files to files stipulating algorithm parameters and names of test machine.
- Error handling, particularly in the face of multi-threading, is done by way of an ErrorLog class – error messages are time-stamped and written to file.

There are three further points worth noting about the Calculator. To build addtional asynchony into the testing, configuration permits including a delay function into any of the calculator processes. This permits modeling of distance between webservers. Also, the system permits modeling of dyanamic page entry or exit. On reaching system-wide PageRank convergence, PageRank information is written to file. We note that configuration allows for reading in of PageRank information relating to certain pages. So, the way dynamic page-entry is modeled by running the system until

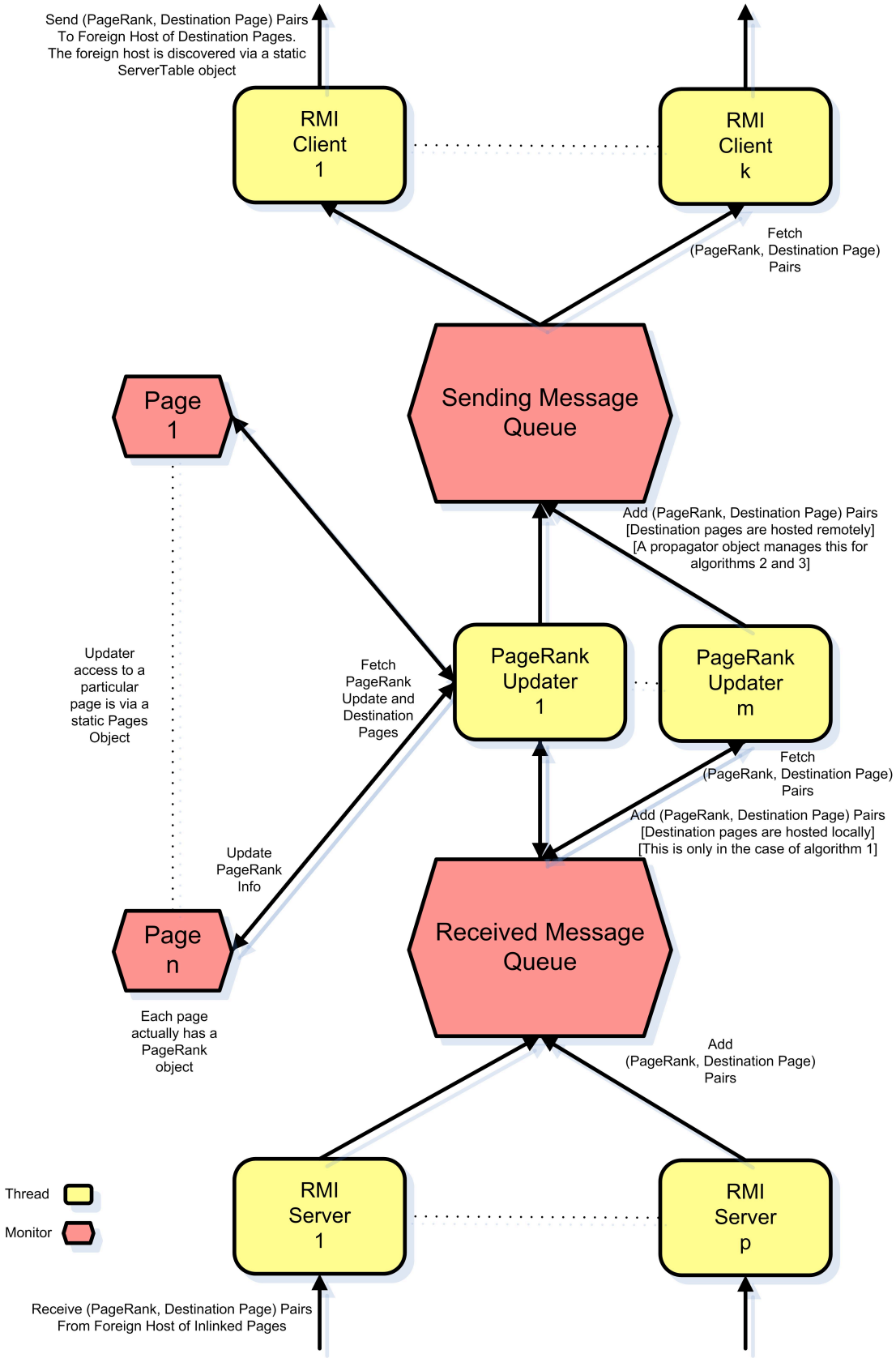

Figure 4.2: PageRanker

convergence and then rerunning it with pages added or removed. The third point concerns cul de sac removal. This is dealt with as follows: all servers send messages to one another so that all pages have a record of their inlinks. Then, if a page is a cul de sac, it informs its inlinks of this fact, and the inlinks mark this page as removed from the PageRank calculation. There are potentially multiple waves of cul de sac messaging. When this finishes, calculation begins. And, when calculation finishes, PageRank is propagated on to cul de sacs. This is, of course, somewhat ugly – and, the memory overheads are unwelcome. It is hoped that future research might render this exercise more pleasant.

4.1.3 Centralised PageRank Calculator

Doogle's centralised PageRank calculator is derivative of the distributed one. It naturally does not have message queues or messager threads. It updates PageRanks in a manner which is analogous to a distributed, algorithm-3, calculator.

4.2 Results

In this section we shall consider a set of Doogle test results. Taking the the second centralised algorithm as our reference point, we compare the PageRanks given us by the three distributed algorithms. We then consider the overheads associated with these algorithms. Throughout the testing, c is set to 0.85, and the dynamic addition of pages into the system is set to fifty-page increments.

4.2.1 Accuracy

Figures 4.3 through 4.11 show PageRank accuracy for the three distributed algorithms. PageRanks are all normalised, and sorted in descending order relative to the centralised algorithm. Note: the centralised algorithm sets $\epsilon = 0.1$.; algorithm 1 sets $\epsilon = 0.01$; and, algorithms 2 and 3 set $\epsilon = 0.1$. This is ascribed to different way in which algorithm 1 measures PageRank change as compared with algorithms 2 and 3 – the former measures change for just a single Page and the latter measures collective change for a set of Pages.

It is clear from these figures that the distributed algorithms all yield *very* similar PageRanks. But, all deviate from the centralised PageRank – and, what is worse, they struggle to preserve the centralised algorithm's PageRank ordering. We know by theorem 3.1 that the algorithms do converge to the centralised PageRanks. But, it would appear that asynchrony demands higher ϵ accuracy than was used – the algorithms were terminated too soon. The key question is: how high? Figure 9.112 shows that even a hundred-fold increase in ϵ accuracy does not yield adequate order-preservation for algorithm 2 (algorithms 1 and 3 yield almost identical graphs).

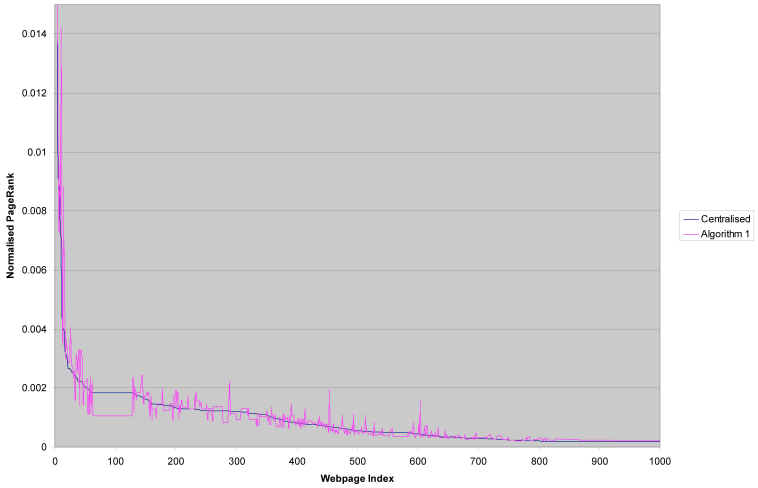

Figure 4.3: Testing Algorithm 1: http://www.google.com/dirhp

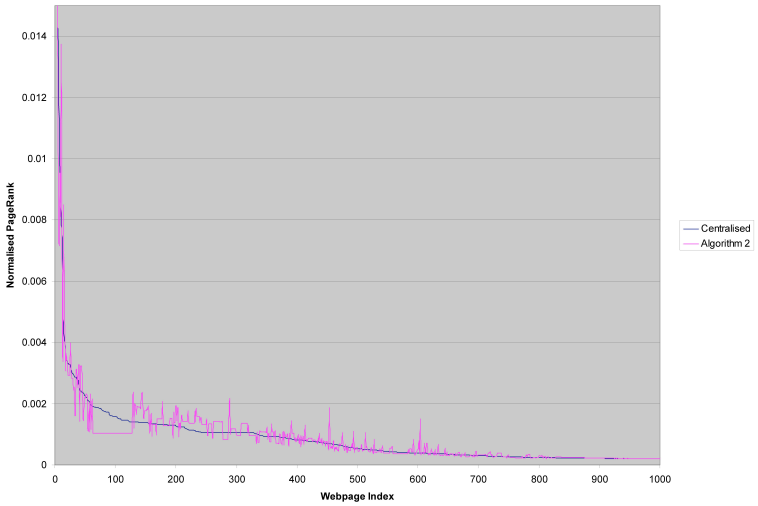

Figure 4.4: Testing Algorithm 2: http://www.google.com/dirhp

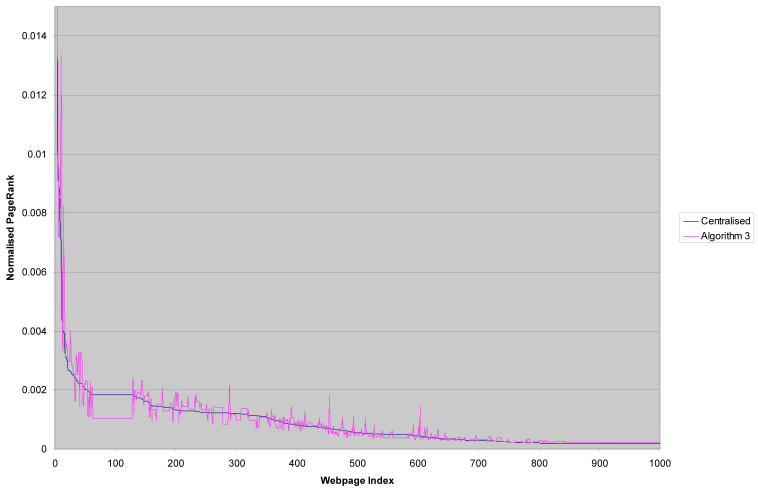

Figure 4.5: Testing Algorithm 3: http://www.google.com/dirhp

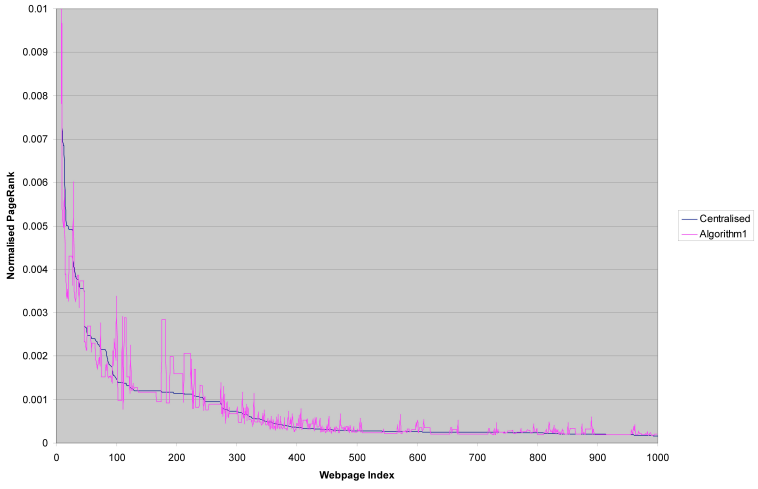

Figure 4.6: Testing Algorithm 1: http://www.ic.ac.uk

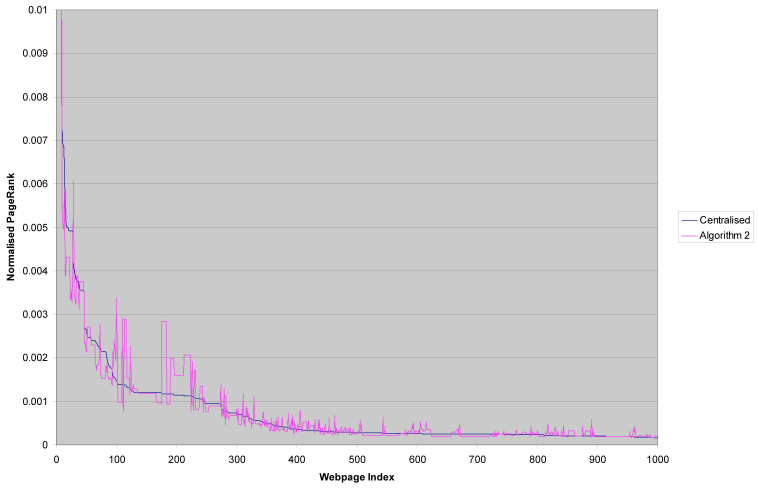

Figure 4.7: Testing Algorithm 2: http://www.ic.ac.uk

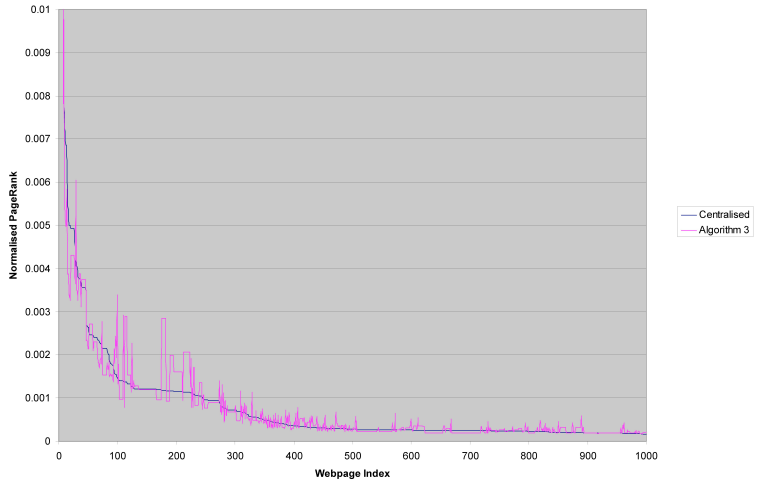

Figure 4.8: Testing Algorithm 3: http://www.ic.ac.uk

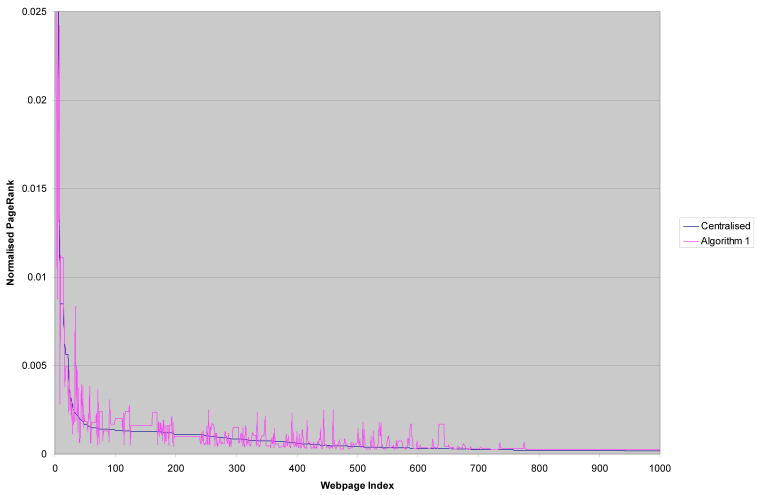

Figure 4.9: Testing Algorithm 1: http://www.doc.ic.ac.uk/˜cj603

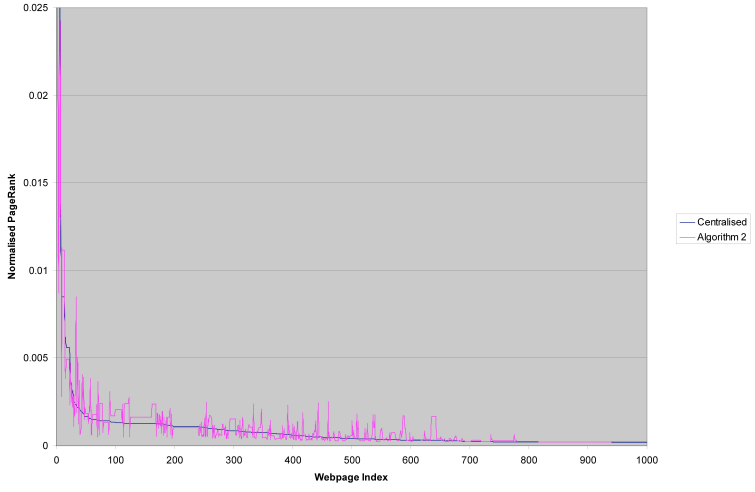

Figure 4.10: Testing Algorithm 2: http://www.doc.ic.ac.uk/˜cj603

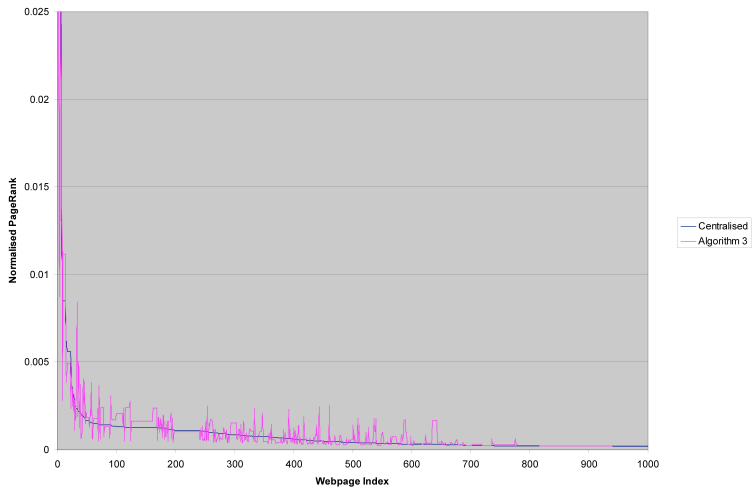

Figure 4.11: Testing Algorithm 3: http://www.doc.ic.ac.uk/˜cj603

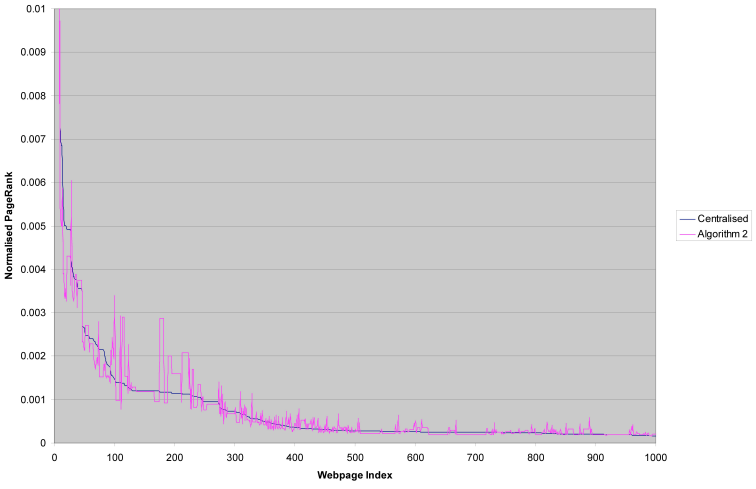

Figure 4.12: Testing Algorithm 2, given $\epsilon = 0.001$: http://www.ic.ac.uk

4.2.2 Communication Overheads

In the preceding subsection, it was suggested that high ϵ accuracy is demanded. Here we shall consider the number of messages sent as the ϵ threshold becomes more precise. We shall also consider the number of messages sent as the set of webpages increases in size.

Messages Sent relative to ϵ-Accuracy

		$\epsilon = 0.1$	$\epsilon = 0.01$	$\epsilon = 0.001$	$\epsilon = 0.0001$
Algorithm 1	http://www.google.com/dirhp		340,769	2,902,405	25,356,520
	http://www.ic.ac.uk/		76,094	680,815	5,946,356
	http://www.doc.ic.ac.uk/~cj603		57,312	451,883	3,694,051
Algorithm 2	http://www.google.com/dirhp	332	2,009	12,604	
	http://www.ic.ac.uk/	292	1,446	6,940	
	http://www.doc.ic.ac.uk/~cj603	391	1,649	7,717	
Algorithm 3	http://www.google.com/dirhp	229	1,292	6,886	
	http://www.ic.ac.uk/	117	554	1,833	
	http://www.doc.ic.ac.uk/~cj603	267	1,084	4,484	

This table shows algorithms 2 and 3 using significantly fewer messages than algorithm 1 – on occasion, nearing $\frac{1}{2000}$ and $\frac{1}{4000}$ of algorithm 1's messages, respectively. The table also shows that as $\frac{1}{\epsilon}$ increases by an order of 10, algorithm 1's messaging increases approximately nine-fold; algorithm 2's messaging increases approximately six-fold; and, algorithm 3's messaging increases approximately three-fold for the two sets with the highest number of links, and approximately 5 times for the third set.

Messages Sent relative to Webpage Number

		250 Pages	500 Pages	1000 Pages
Algorithm 1	http://www.google.com/dirhp	117,714	217,535	340,769
	http://www.ic.ac.uk/	14,198	34,420	76,094
	http://www.doc.ic.ac.uk/~cj603	12,698	32,420	57,312
Algorithm 2	http://www.google.com/dirhp	170	224	332
	http://www.ic.ac.uk/	84	160	292
	http://www.doc.ic.ac.uk/~cj603	109	230	391
Algorithm 3	http://www.google.com/dirhp	111	161	229
	http://www.ic.ac.uk/	61	106	117
	http://www.doc.ic.ac.uk/~cj603	95	175	267

The second table does not as clearly exhibit rates of messaging increase. Again, though, it is clear that algorithm 3 outperforms algorithm 2, which, in turn, significantly outperforms algorithm 1. One needs note, however, that as page numbers increase, the size of message sent within algorithms 2 and 3. This leads to questions of increased message size undoing the advantage of reduced message numbers. This is something that warrants further investigation. Interestingly, none of the rates of messaging increase exceeds the rate of Page number increase.

Chapter 5

Conclusions and Future Work

This paper opened with a review of the PageRank definitions offered in [1], [2] and [5]. Though unstated in the literature, these definitions are different – the PageRank of any given Page could vary according to each of the definitions. Chapter 2 provided us examples of this variation, and it reviewed the literature's justifications of the definitions. These were shown to be insufficient for [1] and [2], and in need of fleshing out for [5]. The remains of the chapter were dedicated to justifying PageRank.

As point of justificatory beginning, the paper provided a review of the requisite Markov theory. This included theorem 2.9, which was rediscovered whilst trying to justify PageRank – it was rediscovered independently of any literature. This review also included a novel proof of theorem 2.20.

Attention was then turned to the crux of the matter: the Random Surfer Model. An intuitive explanation was given for this model, and a novel formalisation was provided. This formalisation was shown to be well-defined, given the earlier Markov review – concerns included convergence, uniqueness and adherence to an intuitive framework. At this point, an assumption was posited: all Pages have outlinks. By using an assortment of linear algebraic techniques, this assumption permitted a formalisation extension to show that the definitions in the literature are all special cases of a particular PageRank type. The assumption was then dropped, and it was shown that there are, in fact, two general PageRank types – types which differ with respect to their treatment of cul de sac pages. In the throws of justifying PageRank, the paper also presented an exciting new theorem showing that Kamvar and Haveliwala's irreducibility requirement is too strong.

With PageRank justified, the aim was then to construct a distributed calculation algorithm. The paper presented and proved part of the Chazan-Miranker theorem [1]. Markov theorem 2.16 was recalled, and we were shown that, from a theoretical point of view, distributed PageRank calculation algorithms are possible.

The search for a good distributed algorithm started with a review of an algorithm in [8]. A detailed analysis showed that this algorithm could be significantly improved upon. Suggestions in [9] were also considered – these were shown to be seriously flawed. Three novel alternatives were presented. The first of these took inspiration from the algorithm in [8], but addressed six of this algorithm's cited difficulties. The second and third algorithms departed significantly from the algorithm in [8]. Some simple matrix manipulation explained their form. They were suggested as responses to the expected communication overheads of the first algorithm.

[1] The proof tidies up some of the forgotten coefficients and index subscripts in the original.

The three algorithms were put through empirical testing. The test set-up was described as an attempt to model communication asymmetry and to incorporate dynamic page-entry. In terms of PageRank accuracy, it was surprising to see that, despite convergence to the centralised PageRank values, all the distributed algorithms permitted a significant amount of variation. Theory showed us that the distributed algorithms do converge to the correct values. Empirical testing showed us that very low ϵ values are demanded. As expected, algorithms 2 and 3 were shown to substantially outperform algorithm 1 in terms of message numbers.

So opens the door to future research. It is unclear from the test results what ϵ values are associated with what level of PagRank accuracy – it is not clear how ϵ would need change in the face of changing rates of dynamic page-entry, or how ϵ would need change in the face of differing degrees of communication asymmetry. There is a suggestion in the data that higher numbers of outlinks leads to higher PageRank accuracy. This need investigation.

As noted in chapter 4, it is also worth investigating an elegant way of removing cul de sacs. The method used here has unwanted memory overheads.

And, whilst algorithms 2 and 3 appear to outperform algorithm 1 in terms of numbers messages sent, algorithms 2 and 3 do send larger messages. It would be of great interest comparing the overall network load, given the different algorithms. It would also be worthwhile investigating the possibilities for optimising message content relative to the different algorithms – sending page indices rather than urls, perhaps.

There are also many linear algebraic investigations which could be undertaken: investigation into link structure properties (with a view to preventing spam, for example); or investigation into alternate algorithm acceleration techniques (particularly for distributed algorithms, perhaps).

One might even wish to pursue the idea of introducing a back-button into the Random Surfer Model. Or, one might wish to investigate ensuring data integrity within a distributed calculation system – ensuring that PageRanks cannot be tampered with. There is much that could be done, and the possibilities are exciting.

Bibliography

[1] Sergey Brin and Lawrence Page. *The Anatomy of a Large-Scale Hypertextual Web Search Engine.* http://www-db.stanford.edu/ backrub/google.html

[2] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Wingrad. *The PageRank Citation Ranking: Bringing Order to the Web.* Stanford Digital Libraries Working Paper, 1998.

[3] A. Langville, C. Meyer. *Deeper Inside PageRank.* mac01-wi428b.math.ncsu.edu/ langville/DeeperInsidePR.pdf

[4] Taher Haveliwala, Sepandar Kamvar, Dan Klein, Chris Manning, Gene Golub. *Computing PageRank using Power Extrapolation.* www.stanford.edu/ taherh/papers/extrapolationII.pdf

[5] Sepandar Kamvar Taher Haveliwala Christopher Manning Gene Golub. *Extrapolation Methods for Accelerating PageRank Computations.* In *Proceedings of the Twelfth Internaltional World Wide Web Conference,* 2003.

[6] T.H. Haveliwala and S.D. Kamvar. *The second eigenvalue of the Google matrix. Stanford University Technical Report,* 2003.

[7] Sepandar Kamvar, Mario Schlosser Hector Garcia-Molina. *The EigenTrust Algorithm for Reputation Management in P2P Networks.* http://www.stanford.edu./ sdkamvar/research.html

[8] K. Sankaralingam, S. Sethumadhavan, J.Browne. *Distributed Pagerank for P2P Systems.* In *12th International Symposium on High Performance Distributed Computing.*

[9] S. Shi, J. Yu, G. Yang, D Wang. *Distributed Page Ranking in Structured P2P Networks.* In *Proceedings of the 2003 International Conference on Parallel Processing.*

[10] D. Chazan, W. Miranker. Chaotic Relaxation. *Linear Algebra Applications*, 2: 199-222, 1969.

[11] D Bertsekas, J. Tsitsiklis. *Parallel and Distributed Computation, Numerical Methods.* 1989.

[12] Olle Haggstrom. *Finite Markov Chains and Algorithmic Applications.* 2002

[13] I. Clarke, O. Sandberg, B. Wiley, T. Hong *Freenet: A distributed anonymous information storage and retrieval system.* In *Proceedings of the Workshop on Design Issues in Anonymity and Unobservability* (July 2000), pp. 46-66.

[14] I. Stoica, R. Morris, D. Karger, F. Kaashoek, 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.

[15] F. Dabek, M. Kaashoek, D. Karger, R. Morris, I Stoica. *Wide-area cooperative storage with CFS* http://pdos.lcs.mit.edu/chord/

[16] A. Rowstron and P. Druschel. *Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems.* In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pp. 329-350, November 2001.

[17] P. Bremaud. *Markov Chains: Gibbs fields, Monte Carlo Simulation, and Queues*, Springer, New York (1998).

[18] M.Richardson, P.Domingos *The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank* In *Advances in Neural Information Processing Systems 14*