A PARALLEL IMPLEMENTATION OF A LINK–BASED RANKING ALGORITHM FOR WEB SEARCH ENGINES

by

Adnan Burak Gürdağ

B.S. in Computer Engineering, Boğaziçi University, 1999

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science
in
Computer Engineering

Boğaziçi University

2002

A PARALLEL IMPLEMENTATION OF A LINK–BASED RANKING ALGORITHM FOR WEB SEARCH ENGINES

APPROVED BY:

Assoc. Prof. Can Özturan (Thesis Supervisor)	.....
Assist. Prof. İlkay Boduroğlu	.....
Prof. M. Ufuk Çağlayan	.....

DATE OF APPROVAL: 06.06.2002

ACKNOWLEDGEMENTS

I would like to thank my supervisor Can Özturan for his guidance and patience during this study. His passion to help his students including me has always been a great encouragement for them. I would also like to thank M. Ufuk Çağlayan for providing me with precious advises and for guiding me with his wisdom in all stages of my graduate study. I would also like to thank İlkay Boduroğlu for his participation to my thesis jury and for his precious feedbacks. I also thank Kuban Altınel for his effective graph theory course.

As Netlab, we have a great source of energy and inspiration who is known as Cem Ersoy. I owe him many thanks for many things that are too many to count here.

I would like to present my sincere thanks to my old friend Berk Gökberk for his great friendship and support. I also give lots of my thanks to Aydın Ulaş, Çiğdem Gündüz and all other CMPE people for their technical and spiritual support. During my assistantship, I feel the privilege to work with the people in CMPE. I thank all in CMPE staff who cause this feeling. I am also grateful to Sadık Gökhan Çağlar for his assistance to run our code on a powerful system. Without his help, our results would lack flavor.

I would like to thank my employer Oksijen Teknoloji for the tolerance they show during my thesis work. I present my sincere thanks to Murat Zeren, our project manager, for his supervision, and encouragements. I also thank to all people at Oksijen for making this company a great working environment for me.

Finally, I would like to thank my beloved, Gülçin Öztürk Gürdağ for her great support, patience, and love. During this thesis study, I, once again, understood that she is the one.

ABSTRACT

A PARALLEL IMPLEMENTATION OF A LINK–BASED RANKING ALGORITHM FOR WEB SEARCH ENGINES

Link–based ranking for Web pages is a technique that orders Web pages based on their linkage information. A popular application for link–based ranking is search engine technology. Google, which is currently considered the best search engine, owes much of its success to its PageRank method, which is a link–based algorithm to approximate the global popularity of Web pages. Considering the current size of the Web, calculation of the popularity of each web page (which is also called PageRank) can take many hours or days on a powerful computer.

In this thesis, we designed and implemented the PageRank algorithm for distributed memory parallel computers using MPI. Our aim is to make PageRank calculations for the growing number of Web pages in a fast and scalable way. Thus, it will be possible to provide central personalized search service that is a feature of PageRank method. PageRank is not currently used to provide personalized search service to a group of subscribers. We performed experiments on two Linux PC clusters to evaluate the performance and scalability of our implementation. As can be seen from our results, our implementation can be used to give personalized search service with an appropriate infrastructure.

ÖZET

WEB ARAMA MOTORLARI İÇİN BAĞLANTI TEMELLİ BİR SIRALAMA ALGORİTMASININ PARALEL GERÇEKLENMESİ

Bağlantı temelli sıralama, Web sayfalarının, yardımlı metin linkleri ile kendi aralarında oluşturdukları bağlantı yapısına göre sıralandırılmasına dayanan bir tekniktir. Bu tekniğin popüler uygulamalarından biri arama motorlarıdır. Şu anda en iyi arama motoru olarak gösterilen Google, başarısının büyük bir bölümünü yine bağlantı temelli sıralamaya dayanan ve esas olarak Web sayfalarının popülerliğini hesaplayan PageRank isimli bir algoritmaya borçludur. Web'in şu andaki boyutu göz önüne alınacak olursa, güçlü bir bilgisayar üzerinde her bir Web sayfasının popülerliğini (ya da PageRank'ini) hesaplamak saatler veya günler sürebilir.

Biz bu tezde PageRank algoritmasını MPI kullanarak dağıtık hafızalı paralel bilgisayarlar için tasarlayıp kodladık. Bunu yapmaktaki amacımız, hızla artan Web sayfaları için PageRank hesaplamalarını hızlı ve ölçeklenebilir şekilde gerçekleştirmek ve böylece PageRank'in yöntem olarak izin verdiği kişisel aramayı merkezi bir arama motoru servisi olarak mümkün kılmaktır. PageRank şu anda bir abone kitlesine kişisel arama servisi vermek için kullanılmamaktadır. Yaptığımız deneylerde, geliştirdiğimiz kodun iki Linux PC öbeği üzerindeki performansını ve ölçeklenebilirliğini gözledik. Çalışmamızın sonuçlarından da görüleceği gibi, gerçeklememiz uygun bir alt yapı ile kişisel arama servisi sağlamak amacıyla kullanılabilir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

b_i	Size of B_i
B_i	Set of backlinks of page i or the set of backlinks of V_i
d_{max}	Maximum out–degree
d_u	Out-degree of page u
E	Set of directed edges(links) in G
E_i	Subset of E on processor i
f_i	Size of F_i
F_i	Set of forelinks of page i or the set of forelinks of V_i
G	Web graph
G_i	Partition of G on processor i
K	thousand ($= 10^3$)
l_i	Number of local links in L_i
L_i	Set of local links on processor i
m	Size of E
m_i	Size of E_i
M	million ($= 10^6$)
n	Size of V
n_i	Size of V_i
n_{B_i}	Size of V_{B_i}
n_{F_i}	Size of V_{F_i}
n_{L_i}	Size of V_{L_i}
p	Number of processors used
p_r	Number of processors to receive data
p_s	Number of processors to send data
$\mathbf{p}$	Personalization vector
$\mathcal{P}$	Page(state) transition matrix
P_i	Processor with ID i
$P_{id}(u)$	Processor ID of page u
$\mathbf{r}$	PageRank vector

V	Set of vertices(pages) in G
V_{B_i}	Set of remote pages that is linked to the pages in V_i
V_{F_i}	Set of remote pages that is linked by the pages in V_i
V_i	Subset of V on processor i
α_{in}	Constant in probability distribution function for in–degrees
α_{out}	Constant in probability distribution function for out–degrees
β_{in}	Power law exponent of in–degree distribution
β_{out}	Power law exponent of out–degree distribution
AS	Autonomous System
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
Gb	Gigabit
I/O	Input output
IR	Information Retrieval
KB	Kilobyte
LBR	Link–Based Ranking
Mb	Megabit
MPI	Message Passing Interface
NIC	Network Interface Card
NFS	Network File System
PC	Personal Computer
PDF	Probability Distribution Function
pid	Processor ID
PL	Power Law
PR	PageRank
WWW	World Wide Web

1. INTRODUCTION

With the advent of World Wide Web, searching in a collection of hypertext documents has attracted great research interest. Considering the current size of the Web and its growth rate, there is a big challenge to locate information in such a huge data source. This challenge resulted in one of the most popular applications of the Internet, search engines.

The state of the art in the search engine technology has seen significant advancements in the second half of 90's. In the most general sense, search engines use classical IR techniques to match the user queries to the web documents [1, 2]. These techniques are based on the relationships between query terms and the web documents.

Besides the classical ones, current popular search engines apply many advanced techniques to improve the quality of the search results, which is generally measured by the level of user satisfaction and the relevance of the returned results to the given query. One of the most notable techniques is to exploit the connectivity information (i.e. link structure) hidden in the web documents to improve the search results.

1.1. Related Work

There are two major algorithms that exploit the link structure to rank the web pages in order to increase the relevance of the search results. One of these algorithms is called Clever (previously known as HITS) [3]. In Clever algorithm, each web page has an authority and a hub score. The authority score of a page is determined by the hub scores of the pages that give link to it. Similarly, the hub score of a page is determined by the authority scores of the pages that this page gives link to. It was stated that hub scores are in fact unnecessary in a large web environment since they eventually collects good authority scores [4]. Therefore, authority score should be sufficient to evaluate the quality of a web page among the other pages. Clever has currently no widely used implementation.

The other algorithm is called PageRank which is used by the Google search engine [5, 6, 7, 8]. PageRank is an iterative algorithm that is based on "importance" flow between pages. A web page divides and transfers its importance (or PageRank) equally to the pages to which it is linked. PageRank can be considered as the extended version of simple citation count which is widely used in academic world to determine the importance or popularity of a published paper. Unlike scientific publications, web pages are published without a control mechanism. In such an environment, it is not so meaningful to accept the number of references as a measure of importance since one can create thousands of web pages which give links to his or her page to give more importance to that page. This is called spamming. PageRank is immune to link spamming by its nature [5] since the importance of a page depends directly on the importance of pages that give links to that page and the importance of arbitrarily created pages will be very low unless they are linked by important pages, which is not quite possible.

Google is currently considered as one of the best web search engines and its popularity comes from its web coverage and high quality search results for the user queries. By "high quality", we mean the most relevant and generally the most "popular" in the subject. The quality of Google's search results is considered to be the direct consequence of its PageRank algorithm.

Haveliwala [9], emphasizes an important aspect of PageRank algorithm: *personalization*. PageRank can be personalized by applying biases during the iterations to increase the importance of certain groups of web pages. Personalization requires running the algorithm for each personal profile. Haveliwala [9] proposes a solution in which the link structure of the web is distributed in a high capacity media such as DVD–ROM and PageRank algorithm is run locally in users' PCs. He also showed for 19 million pages that running time and memory requirement of personalized PageRank algorithm can be reduced to make this computation on a modest machine.

The personalization is an important issue. But the idea of delegating the PageRank calculation to personal computers does not seem practical since the Web is a fast

evolving system and maintaining an up–to–date image of the link structure can be quite hard and expensive due to the frequent changes on the structure of the Web. Moreover, it is questionable that the computation of PageRank distribution for the Web in its current size(approximately two billion pages) can be done in a reasonable time on a modest PC.

1.2. Contribution of the Thesis

In this thesis, we designed and implemented a parallel version of PageRank algorithm for distributed memory multiprocessor systems. Our design is intended to be fast and scalable in order to calculate PageRanks of a very large number of Web pages for each user profile in a reasonable time. Considering the real world, the size of the problem is in billions for the number of pages and in millions for the number of user profiles. The length of the time frame for execution is in days or weeks depending on the need.

Since our program is implemented using open source free software and runs on the systems constructed with commodity off–the–shelf hardware components, the overall cost of the deployment system is quite low.

The process of getting personalization requests from the users is not covered in this work. This is a user interface issue and depends on the environment in which our parallel implementation is deployed. Browser history and personal bookmarks are two possible source of information to determine the personal preferences of the users. Our assumption is that we have a link graph of the web pages and personalization data is already obtained from the users.

1.3. Notations and Definitions

1.3.1. Directed Graphs

We use the following graph terminology throughout this document.

A *directed graph* $G = (V, E)$ consists of a set of vertices V and a set of directed edges E that are ordered pairs of elements of V. Note that multiple edges between two vertices are not allowed in directed graphs.

Let $< u, v >$ is an edge of the graph G. Then u is said to be *linked to* v and v is said to be *linked by* u. Here, u is the *initial vertex* and v is the *terminal vertex* of this edge.

The *in–degree* of a vertex v is the number of edges with v as terminal vertex. The *out–degree* of v is the number of edges with v as initial vertex. Since there are no multiple edges between any two vertices, in–degree is equal to the number of vertices that is linked to v and out–degree is equal to the number of vertices that is linked by v.

A *strongly connected component* of a directed graph G is a maximal subgraph H of G such that for any pair of vertices u and v in the vertex set of H, there is a directed path from u to v, as well as a directed path from j to i.

1.3.2. Web Graphs

A *web graph* is a directed graph $G = (V, E)$ where vertices represent web pages and edges represent hypertext links. The directed edge $< u, v >$ is a *backlink* of page v and a *forward link* (or *forelink*, as we call it) of page u. According to the graph definitions given in Section 1.3.1, number of backlinks of a page u is equal to the number of pages that is linked to u. Similary, number of forelinks of u is equal to the number of pages that is linked by u. The terms “backlinks” and “forelinks” for a page u are sometimes used to define the sets of pages that are linked to and linked by page u, respectively.

1.3.3. Distributed Web Graphs

We use the following terminology when we mention about web graphs partitioned into different processors.

For a processor i, we define the following page and link sets:

- *Local pages* (V_i) are the pages that reside on processor p.
- *Links* (E_i) are the links on local pages.
- *Local links* (L_i) are the links between local pages. L_i is a subset of E_i.
- *Remote backlink pages* (V_{B_i}) are the pages that reside on other processors and are linked to the local pages.
- *Remote backlinks* (B_i) are the links between V_{B_i} and V_{L_i}
- *Remote forelink pages* (V_{F_i}) are the pages that reside on other processors and are linked by the local pages.
- *Remote forelinks* (F_i) are the links between V_i and V_{F_i}. F_i is a subset of E_i.

These sets are depicted in Figure 1.1.

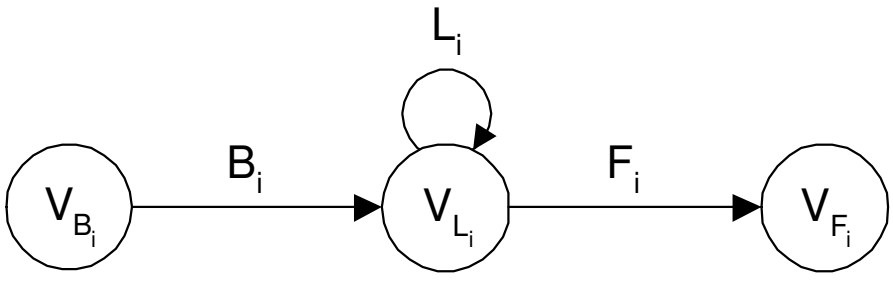


Figure 1.1. Page and link sets with respect to processor i

1.3.4. Markov Chains

The following definitions are from [10] and [11].

Definition 1.1 A *Markov chain* M is a discrete–time stochastic process defined over a set of states S in terms of a matrix $\mathcal{P}$ of transition probabilities.

Definition 1.2 A Markov chain is said to be *irreducible* whenever its underlying graph is strongly connected.

Definition 1.3 The *state probability vector* (also called the *distribution of the chain at time t*) $\mathbf{r}^{(t)} = (r_1^{(t)}, r_2^{(t)}, \ldots, r_n^{(t)})$ is the row vector whose ith component is the probability that the chain is in state i at time t. It can be easily shown that $\mathbf{r}^{(t+1)} = \mathbf{r}^{(t)}\mathcal{P}$.

Definition 1.4 A *stationary distribution* for the Markov chain with transition matrix $\mathcal{P}$ is a probability distribution $\boldsymbol{\pi}$ such that $\boldsymbol{\pi} = \boldsymbol{\pi}\mathcal{P}$.

Definition 1.5 The *periodicity* of a state i is the maximum integer T for which there exists and initial distribution $\mathbf{r}^{(0)}$ and positive integer a such that, for all t, if at time t we have $q_i^{(t)} > 0$, then t belongs to the arithmetic progression $\{a + Ti \mid i \geq 0\}$. A state is said to be *periodic* if it has periodicity greater than 1, and is said to be *aperiodic* otherwise. A Markov chain in which every state is aperiodic is known as an *aperiodic Markov chain*.

Theorem 1.1 (Convergence to equilibrium) Any irreducible and aperiodic Markov chain has a unique stationary distribution π such that for $1 \leq i \leq n, \pi_i > 0$, where n is the number of states in the chain.

Proof. See the proof of Theorem 1.8.3 of [11].

1.3.5. Random Walks

A random walk on a directed graph $G = (V, E)$ is defined as follows,

- Start from a vertex u
- Choose an outgoing edge $< u, v >$ with probability 1/d(u) where d(u) is the out–degree of vertex u,

- Go to the terminal vertex v,
- Continue the walk from v

A random walk can be represented by a Markov chain where the states are the vertices in graph G and for any two vertices $u, v \in V$,

$$\mathcal{P}_{uv} = \begin{cases} \frac{1}{d(u)} & \text{if } < u, v > \in E \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

1.3.6. Notation for Vector–Matrix Multiplication

Throughout this document, we follow the same notation as in [10] when we multiply a probability or PageRank vector $\mathbf{r}$ with a transition probability matrix $\mathcal{P}$. That is, we use $\mathbf{r}\mathcal{P}$ instead of $\mathcal{P}\mathbf{r}$. The reason is as follows. Since r_i is the probability of being in state i and $\mathcal{P}_{ij}$ is the probability of going from state i to state j, the probability of being in state j at time t is the multiplication of the vector $\mathbf{r}^{(t-1)}$ with the jth column of $\mathcal{P}$. For notational convenience, we take the probability vector as a row vector when it is multiplied with $\mathcal{P}$.

1.3.7. Performance and Scalability of Parallel Systems

In this section we give the formal definitions as given by Kumar *et al.* [12]

Definition 1.6 *Serial run time*(T_s) of a program is the time elapsed between the beginning and at the end of its execution on a sequential computer. *Parallel run time* (T_p) is the time that elapses from the moment that a parallel computation starts to the moment that the last processor finishes execution.

Definition 1.7 *Speedup* $(\mathcal{S})$ is the ratio of the serial run time of the best sequential algorithm for solving a problem to the time taken by the parallel algorithm to solve the same problem on p processors. That is, $\mathcal{S} = T_s/T_p$. In an ideal system, speedup is

equal to p.

Definition 1.8 *Efficiency* ($\mathcal{E}$) is a measure of the fraction of time for which a processor is usefully employed; it is defined as the ratio of speedup to the number of processors. That is, $\mathcal{E} = T_s/(pT_p)$. In an ideal system, efficiency is equal to one. Efficiency can also be represented as percentage.

Definition 1.9 *Scalability* of a parallel system is a measure of its capacity to increase speedup in proportion to the number of processors. It reflects a parallel system's ability to utilize increasing processing resources effectively. Therefore, a scalable parallel system is one in which the efficiency can be kept fixed as the number of processors is increased, provided that the problem size is also increased.

1.4. Outline of the Thesis

In this chapter, we give an introduction to the subject. Besides, we give the important notations and terminology used throughout this document. The outline of the remaining chapters are as follows. In Chapter 2, we give the details of serial PageRank algorithm with convergence analysis. Chapter 3 explains our parallel implementation of PageRank algorithm. In Chapter 4, the generation process of sample web graphs which are used in PageRank calculations is explained in detail. In Chapter 5, our experiments and their results are presented. Finally, we draw our conclusion in Chapter 6.

2. LINK–BASED RANKING AND PAGERANK ALGORITHM

2.1. Introduction

Link–based ranking (LBR) is used to rank web pages according to some popularity measure induced by their linkage. There are two major kinds of LBR: query–dependent and query–independent [13].

2.1.1. Query–dependent Link–based Ranking

In query–dependent LBR, given a search query, two sets of inter–related pages are found. These sets are called hubs and authorities. Hub pages are those pages that have good lists of links related to a subject. Authority pages are those that are listed on good hub pages. A good hub page for a subject links to many authoritative pages for that subject. Similarly, a good authority page for a subject is linked by many good hub pages for that subject. "Goodness" is measured by hub and authority scores. Due to the circular definition, these scores are calculated by iteration. This ranking scheme is used in Kleinberg's HITS algorithm [3]. As mentioned in the introduction, there are some weaknesses of HITS algorithm.

In a recent work, Richardson and Domingos [14] proposed to consider page contents in PageRank calculation, which is a query–independent LBR technique. Haveliwala [9] also considered query–specific ordering with PageRank.

Since our focus is not on query–dependent LBR, we do not elaborate further on this subject. Interested readers may find more information on the subject in [3, 15, 16, 4].

2.1.2. Query–independent Link–based Ranking

In query–independent LBR, a universal popularity score is assigned to each page. Given a search query, the result set is found by text retrieval methods and the pages in the set is ordered by their popularity scores. Hence, there is no relation between the popularity score of a page and its relevance to a search query.

In some versions of query–independent LBR, the popularity score is obtained by simply counting links given to a page. That is, the popularity score of a page equals to its in–degree. This kind of popularity is called directed popularity and is a valid measure for high quality documents such as journal papers. However, directed popularity is not a valid measure for Web pages since their quality is not controlled and shows extreme variations.

In some versions, the score is found by summing in– and out–degrees, which is called undirected popularity.

Lawrence and Page proposed an elegant method called PageRank [5, 17] to calculate the link popularity of web pages. PageRank is a method mainly inspired by citation analysis techniques. This method attracts great attention from research community. Our focus in this thesis is on PageRank and in the following sections we explain the PageRank calculation in detail.

2.2. Definition of PageRank

We give the following definition of PageRank (PR) based on [5] and [9]. Let u be a web page. Then let F_u be the set of forward links of u and let B_u be the set of backlinks of u. Let also $d_u = |F_u|$ be the out–degree of u. Let also r_u be the PR of page u. Then the PR values of the pages are calculated as follows,

$$r_v = \sum_{u \in B_v} \frac{r_u}{d_u} \quad \text{for all } v \in V \tag{2.1}$$

Since this is a recursive expression, the PR vector over all pages is calculated by iteration. Initially, each page is assigned a PR value of $1/n$ where n is the number of pages. Each page gives an equal share of its PR value to the pages it is linked. Therefore, PR calculation can be seen as a rank circulation among web pages. At each iteration, the PR values of the previous iteration is used.

The computation of PR values can be expressed as the following eigenvector calculation. Let $\mathcal{P}$ be a row stochastic matrix corresponding to the graph G of the web pages, assuming that all edges have at least one outgoing edge. If there is a link from page i to page j, then let the matrix entry $\mathcal{P}_{ij}$ have the value $1/d_i$. Let all other entries have the value 0. Then, one iteration of the PR computation corresponds to the vector–matrix multiplication $[\mathbf{r}]_{1\times n} \times [\mathcal{P}]_{n\times n}$. Repeated multiplication of $\mathbf{r}$ by $\mathcal{P}$ yields the dominant eigenvector $\mathbf{r}^*$ of the matrix $\mathcal{P}$. Since $\mathcal{P}$ corresponds to the stochastic transition matrix of the graph G, PageRank, which is denoted by $\mathbf{r}^*$, can be viewed as the stationary probability distribution over pages induced by a random walk on the web graph [10]. Because $\mathcal{P}$ is stochastic, its dominant eigenvalue is one. This leads to the following equality:

$$\mathbf{r}^* = \mathbf{r}^* \times \mathcal{P} \tag{2.2}$$

A typical iteration of PR calculation can then be given as follows,

$$\mathbf{r}^{(t)} = \mathbf{r}^{(t-1)} \times \mathcal{P} \tag{2.3}$$

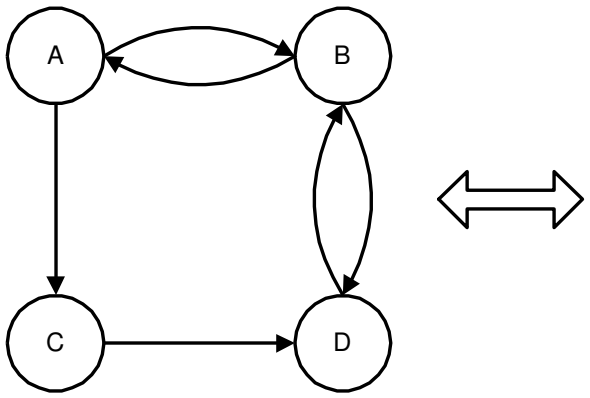
where $\mathbf{r}^{(t)}$ is the rank vector at iteration t. Note that $\mathbf{r}$ is considered as a row vector for notational convenience as explained in Section 1.3.6.

Since PR models the random walk of a web surfer over the web pages, the calculation mentioned above also corresponds to finding a stationary probability distribution for a Markov chain in which web pages represents the states and the matrix $\mathcal{P}$ represents the state transition matrix for the states in the chain. Discrete time steps in the chain correspond to the iterations of PR calculation. See Section 1.3 for detailed

definitions on Markov chains and random walks on directed graphs like web graphs. In Section 2.3.3, we discuss the convergence of this calculation.

2.2.1. An Example

In this section, we illustrate the PageRank calculation on an example. Consider the four–page web graph in Figure 2.1. We initialize the PageRank value of each page to 1/4. Figure 2.2 shows the graphical representation of the first iteration, which is realized as a rank flow defined in Section 2.2. Figure 2.3 shows the first iteration as a vector–matrix multiplication. After 13 iterations, the PageRank vector **r** converges to [0.2 0.4 0.1 0.3].



P	A	B	C	D
A	0	1/2	1/2	0
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1	0	0

Figure 2.1. A simple web graph and its transition probability matrix

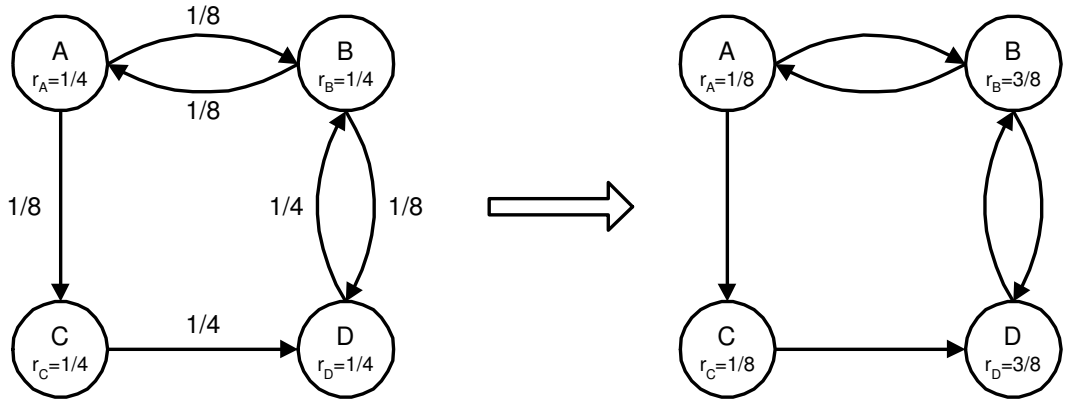


Figure 2.2. Graphical representation of the first iteration of PR calculation

$$\begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{bmatrix} \times \begin{bmatrix} 0 & 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1/8 & 3/8 & 1/8 & 3/8 \end{bmatrix}$$

$$\mathbf{r}^{(0)} \qquad \times \qquad \mathcal{P} \qquad = \qquad \mathbf{r}^{(1)}$$

Figure 2.3. First iteration as a vector–matrix multiplication

2.3. Computational Issues

There are some issues regarding the PR calculation presented in Section 2.2. These issues are addressed in the following sections.

2.3.1. Rank Leakage

First of all, we assumed that all pages have at least one outgoing link. This assumption is made since otherwise ranks of those pages with no outgoing links are lost from the system [5]. This fact is called *rank leakage*. In Figure 2.4, page D collects ranks from pages B and C but distribute no ranks. This causes a loss of some amount of ranks at each iteration. In other words, page D breaks the circulation of ranks inside the system.

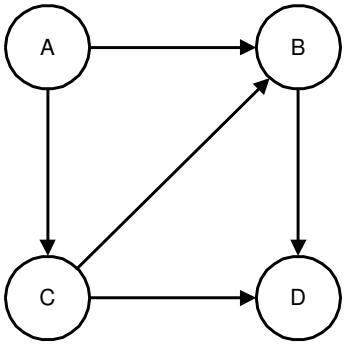


Figure 2.4. System loses ranks through page D

There are three methods proposed to prevent rank leakage. One is to recursively delete all such pages for the calculation and add them back to the system when the ranks are calculated. The other way is to add a link to all pages from the page with

no outgoing links. Doing this, ranks that are normally lost are redistributed back to the system. The third solution is to bound the number of PR iterations. In this work, we generated link graphs in which each page has at least one outgoing link (Chapter 4). Therefore, this assumption is valid for our sample graphs. Furthermore, we favor the second approach since it can be applied without extra cost in storage and computation. At each iteration, we add $z/(n-1)$ to the pages with non–zero out–degree and $(z-1)/(n-1)$ to the pages with zero out–degree where z is the number of pages with zero out–degree. This calculation gives the effect as if there are $n-1$ links from each page with zero out–degree to the other pages. The storage cost is one integer number z and the computational cost is one addition per page.

2.3.2. Rank Sinks

Another issue is the accumulation of ranks onto some group of pages which form a closed loop accepting links from outside of the group but giving links only to each other. An example is shown in Figure 2.5. In this case, pages C and D will accumulate all the ranks during iterations but distribute no ranks to the other pages since there are no links to the other ones.

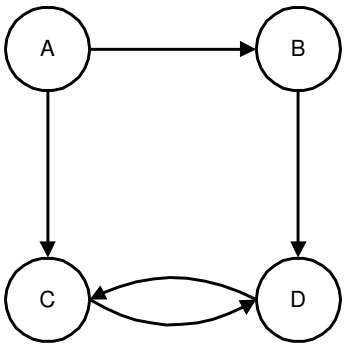


Figure 2.5. C and D forms a rank sink

The solution to this problem is to add a dampening factor D to the rank propagation [5, 9] where D is between zero and one. The state transition matrix is redefined in which the transition edges of probability $(1-D)/n$ are added between every pair of pages in G. The new matrix is defined as follows:

$$\mathcal{P}^{'} = D\mathcal{P} + (1 - D) \times \left[\frac{1}{n}\right]_{n \times n} \tag{2.4}$$

Since $\sum_i r_i = 1$, the iteration of PR becomes as follows:

$$\mathbf{r} \times \mathcal{P}^{'} = \mathbf{r} \times D\mathcal{P} + (1 - D) \times \left[\frac{1}{n}\right]_{1 \times n} \tag{2.5}$$

The introduction of dampening factor reduces the effect of rank sinks since it actually makes the transition matrix more realistically represent a random walk in which the surfer gets bored with probability $(1 - D)$ and jump to a page according to a probability distribution. In the expressions above, the uniform distribution vector $\left[\frac{1}{n}\right]_{1 \times n}$ is used. It is possible to use another distribution vector $\mathbf{p}$ which can be constructed from user preferences. We call $\mathbf{p}$ the personalization vector which is mentioned in Chapter 1. In case of using a personalization vector, the probability of the transition edge from u to v is given by $(1 - D) \times p_v$ where p_v is the probability of being at page v according to the user preference.

Besides solving the rank sink problem, using dampening factor also affects the convergence of PR calculation which is discussed in Section 2.3.3. The typical values of D are 0.85 and 0.90.

2.3.3. Convergence of PageRank

The PageRank iteration is said to converge if $\|\mathbf{r}^{(t)} - \mathbf{r}^{(t-1)}\|_2 < \Delta$ for sufficiently small Δ. In other words, the convergence criteria for PageRank iteration is the Euclidean distance between two successive PR vectors.

Since PR defines a random walk which can be represented by a Markov chain, according to the Theorem 1.1, the PR iteration converges to a rank vector $\mathbf{r}^*$ if the following conditions are satisfied:

- $\mathcal{P}$ is irreducible (i.e. web graph is strongly connected)
- The web graph is aperiodic.
- The initial rank vector is non–degenerate.

The first condition is satisfied by using dampening factor suggested in Section 2.3.2. Transition edges added make the web graph strongly connected since there is no more zero entry in the new transition matrix. The second condition is satisfied since the web graphs are aperiodic by nature. The last condition is satisfied if we choose an initial vector with all entries being non–zero.

2.4. Serial PageRank Algorithm

The serial version of the PR algorithm is given in Figure 2.6. Initially, we assign each page a rank value $1/n$ where n is the number of pages. Then, we iterate over all the pages while distributing their ranks equally to their forward links. The dampening factor is applied at the end of each iteration. This is effectively the same operation as multiplying current rank vector with matrix $\mathcal{P}$.

2.4.1. Time Complexity of Serial PageRank

There are total of $|V|$ floating point numbers read from personalization file at the beginning of an execution. The size of the data read from the graph file during an iteration is $|E| + 2|V|$ integers (Section 2.5.1). At the end of the execution, total of $|V|$ floating point numbers representing the resultant PR vector are dumped into disk. Hence, the total volume of I/O operations performed during an execution is $2|V| + c \times (|E| + 2|V|)$ integers where c is the number of iterations. Since there is a linear relationship between the number of links and the number of pages in a web graph (Section 4.2) and since the number of iterations is nearly constant (around 10 in our experiments), the total time spent on disk I/O during an execution is $O(n)$ where $n = |V|$. Here, we assume that unit times spent on I/O read and write operations are constant multiples of each other. There are $|E|$ rank additions and $|V|$ rank updates during an iteration. Therefore, computational complexity is also $O(n)$.

```
/* Initialization */
for all u ∈ V
    r_u^(0) = 1/n                    /* a non–degenerate initial PR vector */
    p_u = read from the personalization file
Residual = ∞
t = 0
/* PR iterations */
while (Residual > Δ)
    /* STAGE-1 : distribute ranks */
    for all u ∈ V
        for all v ∈ F_u
            r_v^(t+1) = r_v^(t+1) + r_u^(t)/d_u
    /* STAGE-2 : apply dampening factor */
    sum = 0
    for all v ∈ V
        r_v^(t+1) = D * r_v^(t+1) + (1 − D) × p_v    /* dampening */
        sum = sum + (r_v^(t+1) − r_v^(t))^2          /* for residual calculation */
    /* STAGE-3 : calculate residual */
    Residual = √sum
    /* STAGE-4 : transition to next iteration */
    r^(t) = r^(t+1)
    t = t + 1
```

Figure 2.6. Serial PageRank Algorithm

Under the assumption that the unit time for disk I/O is a constant multiple of the unit time for computation, we can say that the overall time complexity of serial PR algorithm is O(n).

2.5. Implementation of Serial PageRank Algorithm

2.5.1. Input and Output

Serial PR algorithm accepts a binary graph file and a personalization file as input. It produces a PR file as output.

The personalization file contains an integer number holding the number of pages and a list of 4–byte floating point numbers representing the personalization vector mentioned in Section 2.3.2. Therefore, its size is $4 \times (1 + |V|)$ bytes. The PR file contains the calculated PR vector and it has the same file structure and size as the personalization file.

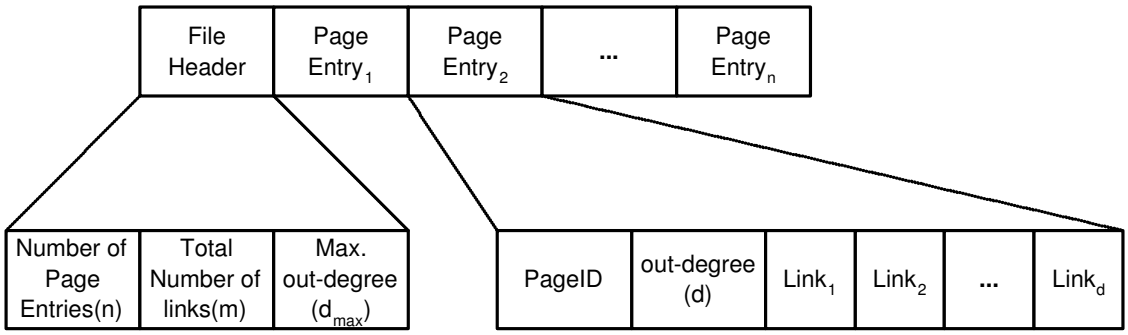


Figure 2.7. Structure of web graph file

The structure of a graph file is illustrated in Figure 2.7. This file consists of a header and page entries for the pages in the graph. The header is composed of three numbers: total number of pages(n), total number of links(m) and the maximum out–degree(d_{max}) in the graph. A page entry stores the necessary structural information for a page. It consists of a page ID u, the out–degree d_u, and the list of forward links of u. The list is an array of d page IDs. All numbers are stored in 4–byte integers. Therefore, the total size of the graph file is $4 \times (3 + 2n + m)$ bytes. Page entries and the page lists in the entries are sorted by page IDs. An example mapping between a web graph and the graph file is illustrated in Figure 2.8.

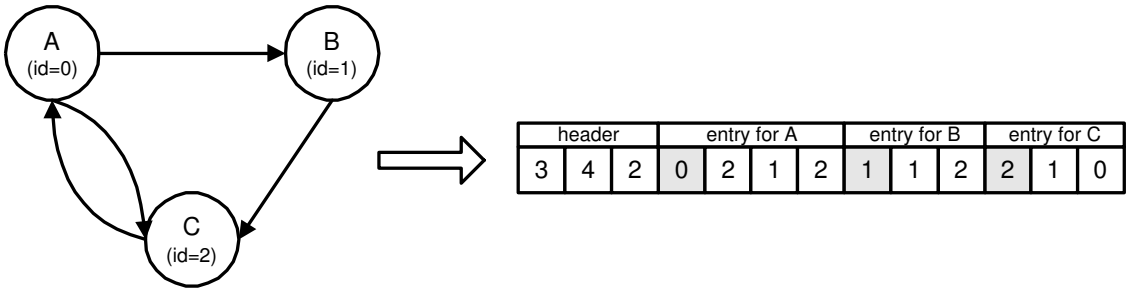


Figure 2.8. Graph file example

Note that this storage scheme is perfectly suitable for PR algorithm since the link information is read and processed sequentially from the graph file.

2.5.2. Memory Usage

In our implementation, the two PR vectors and the personalization vector used in the iterations are held in memory buffers (i.e. arrays). As Haveliwala [9] showed, four–byte floating point numbers has enough precision for PR calculations and we use four–byte floating point numbers, as well. Therefore, the memory allocated for these three buffers are $3 \times 4 \times n = 12n$ bytes, where n is the number of pages.

Since the graph file is very large in practice, we do not hold it in the memory. We use a buffer to hold the adjacency list in the largest page entry mentioned in Section 2.5.1. In each iteration, the entries in the graph file are read sequentially and analyzed through this buffer. The size of the buffer is $4 \times d_{max}$ bytes.

We used an extra I/O buffer to prevent large number of disk accesses and inefficient disk I/O that is caused by reading data in chunks smaller than the disk page size of the operating system [18]. Especially in our case, the pages that have small out–degree constitute the majority of all pages. Since the page entries of these pages are very small (e.g. 12 bytes for a page with out–degree=1), individual read operations for these entries will result in enormous lost in I/O efficiency. In practice, the page size for most Unix systems is 4KB or 8KB (in our case, Linux uses 4KB for disk pages). We use this buffer to read the data in bulk from the disk. Its size is $(d_{max} + 2)$ integers (i.e. size of the largest page entry), which is guaranteed to be larger than 4KB and not more than several hundred kilobytes due the nature of web graphs(Section 4.2). Note that, increasing the buffer size beyond 4KB has no effect on I/O performance in the limits of our usage.

3. PARALLEL PAGERANK

In this chapter, we give the details of our parallel PR algorithm and its implementation.

3.1. Parallel PageRank Algorithm

Let p be the number of processors. A web graph $G(V, E)$ is partitioned into p parts and each part is assigned to a processor. Let $G_i(V_i, E_i)$ be the partition assigned to processor P_i. Let also V_{F_i} be the set of remote pages linked by the local pages in V_i.

Along with G_i, each processor has a partition vector of $(u, P_{id}(u))$ pairs where $u \in V_i \bigcup V_{F_i}$, and $P_{id}(u)$ is the processor ID of the page u. This vector is used to determine the processor of a page to send ranks addressed to that page.

Each processor has enough information to exchange rank data with other processors. This information is made available to each processor before the execution of parallel PR algorithm. In Section 3.3.4, we explain how to setup this information.

The parallel version of PR uses the same realization of vector–matrix multiplication as in the serial version. The ranks destined to local pages are handled in the same manner whereas a processor has to send those ranks for remote pages to the processors on which these pages reside. Similarly, a processor has to receive those ranks for local pages from the processors on which the sender pages of these ranks reside.

The parallel PR algorithm is outlined in Figure 3.1. As can be seen, the first three stages of the parallel version correspond to the first stage of the serial version (Figure 2.6). Here, we can see two obvious overheads caused by the parallelization. One is the preparation of send buffers and the other is the exchange of send– and receive–buffers with the other processors.

for all P_i, do in parallel

/* Initialization */

read the communication information

read the personalization vector **p**

for all $u \in V_i$

$r_u^{(0)} = 1/n$ /* **a non–degenerate initial PR vector** */

Residual $= \infty$

$t = 0$

/* PR iterations */

while (Residual $> \Delta$)

/* STAGE–1 : distribute local ranks and prepare send–buffers*/

for all $u \in V_i$

for all $v \in F_u$

if $P_i == P_{id}(v)$ /* $v \in V_i$ **, update immediately** */

$r_v^{(t+1)} = r_v^{(t+1)} + r_u^{(t)}/d_u$

else /* v **is a remote page** */

store $r_u^{(t)}/d_u$ into the send–buffer for $P_{id}(v)$

/* STAGE–2 : exchange rank data with other processors */

send all buffers to the corresponding processors

receive all buffers destined to P_i

/* STAGE–3 : process ranks received from other processors */

for all buffers received

for all $r_u^{(t)}/d_u$ destined to v

$r_v^{(t+1)} = r_v^{(t+1)} + r_u^{(t)}/d_u$

/* STAGE–4 : apply dampening factor */

lsum $= 0$

for all $v \in V$

$r_v^{(t+1)} = D * r_v^{(t+1)} + (1 - D) \times p_v$ /* **dampening** */

$lsum = lsum + (r_v^{(t+1)} - r_v^{(t)})^2$ /* **for residual calculation** */

/* STAGE–5 : calculate global residual */

$gsum = lsum + \sum(lsums$ from all processors$)$

Residual $= \sqrt{gsum}$

/* STAGE–6 : transition to next iteration */

$\mathbf{r}^{(t)} = \mathbf{r}^{(t+1)}$

$t = t + 1$

Figure 3.1. Parallel PageRank Algorithm

3.2. Time Complexity of Parallel PageRank

In this section we examine the time complexity of PR execution performed during an execution. We do not consider the time spent in reading the communication information since this information can be read from the disk once before a batched execution of PR executions for large number of personalization vectors.

There are two kinds of I/O made in our algorithm: disk I/O and communication I/O. The volume of disk I/O operations is $2n_i + c \times (2n_i + m_i)$ due to the reading of personalization vector, the reading of local page entries, and the writing of resultant PR vector. The volume of communication I/O is $b_i + f_i$ where b_i is the number of ranks received and f_i is the number of ranks sent. Under the assumption that the load distribution is done well and considering the linear relationship between n and m, n_i, m_i, b_i and f_i can be said to be proportional to n/p. Therefore, the overall I/O complexity is $O(n/p)$.

The number of divisions to calculate the ranks to flow to both local pages and remote pages are $f_i + l_i$, where l_i is the number of local links between the local pages. The number of additions used in rank calculations is $b_i + l_i$. Finally, the total number of floating point operations in dampening and residual calculations is $6n_i$. The total time complexity is therefore $O(n_i + b_i + f_i + l_i)$. Again, this complexity can be said to be $O(n/p)$.

In stage one, we have to lookup the processor ID for a page. The lookup operation is performed on the partition array which has $|V_i \bigcup V_{F_i}| = n_i + n_{F_i}$ pairs. We use binary search for this lookup operation. Since there are total of n_i processor lookups in stage one, the total time spent on processor lookups has $O(n_i \times \log(n_i + n_{F_i}))$ complexity which can be approximated by $O((n/p)\log(n/p))$.

We need to map global page IDs to local page array. This mapping is done by performing binary search on an array which holds the global IDs of the local pages. This operation is performed m_i times on stage one and b_i times on stage three. Since

E_i is the superset of B_i, the total time spent in ID lookups has $O(m_i \log n_i)$ complexity, which is again $O((n/p) \log (n/p))$ if the load is distributed well.

If it is guaranteed that we use simple partitioning(Section 3.3.3.1) all the time, the binary search operations mentioned above become unnecessary because, in this case, we can find the local indices by simple arithmetic. This reduces the time complexity to $O(n/p)$. However, making our algorithm depend on a single partitioning strategy is not a flexible approach. Besides, simple partitioning is not an optimum strategy. Therefore, we are likely to use a better strategy if we find one.

To sum up, the overall time complexity of our parallel PR algorithm is $O((n/p) \log(n/p))$.

3.2.1. Time Spent per Processor

In this section we give the formulation of the time spent by each processor at each iteration. As mentioned in Section 3.2, there are mainly six operations. These are disk I/O, processor lookups, global index lookups, arithmetic operations, and data exchange with other processors. The times spent on these operations are given, respectively, as follows:

$$T_1 = (2n_i + m_i) \times T_{\text{io}} \tag{3.1}$$

$$T_2 = n_i \log (n_i + n_{f_i}) \times T_{\text{pl}} \tag{3.2}$$

$$T_3 = m_i \log n_i \times T_{\text{il}} \tag{3.3}$$

$$T_4 = (6n_i + 2l_i + f_i + b_i) \times T_{\text{ar}} \tag{3.4}$$

$$T_5 = (p_s + p_r) \times T_{\text{lt}} + (f_i + b_i) \times T_{\text{tr}} \tag{3.5}$$

where T_{io} is the unit time spent in disk I/O, T_{pl} is the average unit time spent in processor lookup, T_{il} is the average unit time spent in global index lookup, T_{ar} is the average unit time spent in a floating point operation, T_{lt} is the message latency, and T_{tr} is the unit time to send and receive a floating point number. Here, p_s is the number

of processors to send ranks, and p_r is the number of processors to receive ranks. The total time spent by a processor is $T_{\text{tot}} = T_1 + T_2 + T_3 + T_4 + T_5$.

3.3. Implementation of Parallel PageRank

3.3.1. Input and Output

Each processor P_i uses four files as input: a graph file, a partition file, a communication data file, and a personalization file. After execution, each processor produces a file that contains the PR values of the local pages. Its size is $4 \times (|V_i|)$ bytes.

The graph file holds the local pages and their links. It is the distributed version of the graph file mentioned in Section 2.5.1 and has the same structure with that file. It has a size of $4 \times (3 + 2|V_i| + |E_i|)$ bytes. Similarly, the personalization file is the distributed version of that used in the serial version and its size is $4 \times |V_i|$ bytes.

The partition file holds the partition vector mentioned in Section 3.1. It contains sorted list of (page,processor) pairs for all local pages and remote pages that are linked by the local pages. It has a size of $2 \times (|V_i| + |V_{F_i}|)$. This file is supposed to be created initially with the graph file.

The communication data file holds the necessary information for data exchange between the other processors. This information consists of the following:

- The number of processors to which this processor will send ranks (**numS**)
- The list of processors to which this processor will send ranks (**dst**)
- The list of the number of ranks for each processor in **dst** (**lenS**)
- The number of processors from which this processor will receive ranks (**numR**)
- The list of processors from which this processor will receive ranks (**src**)
- The list of the number of ranks for each processor in **dst** (**lenR**)
- For each processor P_j in **src**, a list of local pages for which P_j sends ranks to this processor ($\mathbf{rlist}_j$)

- The number of local pages (**numP** $= |V_i| = n_i$)
- The list of local pages (**plist**)

Note that, the last two items are not related to communication. They are stored to prevent an unnecessary pass over the graph file in PR execution. The setup process of communication data file is outlined in Section 3.3.4. The file layout is depicted in Figure 3.2.

numP	plist[0]	...	plist[numP-1]
numS	dst[0]	...	dst[numS-1]
numS	lenS[0]	...	lenS[numS-1]
numR	src[0]	...	src[numR-1]
lenR[0]	rlist[0][0]	...	rlist[0][lenR[0]-1]
⋮	⋮	...	⋮
lenR[numR-1]	rlist[numR-1][0]	...	rlist[numR-1][lenR[numR-1]-1]

Figure 3.2. Communication file layout

Since all numbers are stored as 4–byte integers, the size of the communication file is given by the following equation:

$$s = 4 \times [\, 3 + 2 \times \mathbf{numS} + 2 \times \mathbf{numR} + n_i + b_i \,] \tag{3.6}$$

where b_i is the number of links from remote pages to the local ones, which is also equal to the number of ranks to be received from the other processors. Note that we do not store the second **numS** in Figure 3.2 actually. We put it in the figure for clarity.

In the experiments that we used simple partitioning mentioned in Section 3.3.3.1, we observed that **numS** = **numR** $= p - 1$, where p is the number of processors. Besides, in simple partitioning, we have $n_i = n/p$, where $n = |V|$. According to this, Equation 3.6 becomes

$$s = 4 \times \left[4p - 1 + \frac{n}{p} + b_i \right] \tag{3.7}$$

3.3.2. Memory Usage

Out parallel PR implementation uses the following data buffers:

- Buffers to hold the data in the communication file (Section 3.3.1)
- A list of buffers to send ranks to the other processors
- A buffer to hold inverted index of **dst** (Section 3.3.1)
- Local personalization vector read from the personalization file
- PR vector to use in the current iteration
- PR vector to use in the next iteration
- Two read–buffers mentioned in Section 2.5.2

The expression for the size of memory allocated for these buffers are as follows:

$$4 \times [5 + 3 \times \mathbf{numS} + 2 \times \mathbf{numR} + b_i + f_i + 4n_i + 2 \times d_{max}] \tag{3.8}$$

where f_i is the number of links from local pages to the remote ones, which is also equal to the number of ranks to be sent to the other processors. If we apply the same replacements in Section 3.3.1, Equation 3.8 becomes

$$4 \times \left[5p + \frac{4n}{p} + b_i + f_i + 2 \times d_{max}\right] \tag{3.9}$$

Per–processor memory usage of our serial and parallel PR implementations for a set of sample web graphs are depicted in Figure 3.3. In these runs, we did not allocate space for personalization vector since we used uniform distribution.

3.3.3. Partitioning the Web Graph for Parallel Execution

Load of a parallel program must be balanced among processors to achieve higher utilization of processing resources. In our problem, local computation time is determined by two load factors: the total number of forelinks of local pages and the total

number of backlinks of local pages.

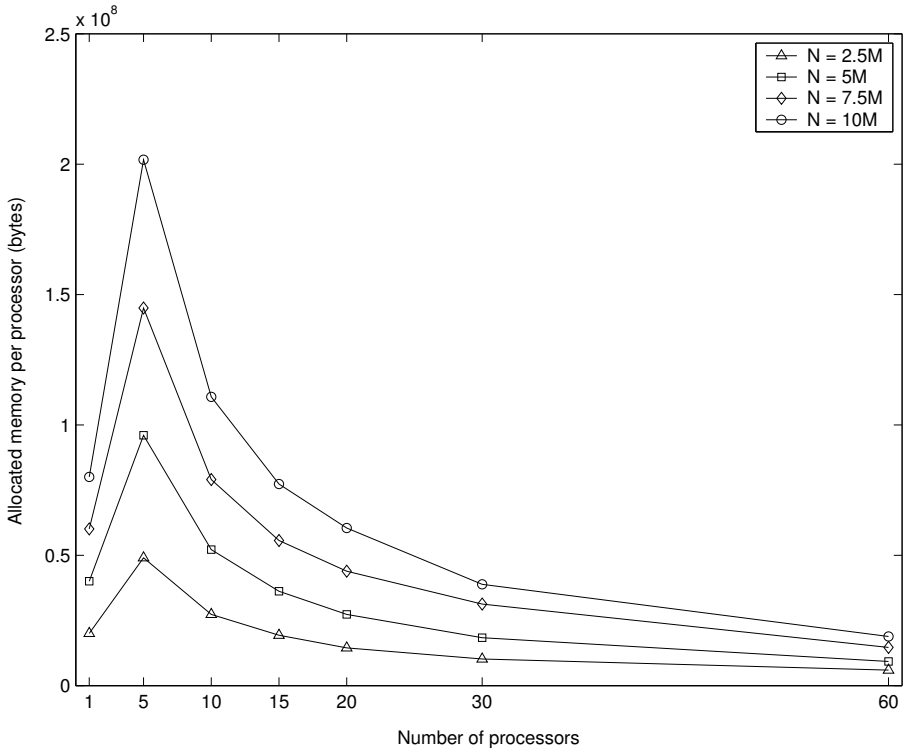


Figure 3.3. Memory usage of parallel PageRank implementation

3.3.3.1. Simple Partitioning. By simple partitioning, we mean the even distribution of pages and their adjacency lists to the processors. A processor P_i gets the pages with index j given that $i \times n_i \leq j < (i+1) \times n_i$ where $0 \leq i < p$. Here, the number of pages n_i on P_i is found by the following equation:

$$n_i = \left\{ \begin{array}{ll} \lfloor \frac{n}{p} \rfloor + 1 & \text{if } 0 \leq i < (n \bmod p) \\ \lfloor \frac{n}{p} \rfloor & \text{otherwise} \end{array} \right\} \tag{3.10}$$

According to our observations, this partitioning scheme yields a nice load balancing. This is mainly due to the uniform distribution of pages within the graph file. That is, we do not order the page entries in the file based on their out–degrees.

3.3.3.2. Partitioning using ParMETIS. At early phases, we considered using a graph partitioner to reduce the communication volume among the processors. We chose ParMetis [19] for this purpose. We implemented a partitioner using ParMetis library. However, two observations caused us to give up using ParMetis. The first one is the large memory requirement of PartKway function, which is the only partitioning function we used in ParMetis library. On ASMA (Section 5.3.1), we could partition up to 750 thousand–page web graphs on 20 machines with using nearly all the physical and the swap memory. Moreover, this is the case when we do not apply vertex and edge weighting in ParMetis. If we apply weighting, the memory requirement will be much larger. On the other hand, we can easily handle more than 10 million–page web graph in our parallel PR implementation.

The second observation is the load imbalance caused by k–way function. Unlike the simple partitioning scheme mentioned in Section 3.3.3.1, partitioning with ParMetis results in large differences in the number of pages and number of links on the processors. Although the communication volume is reduced by nearly 30 per cent, the difference in the sizes of local data to be handled by processors causes large imbalance in the processing times. In this case, the processor that must handle the largest data determines the overall execution time, since it spends longer time than the others.

Due the reasons explained above, we gave up considering ParMetis for our problem. Note that, we could not have chance to experiment partitioning with ParMetis on MYRI (Section 5.3.1) due to the non–technical reasons.

3.3.4. Establishing Communication Information

Each processor needs a file that consists of certain information to exchange rank values with other processors. This information is described in Section 3.3.1. In this section, we describe the process to setup this information. This setup process is performed only once for each graph file. Figure 3.4 outlines the procedure for this process.

3.3.5. Parallel Execution with MPI

We use non–blocking send and receive operations of MPI library to perform exchange operations in the second stage of our parallel PR algorithm (Figure 3.1). The asynchronous behavior of these operations simplifies the rank exchange operation between the processors.

```
/* determining the send–buffer sizes and the recipient processors */
for all u ∈ V_i
    for all v ∈ F_u
        if P_i ≠ P_id(v)            /* v is on a remote processor */
            if P_id(v) is not marked
                mark P_id(v) as recipient
                save P_id(v) in dst list
                increment numS
            increment lenS[P_id(v)]
/* exploration of sender processors and the receive–buffer sized */
sum all the recipient marks of the processors (AllReduce)
    numR is the total number of marks for this processor
for all P_j in dst
    send lenS[P_j] to P_j
for numR times
    for all senders P_k
        receive lenR[P_k] from P_k
        save P_k in src
/* determining the the local pages that will send ranks to remote pages*/
for all u ∈ V_i
    for all v ∈ F_u
        if P_i ≠ P_id(v)            /* v is on a remote processor */
            save u in send–buffer slist[] for P_id(v)
/* exploration of the remote pages that will send ranks to local pages */
for all receivers P_j in dst
    send buffer slist[P_j] to P_j
for all senders P_k in src
    receive buffer rlist[P_k] from P_k
dump numS, dst[],lenS[], numR, src[], lenR[], rlist[][] to file
dump n_i as numP and V_i as plist[] to file
```

Figure 3.4. Setup procedure for communication information

4. SAMPLE WEB GRAPH GENERATION

4.1. Introduction

The workload for our parallel PR algorithm is web graphs. The number of pages and the number links are our workload parameters. We know that the structure of a web graph has certain characteristics that affects the performance and scalability. A web graph is not a random graph. It has a certain structure. Therefore, we have to choose our workload carefully and accordingly.

In order to obtain sample web graphs, we implemented a web graph generator. In the following subsections, we mention about the structural properties of the web graphs that interest us, the reason why we chose to implement a generator, the outline of our generation algorithm and some sample outputs of our program.

4.2. Structure of a Web Graph

Barabási *et al.* [20, 21] showed that the distribution of in– and out–degrees of web pages follows a power law. That is, the probability of a page has in–degree (or out–degree) d is inversely proportional to d^{β} for some $\beta > 1$. Barabási *et al.* [22] observed the power law exponent 2.1 for in–degrees and 2.45 for out–degrees on a relatively small sample(325,729 pages and 1,469,680 links). Broder *et al.* [23] confirmed the power law on node degrees by analyzing a large (over 200 million pages and 1.5 billion links) crawl of Altavista Search Engine [24]. They observed the power law exponent 2.1 for in–degrees and 2.72 for out–degrees. In fact, most of the real self evolving natural networks such as genetic networks, WWW, Internet topology, social networks, scientific citation index and movie actor relationships exhibits power law in node degrees with exponents between two and three [20, 25, 26]. These networks are said to have stationary scale–free distributions which implies that these networks have robust and self-organizing development mechanisms [20].

A good theoretical analysis about the structure of the Web and similar networks is given by Dorogovtsev *et al.* [27, 28].

Another interesting observation about the number of links in a web graph is that average number of links on a page is constant. Shiode and Batty [29] observed 3.92 as an average number of links per web page on over 75 million pages. We also observe constant average number of links in the output graphs of our generator (Section 4.5).

Besides those mentioned above, there are many works on modelling the WWW in the literature [30, 31, 32, 33, 34, 35]

4.3. Why to Use a Generator?

At the beginning of our work, we intended to crawl the Web to construct a sample web graph. After some initial work, however, it proved to be impractical to crawl the Web to obtain a representative web graph. First of all, our bandwidth is not enough to crawl the Web. Second, obtaining a representative web graph from the Web is quite hard. After crawling for days, there is a big risk for the collected data to be useless. Third, crawling the pages and analyzing the collected pages to produce web graph becomes overwhelming if you need different sizes of web graphs. Lastly, after examining the literature about modelling the WWW, we saw that the Web has a certain structure (Section 4.2) and it is possible to produce web graphs as workload for our parallel PR implementation. Due to these reasons, we concluded that having a web graph generator is much more flexible and time saving.

The AS-level topology of Internet also shows power laws [36]. There are several tools [37, 38] to generate sample Internet topologies that produces graphs similar to web graphs in structure. These tools, however, generates undirected graphs, which is not appropriate for our purpose.

There are several models proposed to generate web graphs that exhibit power laws [21, 31]. These models are quite complex and obtaining system parameters to

produce a web graphs is not trivial.

Since all we want to do is to generate sample web graphs with different number of pages, we decided to implement our own generator which produces graphs according to observed power law exponentials for in– and out–degrees. This means, our generator does not consider the dynamics of the Web. Its only purpose is to generate representative workload for our parallel code.

Far after we implemented the generator, we noticed an important advantage of having a generator when we need to run our code on another cluster with the same data set. Due to time restrictions, it would simply be impossible for us to send the graphs to the other site if we obtain our samples by crawling. We did not think of this possibility when we were considering using a generator. It would certainly speed up the decision if we did.

4.4. Generation Algorithm

Our generator takes three inputs: the number of pages(n), power law exponent for in–degrees (β_{in}), and power law exponent for out–degrees (β_{out}). We used an implementation of a high quality pseudo random number generator [39] to generate node degrees and links among pages.

Figure 4.1 illustrates the pseudocode of our generation algorithm. Here, α_{in} and α_{out} is the constants such that the following probability sum is satisfied:

$$\sum_{d=1}^{n-1} \frac{\alpha_{in/out}}{d^{\beta_{in/out}}} = 1 \tag{4.1}$$

where d is (in/out)–degree and the expression $\alpha_{in/out}/d^{\beta}_{in/out}$ is the probability that a page has an (in/out)–degree d. Note that, since α_{in} and α_{out} is dependent to n, we calculate them at the beginning of each generation.

α_{in} = calculateAlpha(β_{in}, d, n)

α_{out} = calculateAlpha(β_{out}, d, n)

/* generate in–degrees */

$S_{in} = 0$ for all pages $u = 0$ to $n - 1$

    D_u^{in} = chooseRandomIndegree(α_{in}, β_{in})

    $S_{in} = S_{in} + D_u^{in}$

/* generate out–degrees */

$S_{out} = 0$

for all pages $u = 0$ to $n - 1$

    D_u^{out} = chooseRandomOutdegree($\alpha_{out}, \beta_{out}$)

    $S_{out} = S_{out} + D_u^{out}$

/* calculate cumulative distribution function of current D^{out} */

cdf_{out} = calculateCDF(D^{out})

/* generate pseudo–links on pages to equalize S_{in} and S_{out} */

for all pages $u = 0$ to $n - 1$

    $D_u^{out} = 0$

for S_{in} times

    u = chooseRandomPage(cdf_{out})

    $D_u^{out} = D_u^{out} + 1$

/* make all zero out–degrees one */

added = 0

for all pages $u = 0$ to $n - 1$

    if $D_u^{out} == 0$ **/* zero out–degree is not allowed */**

        $D_u^{out} = 1$

        added = added + 1

$S_{out} = S_{out}$ + added **/* adjust S_{out} */**

/* Real generation of links using in–degrees as popularity */

cdf_{in} = calculateCDF(D^{in})

for all pages $u = 0$ to $n - 1$

    adjlist = Ø

    for $d = 0$ to D_u^{out}

        v = chooseRandomPage(cdf_{in})

        put v into adjlist

    sort(adjlist)

    write page entry with (u,D_u^{out},adjlist) to the file

Figure 4.1. Web graph generation algorithm

4.5. Sample Graphs

In this section, we present two sample web graphs generated by our graph generator. Both samples are generated with $\beta_{in} = 2.1$ and $\beta_{out} = 2.7$.

The first sample graph G_1 has five million pages and 25,927,066 links. In– and out–degree histograms of G_1 are illustrated in Figure 4.2 and Figure 4.3, respectively. Our second sample G_2 has ten million pages and 53,435,470 links. In– and out–degree histograms of G_2 are illustrated in Figure 4.4 and Figure 4.5, respectively.

The deviations in Figures 4.5 and 4.3 for pages with small out–degrees is probably caused by the CDF we setup from the initial out–degree distribution. During the re–generation, the CDF could not represent the pages with very small out–degrees and pages are assigned to larger out–degrees. We think this unrepresentative nature comes from the relatively small out–degrees generated by our code. On the other hand, the resultant in–degrees shows a nicer fit. This is probably due to the large in–degrees which leads to a representative CDF for pages with small out–degrees. However, the deviation in out–degree distributions does not affect the overall structure other than a slight increase in connectivity. Therefore, our generator can be said to produce representative load for our algorithm.

4.5.1. Number of Pages vs. Number of Links

Figure 4.6 demonstrates the linear relationship between the number of pages and the total number of links in generated web graphs with different sizes. We generate ten different graphs for ten different sizes and plot the means and the standart deviations. The numerical values is presented in Table 4.1. This shows that our generator produces web graphs that is consistent with the structure of real web graphs that are mentioned in Section 4.2.

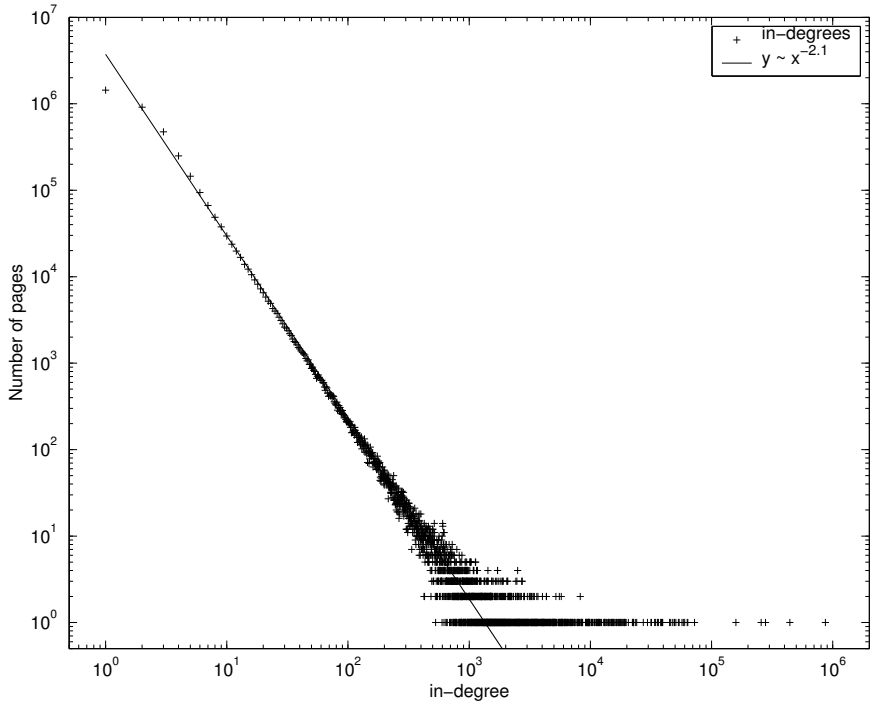


Figure 4.2. In–degree distribution of the sample web graph G_1 (n = 5M)

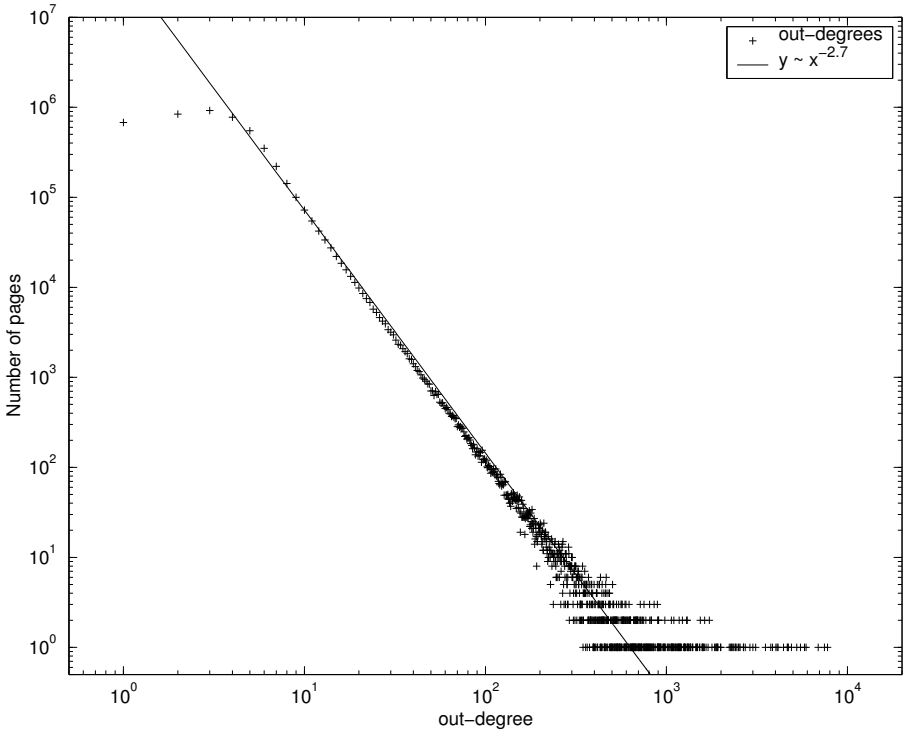


Figure 4.3. Out–degree distribution of the sample web graph G_1 (n = 5M)

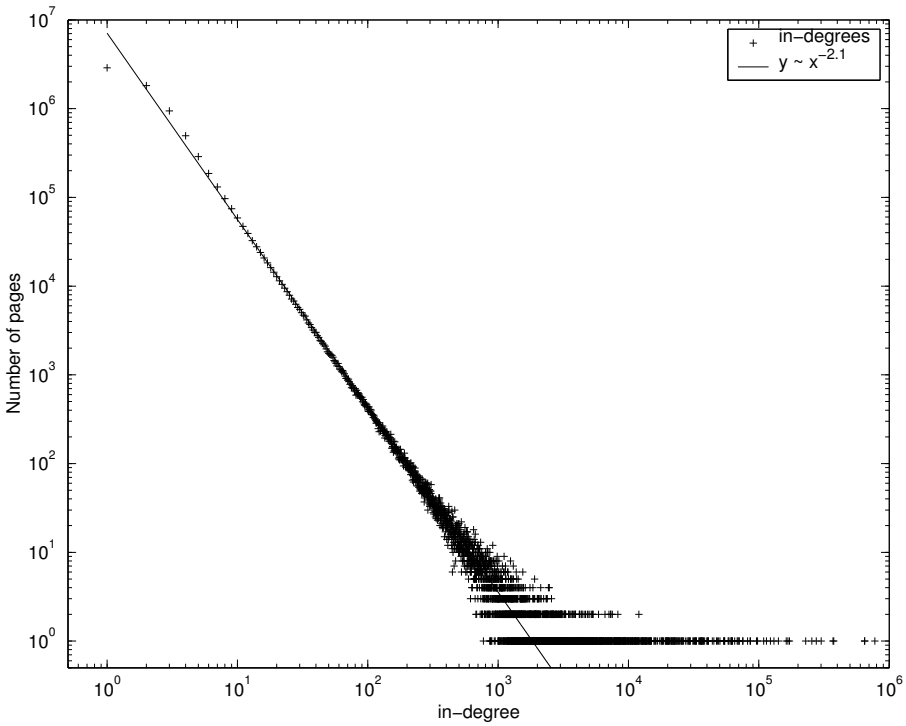


Figure 4.4. In–degree distribution of the sample web graph G_2 (n = 10M)

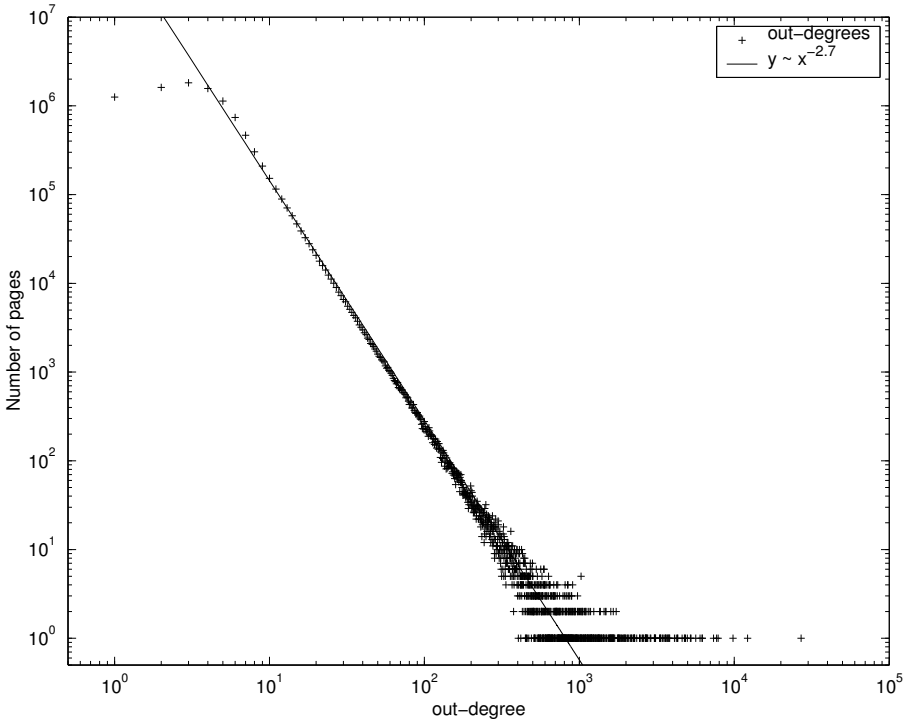


Figure 4.5. Out–degree distribution of the sample web graph G_2 (n = 10M)

Table 4.1. Number of links in some of the generated sample graphs

n	**Mean(e)**	**StdDev(e)**
250K	1303488.5	152859.2
500K	2636821.9	244474.7
750K	3882198.6	317979.6
1M	5168221.0	388518.1
2.5M	13506612.1	1377271.7
5M	26447850.6	1435122.6
7.5M	40947934.8	2517087.3
10M	53661499.9	3244825.4
20M	112201818.0	7015744.8
40M	225009693.5	10324001.9

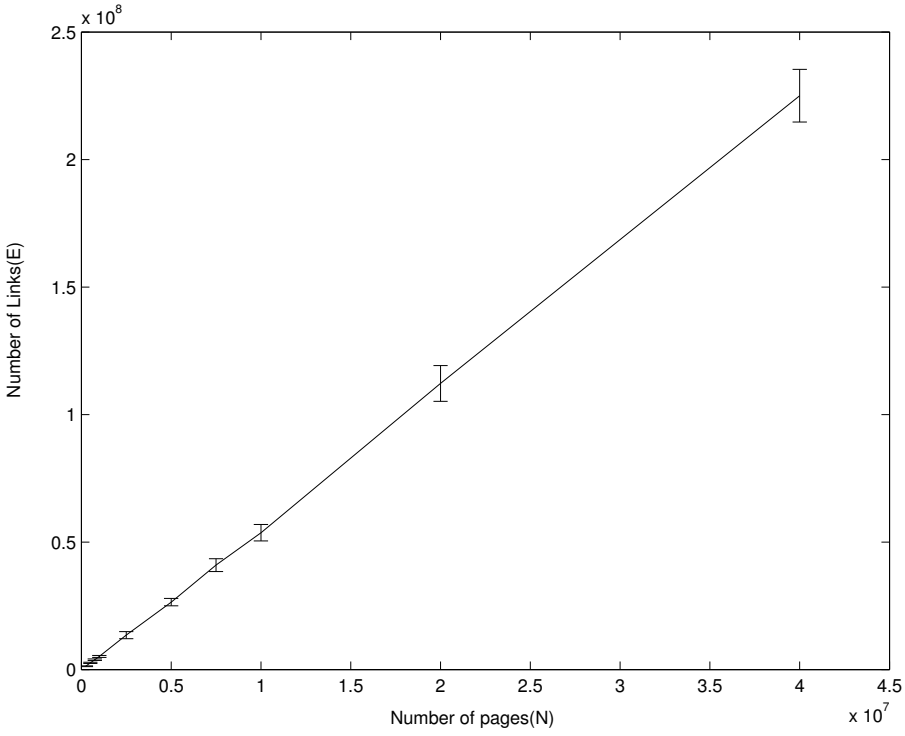


Figure 4.6. Number of links in some of the generated sample graphs

5. EXPERIMENTS AND RESULTS

In this section, we define our goal and metrics that we consider when doing our experiments. We also define the system and workload parameters that may affect our metrics. Then, we identify which parameters to fix and which to change in our experiments. Finally, we present the results.

5.1. Goals

In the experiments, our aim is to examine the performance and scalability of our parallel PR implementation. We use our serial PR implementation as a reference in order to evaluate our parallel program. Since the performance of a parallel code strictly dependent on the architecture it runs, we will evaluate our code on two different distributed memory parallel computers two make healthy observations.

5.2. Metrics

Since we will evaluate our parallel code in terms of performance and scalability, we have mainly two kinds of metrics: performance metrics and scalability metrics.

We define our performance metrics as follows:

- Total time spent in the PR iterations
- Speedup gained over the serial program by running parallel
- Efficiency of the parallel code

In order to determine the scalability, we observe the constancy in efficiency when we increase the size of the graphs as well as the number of processors.

5.3. Parameters

5.3.1. System Parameters

We define 15 system parameters that may affect our results:

- OS and kernel version
- C Compiler
- Compiler options
- MPI library used
- Network switch environment
- Network Interface Card(NIC) properties
- Number of processors used
- Per–node memory
- Per–node CPU
- Node disk access type
- Disk read/write bandwidth
- Loads on the processors during experiments
- Loads on the network during the experiments
- Power law exponents for workload generation (β_{in}, β_{out})
- Minimum vector distance to determine convergence (Δ)

We tested our code on two Linux PC clusters. One of the clusters, ASMA, is located in the Department of Computer Engineering in Boğaziçi University. The other that we call MYRI is located in USA. The values of the system parameters for these clusters are shown in Table 5.1.

5.3.2. Workload Parameters

We define three workload parameters. These are the number of links in the web graph (m), the number of pages in the web graph(n), and the usage of Personalization vector.

Table 5.1. System parameter values for two clusters

System Parameter	**ASMA**	**MYRI**
OS	RedHat Linux 6.1	RedHat Linux 7.2
Linux kernel	2.2.14	2.4.12 SMP
C Compiler	egcs 2.91.66	gcc 3.0.2
Compiler options used	optimization level 2	optimization level 2
MPI library used	MPICH 1.2.1 over TCP	MPICH–GM 1.2.1
Network Switch Environment	HP4000M Fast Ethernet Switch	M3–E64 Myrinet Switch
Network Interface Card(NIC)	Intel EtherExpress 100	Myrinet M3S–PCI64B
Raw network bandwidth	100 Mb/s	1.92 Gb/s
MPI bandwidth	60 Mb/s	0.5–1 Gb/s
MPI latency	120 μs	10 μs
Number of processors used	5, 10, 15, 20	5, 10, 15, 20, 30, 60
Per–node memory	128 MB	1000 MB
Per–node CPU	PII–400 Mhz	Dual PIII–1000 Mhz
Node disk access type	local	central over NFS
Disk bandwidth(read/write)	(20 MB/s) / (6 MB/s)	not measured
Processor load	idle	idle
Network load	idle	idle
PL exponents(β_{in}/β_{out})	2.1/2.7	2.1/2.7
Convergence criteria(Δ)	0.0001	0.0001
Dampening factor(D)	0.85	0.85

We used our generator (Chapter 4) to produce workload for our experiments. The number of links in a generated graph depends on the number of pages and power law exponents. Since we fixed PL exponents, we chose different values only for the number of pages. The graphs used in the experiments are shown in Table 5.2.

We used the same personalization vector in all experiments. This is the uniform vector $\mathbf{p}$ where $p_u = 1/n$ for all $u \in V$. Due to this uniformity, we did not read the values from a personalization file. Instead, we used the value directly. This does not have an effect on the PR execution time since the personalization vector is read from the disk before the PR iterations begin.

Table 5.2. The generated graphs used in the experiments

n	m	d_{max}
250K	1,378,610	3,067
500K	2,549,871	11,337
1.25M	5,895,383	498,625
2.5M	12,748,989	18,785
3.75M	19,795,627	11,021
5M	25,927,066	7,764
7.5M	37,250,620	41,792
10M	53,435,470	27,007

5.4. Experimental Design

For performance measurements, we used the graphs with n={2.5M, 5M, 7.5M, 10M} pages. on both ASMA and MYRI. We run the tests on p={1, 5, 10, 15, 20} processors of ASMA and p={1, 5, 10, 15, 30, 60} processors of MYRI.

For scalability measurements, we run experiments for k=250K, 500K pages per processor on p={1, 5, 10, 15, 20} processors. This corresponds to the following (n, p) pairs: (250K, 1), (500K, 1), (1.25M, 5), (2.5M, 5), (2.5M, 10), (3.75M, 15), (5M, 10), (5M, 20), (7.5M, 15), (7.5M, 30), and (10M, 20).

Each test is repeated five times on ASMA and two times on MYRI. The presented timing results are the averages of these repetitions. Note that we could not find a chance to make more repetitions on MYRI due to time and resource restrictions.

5.5. Results

5.5.1. MYRI Results

5.5.1.1. Run Time. The PR execution times on MYRI are presented in Table 5.3. We also give two views of this table in Figures 5.1 and 5.2. The Figure 5.1 emphasizes the

speedup in execution time while the number of processors increases. The computational overhead due to parallelization can be seen in the sharp increase from single–processor to five–processor execution in all sizes.

The Figure 5.2 shows the increase rate in execution time on different number of processors when the problem size increases. Since the time complexity is $O((n/p)\log(n/p))$, the increase rate is smaller when we use more processors.

Table 5.3. PR execution times on MYRI (sec)

p	n			
	2.5M	**5M**	**7.5M**	**10M**
1	30.57	59.62	87.58	125.56
5	49.63	116.37	165.69	246.55
10	23.40	57.74	74.68	111.84
15	14.53	33.95	47.13	69.83
30	6.63	16.70	22.71	33.46
60	3.59	9.65	12.30	17.70

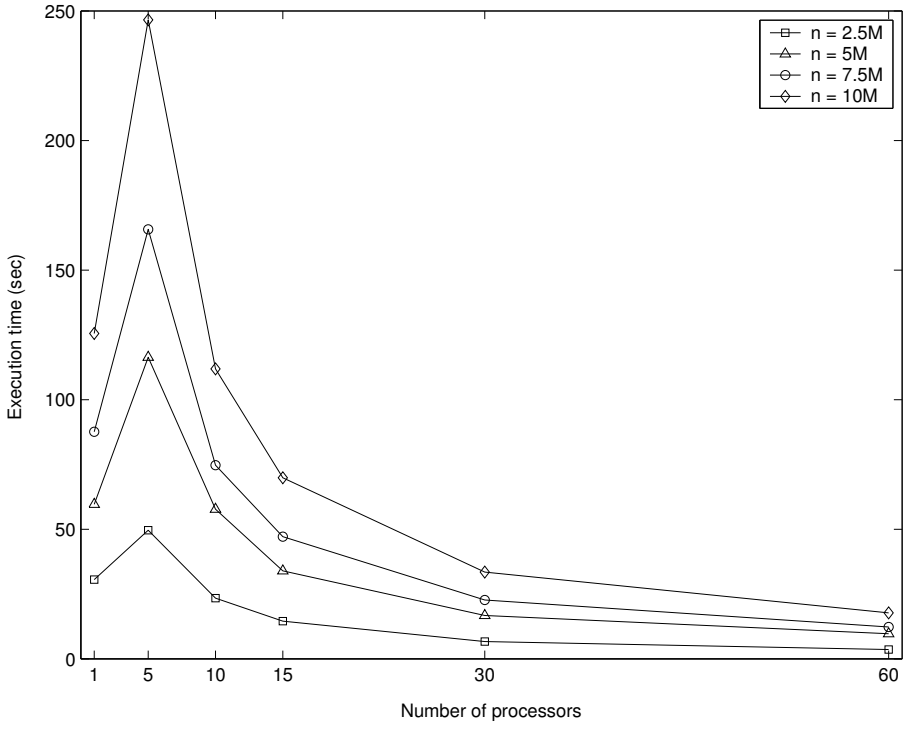


Figure 5.1. PR execution times on MYRI

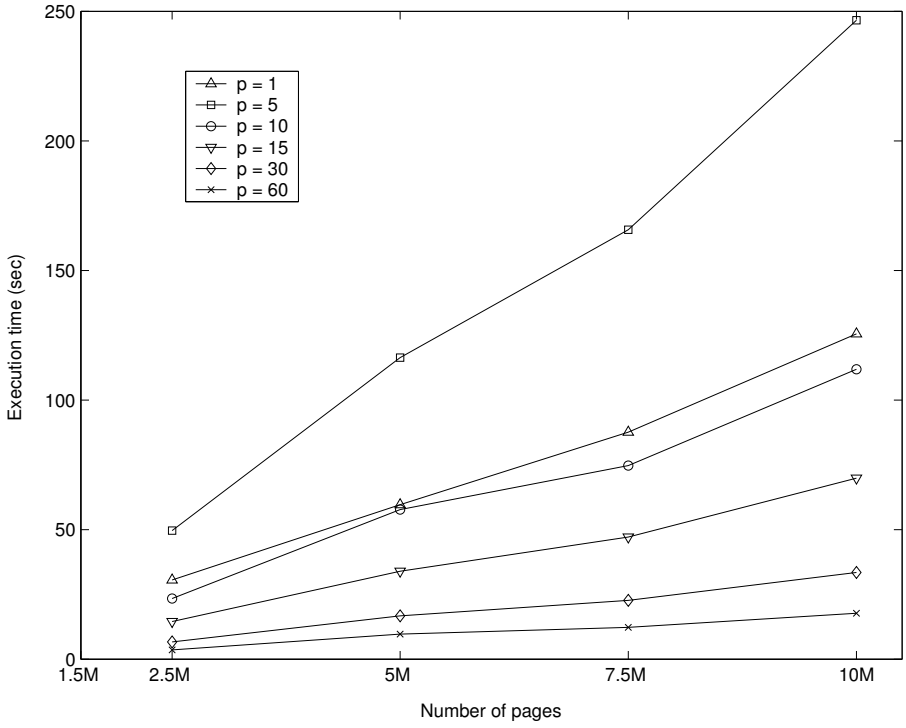


Figure 5.2. Number of Pages vs. PR execution times on MYRI

5.5.1.2. Speedup. The speedups observed in PR execution times on MYRI are presented in Table 5.4 and in Figure 5.3. As can be seen in the figure, speedup increases with the number of processors. We observed maximum speedup of 8.51 for 2.5M pages on 60 processors. The low speedup values are due to the overhead caused by parallelization.

Another observation here is that the speedup values for 5M pages are less than or equal to those for 7.5M and 10M pages. Normally, the speedup drops as the problem size increases while the number of processors is kept constant. This is because the serial execution time increases at slower rate than parallel one when the number of processors is fixed. The exception for 5M–page case is due to the load imbalance in the backlink and local link distributions. As a specific example, when 60 processors are used, P_{49} is extremely loaded with backlinks and local links compared to the other processors (Table A.13). Since the execution time is determined by the longest time spent by the individual processors, P_{49} determines the execution time in this case. Similarly, P_{49} in 30–processor case, P_{12} in 15–processor case, and P_8 in 10–processor case can be

considered the points of imbalance.

Table 5.4. Speedups in PR execution times on MYRI

p	n			
	2.5M	**5M**	**7.5M**	**10M**
1	1.00	1.00	1.00	1.00
5	0.62	0.51	0.53	0.51
10	1.31	1.03	1.17	1.12
15	2.10	1.76	1.86	1.80
30	4.61	3.57	3.86	3.75
60	8.51	6.18	7.12	7.10

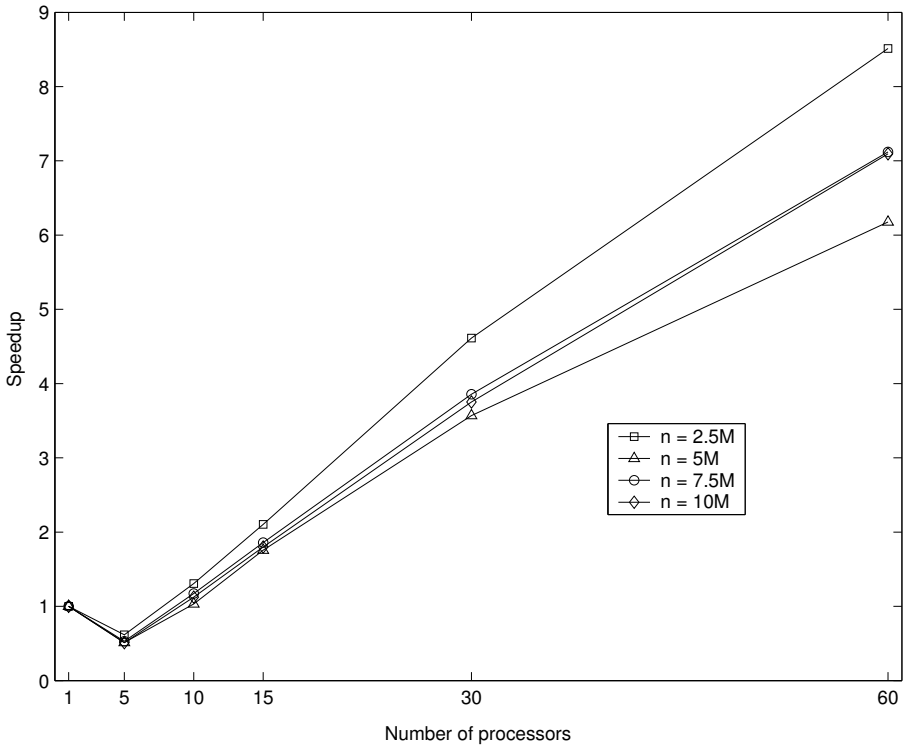


Figure 5.3. Speedups in PR execution times on MYRI

5.5.1.3. Efficiency. The efficiency of our parallel PR implementation is shown in Table 5.5 and in Figure 5.4. The special case for n=5M explained in Section 5.5.1.2 can also be observed here.

Since the efficiency is the speedup divided by the number of processors, the efficiency drops when the problem size increases and the number of processors is fixed.

Table 5.5. Efficiency of parallel PR on MYRI (per cent)

p	n			
	2.5M	**5M**	**7.5M**	**10M**
1	100	100	100	100
5	12.32	10.25	10.57	10.19
10	13.06	10.33	11.73	11.23
15	14.03	11.71	12.39	11.99
30	15.37	11.90	12.86	12.51
60	14.19	10.29	11.87	11.83

5.5.1.4. Scalability. As can be seen in Figure 5.5, the efficiency remains almost constant at a level between 12 and 13 per cent, which shows that our code is scalable to at least 60 processors for 250K and 500K pages per processor on MYRI system. The numerical data for this plot can be found in Table 5.10.

5.5.1.5. Time Spent per Processor. In Section 3.2.1, we give the general formulation of the time spent at an iteration on a processor. Here, we give the formulations according to the observations on the experimental data and make predictions for the optimum number of processors for MYRI system. We observed the following relations on the partitioned graphs in our experiments (Appendix A).

$$m = kn \tag{5.1}$$

$$n_i = n/p \tag{5.2}$$

$$m_i = m/p = kn/p \tag{5.3}$$

$$l_i = m_i/p = kn/p^2 \tag{5.4}$$

$$f_i = b_i = (p-1) \times l_i = (p-1)kn/p^2 \tag{5.5}$$

$$n_{l_i} \approx 2n_i/p = 2n/p^2 \tag{5.6}$$

$$n_{f_i} = (p-1) \times n_{l_i} = 2n(p-1)/p^2 \tag{5.7}$$

For the generated graphs, we observed $k \approx 5.3$(Table 4.1 and Table 5.2). If we

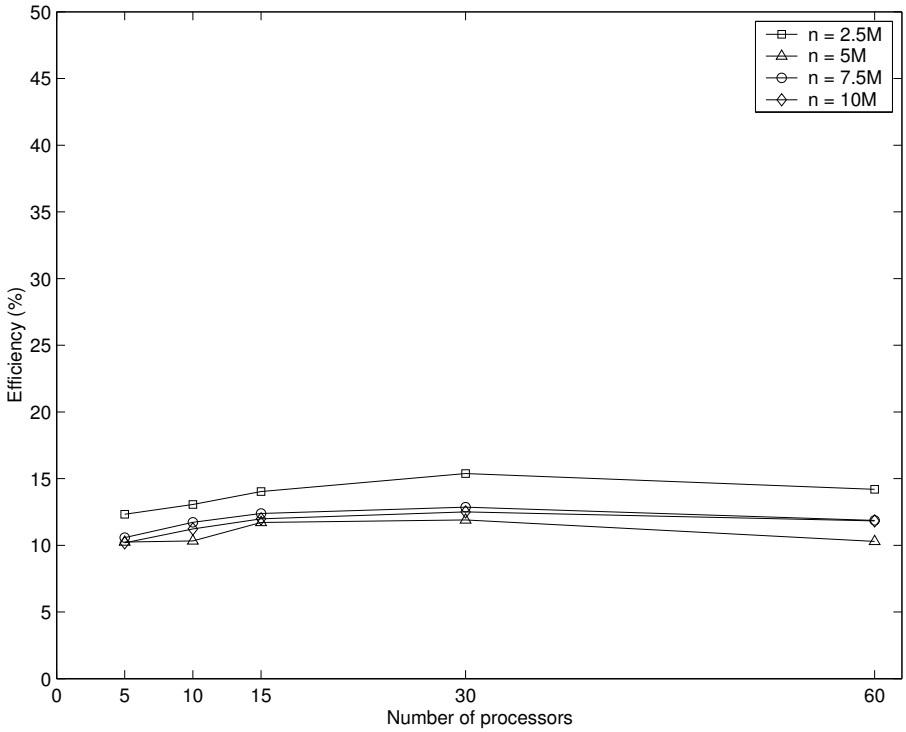


Figure 5.4. Efficiency of parallel PR on MYRI

substitute these expressions into the Equations 3.1–3.5, we get the following equations:

$$T_1 = \left(\frac{7.3n}{p}\right) \times T_{\text{io}} \tag{5.8}$$

$$T_2 = \frac{n}{p} \log \left(\frac{n}{p} + \frac{2n(p-1)}{p^2}\right) \times T_{\text{pl}} \tag{5.9}$$

$$T_3 = \frac{5.3n}{p} \log \frac{n}{p} \times T_{\text{il}} \tag{5.10}$$

$$T_4 = \frac{16.6n}{p} \times T_{\text{ar}} \tag{5.11}$$

$$T_5 = 2(p-1) \times T_{\text{lt}} + \frac{10.6n(p-1)}{p^2} \times T_{\text{tr}} \tag{5.12}$$

We consider the time spent in local computations(i.e. T_1, T_2, T_3, and T_4) and the time spent in communications (i.e. T_5) separately. We denote the total local computation time with T_{local} and the communication time with T_{comm}. Since we do not know the constants T_{io}, T_{pl}, T_{il}, T_{ar}, T_{lt}, and T_{tr} for MYRI, we use least squares approximation to fit the experimental values into a formulation. We use the timing data in Table 5.6.

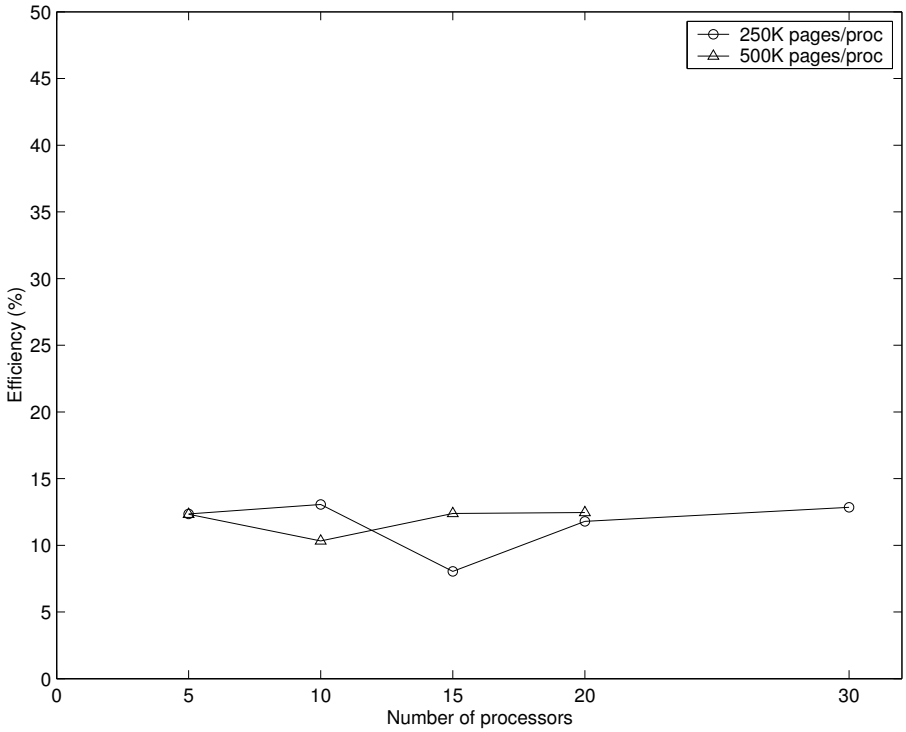


Figure 5.5. Scalability of parallel PR on MYRI

Using the (n, p) pairs and Equations 5.8–5.12, we establish two over–determined equation systems for T_{local} and T_{comm}. We then calculate the constants by least squares approximation. For MYRI, The resultant values of T_{io}, T_{pl}, T_{il}, T_{ar}, T_{lt}, and T_{tr} are 0, 4.75×10^{-6}, -5.87×10^{-7}, -1.69×10^{-6}, 4.54×10^{-4}, and 3.12×10^{-8}, respectively. The actual values of these constants cannot be zero or negative. However, these values are the results of an approximation and are used to predict the non–experimental timing values. As an example, we look for the optimum number of processors to minimize the time to process two billion pages. As can be seen in Figure 5.6, the processing time per iteration is minimized at 8.696 seconds with approximately 5280 processors.

Myrinet is a high quality network infrastructure. Therefore this prediction may be a good approximation for a real life deployment. However, our extrapolation must be considered with caution since the number of processors that we use is quite small compared to the number of processors predicted for optimum execution. Currently, Myrinet supports up to 128 ports to connect the machines. In order to connect 5280 dual processor machines, we need 43 switches (42 to connect the machines and one

to connect the switches). Intra–switch communication will increase the message latency and reduces the non–blocking communication capability of the overall system. Therefore, our prediction must be taken as an optimistic approximation.

Table 5.6. Average times spent by each processor on MYRI (sec/iteration)

n	p	T_{comm}	T_{local}
1.25M	5	0.318	1.953
2.5M	5	0.078	4.834
2.5M	10	0.103	2.167
2.5M	15	0.047	1.335
2.5M	30	0.030	0.575
2.5M	60	0.028	0.271
3.75M	15	0.557	2.754
5M	5	0.172	11.236
5M	10	0.418	5.072
5M	15	0.074	3.133
5M	20	0.093	2.232
5M	30	0.075	1.382
5M	60	0.056	0.675
7.5M	5	0.273	17.897
7.5M	10	0.153	7.976
7.5M	15	0.115	4.973
7.5M	30	0.096	2.206
7.5M	60	0.081	1.115
10M	5	0.539	26.368
10M	10	0.443	11.740
10M	15	0.169	7.299
10M	20	0.133	5.213
10M	30	0.114	3.260
10M	60	0.105	1.699

5.5.2. ASMA Results

5.5.2.1. Run Time. The PR execution times on ASMA are presented in Table 5.7 and in Figure 5.7. The bold numbers are from the tests where swap memory is used due to insufficient physical memory.

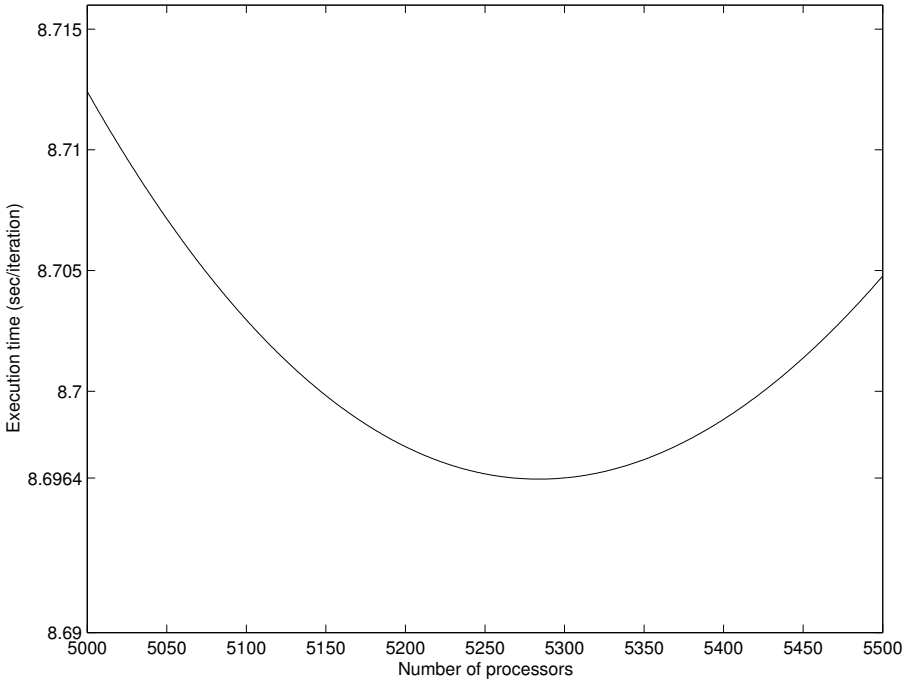


Figure 5.6. Estimation for the optimum number of processors to process two billion pages on MYRI

Note the anomaly observed in the serial execution times (shown in italics in Figure 5.7). Since the execution of serial PR is O(n), we expect linear increase in these times as we observed in MYRI. When we analyzed this situation, we found that after a certain number of I/O operations the system spends more time on read operations than expected. This should be a problem with read() system call of Linux Kernel 2.2.14. We did not observe this behavior on Kernel 2.4.

Table 5.7. PR execution times on ASMA (sec)

p	n			
	2.5M	**5M**	**7.5M**	**10M**
1	53.38	*546.30*	*737.98*	*1044.57*
5	129.24	341.97	**827.66**	**2015.39**
10	65.41	145.73	195.35	**351.02**
15	43.45	99.22	130.06	191.61
20	39.43	75.09	96.29	147.11

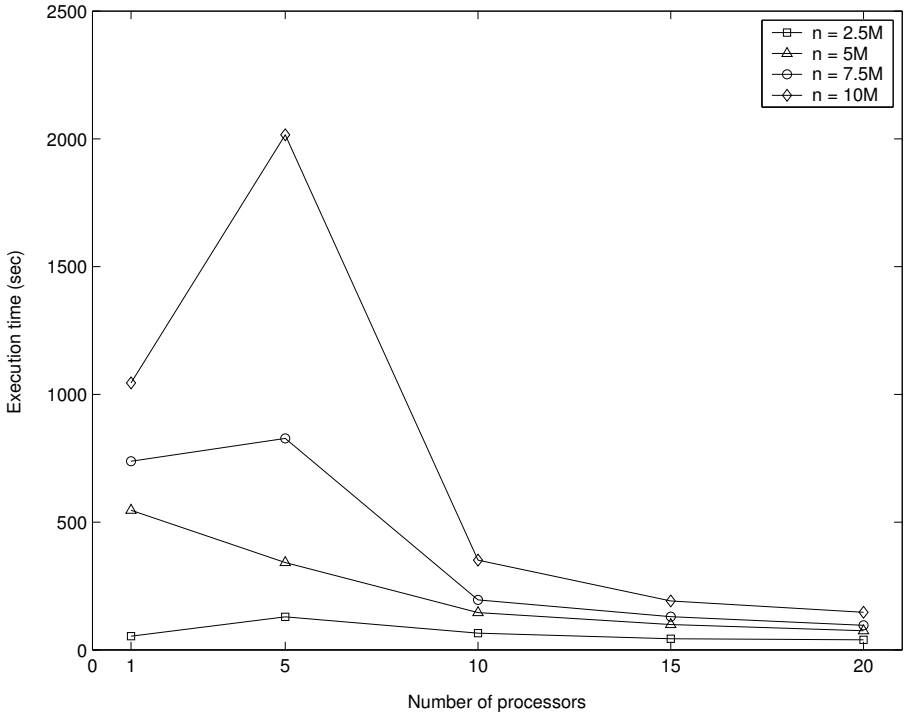


Figure 5.7. PR execution times on ASMA

5.5.2.2. Speedup. Due to the anomaly observed in serial PR execution, the speedup values are above normal. We supply the speedup results on ASMA for the sake of completeness. These results are given in Table 5.8 and in Figure 5.10.

Table 5.8. Speedups in PR execution times on ASMA

p	n			
	2.5M	**5M**	**7.5M**	**10M**
1	1.00	1.00	1.00	1.00
5	0.41	1.60	0.89	0.52
10	0.82	3.75	3.78	2.98
15	1.23	5.51	5.67	5.45
20	1.35	7.28	7.66	7.10

5.5.2.3. Efficiency. Since efficiency is a function of speedup, the effect of anomaly mentioned in Section 5.5.2.1 can be seen in the high percentages for n = 5M,7.5M,10M.

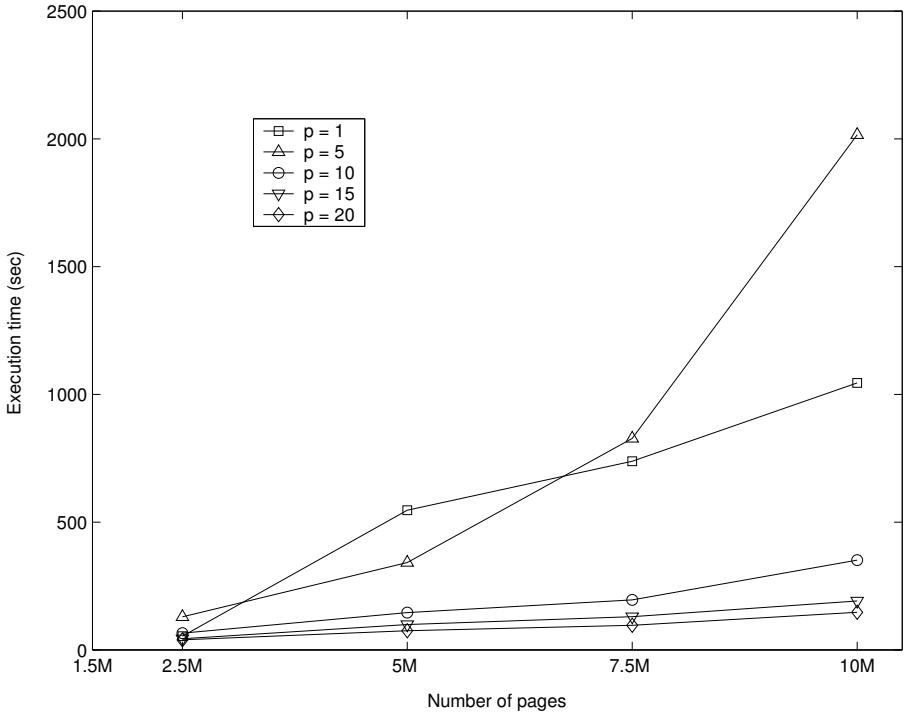


Figure 5.8. Number of Pages vs. PR execution times on ASMA

5.5.2.4. Scalability. The changes in efficiency while keeping the per–node load constant are illustrated in Figure 5.11. The numerical data for this plot can be found in Table 5.10.

Figure 5.11, the serial execution anomaly shows itself as the sharp increase in efficiencies. However, we can see that there are two regions in where the efficiency tends to remain constant. The first region is where the anomaly is not seen, that is where the n is smaller than or equal to 2.5M. The second region is where the anomaly occurs. However, since the times in serial execution increases linearly, which is consistent with the algorithmic complexity, we are also observing scalability in this region.

Based on our observations, we can conclude that our parallel code is scalable to at least 20 processors with 250K and 500K pages per processor on ASMA cluster.

5.5.2.5. Time Spent per Processor. We apply the same methodology as in Section 5.5.1.5 in order to approximate the time spent by each processor at a PageRank itera-

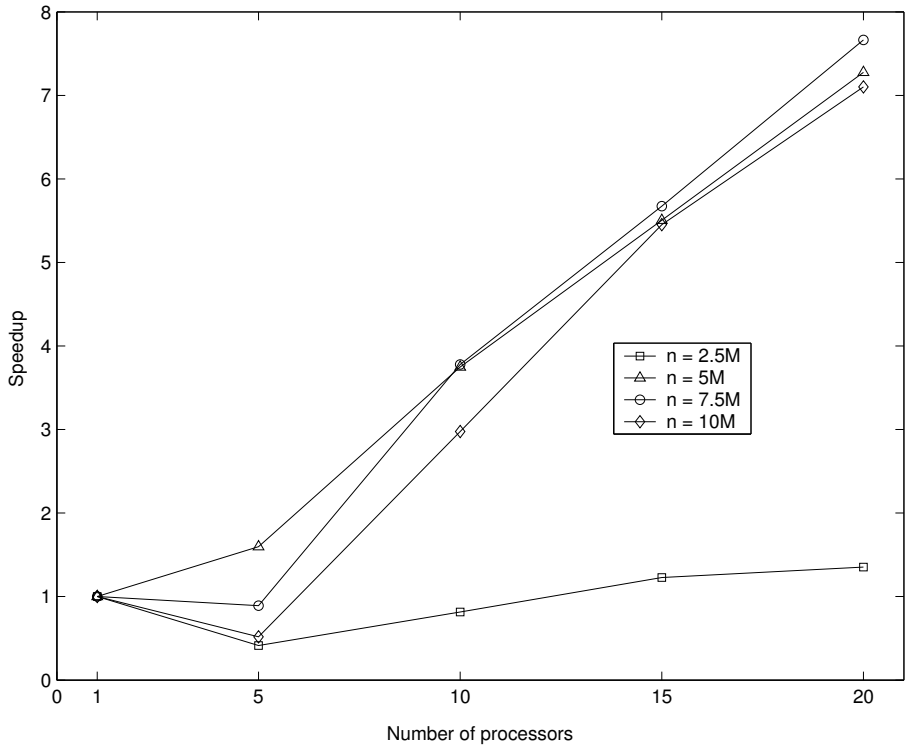


Figure 5.9. Speedups in PR execution times on ASMA

tion. We use Table 5.12 for least squares approximation. We ignored the experiments for (n=7.5M, p=5) and (n=10M, p=5) because of the significant deviation in execution time that is caused by using swap memory. The approximated values of T_{io}, T_{pl}, T_{il}, T_{ar}, T_{lt}, and T_{tr} are 0, 1.38×10^{-5}, -1.65×10^{-6}, -5.76×10^{-6}, 2.81×10^{-3}, and 7.99×10^{-7}, respectively. Again, we fixed the problem size at two billion pages and looked for the optimum number of processors that minimizes the execution time. As can be seen in Figure 5.12, the time to process two billion pages at an iteration is minimized at 37.834 seconds with approximately 3720 processors. The two curves estimated for MYRI and ASMA can be compared in Figure 5.13.

This prediction is not so realistic since the Fast Ethernet network infrastructure cannot be extended to support this number of processors without serious compromises in communication performance. Our HP4000M ProCurve Fast Ethernet switch can support up to 80 ports. In order to connect 3720 single processor machines, 48 switches are needed. We need more analysis to make better prediction for the actual network performance on this infrastructure.

Table 5.9. Efficiency of parallel PR on ASMA (per cent)

p	n			
	2.5M	**5M**	**7.5M**	**10M**
1	100	100	100	100
5	8.26	31.95	17.83	10.37
10	8.16	37.49	37.78	29.76
15	8.19	36.70	37.83	36.34
20	6.77	36.38	38.32	35.50

Table 5.10. Efficiency of parallel Pagerank with fixed load per processor (per cent)

p	k			
	ASMA		**MYRI**	
	250K	**500K**	**250K**	**500K**
1	100	100	100	100
5	9.89	8.26	12.34	12.32
10	8.16	37.49	13.06	10.33
15	36.65	37.83	8.05	12.39
20	36.38	35.50	11.79	12.45
30	N/A	N/A	12.85	N/A

5.5.3. Other Results

In this section, we present some observations and facts from the experiments we performed on both clusters.

5.5.3.1. Number of Iterations. Keeping Δ fixed at 0.0001, the numbers of PR iterations for the graphs used in the experiments are given in Table 5.13. Note that the number of iterations is independent of the number of processors used.

5.5.3.2. Communication Volume. Figure 5.14 and Table 5.14 show the total number of PageRanks that are flowed between the processors during a parallel PR iteration. As the number of processors increase, the communication volume approaches to the

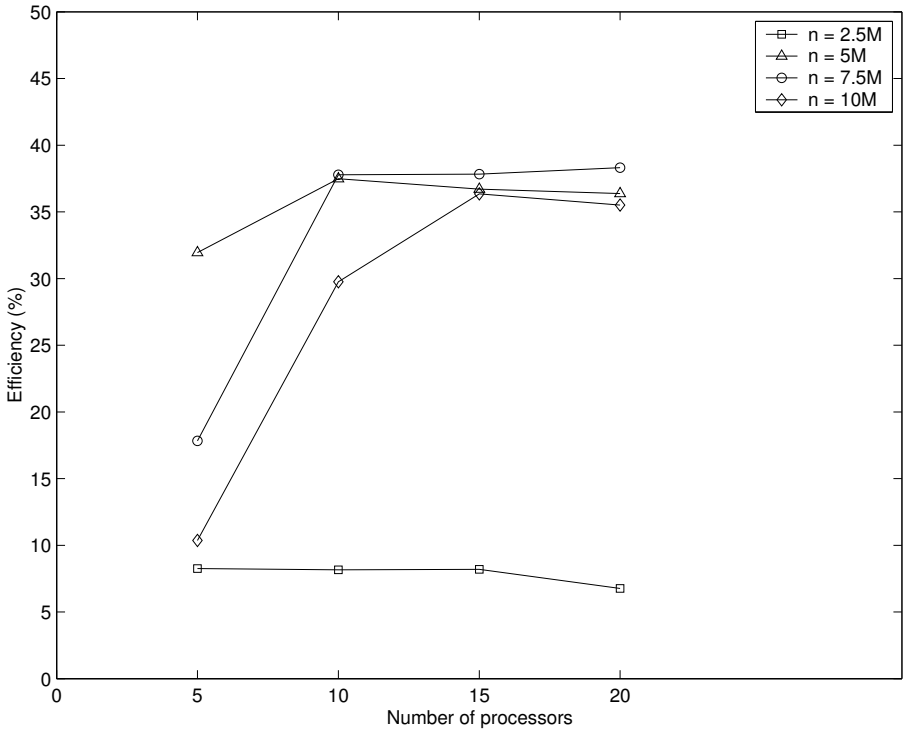


Figure 5.10. Efficiency of parallel PR on ASMA (per cent)

total number of links.

5.5.3.3. Communication Pattern. Since the number of pages are much larger than the number of processors, all processors interacted with all the others in the experiments. Therefore, for all experiments, the communication pattern is all–to–all.

5.5.3.4. Read/Write Performance over NFS. Under normal circumstances, the dumping of PR vector to the disk is done in approximately 0.5 percent of the total execution time on ASMA. On the other hand, it takes three percent of the total execution time on MYRI to dump the vector. This difference is caused by disk I/O over NFS. Lever and Honeyman [40] has a recent study on Linux NFS write performance.

We could not have a chance to observe NFS read performance in our application since we cannot apply profiling without affecting the overall run time. However, it is obvious that doing file I/O over network can be deadly for our application. We would gain much better results on MYRI, if the disk access was local.

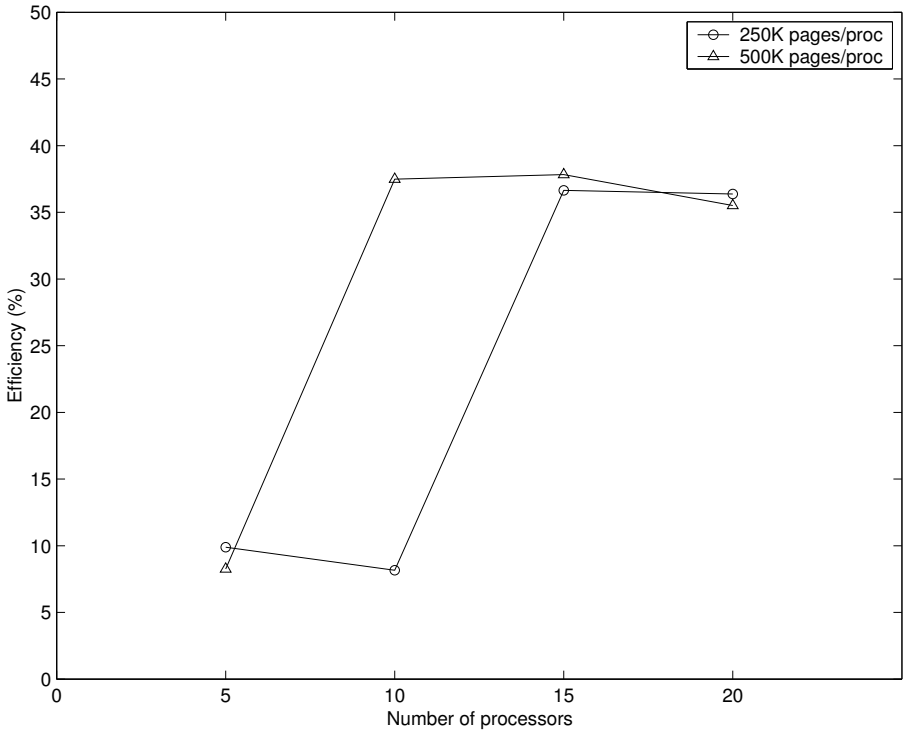


Figure 5.11. Scalability of parallel PR on ASMA

Table 5.11. Execution times of parallel PR with fixed load per processor (sec)

p	k			
	ASMA		**MYRI**	
	250K	**500K**	**250K**	**500K**
1	5.71	10.53	2.81	5.61
5	59.85	129.24	25.44	49.63
10	65.41	145.73	23.40	57.74
15	66.25	130.06	32.94	47.13
20	75.09	147.11	25.29	50.43
30	N/A	N/A	22.71	N/A

Table 5.12. Average times spent by each processor on ASMA (sec/iteration)

n	**p**	**Tcomm**	**Tcio**
1.25M	5	1.128	4.045
2.5M	5	2.554	9.993
2.5M	10	1.730	4.497
2.5M	15	1.319	2.802
2.5M	20	1.882	2.000
3.75M	15	1.960	4.683
5M	5	7.849	24.418
5M	10	3.434	10.317
5M	15	2.719	6.457
5M	20	2.220	4.628
7.5M	10	4.663	15.971
7.5M	15	3.548	10.129
7.5M	20	2.958	7.568
7.5M	5	**23.879**	**62.092**
10M	5	**35.262**	**177.078**
10M	10	8.454	28.619
10M	15	5.214	14.776
10M	20	4.567	10.718

Table 5.13. Number of PR iterations (c) until convergence

n	250K	500K	1.25M	2.5M	3.75M	5M	7.5M	10M
c	12	11	11	10	9	10	9	9

Table 5.14. Total communication volume in a parallel PR iteration

p	n			
	2.5M	**5M**	**7.5M**	**10M**
5	10,199,844	20,741,630	29,801,451	42,746,509
10	11,475,299	23,336,036	33,524,118	48,092,002
15	11,900,270	24,199,203	34,767,180	49,872,555
20	12,112,158	24,631,981	35,388,640	50,762,990
30	12,324,619	25,062,942	36,007,825	51,654,247
60	12,536,485	25,496,001	36,629,820	52,544,551

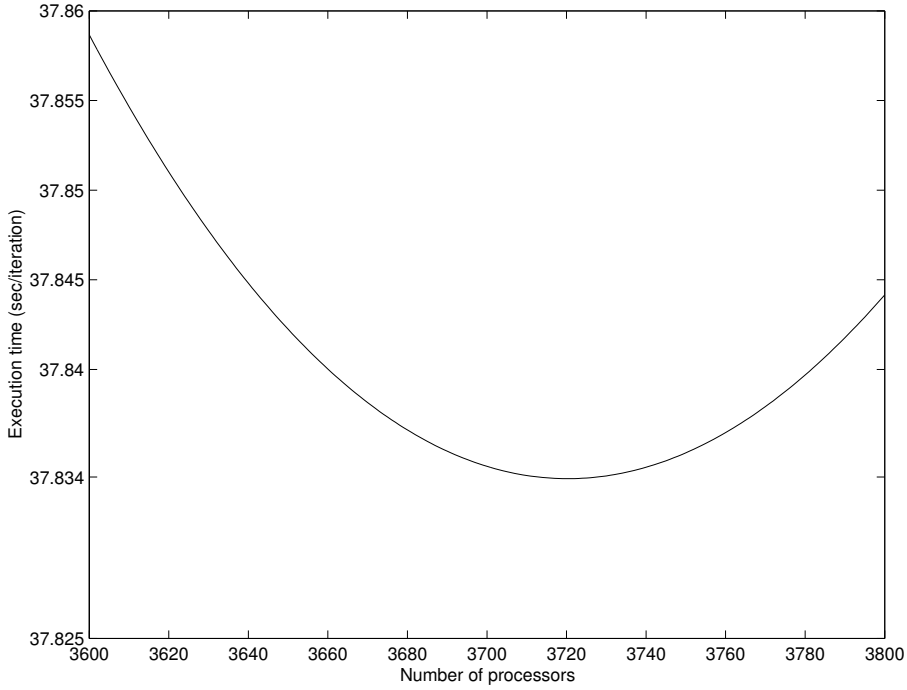


Figure 5.12. Estimation for the optimum number of processors to process two billion pages on ASMA

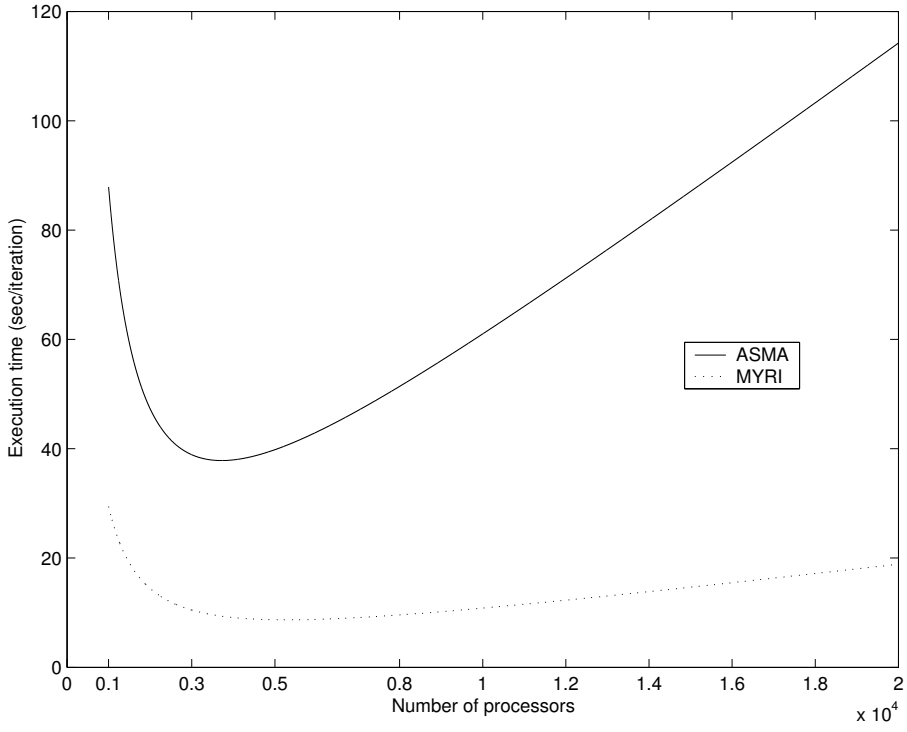


Figure 5.13. Comparison of the processor estimations made for MYRI and ASMA

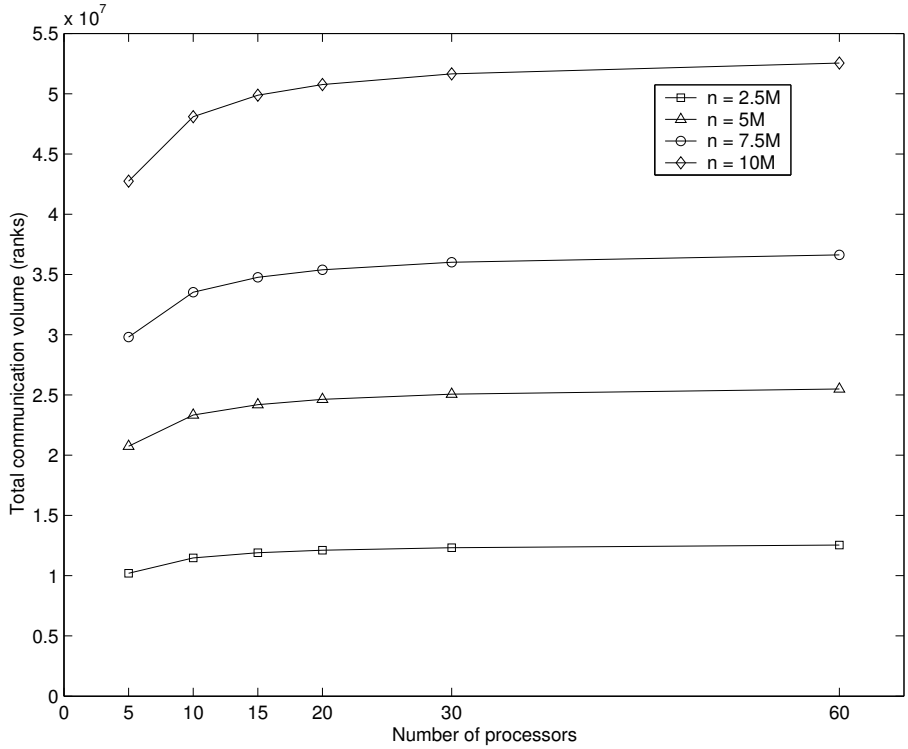


Figure 5.14. Communication volume in a parallel PR iteration

6. CONCLUSION

In this thesis, we developed a distributed memory parallel implementation of Google's PageRank algorithm. We tested our implementation on two Linux PC clusters with different configurations. As a result of our experiments, our implementation is proven to be scalable for 250K and 500K pages per node. 500K, however, is not the upper bound since we could not perform experiments with larger per–node graph sizes due to the lack of resources on ASMA and due to the usage policies on MYRI. We guess our implementation is scalable for a much larger number of pages per node.

We observed significant speedups in our experiments. Although the efficiency of our parallel code is nearly 10 per cent, which is quite low, we state two crucial points where we lose the efficiency: disk access to a central disk over NFS and $\log n$ memory access to the elements of distributed vectors. We need to optimize the code to reduce the computational overhead caused by the parallelization. Optimization may involve more preprocessing to reduce the time spent in PR iterations. Hence, it is possible to make significant improvement in efficiency by further optimizing the code and running on a fully distributed disk environment.

We also supplied an analysis to estimate the optimum number of processors to minimize the time spent on a PageRank iteration. We used least squares approximation to derive a formula for the iteration time based on the experimental timing data. We made this analysis for both MYRI and ASMA.

Although our simple partitioning scheme seems to work fine, it does not guarantee to distribute the load equally. It can be beneficial to research the partitioning strategies specific to the web graphs for the applications like ours. We need a sophisticated partitioning algorithm that will both minimize the inter–processor communication and distribute the load uniformly.

By performing experiments on two significantly different clusters, we demon-

strated how a parallel code like ours behave on such systems. In our experiments, we saw the system parameters that affect the performance significantly. For example, the difference on communication performance between two systems shows the insufficiency of Fast Ethernet networks for problems that requires dense all–to–all communication. On the other hand, according to our experience, the communication performance of Myrinet network infrastructure is perfectly suitable for our implementation. It should be also stated that, in MYRI, if the processors used local disks instead of accessing a remote one over NFS, the performance of our application would be much higher. Although the read performance over NFS was quite good, our application strictly requires low latency disk access to gain high performance.

The execution time of PageRank is significantly affected by the number of iterations made until convergence. We did not attempt to speedup the convergence rate. For example, it is known that the initial PageRank vector affects the convergence rate. We used a uniform probability vector as the initial PageRank vector. Although the number of iterations are quite small, it might be beneficial to investigate the ways to achieve the convergence in less number of iterations.

APPENDIX A: DISTRIBUTED GRAPH INFORMATION

In this Appendix, we give the partitioned graph information for the graphs used in the experiments. The legend of the table headers is as follows:

P_i	:	Processor ID
n_i	:	Number of pages on P_i
m_i	:	Number of links on P_i
d_{max}	:	Max. out–degree on the corresponding processor
$\|V_{F_i}\|$	:	Number of remote pages that is linked by the local pages
$\|V_{B_i}\|$	:	Number of remote pages that is linked to the local pages
$\|V_{L_i}\|$	:	Number of local pages that is linked to the local pages
$\|F_i\|$	:	Number of links from local pages to the remote pages
$\|B_i\|$	:	Number of links from remote pages to the local pages
$\|L_i\|$	:	Number of links between local pages
F/L	:	The ratio $\|F_i\|/\|L_i\|$
B/L	:	The ratio $\|B_i\|/\|L_i\|$

Table A.1. Graph information for n=10M and p=60

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	166667	890641	3319	440820	123568	7424	877703	761527	12938	67.84	58.86
1	166667	899268	4846	445678	123101	7627	881529	1030374	17739	49.69	58.09
2	166667	891974	1644	440244	123245	7544	869492	1325349	22482	38.68	58.95
3	166667	906677	4772	448729	123202	7719	893411	760790	13266	67.35	57.35
4	166667	880362	4324	435560	123138	7459	867151	785675	13211	65.64	59.47
5	166667	891022	3074	440680	123256	7390	875750	903542	15272	57.34	59.16
6	166667	892781	2276	442312	123223	7426	877655	887841	15126	58.02	58.70
7	166667	897755	3295	444863	123483	7498	884119	790164	13636	64.84	57.95
8	166667	898340	5643	444553	123459	7616	883711	854613	14629	60.41	58.42
9	166667	892599	4305	442168	123457	7550	879947	747389	12652	69.55	59.07
10	166667	895018	5255	442948	123271	7664	882779	710592	12239	72.13	58.06
11	166667	882797	1857	437676	123510	7637	869516	775066	13281	65.47	58.36
12	166667	899496	6192	444866	123078	7568	884812	846644	14684	60.26	57.66
13	166667	897291	3557	443708	123264	7530	883709	796366	13582	65.06	58.63
14	166667	886037	1390	438289	123416	7435	866664	1135451	19373	44.74	58.61
15	166667	890786	3356	441070	123253	7388	872112	1097736	18674	46.70	58.78
16	166667	879942	2494	435821	123319	7337	864273	949971	15669	55.16	60.63
17	166667	885121	1053	438103	123376	7355	871205	827717	13916	62.60	59.48
18	166667	883456	2054	436628	123254	7514	870585	764106	12871	67.64	59.37
19	166667	884514	4609	437679	123383	7483	871668	765710	12846	67.86	59.61
20	166667	903895	12180	448691	123202	7531	890798	761172	13097	68.02	58.12
21	166667	886123	1762	439067	123266	7498	874462	693672	11661	74.99	59.49
22	166667	886743	1313	439769	123416	7506	874178	745368	12565	69.57	59.32
23	166667	905370	3450	447952	123059	7546	888933	955823	16437	54.08	58.15
24	166667	884829	1972	438026	123080	7254	872748	725650	12081	72.24	60.07
25	166667	885775	4981	438238	123420	7391	868521	1021443	17254	50.34	59.20
26	166667	889967	3346	440297	123334	7378	877552	736632	12415	70.68	59.33
27	166667	878686	1935	435185	123044	7379	865375	802184	13311	65.01	60.26
28	166667	886712	3046	439376	123219	7429	871264	912521	15448	56.40	59.07
29	166667	884431	1657	438143	123188	7544	869952	855721	14479	60.08	59.10

Table A.1. Graph information for n=10M and p=60 (continued)

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
30	166667	892885	3474	441682	123433	7648	878433	839272	14452	60.78	58.07
31	166667	883712	1800	437200	123536	7544	870607	773053	13105	66.43	58.99
32	166667	902786	7688	446381	123579	7569	890021	743174	12765	69.72	58.22
33	166667	884603	1679	437726	123892	7383	865889	1106932	18714	46.27	59.15
34	166667	884323	2727	438490	122868	7311	871301	782164	13022	66.91	60.06
35	166667	890055	7866	440687	123511	7442	865497	1449146	24558	35.24	59.01
36	166667	893820	4795	441863	123376	7451	880046	815301	13774	63.89	59.19
37	166667	900355	6219	444823	123201	7629	888081	707580	12274	72.35	57.65
38	166667	892971	3799	441685	122912	7400	881194	702530	11777	74.82	59.65
39	166667	882168	2568	437453	123040	7300	863613	1116350	18555	46.54	60.16
40	166666	890013	3781	440773	123416	7425	873023	1006215	16990	51.38	59.22
41	166666	892565	2891	441472	123078	7344	877806	871674	14759	59.48	59.06
42	166666	886476	1285	438765	123311	7396	873584	770783	12892	67.76	59.79
43	166666	890680	1484	440954	123194	7502	876914	810471	13766	63.70	58.87
44	166666	892744	3086	441209	123195	7479	879803	764326	12941	67.99	59.06
45	166666	886665	5214	439450	122933	7443	875089	687799	11576	75.60	59.42
46	166666	883376	1167	436994	123445	7377	870229	779262	13147	66.19	59.27
47	166666	893614	3632	442064	123043	7545	877807	927795	15807	55.53	58.70
48	166666	892882	9823	442709	123236	7579	879761	772363	13121	67.05	58.86
49	166666	893245	6011	442328	123412	7527	867630	1528019	25615	33.87	59.65
50	166666	882685	1153	437393	123381	7307	869118	823207	13567	64.06	60.68
51	166666	891957	2554	441380	123185	7564	876003	932060	15954	54.91	58.42
52	166666	885192	2049	438359	123195	7384	872618	745769	12574	69.40	59.31
53	166666	890390	1508	440951	123195	7491	877935	727468	12455	70.49	58.41
54	166666	916643	27007	456722	123042	7847	904038	712533	12605	71.72	56.53
55	166666	883889	3778	438614	123329	7429	869428	861140	14461	60.12	59.55
56	166666	895787	4109	443069	123227	7614	879891	929971	15896	55.35	58.50
57	166666	884183	1815	437718	123204	7469	869898	844677	14285	60.90	59.13
58	166666	885034	2668	437989	123385	7442	860946	1433053	24088	35.74	59.49
59	166666	891364	1659	441111	123414	7549	876774	853655	14590	60.09	58.51

Table A.2. Graph information for n=10M and p=30

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	333334	1789909	4846	774000	245032	26605	1729064	1761733	60845	28.42	28.95
1	333334	1798651	4772	776490	244759	26945	1727052	2050288	71599	24.12	28.64
2	333334	1771384	4324	765728	244788	26276	1714790	1661106	56594	30.30	29.35
3	333334	1790536	3295	775423	245053	26653	1733159	1649390	57377	30.21	28.75
4	333334	1790939	5643	774344	245235	26823	1736505	1574849	54434	31.90	28.93
5	333334	1777815	5255	769251	245165	26982	1727209	1460572	50606	34.13	28.86
6	333334	1796787	6192	776088	244625	26744	1740512	1615001	56275	30.93	28.70
7	333334	1776823	3356	768202	244953	26491	1700819	2195230	76004	22.38	28.88
8	333334	1765063	2494	763373	245117	26180	1705818	1748028	59245	28.79	29.51
9	333334	1767970	4609	764024	245032	26638	1716346	1503909	51624	33.25	29.13
10	333333	1790009	12180	775624	244789	26711	1740605	1430197	49404	35.23	28.95
11	333333	1792117	3450	775225	244752	26722	1734184	1672250	57933	29.93	28.87
12	333333	1770589	4981	765818	244860	26130	1711920	1717765	58669	29.18	29.28
13	333333	1768647	3346	764703	244718	26349	1716834	1512729	51813	33.14	29.20
14	333333	1771153	3046	766674	244730	26532	1711332	1738342	59821	28.61	29.06
15	333333	1776581	3474	767925	245282	26924	1721607	1584912	54974	31.32	28.83
16	333333	1787393	7688	772132	245870	26483	1724669	1818798	62724	27.50	29.00
17	333333	1774367	7866	768197	244653	26462	1698830	2193408	75537	22.49	29.04
18	333333	1794184	6219	774134	244935	26628	1742351	1497098	51833	33.61	28.88
19	333333	1775108	3799	767989	244228	26399	1713753	1787842	61355	27.93	29.14
20	333333	1782600	3781	770443	244886	26548	1718723	1845736	63877	26.91	28.90
21	333333	1777170	1484	768555	244819	26487	1723829	1554618	53341	32.32	29.14
22	333333	1779400	5214	769228	244533	26517	1730406	1427645	48994	35.32	29.14
23	333333	1776985	3632	768407	244838	26421	1719115	1678132	57870	29.71	29.00
24	333333	1786131	9823	773231	244979	26714	1708347	2261343	77784	21.96	29.07
25	333333	1774664	2554	768019	244827	26475	1715425	1725552	59239	28.96	29.13
26	333333	1775586	2049	768242	244766	26546	1725592	1448258	49994	34.52	28.97
27	333333	1800534	27007	782195	244682	26976	1746470	1546693	54064	32.30	28.61
28	333333	1779976	4109	769639	244761	26590	1719881	1744735	60095	28.62	29.03
29	333333	1776399	2668	768247	245154	26491	1699100	2248088	77299	21.98	29.08

Table A.3. Graph information for n=10M and p=20

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	500K	2681880	4846	1051774	364950	55240	2522947	3011472	158933	15.87	18.95
1	500K	2678059	4772	1050535	364582	55067	2553444	2367140	124615	20.49	19.00
2	500K	2688877	5643	1055909	365148	55414	2559035	2446166	129842	19.71	18.84
3	500K	2670393	5255	1048288	365246	55771	2556317	2157145	114076	22.41	18.91
4	500K	2682832	6192	1051695	364673	55024	2540755	2684004	142077	17.88	18.89
5	500K	2655842	3356	1042115	365072	54525	2510772	2778627	145070	17.31	19.15
6	500K	2671857	12180	1049059	364947	55126	2555404	2213303	116453	21.94	19.01
7	500K	2678228	3450	1051635	364623	55551	2556142	2313477	122086	20.94	18.95
8	500K	2660579	4981	1043649	364761	54759	2535021	2399918	125558	20.19	19.11
9	500K	2649827	3046	1040656	364326	54918	2519837	2483654	129990	19.38	19.11
10	500K	2679378	7688	1050361	365407	55779	2558854	2275305	120524	21.23	18.88
11	500K	2658960	7866	1044563	365216	54857	2489858	3225435	169102	14.72	19.07
12	500K	2687152	6219	1052573	364418	55320	2573409	2149484	113743	22.62	18.90
13	500K	2664753	3781	1046364	364644	54674	2513137	2892895	151616	16.58	19.08
14	500K	2669906	3086	1046753	364596	55329	2550470	2265793	119436	21.35	18.97
15	500K	2663644	5214	1045877	364497	54860	2542522	2314256	121122	20.99	19.11
16	500K	2668847	9823	1048855	364943	55132	2510885	3017952	157962	15.90	19.11
17	500K	2667541	2554	1046626	364591	55362	2544603	2323322	122938	20.70	18.90
18	500K	2696308	27007	1060683	364471	55914	2567590	2417912	128718	19.95	18.78
19	500K	2660607	2668	1044045	364992	54931	2501988	3025730	158619	15.77	19.08

Table A.4. Graph information for n=10M and p=15

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	666667	3588544	4846	1298511	482996	92756	3323580	3679500	264964	12.54	13.89
1	666667	3561929	4324	1290585	483125	91895	3333693	3196211	228236	14.61	14.00
2	666667	3568740	5643	1292171	483616	92875	3358745	2930485	209995	15.99	13.96
3	666667	3573579	6192	1293007	482762	91950	3309548	3678473	264031	12.53	13.93
4	666667	3533067	4609	1279766	483420	91238	3311445	3141180	221622	14.94	14.17
5	666667	3582119	12180	1298443	482808	92716	3366923	2994594	215196	15.65	13.92
6	666667	3539250	4981	1281892	482686	91397	3317874	3119605	221376	14.99	14.09
7	666667	3547736	3474	1285105	483242	92252	3318999	3209308	228737	14.51	14.03
8	666667	3561768	7866	1290058	483693	92091	3284901	3873604	276867	11.86	13.99
9	666667	3569293	6219	1291222	482245	92255	3342276	3171112	227017	14.72	13.97
10	666666	3559770	3781	1287905	482811	92183	3325067	3282869	234703	14.17	13.99
11	666666	3556385	5214	1288020	482577	91823	3343165	2999421	213220	15.68	14.07
12	666666	3560795	9823	1290832	482915	92285	3286320	3849443	274475	11.97	14.02
13	666666	3576120	27007	1298187	482578	92643	3368148	2891037	207972	16.20	13.90
14	666666	3556375	4109	1287895	483046	92001	3281871	3855713	274504	11.96	14.05

Table A.5. Graph information for n=10M and p=10

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	1M	5359939	4846	1695969	713796	188058	4791663	5093884	568276	8.43	8.96
1	1M	5359270	5643	1697287	714576	188894	4872821	4360780	486449	10.02	8.96
2	1M	5338674	6192	1689599	713775	186944	4764336	5175440	574338	8.30	9.01
3	1M	5350085	12180	1694403	713826	188380	4872940	4288174	477145	10.21	8.99
4	1M	5310406	4981	1682532	713493	186572	4799773	4628487	510633	9.40	9.06
5	1M	5338338	7866	1690298	714911	188392	4758643	5210671	579695	8.21	8.99
6	1M	5351905	6219	1692882	713168	187594	4820915	4776748	530990	9.08	9.00
7	1M	5333550	5214	1688650	713255	187411	4852865	4339922	480685	10.10	9.03
8	1M	5336388	9823	1691012	713770	187692	4774402	5060188	561986	8.50	9.00
9	1M	5356915	27007	1698510	713702	188466	4783644	5157708	573271	8.34	9.00

Table A.6. Graph information for n=10M and p=5

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	2M	10719209	5643	2467020	1358007	616367	8610271	8400451	2108938	4.08	3.98
1	2M	10688759	12180	2461334	1357279	614710	8585984	8412322	2102775	4.08	4.00
2	2M	10648744	7866	2453328	1358706	613632	8469186	8749928	2179558	3.89	4.01
3	2M	10685455	6219	2459313	1356415	614185	8660395	8103285	2025060	4.28	4.00
4	2M	10693303	27007	2465152	1357304	615907	8420673	9080523	2272630	3.71	4.00

Table A.7. Graph information for n=7.5M and p=60

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	125K	635133	20847	338471	92411	5799	623841	643682	11292	55.25	57.00
1	125K	614701	2389	326388	92362	5609	596988	1048683	17713	33.70	59.20
2	125K	620946	3308	329833	92333	5530	611146	586529	9800	62.36	59.85
3	125K	619308	1390	328783	92120	5559	610481	522168	8827	69.16	59.16
4	125K	621894	3038	330185	92404	5470	612712	545382	9182	66.73	59.40
5	125K	623314	3777	330985	92796	5627	613626	574120	9688	63.34	59.26
6	125K	621770	5290	330189	92237	5571	612187	562833	9583	63.88	58.73
7	125K	615250	1550	326397	92349	5464	605314	594131	9936	60.92	59.80
8	125K	620638	3364	329771	92343	5572	611377	553812	9261	66.02	59.80
9	125K	617196	1686	327492	91803	5579	607851	557032	9345	65.05	59.61
10	125K	614938	1228	326704	92218	5545	603703	667530	11235	53.73	59.42
11	125K	622006	1819	329596	92524	5645	611424	621315	10582	57.78	58.71
12	125K	619589	3058	328328	92572	5665	610197	551351	9392	64.97	58.70
13	125K	622397	2442	329904	92442	5621	609173	768158	13224	46.07	58.09
14	125K	617579	5062	328117	92088	5646	608593	527697	8986	67.73	58.72
15	125K	615513	2231	326882	92459	5455	606277	554414	9236	65.64	60.03
16	125K	618155	2627	328071	92135	5581	607581	628971	10574	57.46	59.48
17	125K	614335	2649	326171	92115	5509	605536	525185	8799	68.82	59.69
18	125K	619956	3717	328913	92613	5714	610397	555699	9559	63.86	58.13
19	125K	617167	1270	327547	92265	5607	607372	575438	9795	62.01	58.75
20	125K	627346	5742	333304	92255	5644	616959	611035	10387	59.40	58.83
21	125K	617788	2330	328006	92291	5495	606940	646721	10848	55.95	59.62
22	125K	627763	1846	332394	92362	5660	617894	564705	9869	62.61	57.22
23	125K	617471	2250	327996	92224	5515	607175	601492	10296	58.97	58.42
24	125K	622166	4329	329830	92636	5633	612065	595239	10101	60.59	58.93
25	125K	616547	1042	327966	92206	5425	605365	670905	11182	54.14	60.00
26	125K	612822	824	325416	92294	5460	603303	574000	9519	63.38	60.30
27	125K	614053	1978	326597	92477	5553	603487	628266	10566	57.12	59.46
28	125K	627109	4021	332233	92582	5717	617760	541351	9349	66.08	57.90
29	125K	630337	5770	334520	92278	5724	614962	885862	15375	40.00	57.62

Table A.7. Graph information for n=7.5M and p=60 (continued)

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
30	125K	620540	1961	328998	92464	5494	611106	561874	9434	64.78	59.56
31	125K	612123	1330	325494	92420	5423	603012	554672	9111	66.19	60.88
32	125K	629538	12748	335058	92017	5708	620764	510893	8774	70.75	58.23
33	125K	632641	7848	335790	92464	5623	623081	560793	9560	65.18	58.66
34	125K	614069	1465	326610	92206	5551	604343	578494	9726	62.14	59.48
35	125K	626210	7372	332961	92315	5643	615755	614126	10455	58.90	58.74
36	125K	615103	1330	326478	92529	5582	605653	565036	9450	64.09	59.79
37	125K	614241	1430	326920	92332	5605	604527	569896	9714	62.23	58.67
38	125K	621878	4622	329743	92523	5670	612763	531339	9115	67.23	58.29
39	125K	632394	6587	335817	91941	5676	620561	693030	11833	52.44	58.57
40	125K	622962	1250	329743	91992	5772	612792	582834	10170	60.25	57.31
41	125K	620655	1585	329369	92035	5635	610989	571091	9666	63.21	59.08
42	125K	618059	1756	327976	92464	5495	607808	618105	10251	59.29	60.30
43	125K	614777	1572	326910	92448	5513	605975	526852	8802	68.85	59.86
44	125K	618922	4044	328044	92532	5685	609236	558437	9686	62.90	57.65
45	125K	616020	3068	327273	92526	5602	604567	681089	11453	52.79	59.47
46	125K	621957	2552	329877	92351	5547	611445	624664	10512	58.17	59.42
47	125K	615507	2264	326900	92566	5521	605413	601735	10094	59.98	59.61
48	125K	615048	1961	327206	92279	5579	604289	633668	10759	56.17	58.90
49	125K	614729	1136	326116	92290	5596	603071	697057	11658	51.73	59.79
50	125K	624948	2024	331186	92424	5521	611974	764764	12974	47.17	58.95
51	125K	616053	1050	327251	92311	5502	605189	655917	10864	55.71	60.38
52	125K	614798	2616	327188	92244	5583	605825	531630	8973	67.52	59.25
53	125K	613717	1128	326407	92605	5405	603670	602947	10047	60.08	60.01
54	125K	622912	3323	330344	92392	5431	613441	574389	9471	64.77	60.65
55	125K	663487	41792	356527	92211	6124	652787	591803	10700	61.01	55.31
56	125K	618673	1673	328494	92360	5603	608975	572960	9698	62.79	59.08
57	125K	613200	1623	325845	92223	5512	601130	730287	12070	49.80	60.50
58	125K	629313	4456	334464	92334	5571	618699	617938	10614	58.29	58.22
59	125K	630959	8744	335077	92180	5801	619294	667794	11665	53.09	57.25

Table A.8. Graph information for n=7.5M and p=30

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	250K	1249834	20847	580827	183519	20012	1192158	1663694	57676	20.67	28.85
1	250K	1240254	3308	575733	183251	19684	1202905	1089975	37349	32.21	29.18
2	250K	1245208	3777	577450	183946	19875	1207326	1100490	37882	31.87	29.05
3	250K	1237020	5290	573694	183335	19687	1198114	1137577	38906	30.80	29.24
4	250K	1237834	3364	574409	182960	19686	1200735	1092351	37099	32.37	29.44
5	250K	1236944	1819	573730	183529	19841	1193241	1266959	43703	27.30	28.99
6	250K	1241986	3058	574632	183758	20055	1196866	1297005	45120	26.53	28.75
7	250K	1233092	5062	572305	183332	19763	1196581	1063822	36511	32.77	29.14
8	250K	1232490	2649	571688	183017	19699	1193581	1134620	38909	30.68	29.16
9	250K	1237123	3717	573835	183600	19955	1198799	1112167	38324	31.28	29.02
10	250K	1245134	5742	577779	183299	19969	1202186	1236043	42948	27.99	28.78
11	250K	1245234	2250	577040	183324	19995	1205054	1146182	40180	29.99	28.53
12	250K	1238713	4329	575085	183629	19770	1196178	1244892	42535	28.12	29.27
13	250K	1226875	1978	570029	183534	19760	1186474	1181950	40401	29.37	29.26
14	250K	1257446	5770	582288	183563	20240	1208384	1402875	49062	24.63	28.59
15	250K	1232663	1961	571796	183610	19556	1195488	1097916	37175	32.16	29.53
16	250K	1262179	12748	585662	183232	20382	1225131	1052972	37048	33.07	28.42
17	250K	1240279	7372	576378	183244	19952	1199692	1172214	40587	29.56	28.88
18	250K	1229344	1430	570942	183641	19799	1191166	1115918	38178	31.20	29.23
19	250K	1254272	6587	581441	183222	20151	1212282	1203327	41990	28.87	28.66
20	250K	1243617	1585	575634	182800	20024	1204217	1134361	39400	30.56	28.79
21	250K	1232836	1756	572160	183635	19743	1194585	1125759	38251	31.23	29.43
22	250K	1234942	4044	572749	183885	19839	1193094	1218817	41848	28.51	29.12
23	250K	1237464	2552	573977	183698	19948	1195853	1205394	41611	28.74	28.97
24	250K	1229777	1961	570672	183325	19693	1185180	1308545	44597	26.58	29.34
25	250K	1241001	2024	574841	183419	19936	1192575	1396093	48426	24.63	28.83
26	250K	1228515	2616	571215	183629	19722	1190376	1115458	38139	31.21	29.25
27	250K	1286399	41792	599246	183304	20687	1245799	1145763	40600	30.68	28.22
28	250K	1231873	1673	571731	183308	19763	1188082	1281224	43791	27.13	29.26
29	250K	1260272	8744	584674	183264	20206	1215723	1263462	44549	27.29	28.36

Table A.9. Graph information for n=7.5M and p=20

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	375K	1870780	20847	788444	273240	41479	1754516	2201435	116264	15.09	18.93
1	375K	1864516	3777	784831	273588	41105	1781275	1586126	83241	21.40	19.05
2	375K	1857658	5290	782102	273102	40993	1771099	1652997	86559	20.46	19.10
3	375K	1854140	1819	780599	272750	41003	1760756	1783655	93384	18.86	19.10
4	375K	1859565	5062	781670	273332	41418	1765614	1784857	93951	18.79	19.00
5	375K	1848003	2649	778328	273002	40782	1761942	1651118	86061	20.47	19.19
6	375K	1864469	5742	785061	273239	41449	1775889	1683333	88580	20.05	19.00
7	375K	1863022	2330	783912	273103	41088	1770434	1751343	92588	19.12	18.92
8	375K	1851535	4329	779995	273433	40858	1759095	1778506	92440	19.03	19.24
9	375K	1871499	5770	787792	273558	41546	1766349	1985619	105150	16.80	18.88
10	375K	1862201	12748	784158	273092	41190	1779947	1572504	82254	21.64	19.12
11	375K	1872920	7848	789175	273161	41521	1783148	1693382	89772	19.86	18.86
12	375K	1851222	4622	779298	273712	41077	1766981	1610309	84241	20.98	19.12
13	375K	1876011	6587	788629	272265	41668	1781080	1783693	94931	18.76	18.79
14	375K	1851758	4044	779314	273637	41275	1765627	1646002	86131	20.50	19.11
15	375K	1853484	3068	780186	273601	41414	1756693	1842756	96791	18.15	19.04
16	375K	1854725	2024	780348	273219	40783	1749046	2025201	105679	16.55	19.16
17	375K	1844568	2616	778358	273452	40946	1754350	1730160	90218	19.45	19.18
18	375K	1905072	41792	803805	273119	42364	1814762	1678711	90310	20.09	18.59
19	375K	1873472	8744	788886	272944	41537	1770037	1946933	103435	17.11	18.82

Table A.10. Graph information for n=7.5M and p=15

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	500K	2490088	20847	968072	361542	69027	2299803	2658409	190285	12.09	13.97
1	500K	2482228	5290	964370	362209	68779	2328503	2161130	153725	15.15	14.06
2	500K	2474778	3364	962254	361344	68218	2313629	2278963	161149	14.36	14.14
3	500K	2475078	5062	960305	362081	68871	2312546	2279926	162532	14.23	14.03
4	500K	2469613	3717	959588	361480	68780	2315273	2169680	154340	15.00	14.06
5	500K	2490368	5742	967243	361462	69130	2324903	2299888	165465	14.05	13.90
6	500K	2465588	4329	959425	362128	68462	2299655	2343845	165933	13.86	14.13
7	500K	2490109	5770	966088	361991	69187	2316965	2413884	173144	13.38	13.94
8	500K	2502458	12748	972794	361359	69652	2347429	2147792	155029	15.14	13.85
9	500K	2483616	6587	964740	361852	68821	2323589	2239386	160027	14.52	13.99
10	500K	2476453	1756	961485	361459	68854	2321341	2182659	155112	14.97	14.07
11	500K	2472406	4044	959976	362339	69138	2305438	2340702	166968	13.81	14.02
12	500K	2470778	2024	959941	361590	68408	2285405	2612288	185373	12.33	14.09
13	500K	2514914	41792	979365	361766	69959	2357274	2182320	157640	14.95	13.84
14	500K	2492145	8744	968262	361409	69133	2315427	2456308	176718	13.10	13.90

Table A.11. Graph information for n=7.5M and p=10

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	750K	3735296	20847	1269153	534982	140924	3335548	3587318	399748	8.34	8.97
1	750K	3711798	5290	1261934	534040	139419	3352476	3257273	359322	9.33	9.07
2	750K	3707568	5062	1258343	534669	140164	3346980	3255399	360588	9.28	9.03
3	750K	3727491	5742	1265968	534461	140912	3365313	3253666	362178	9.29	8.98
4	750K	3723034	5770	1265123	535195	140825	3327165	3565846	395869	8.40	9.01
5	750K	3735121	12748	1269221	534427	140946	3390278	3093069	344843	9.83	8.97
6	750K	3727233	6587	1265202	534124	140655	3369089	3215030	358144	9.41	8.98
7	750K	3705242	4044	1258043	535415	140625	3339263	3305701	365979	9.12	9.03
8	750K	3699293	2616	1258079	534767	139515	3306793	3558758	392500	8.42	9.07
9	750K	3778544	41792	1283782	533966	142545	3391213	3432058	387331	8.76	8.86

Table A.12. Graph information for n=7.5M and p=5

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	1.5M	7447094	20847	1840848	1016329	459850	5928158	6084725	1518936	3.90	4.01
1	1.5M	7435059	5742	1837001	1016954	459324	5991643	5788415	1443416	4.15	4.01
2	1.5M	7458155	12748	1843491	1017048	460732	5976202	5917674	1481953	4.03	3.99
3	1.5M	7432475	6587	1836067	1017330	459299	5986917	5799296	1445558	4.14	4.01
4	1.5M	7477837	41792	1848914	1015949	461749	5918531	6211341	1559306	3.80	3.98

Table A.13. Graph information for n=5M and p=60

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	83334	427368	1859	219646	61684	3755	420950	380964	6418	65.59	59.36
1	83334	436790	5495	224665	61647	3788	429820	405020	6970	61.67	58.11
2	83334	435647	3771	223847	61918	3774	428194	434217	7453	57.45	58.26
3	83334	426487	979	218764	62170	3729	418763	462439	7724	54.22	59.87
4	83334	430710	1883	220992	61770	3800	424283	379148	6427	66.02	58.99
5	83334	424157	863	217524	61859	3741	417757	388520	6400	65.27	60.71
6	83334	441871	4430	226665	61669	3861	435954	344499	5917	73.68	58.22
7	83334	433225	1922	221954	61914	3763	426791	378845	6434	66.33	58.88
8	83334	428717	957	220092	61573	3707	423078	331789	5639	75.03	58.84
9	83334	432280	2282	221712	61829	3738	426079	369679	6201	68.71	59.62
10	83334	436264	5247	223469	61478	3801	429134	416050	7130	60.19	58.35
11	83334	431864	1727	221965	61754	3668	426414	331488	5450	78.24	60.82
12	83334	437190	7764	224008	61869	3859	429879	419036	7311	58.80	57.32
13	83334	433689	1354	222683	62162	3705	427493	374604	6196	68.99	60.46
14	83334	437396	4710	224948	61862	3792	430536	397947	6860	62.76	58.01
15	83334	429461	1209	220105	61882	3735	422861	391343	6600	64.07	59.29
16	83334	427433	911	219612	61995	3723	421132	378473	6301	66.84	60.07
17	83334	425540	1096	218265	61885	3680	419463	366660	6077	69.02	60.34
18	83334	428463	3850	220206	61712	3761	422024	383424	6439	65.54	59.55
19	83334	424823	1150	218002	62069	3621	418520	387647	6303	66.40	61.50
20	83333	440059	6897	225652	61967	3747	433886	357162	6173	70.29	57.86
21	83333	424860	1074	217994	61714	3670	418807	370702	6053	69.19	61.24
22	83333	426463	1469	218551	61822	3661	415520	648063	10943	37.97	59.22
23	83333	433903	1315	222310	61974	3744	427019	404642	6884	62.03	58.78
24	83333	427886	1032	219648	61951	3759	420863	426637	7023	59.93	60.75
25	83333	425158	1727	218701	61929	3706	418458	405699	6700	62.46	60.55
26	83333	435751	1585	223441	61772	3758	428971	397001	6780	63.27	58.55
27	83333	435720	1845	223360	61826	3668	429531	372576	6189	69.40	60.20
28	83333	440550	7457	226243	61900	3841	430617	574812	9933	43.35	57.87
29	83333	428295	992	219520	61799	3697	422428	347164	5867	72.00	59.17

Table A.13. Graph information for n=5M and p=60 (continued)

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
30	83333	428336	1483	219587	61719	3718	421755	392643	6581	64.09	59.66
31	83333	430094	3526	220981	62002	3733	422754	439442	7340	57.60	59.87
32	83333	429518	1311	220495	61927	3759	423215	370520	6303	67.15	58.78
33	83333	431740	1578	220938	61634	3688	426203	332856	5537	76.97	60.11
34	83333	428994	2451	220253	61585	3721	421595	444297	7399	56.98	60.05
35	83333	427100	1142	218979	61793	3705	418741	503236	8359	50.09	60.20
36	83333	430975	1298	221205	61972	3674	424972	364676	6003	70.79	60.75
37	83333	440006	2553	225317	62042	3930	432688	422980	7318	59.13	57.80
38	83333	432109	1240	221637	61947	3709	420561	674934	11548	36.42	58.45
39	83333	431770	1538	221491	61757	3712	424754	414582	7016	60.54	59.09
40	83333	427786	1997	219535	61907	3830	422165	333887	5621	75.10	59.40
41	83333	430546	1989	220522	61779	3664	423530	423746	7016	60.37	60.40
42	83333	441116	4908	226775	61932	3814	433500	439446	7616	56.92	57.70
43	83333	440283	4690	225353	61933	3786	434499	337898	5784	75.12	58.42
44	83333	435600	4023	223533	62001	3782	428762	402675	6838	62.70	58.89
45	83333	429974	1320	219933	61957	3715	416663	787239	13311	31.30	59.14
46	83333	434324	1698	222810	61917	3775	426654	448737	7670	55.63	58.51
47	83333	434453	4138	222988	61949	3726	427347	422811	7106	60.14	59.50
48	83333	433744	2662	222975	61841	3792	427317	376733	6427	66.49	58.62
49	83333	434515	4087	223064	61811	3747	413948	1222444	20567	20.13	59.44
50	83333	436331	4647	224170	61649	3884	430268	342338	6063	70.97	56.46
51	83333	426034	1359	218874	61986	3758	419435	395556	6599	63.56	59.94
52	83333	430855	5907	220934	61767	3782	423324	448402	7531	56.21	59.54
53	83333	430881	1051	221093	61866	3807	424209	392482	6672	63.58	58.83
54	83333	439682	2726	225496	61752	3797	433927	332938	5755	75.40	57.85
55	83333	435461	1608	222817	61897	3822	428196	422199	7265	58.94	58.11
56	83333	425478	923	218427	61822	3698	418792	402677	6686	62.64	60.23
57	83333	442750	5790	227057	61882	3831	435087	439609	7663	56.78	57.37
58	83333	429694	1786	220325	61661	3803	422798	399421	6896	61.31	57.92
59	83333	428897	1204	220042	61836	3754	423117	338347	5780	73.20	58.54

Table A.14. Graph information for n=5M and p=30

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	166667	864156	5495	388089	122489	13493	837246	772457	26910	31.11	28.71
1	166667	862130	3771	386703	123227	13458	831796	881502	30334	27.42	29.06
2	166667	854868	1883	383330	122788	13331	829209	754837	25659	32.32	29.42
3	166667	875059	4430	392223	122742	13588	850323	710953	24736	34.38	28.74
4	166667	861019	2282	386284	122511	13254	837240	689526	23779	35.21	29.00
5	166667	868127	5247	389047	122427	13309	842758	734736	25369	33.22	28.96
6	166667	870874	7764	390311	123187	13458	843812	780069	27062	31.18	28.83
7	166667	866843	4710	388573	122894	13583	839779	775709	27064	31.03	28.66
8	166667	852975	1096	382722	123014	13215	828093	732630	24882	33.28	29.44
9	166667	853265	3850	383054	122913	13160	827631	758121	25634	32.29	29.57
10	166667	864947	6897	387742	122827	13363	840255	715461	24692	34.03	28.98
11	166667	860377	1469	385254	122999	13291	824883	1034867	35494	23.24	29.16
12	166667	853036	1727	383301	123049	13217	825454	818643	27582	29.93	29.68
13	166667	871452	1845	390426	122772	13301	845362	756461	26090	32.40	28.99
14	166667	868861	7457	389358	122856	13440	837292	906209	31569	26.52	28.71
15	166667	858446	3526	385053	122914	13306	830345	817911	28101	29.55	29.11
16	166667	861262	1578	385931	122749	13133	837738	691694	23524	35.61	29.40
17	166667	856096	2451	383730	122595	13153	824490	931678	31606	26.09	29.48
18	166667	870986	2553	390217	123202	13505	844209	774207	26777	31.53	28.91
19	166667	863883	1538	387408	122904	13160	826885	1071082	36998	22.35	28.95
20	166666	858332	1997	384689	122844	13370	832875	744813	25457	32.72	29.26
21	166666	881399	4908	394679	123019	13489	854570	763915	26829	31.85	28.47
22	166666	865574	4023	387722	123149	13409	825222	1169711	40352	20.45	28.99
23	166666	868777	4138	389513	123030	13367	839190	856737	29587	28.36	28.96
24	166666	868259	4087	389765	122829	13486	813934	1571846	54325	14.98	28.93
25	166666	862365	4647	387096	122782	13501	837099	725290	25266	33.13	28.71
26	166666	861736	5907	386326	122842	13333	833390	826741	28346	29.40	29.17
27	166666	875143	2726	391471	122789	13586	849045	742059	26098	32.53	28.43
28	166666	868228	5790	389385	122854	13228	839578	827985	28650	29.30	28.90
29	166666	858591	1786	384740	122637	13393	833239	725092	25352	32.87	28.60

Table A.15. Graph information for n=5M and p=20

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	250K	1299797	5495	529483	182678	27869	1237288	1178531	62509	19.79	18.85
1	250K	1281355	1883	521548	183265	27611	1219635	1188936	61720	19.76	19.26
2	250K	1303803	4430	530388	182600	27926	1249835	1019007	53968	23.16	18.88
3	250K	1300396	5247	528833	182558	27621	1243467	1079197	56929	21.84	18.96
4	250K	1308254	7764	532094	183310	28009	1247107	1150809	61147	20.40	18.82
5	250K	1282436	1209	521806	183233	27512	1225138	1098158	57298	21.38	19.17
6	250K	1293347	6897	526450	183118	27618	1236291	1090067	57056	21.67	19.11
7	250K	1285209	1469	522483	183051	27398	1213718	1375541	71491	16.98	19.24
8	250K	1288818	1727	524930	183202	27542	1227024	1188332	61794	19.86	19.23
9	250K	1304554	7457	530120	183021	27810	1238361	1250342	66193	18.71	18.89
10	250K	1287949	3526	524299	183101	27591	1227191	1162056	60758	20.20	19.13
11	250K	1287852	2451	523517	182575	27253	1223476	1237331	64376	19.01	19.22
12	250K	1303088	2553	529538	183450	28028	1228281	1412402	74807	16.42	18.88
13	250K	1290112	1997	524754	182924	27593	1230929	1132931	59183	20.80	19.14
14	250K	1317003	4908	535218	183314	27954	1256123	1139387	60880	20.63	18.72
15	250K	1298756	4138	527677	183346	27657	1214510	1602625	84246	14.42	19.02
16	250K	1304596	4647	530867	182811	28006	1205596	1875575	99000	12.18	18.95
17	250K	1287771	5907	524106	183198	27581	1225742	1195213	62029	19.76	19.27
18	250K	1300625	2726	528686	182986	27855	1241503	1118400	59122	21.00	18.92
19	250K	1301345	5790	529084	182792	27868	1240766	1137141	60579	20.48	18.77

Table A.16. Graph information for n=5M and p=15

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	333334	1726286	5495	648590	242236	46591	1612274	1597191	114012	14.14	14.01
1	333334	1729927	4430	649592	242191	46633	1628801	1415059	101126	16.11	13.99
2	333334	1729146	5247	649218	241508	46105	1630858	1375122	98288	16.59	13.99
3	333334	1737717	7764	651910	242681	46772	1629510	1501697	108207	15.06	13.88
4	333334	1706240	3850	641652	242568	46056	1604896	1439923	101344	15.84	14.21
5	333333	1725315	6897	647308	242397	46015	1605020	1690106	120295	13.34	14.05
6	333333	1724485	1845	647937	242428	45954	1616724	1521128	107761	15.00	14.12
7	333333	1727309	7457	648512	242368	46322	1608286	1664764	119023	13.51	13.99
8	333333	1717353	2451	644725	241947	45621	1606553	1567703	110800	14.50	14.15
9	333333	1734863	2553	650709	242656	46493	1607123	1781322	127740	12.58	13.94
10	333333	1739734	4908	652038	242461	46468	1635455	1456734	104279	15.68	13.97
11	333333	1734361	4138	650356	242875	46427	1594543	1956566	139818	11.40	13.99
12	333333	1730629	4647	650036	242237	46638	1571632	2217729	158997	9.88	13.95
13	333333	1736880	5907	651221	242165	46541	1627903	1514276	108977	14.94	13.90
14	333333	1726821	5790	648305	242108	46108	1619625	1499883	107196	15.11	13.99

Table A.17. Graph information for n=5M and p=10

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	500K	2581152	5495	847838	358073	94611	2333016	2243560	248136	9.40	9.04
1	500K	2604199	5247	854411	357140	94662	2382192	1987094	222007	10.73	8.95
2	500K	2590690	7764	850022	358569	94935	2353729	2130451	236961	9.93	8.99
3	500K	2578556	6897	846975	358365	93779	2321226	2336825	257330	9.02	9.08
4	500K	2593372	7457	850802	358204	94400	2336680	2309969	256692	9.10	9.00
5	500K	2575801	3526	846084	357796	93203	2325545	2274265	250256	9.29	9.09
6	500K	2593200	2553	850490	358509	94628	2325253	2411376	267947	8.68	9.00
7	500K	2615759	4908	856797	358777	94962	2325171	2596550	290588	8.00	8.94
8	500K	2592367	5907	851030	358074	94776	2269254	2908704	323113	7.02	9.00
9	500K	2601970	5790	853820	357954	94281	2363970	2137242	238000	9.93	8.98

Table A.18. Graph information for n=5M and p=5

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	1M	5185351	5495	1237555	680094	309322	4244334	3759780	941017	4.51	4.00
1	1M	5169246	7764	1233503	681292	307664	4154457	4076466	1014716	4.09	4.02
2	1M	5169173	7457	1234182	682102	308969	4179820	3972141	989426	4.22	4.01
3	1M	5208959	4908	1239851	680793	308898	4071600	4484322	1122737	3.63	3.99
4	1M	5194337	5907	1239897	682077	310074	4091419	4448921	1117540	3.66	3.98

Table A.19. Graph information for n=3.75M and p=15

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	250K	1318262	2378	486683	181963	34682	1241923	1070074	76339	16.27	14.02
1	250K	1318524	2369	486377	182080	34679	1239382	1112981	79142	15.66	14.06
2	250K	1304817	2169	482465	181654	34320	1163962	1988085	140855	8.26	14.11
3	250K	1311079	5190	484858	182312	34629	1226641	1200514	84438	14.53	14.22
4	250K	1325067	3502	489370	181627	34979	1245215	1113677	79852	15.59	13.95
5	250K	1334485	11021	493222	182024	35142	1244482	1249400	90003	13.83	13.88
6	250K	1325859	4155	488694	181569	35093	1220491	1465637	105368	11.58	13.91
7	250K	1326434	5183	490141	181203	35032	1237498	1230137	88936	13.91	13.83
8	250K	1327333	4945	490682	181772	35051	1252831	1031509	74502	16.82	13.85
9	250K	1304728	1420	482123	181587	34503	1230421	1054565	74307	16.56	14.19
10	250K	1314063	2984	485709	181932	34432	1219563	1340709	94500	12.91	14.19
11	250K	1327378	8052	490224	181953	34814	1252124	1049645	75254	16.64	13.95
12	250K	1311939	1997	485475	181657	34693	1235502	1075524	76437	16.16	14.07
13	250K	1322794	6496	488203	181729	35041	1226510	1338514	96284	12.74	13.90
14	250K	1322865	5804	488653	181772	34710	1240348	1155922	82517	15.03	14.01

Table A.20. Graph information for n=2.5M and p=60

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	41667	214369	4071	111891	30873	1890	208342	349328	6027	34.57	57.96
1	41667	214956	3907	111509	30823	1839	211956	176665	3000	70.65	58.89
2	41667	211570	807	109864	30855	1872	208236	199487	3334	62.46	59.83
3	41667	212525	867	110450	30972	1865	207705	287357	4820	43.09	59.62
4	41667	213172	2331	110601	30755	1849	209841	197184	3331	63.00	59.20
5	41667	209859	642	108831	30838	1828	206494	197064	3365	61.37	58.56
6	41667	209995	1041	108962	30778	1768	206488	215049	3507	58.88	61.32
7	41667	207735	701	108083	30872	1794	205042	164742	2693	76.14	61.17
8	41667	214623	972	111106	30834	1903	211221	194514	3402	62.09	57.18
9	41667	207115	605	107548	30790	1858	204242	168760	2873	71.09	58.74
10	41667	212404	1124	110077	30882	1857	209012	197784	3392	61.62	58.31
11	41667	209557	582	108926	30982	1861	206463	187594	3094	66.73	60.63
12	41667	216913	3935	112776	30750	1863	214177	157616	2736	78.28	57.61
13	41667	215459	1877	111783	30825	1928	212174	188204	3285	64.59	57.29
14	41667	210054	679	109088	30882	1899	206951	186272	3103	66.69	60.03
15	41667	209350	1819	108610	30893	1856	205482	232603	3868	53.12	60.14
16	41667	214220	2001	110891	30969	1918	210952	194911	3268	64.55	59.64
17	41667	215211	1775	111920	30933	1898	211234	230217	3977	53.11	57.89
18	41667	217332	3444	112772	30856	1966	211913	318582	5419	39.11	58.79
19	41667	208878	533	108741	30922	1836	204487	265798	4391	46.57	60.53
20	41667	211055	990	109487	30789	1924	207894	182769	3161	65.77	57.82
21	41667	209300	841	108249	30763	1881	204776	271562	4524	45.26	60.03
22	41667	213296	5158	111333	30942	1875	210267	177915	3029	69.42	58.74
23	41667	214035	812	110725	30821	1859	211045	174519	2990	70.58	58.37
24	41667	207956	819	108108	30788	1787	204246	228421	3710	55.05	61.57
25	41667	210509	701	109084	30670	1862	207156	200541	3353	61.78	59.81
26	41667	212252	1030	110420	30826	1977	208440	221498	3812	54.68	58.11
27	41667	219748	2259	113500	30920	1901	216684	173928	3064	70.72	56.77
28	41667	208355	656	108239	30879	1868	205197	189164	3158	64.98	59.90
29	41667	213504	1535	110297	30803	1807	210798	163675	2706	77.90	60.49

Table A.20. Graph information for n=2.5M and p=60 (continued)

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
30	41667	230613	18785	121704	30967	2058	227279	175774	3334	68.17	52.72
31	41667	212630	1412	110153	30819	1939	208857	221135	3773	55.36	58.61
32	41667	209538	893	109067	30837	1824	206396	188022	3142	65.69	59.84
33	41667	212323	1033	110207	30768	1841	209588	162320	2735	76.63	59.35
34	41667	218332	4166	113147	30970	1980	214044	244002	4288	49.92	56.90
35	41667	211203	935	109311	30783	1859	207348	230147	3855	53.79	59.70
36	41667	212652	1069	110180	30898	1931	208900	226025	3752	55.68	60.24
37	41667	211879	1215	110094	31046	1825	208744	186901	3135	66.59	59.62
38	41667	212692	991	110581	30857	1859	209983	169820	2709	77.51	62.69
39	41667	214526	4106	111368	30895	1899	210093	250720	4433	47.39	56.56
40	41666	211422	1323	109857	30797	1844	208581	171541	2841	73.42	60.38
41	41666	210499	619	109105	30907	1925	207069	201904	3430	60.37	58.86
42	41666	208132	630	108236	30731	1811	203021	306415	5111	39.72	59.95
43	41666	213421	618	110304	30997	1833	210078	198358	3343	62.84	59.34
44	41666	208496	881	108399	30842	1829	204714	227655	3782	54.13	60.19
45	41666	213049	1926	110208	30932	1941	209061	231076	3988	52.42	57.94
46	41666	214395	1631	110895	30667	1910	210870	201386	3525	59.82	57.13
47	41666	210948	844	109130	30963	1889	207829	185592	3119	66.63	59.50
48	41666	217397	7710	113337	30951	1924	213945	198229	3452	61.98	57.42
49	41666	212979	1172	110337	30832	1931	209632	193016	3347	62.63	57.67
50	41666	212360	942	110086	31033	1910	207636	277113	4724	43.95	58.66
51	41666	207884	978	107683	30991	1799	204652	201482	3232	63.32	62.34
52	41666	215292	4335	111821	30841	1930	211954	189746	3338	63.50	56.84
53	41666	213154	877	110488	30912	1827	209338	228953	3816	54.86	60.00
54	41666	208365	843	108326	30859	1869	203745	270894	4620	44.10	58.64
55	41666	209543	934	108583	30860	1822	206746	171185	2797	73.92	61.20
56	41666	213512	986	110478	30793	1875	210340	192064	3172	66.31	60.55
57	41666	214140	1690	110575	30787	1977	211006	177823	3134	67.33	56.74
58	41666	211445	1038	110074	30787	1816	208671	166336	2774	75.22	59.96
59	41666	210861	2126	109338	30986	1869	207450	197128	3411	60.82	57.79

Table A.21. Graph information for n=2.5M and p=30

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	83334	429325	4071	195088	61296	6662	411353	517048	17972	22.89	28.77
1	83334	424095	867	192353	61401	6729	407610	478513	16485	24.73	29.03
2	83334	423031	2331	191745	61199	6578	409757	387670	13274	30.87	29.21
3	83334	417730	1041	189775	61264	6432	405200	373461	12530	32.34	29.81
4	83334	421738	972	191102	61182	6626	409370	357181	12368	33.10	28.88
5	83334	421961	1124	191605	61471	6592	409117	379020	12844	31.85	29.51
6	83334	432372	3935	196093	61124	6760	420357	339826	12015	34.99	28.28
7	83334	419404	1819	190436	61368	6542	405470	411912	13934	29.10	29.56
8	83334	429431	2001	194527	61474	6834	414821	417763	14610	28.39	28.59
9	83334	426210	3444	193510	61346	6748	406480	574460	19730	20.60	29.12
10	83333	420354	990	190090	61142	6746	404964	446625	15390	26.31	29.02
11	83333	427325	5158	193992	61363	6745	415198	346324	12127	34.24	28.56
12	83333	418466	819	190053	61035	6454	404327	421880	14139	28.60	29.84
13	83333	432000	2259	195507	61357	6800	418247	388541	13753	30.41	28.25
14	83333	421836	1535	190993	61263	6522	410145	347020	11691	35.08	29.68
15	83333	443244	18785	202308	61348	7011	429208	389981	14036	30.58	27.78
16	83333	421873	1033	191514	61172	6589	410045	344383	11828	34.67	29.12
17	83333	429482	4166	194284	61306	6797	413314	466131	16168	25.56	28.83
18	83333	424562	1215	192433	61525	6709	410728	405945	13834	29.69	29.34
19	83333	427219	4106	193948	61353	6637	413061	413316	14158	29.18	29.19
20	83333	421928	1323	191489	61304	6620	409488	367517	12440	32.92	29.54
21	83333	421541	630	191151	61312	6442	404614	496293	16927	23.90	29.32
22	83333	421554	1926	191040	61353	6706	406115	451074	15439	26.30	29.22
23	83333	425324	1631	192198	61201	6717	412154	380454	13170	31.29	28.89
24	83333	430378	7710	195339	61384	6772	416849	384510	13529	30.81	28.42
25	83333	420268	978	190398	61630	6586	404317	470558	15951	25.35	29.50
26	83333	428450	4335	194375	61332	6683	414075	411523	14375	28.81	28.63
27	83333	417925	934	189561	61297	6570	403182	434754	14743	27.35	29.49
28	83333	427652	1690	193165	61153	6758	415012	363557	12640	32.83	28.76
29	83333	422311	2126	192038	61372	6583	410041	357379	12270	33.42	29.13

Table A.22. Graph information for n=2.5M and p=20

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	125K	640892	4071	263925	91283	13924	603781	700729	37111	16.27	18.88
1	125K	635555	2331	261457	91288	13754	600753	658317	34802	17.26	18.92
2	125K	632344	1041	260573	91327	13525	603514	555074	28830	20.93	19.25
3	125K	629077	1124	259271	91451	13696	601258	535674	27819	21.61	19.26
4	125K	642422	3935	264301	91181	13926	615155	513952	27267	22.56	18.85
5	125K	638776	2001	262534	91528	13943	605370	635439	33406	18.12	19.02
6	125K	637264	3444	262492	91298	13882	598366	741219	38898	15.38	19.06
7	125K	636618	5158	261758	91275	13969	604797	602709	31821	19.01	18.94
8	125K	630717	1030	260061	91022	13715	598136	628759	32581	18.36	19.30
9	125K	641598	2259	263117	91323	13800	614755	508618	26843	22.90	18.95
10	125K	652788	18785	270121	91350	14210	622356	564977	30432	20.45	18.57
11	125K	641808	4166	263600	91216	13966	609295	614839	32513	18.74	18.91
12	125K	637240	1215	262275	91494	14011	607897	562965	29343	20.72	19.19
13	125K	636478	4106	262433	91330	13848	604750	603177	31728	19.06	19.01
14	125K	630041	881	259768	91352	13452	593255	707861	36786	16.13	19.24
15	125K	638380	1926	261536	91293	14005	606780	597099	31600	19.20	18.90
16	125K	642774	7710	264843	91611	13831	608623	645624	34151	17.82	18.90
17	125K	636326	4335	262012	91515	13629	604977	599334	31349	19.30	19.12
18	125K	631438	986	259657	91268	13798	599448	612744	31990	18.74	19.15
19	125K	636453	2126	262022	91293	13805	608892	523048	27561	22.09	18.98

Table A.23. Graph information for n=2.5M and p=15

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	166667	853415	4071	324063	120913	23298	784652	961254	68763	11.41	13.98
1	166667	840758	2331	319783	120771	22583	788990	735167	51768	15.24	14.20
2	166667	843699	1124	320515	120977	22924	793435	711147	50264	15.79	14.15
3	166667	851765	3935	323630	120823	23078	799785	725704	51980	15.39	13.96
4	166667	855636	3444	324939	121145	23495	786997	957636	68639	11.47	13.95
5	166667	847685	5158	321578	120785	23258	792761	765829	54924	14.43	13.94
6	166667	850472	2259	322991	120716	22965	794651	782476	55821	14.24	14.02
7	166667	865079	18785	329363	120814	23710	813401	711057	51678	15.74	13.76
8	166667	851363	4166	323265	120735	23120	795585	782748	55778	14.26	14.03
9	166667	851786	4106	323700	121134	23227	795641	791108	56145	14.17	14.09
10	166666	843469	1323	320593	120880	22911	784646	834354	58823	13.34	14.18
11	166666	846878	1926	320726	120890	23249	789513	802772	57365	13.76	13.99
12	166666	850646	7710	323280	121361	22963	791986	825888	58660	13.50	14.08
13	166666	846375	4335	321531	120808	23071	788077	817097	58298	13.52	14.02
14	166666	849963	2126	322847	120827	23042	800150	696033	49813	16.06	13.97

Table A.24. Graph information for n=2.5M and p=10

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	250K	1276447	4071	423888	178566	47189	1133346	1287858	143101	7.92	9.00
1	250K	1261421	1124	419643	178863	46441	1148350	1034326	113071	10.16	9.15
2	250K	1281198	3935	424972	178822	47585	1159619	1088485	121579	9.54	8.95
3	250K	1273882	5158	422850	178566	47480	1132508	1273273	141374	8.01	9.01
4	250K	1272315	2259	422511	178478	46991	1153242	1077728	119073	9.69	9.05
5	250K	1294596	18785	430162	178371	48018	1168272	1116437	126324	9.25	8.84
6	250K	1273718	4106	423683	178997	47106	1151818	1105313	121900	9.45	9.07
7	250K	1268421	1926	420836	178725	46764	1131525	1236450	136896	8.27	9.03
8	250K	1279100	7710	424761	179111	47017	1148114	1179472	130986	8.77	9.00
9	250K	1267891	2126	421005	178516	47131	1148505	1075957	119386	9.62	9.01

Table A.25. Graph information for n=2.5M and p=5

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	500K	2537868	4071	613988	340082	152876	2024996	2065484	512872	3.95	4.03
1	500K	2555080	5158	616240	339898	155060	2029062	2098693	526018	3.86	3.99
2	500K	2566911	18785	619965	339079	155154	2075838	1948489	491073	4.23	3.97
3	500K	2542139	4106	614132	340271	153619	2023578	2081998	518561	3.90	4.01
4	500K	2546991	7710	615140	340117	154074	2046370	2005180	500621	4.09	4.01

Table A.26. Graph information for n=1.25M and p=5

P_i	n_i	m_i	d_{max}	$\|V_{F_i}\|$	$\|V_{B_i}\|$	$\|V_{L_i}\|$	$\|F_i\|$	$\|B_i\|$	$\|L_i\|$	F/L	B/L
0	250K	1088020	15553	293998	189052	72885	877560	934530	210460	4.17	4.44
1	250K	1070269	7973	289154	189448	72215	846834	1006851	223435	3.79	4.51
2	250K	1591900	498625	532054	163456	132910	1275887	849500	316013	4.04	2.69
3	250K	1078105	7184	291271	188559	72716	844353	1035315	233752	3.61	4.43
4	250K	1067089	3828	288465	189232	72525	872379	890817	194710	4.48	4.58

REFERENCES

1. Baeza-Yates, R. and B. Ribeiro-Neto, *Modern Information Retrieval*, ACM Press, New York, 1999.

2. Berry, M. W. and M. Browne, *Understanding Search Engines*, SIAM, Philadelphia, 1999.

3. Kleinberg, J. M., "Authoritative Sources in a Hyperlinked Environment", *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms(SODA)*, pp. 668–677, 1998.

4. Lifantsev, M., *Rank Computation Methods for Web Documents*, Technical Report TR-76, ECSL, Department of Computer Science, SUNY at Stony Brook, 1999.

5. Page, L., S. Brin, R. Motwani and T. Winograd, *The PageRank Citation Ranking: Bringing Order to the Web*, Technical Report, Stanford University, 1998.

6. Brin, S. and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", *Proceedings of the 7th International World Wide Web Conference*, 1998.

7. Google Search Engine, http://www.google.com.

8. Page, L., *Method for Node Ranking in a Linked Database*, United States Patent 6285999, September 2001.

9. Haveliwala, T. H., *Efficient Computation of PageRank*, Technical Report, Stanford University, 1999.

10. Motwani, R. and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, United Kingdom, 1995.

11. Norris, J. R., *Markov Chains*, Cambridge University Press, United Kingdom, 1997.

12. Kumar, V., A. Grama, A. Gupta and G. Karypis, *Introduction to Parallel Computing : Design and Analysis of Parallel Algorithms*, Benjamin/Cummings Publishing Company, Inc., 1994.

13. Raghavan, P., *Link–based Ranking in Web Search Engines*, CS347 Lecture Notes, April 2001.

14. Richardson, M. and P. Domingos, "The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank", *to appear in Neural Information Processing Systems*, 2002.

15. Kleinberg, J., *The Small–World Phenomenon: An Algorithmic Perspective*, Technical Report 99-1776, Cornell Computer Science, 1999.

16. Kleinberg, J. M., R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, A. Tomkins and J. Wiener, "The Web as a Graph: Measurements, Models, and Methods", *Proceedings of the 5th International Conference on Computing and Combinatorics*, July 1999.

17. Brin, S., R. Motwani, L. Page and T. Winograd, "What can you do with a Web in your Pocket?", *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, Vol. 21, pp. 37–47, 1998.

18. Stevens, W. R., *Advanced Programming in the Unix Environment*, Addison–Wesley Publishing Company, 1993.

19. Karypis, G., K. Schloegel and V. Kumar, *PARMETIS: Parallel Graph Partitioning and Sparse Matrix Ordering Library version 2.0*, September 1998.

20. Barabási, A.-L. and R. Albert, "Emergence of Scaling in Random Networks", *Science*, Vol. 286, pp. 509–512, 1999.

21. Barabási, A.-L., R. Albert and H. Jeong, "Scale–free Characteristics of Random Networks: the Topology of the World Wide Web", *Physica A: Statistical Mechanics*

and its Applications, Vol. 281, pp. 69–77, 2000.

22. Albert, R., H. Jeong and A.-L. Barabási, "Diameter of the World Wide Web", *Nature*, Vol. 401, pp. 130–131, September 1999.

23. Broder, A., R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins and J. Wiener, "Graph Structure in the Web", *Proceedings of the 9th International World Wide Web Conference*, 2000.

24. Altavista Search Engine, http://www.altavista.com.

25. Hayes, B., "Graph Theory in Practice: Part 1", *American Scientist*, Vol. 88, No. 1, pp. 9–13, January–February 2000.

26. Hayes, B., "Graph Theory in Practice: Part 2", *American Scientist*, Vol. 88, No. 2, pp. 104–109, March–April 2000.

27. Dorogovtsev, S. N., J. F. F. Mendes and A. N. Samukhin, "WWW and Internet Models from 1955 till Our Days and "Popularity is Attractive" Principle", *e–print cond–mat/0009090*, September 2000.

28. Dorogovtsev, S. N., J. F. F. Mendes and A. N. Samukhin, "Structure of Growing Networks: Exact Solution of the Barabási–Albert's Model", *e–print cond–mat/0004434*, April 2000.

29. Shiode, N. and M. Batty, "Power Law Distributions in Real and Virtual Worlds", *Proceedings of INET 2000*, July 2000.

30. Adamic, L. A., *The Small World Web*, Technical Report, Xerox PARC, Palo Alto, CA, 1999.

31. Aiello, W., F. Chung and L. Lu, "A Random Graph Model for Massive Graphs", *Proceedings of 32nd Annual ACM Symposium on the Theory of Computing*, 2000.

32. Albert, R. and A.-L. Barabási, "Topology of Evolving Networks: Local Events and Universality", *e–print cond–mat/0005085*, May 2000.

33. Adamic, L. A. and B. A. Huberman, "The Web's Hidden Order", *Communications of the ACM*, Vol. 44, No. 9, September 2001.

34. Adamic, L. A., R. M. Lukose, A. R. Puniyani and B. Huberman, "Search in Power–Law Networks", *e–print cond–mat/0103016*, March 2001.

35. Tadić, B., "Dynamics of Directed Graphs: the World–Wide Web", *Physica A: Statistical Mechanics and Its Applications*, Vol. 293, pp. 273–284, 2001.

36. Faloutsos, M., P. Faloutsos and C. Faloutsos, "On Power–Law Relationships of the Internet Topology", *Proceedings of ACM SIGCOMM*, 1999.

37. Jin, C., Q. Chen and S. Jamin, *Inet: Internet Topology Generator*, Technical Report CSE–TR–433–00, University of Michigan at Ann Arbor, 2000.

38. Medina, A., A. Lakhina, I. Matta and J. Byers, *BRITE: Universal Topology Generation from a User's Perspective*, Technical Report BUCS–TR–2001–003, Boston University, April 2001.

39. Kirkpatrick, S. and E. Stoll, "A Very Fast Shift–Register Sequence Random Number Generator", *Journal of Computational Physics*, Vol. 40, pp. 517–526, 1981.

40. Lever, C. and P. Honeyman, *Linux NFS Client Write Performance*, CITI Technical Report TR-01-02, University of Michigan, September 2001.