Efficient Sampling from Random Web Graphs and its Application

by

Yan Zhuang

BSc, University of Winnipeg, 2006

A Master Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

in the Department of Computer Science

University of Victoria

Efficient Sampling from Random Web Graphs and its Application

by

Yan Zhuang

BSc, University of Winnipeg, 2006

Supervisory Committee

Dr. V. King, Supervisor (Department of Computer Science)

Dr. B. Kapron, Member (Department of Computer Science)

Dr. K. Wu, Member (Department of Computer Science)

Supervisory Committee

Dr. V. King, Supervisor (Department of Computer Science)

Dr. B. Kapron, Member (Department of Computer Science)

Dr. K. Wu, Member (Department of Computer Science)

Abstract

This thesis presents space-efficient algorithms to sample from random web graphs generated by two important stochastic graph models based on concept of copying: the linear copy model and the hostgraph model. The goal is to avoid constructing the entire random graph, and instead use an amount of space nearer to the desired (smaller) sample size. The efficiency of our algorithms is achieved by refraining from making unnecessary random decisions when constructing the sample. The construction of a sample subgraph from a random graph with n nodes and k outgoing links on each node based on the linear copying model uses an expected $O(k \ln n)$ words for each node in the sample subgraph. The construction of a sample subgraph from a random graph with n nodes based on the hostgraph model uses, for any small sample size, an expected $n + o(n)$ words.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank all the persons who provided help throughout this work. In particular, I would like to thank my supervisor Valerie King for her suggestions and support. I also thank my wife for her encouragement.

Chapter 1

Introduction

The World Wide Web can be viewed as a directed graph in which the pages (or web sites) are nodes and the hyperlinks between them are edges. This induced graph is referred as the *web graph*. The techniques described in this thesis were developed to facilitate our own exploration of the link structure of the web graph (KYZ08), and hopefully can be useful to the many others who study the link structure of the web graph.

In our previous research(KYZ08), we have discovered some special patterns in the Chinese web graph, e.g. mutual links. We hypothesized that these special patterns represent a special social interaction between Chinese web sites, which is not captured by the existing random graph models that model the web.

To demonstrate our hypothesis, we compared samples of randomly generated graphs with samples of the actual web graph. Naively, we can implement this by generating random graphs of size similar to the actual web graph, and then applying the same sampling technique to the random graphs as used for sampling the actual web graph. As the representation of the web graph and therefore the stochastic model of the web graph are very large, computations necessarily involve external memory references and can be very time consuming. The goal of the research described here is to produce samples of large random graphs in much smaller space than needed to represent such graphs, i.e., small enough to be stored in main memory. In particular, we show space-efficient techniques for sampling using typical sampling methods from random graphs generated by commonly used stochastic models: the linear copying model and the hostgraph model. The samples we generate are provably distributed identically to the samples generated with the naive method.

The main idea of our approach is to refrain from making random choices which are not needed to produce the sample. This thesis presents two basic techniques to be used as subroutines in the various sampling methods, which are called NEC and NEH. Let n be the number of nodes in the random graph. We assume a word is large enough to hold the name of a node. For constructing a sample subgraph from a random graph with k outgoing links on each node based on the copying model, the NEC technique only used $O(k \ln n)$ words per node in the sample subgraph. For the hostgraph model, we give the NEH method, which uses, for any small sample size, $n + o(n)$ words.

The linear copying and hostgraph models and the typical sampling techniques

along with other related background are described in Chaper 2. Techniques and analytical results for the linear copying model and the hostgraph model are described in Chapers 3 and 4, respectively. Chaper 5 shows some experimental findings. Chaper 6 gives an application of one of these new techniques. Chapter 7 gives the conclusion.

Chapter 2

Background

2.1 The Web as a Graph

2.1.1 Degree Distribution of the Web Graph

Many experiments have been conducted regarding the structure of the web graph (e.g., see (BKM$^+$00), (AJB99)) One important property of the web graph is that it follows the power-law in the in-degree and out-degree distribution of nodes (See (BKM$^+$00), (KRRT99), (BA99)). A power-law distribution means that the fraction of nodes with degree i is proportional to $\frac{1}{i^c}$ for some positive constant c. Kumar et al. (KRR$^+$) showed the exponents of the power law distribution for in-degree and out-degree in the web graph constructed by using web pages as the nodes are 2.1 and 2.72 respectively. Bharat et al. (BCHR01) showed these exponents are 1.62 and 1.67 on the web graph which is constructed by using the web sites as the nodes.

2.1.2 The Usage of the Link Structure of the Web Graph

Web Communities One of the earlier uses of link structure is found in the analysis of web communities, where properties such as cliques are identified. Kumar et al. (KRRT99) defined topic enumeration, which seek to enumerate all topics that are well represented on the web by defining dense bipartite cores. The intuition is that a community emerges when many (hub) pages link to many of the same (authority) pages. Another more general definition of web community is a set of web pages that link (in either direction) to more web pages in the community than to pages outside of the community(FLG00)(IK03). Communities like these are identified by determining components separated by minimum cuts in the web graph.

PageRank PageRank is another important application which makes usage of the link structure of the web graph. The PageRank algorithm calculates the stationary distribution of a random walk on the web graph which is used as a query-independent measure of the importance of web pages, based on the notion of peer-endorsement: a hyperlink from page A to page B is an endorsement of page B's content by page A's author.

2.2 Random Graph Models of the World Wide Web

Many stochastic models have been created to generate random graphs that have certain attributes which resemble the structure of the web (see (Bon04) and (ACL02)

for surveys). Two main concepts have been used: preferential attachment (BA99) and copying (KRR$^+$00).

2.2.1 The Preferential Attachment

The BA Model The first evolving graph model explicitly designed to model the web was given in (BA99). The idea behind the model is that new nodes are more likely to link to existing nodes with high degree. This model is referred as *the BA model or the preferential attachment model*. The detail of the BA model is as follows:

1. At each time step, a new node with k edges is added to the existing graph.
2. When choosing the destination of each edge, the probability that a node with degree d_i will be chosen is $\frac{d_i}{\sum_j d_j}$.

The authors of (BA99) analyzed such a model, and concluded that it generates graphs whose in-degree distribution follows a power law with a constant exponent of three, independent of parameters chosen.

The Generalized Preferential Attachment Model Dorogoytsev et al.(DMS00) generalized the BA model as follows: at each time step a new node is added and m new directed links are added to new nodes and existing nodes. Each site is assigned with an initial attractiveness score. As time increases, the attractiveness score of a site S is equal to the initial attractiveness score plus the in-degree of S. The initial attractiveness score governs the probability for newly added sites to get new links, but afterwards the attractiveness of a site increases as the in-degree of the site increases. The original preferential attachment model (BA99) specifies that at each time step, directed links are only assigned to newly added nodes, whereas in the model by Dorogovtsev et al., directed links are added to both old and new nodes. We therefore refer it as *the generalized preferential attachment model*.

2.2.2 The Copying

The Linear Copying Model The first model based on the concept of *copying* was introduced by Kleinberg et al. (KKR$^+$99). Later, the model was rigorously analyzed by Kumar et al. (KRR$^+$00) and it is referred as *the linear copying model*. The copying mechanism is motivated by the intuition that authors of web pages will randomly find a web page and then copy some portion of the links to their own page. A detail description of the linear copying model is given in (KRR$^+$00) and we summarize it as follows: at each time step, a new node with k outgoing edges is added. The k edges are added as following:

1. Pick a prototype node uniformly at random among all existing nodes.
2. With probability α, the destination of the i^{th} outgoing link is chosen uniformly at random from all the existing nodes. With the remaining probability $1-\alpha$, the destination of the i^{th} outgoing link is chosen to be the destination of the i^{th} outgoing link picked from the prototype.

The HostGraph Model A notable variation of the copying model is *the hostgraph model* by Bharat et al. (BCHR01) which models the linking structure between web sites. The hostgraph model is defined as follows: at each time step, with probability β, a new node with k edges is added. With the remaining probability $1-\beta$, k edges are added to an existing node chosen uniformly at random. The k edges are added as following:

1. Pick a prototype uniformly at random among all existing nodes. Then pick k outgoing edges from the prototype uniformly at random without replacement. Then we randomly number them from 1 to k.

2. With probability α, the destination of the i^{th} outgoing link is chosen uniformly at random from all the existing nodes. With the remaining probability $1-\alpha$, the destination of the i^{th} outgoing link is chosen to be the destination of the i^{th} outgoing link picked from the prototype.

In the hostgraph model, the authors modify the copying model by introducing a new parameter β, depending on which the links can be added between old nodes. This new parameter β can be interpreted as the cost of establishing a new host versus creating new links from an existing host. Random graphs with different power law distributions can be obtained from the hostgraph model by choosing different β, the smaller the β value, the lower the power law exponent. The α and the β typically range from 0.1 to 0.9. We assume they are constants.

2.3 Graph Sampling Method

In graph sampling we obtain a subgraph by taking samples (nodes and edges) from a large target graph such that the subgraph will have similar properties as the target graph. There are mainly three families of graph sampling methods: sampling by randomly selecting nodes, sampling by randomly selecting edges and sampling by exploration (LF06).

2.3.1 Sampling by Randomly Selecting Nodes and by Randomly Selecting Edges

The ideas of these two families of sampling algorithm are very simple and straightforward. Two typical examples in these two families are the Random Node (RN) (SWM05) and the Random Edge (RE) sampling methods.

In RN, a sample graph is obtained by selecting nodes uniformly at random and including the edges if both of the endpoints are in the sample graph. The sample graph obtained by the RN method does not match the degree distributions of the original graph.(SWM05).

In RE, a sample graph is obtained by selecting edges uniformly at random and including all the endpoints of the selected edges. The sample graph obtained by the RE method is very sparsely connected, and therefore it loses the community structures of the original graph.

There are many variations in these two families of sampling methods but none of them perform well on the real world large graphs (LF06).

2.3.2 Sampling by Exploration

The three typical examples in sampling by exploration are Random Node Neighbor (RNN), Random Walk (RW), and Random Jump (RJ) sampling methods.

In RNN, we first choose a node uniformly at random and then we obtain all its outgoing neighbors by exploring all its outgoing edges. The sample graph obtained by the RNN method agrees on the outdegree distribution of the original graph but it does not match the indegree distribution. The RNN can also capture the connectivity features such as the mutual links between a node and its neighbors.

The RW and the RJ are based on the same idea. Both simulate a random walk on the target graph from a randomly selected node; the nodes and the edges along the path of this walk form the sample graph. The only difference is with some probability c, the RJ will jump to a randomly selected node but the RW will go back to the starting node.

Chapter 3

Sampling from the Linear Copying Model

3.1 Intuition

In the linear copying model, a new node is established at each time step by a sequence of coin flips. We first choose a prototype node uniformly at random from the existing nodes. Then for each of its outgoing links, a biased coin is flipped for determining whether to copy the corresponding edge from the prototype or to link to a random destination. Figure 3.1 illustrates an example of this process. The process starts with two nodes in the graph and another 16 nodes are added in the rest of 16 time steps. Each column in the table corresponds to a sequence of coin flipping at each time step. The node field represents the new node being added. The prototype field represents the chosen prototype node. The destinations of outgoing links are recorded in the random bit field, 0 representing to copy the corresponding outgoing link of the prototype node and a non-zero positive integer r representing the random selected destination of the link.

Generating a random graph according to the copying model is equivalent to generating the table in Figure 3.1 by making all the random choices in each time step. The most straightforward way to sample from a random graph based on the linear copying model is to first generate an instance of random graph using space equal to $\theta(nk)$ words and then choose a subgraph from it. We refer this as the *regular random graph sampling method*. We ask if we can do better than this, if there is an efficient way in terms of both time and space to sample from such random graphs. We show the answer is yes, using the principle of deferred decisions.

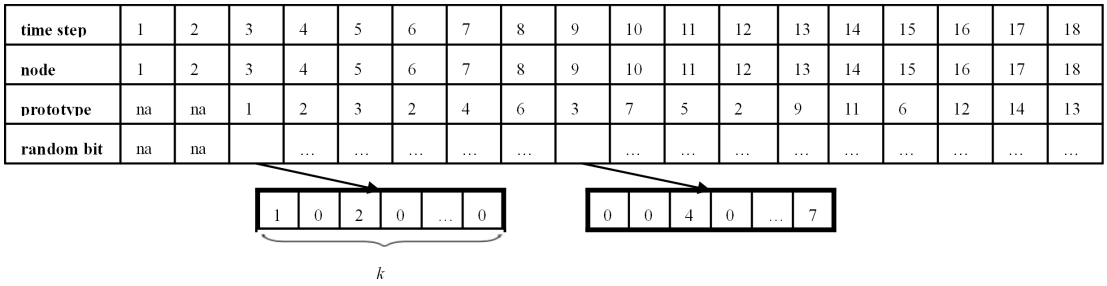

Figure 3.1: Generation of the copying model, where we use non-zero positive integers to represent nodes in the graph and each column corresponds to one time step.

The *principle of deferred decisions* states that rather than making all random choices in advance, in each step of the process, the algorithm only makes random choices as needed (MR95). The regular random graph sampling method as motioned before can be considered as an algorithm which first generates the random graph by making all the random choices in advance , and then takes samples from the generated graph. Instead, based on the principle of deferred decisions, we only generated a node and its outgoing links if the node is needed for constructing the sample graph. This will serve as a general framework of our efficient sampling method and we will elaborate it in detail in the following sections.

We introduce some basic definitions. In the linear copying model, a node joins the graph by linking to some existing nodes ,and we refer this process as *node expanding*. We say *a node is expanded* if the node along with its entire set of outgoing links is already in the graph, and *a node is partially expanded* if the node along with some of its outgoing links is already in the graph.

Before we explore the efficient ways to expand nodes to obtain sample graphs based on different sampling strategies, we first introduce an algorithm, *NEC (Node Expanding based on linear Copying model)*, which can expand any given node by only expanding a subset of nodes of a random graph.

3.2 NEC algorithm

Algorithm 1 The NEC algorithm

Require: parameters needed by the linear copying model, the node r needs to be expanded

Ensure: node r is expanded

1: choose an integer p uniformly at random from 1 to $r-1$ as the prototype of r
2: **if** node p has not yet been expanded **then**
3: expand node p using NEC algorithm
4: expand node r by copying links from node p according to the definition of the linear copying model

In the NEC algorithm, we use positive integer i to represent the i^{th} node added to the random graph. The NEC algorithm only expands a node when necessary. That is when expanding node r, the algorithm holds off on all the random decisions for other nodes and first determines the prototype node p by choosing an integer uniformly at random from 1 to $r-1$. If the prototype node p has been expanded, the algorithm determines the outgoing links of the node r by copying links from its prototype node p and terminates the process. If the prototype node p has not yet been expanded, the algorithm expands the node p by calling the NEC algorithm again. The recursive calls terminate until an expanded node is selected as the prototype node.

Figure 3.2 illustrates a recursive expanding process of a node n, where there are two nodes in the graph initially. To demonstrate the efficiency of the NEC algorithm, we need to show the expected number of nodes that need to be expanded in this process.

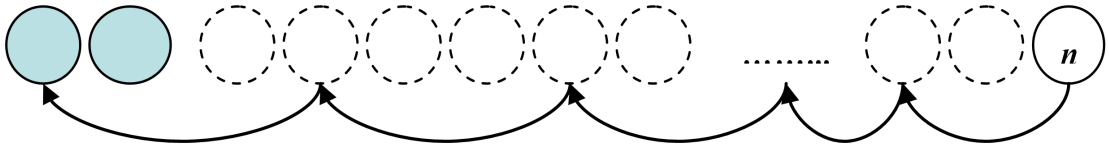

Figure 3.2: Node expanding process.

Theorem 3.2.1 *Define S_i to be the number of nodes need to be expanded for expanding the i^{th} node in the random graph based on the linear copying model. Then to expand the n^{th} node in the graph, the expectation $E(S_n) < 2 \ln n$*

Proof Without loss of generality, let us assume there is only one node in the initial graph. The problem can be viewed as a special case of the *moving particle problem*(MR95): a particle changes its position at discrete time steps and is always at a positive integer. The particle starts at position $n > 1$ and it moves backwards (towards position 1) x positions at each time step, where x is a random variable ranging over 1 and $n-1$. The particle stops when it is at position 1. We use random variable T to denote the number of steps in which the particle reaches position 1. Let $g(n)$ be a monotone non-decreasing function. (MR95) shows that if $E(x) \geq g(n)$ then $E(T) \leq \int_1^n dx/g(x)$.

The node expanding process is exactly an instance of the moving particle problem, where the random variable x is distributed uniformly between 1 and $n-1$ and therefore $E(x) = n/2$. The expected number of nodes that need to be expanded $E(S_n)$ is then the same as E(T) and so we have $E(S_n) \leq \int_1^n \frac{2}{x} dx < 2 \ln n$. ∎

3.3 Sampling with the NEC Algorithm

In the Random Node Neighbor (RNN) sampling method, a sample node is selected uniformly at random along with all its neighbors reachable by its out-going links. Since our NEC algorithm is able to expand any node based on the linear copying model, the implementation of the RNN sampling method is therefore very simple: For every sample node in the sample set S (if the number of nodes in the random graph is n, S contains a set of integers chosen uniformly at random between 1 and n without replacement), we expand the sample node and then expand all its outgoing neighbors.

The ideas of Random Walk (RW) and Random Jump (RJ) sampling method are very similar. Both explore the graph by performing a random walk starting from a randomly selected node. The only difference is with some probability c, the RJ will jump to a randomly selected node but the RW will go back to the starting node. Using the NEC algorithm, we expand a node only when it is the next stop of the random walk.

Theorem 3.3.1 *Let n be the total number of nodes in a random graph generated by the linear copying model and suppose k edges are added at each time step. Based on the NEC algorithm, for constructing the sample subgraph, the RNN, the RW and the RJ use an expected $O(k \ln n)$ words for each sample node in the sample subgraph.*

Proof According to Theorem 3.2.1, $E(S_i) \leq E(S_n) \leq 2 \ln n$ for all $i \leq n$. Since $O(k)$ words are needed to store an expanded node, the NEC algorithm uses an expected $O(k \ln n)$ words for expanding any node i in the random graph. In the RNN, the RW and the RJ, the NEC algorithm is only applied to those nodes that are in the sample subgraph, therefore these methods uses an expected $O(k \ln n)$ words for each sample node in the sample subgraph. ∎

Note that for the RNN, the size of the sample subgraph generated using sample set of size s can be as large as $s(k+1)$.

Chapter 4

Sampling from the Hostgraph Model

4.1 Intuition

In the hostgraph model, links can be added to the same node at different time steps. This presents added difficulties for our method. For example, there needs to be a consistent view of the size of the random graph at each time step.

Figure 4.1 shows an example of the node expanding process based on the hostgraph model. In the example, the process begins with two nodes in the initial graph. Another seven different nodes are added in 18 time steps. The table in Figure 4.1 is similar to the table in Figure 3.2, but we omitted the random bit field. It specifies the selected source node and the prototype node at each time step. As illustrated in Figure 4.1, for expanding the node 9, all the time steps where node 9 is selected as source node need to be considered, in this case time step 14 and 17. When expanding node 9 at time step 17, node 5 is chosen as the prototype. Since the outgoing links of node 5 are also undetermined, we need to expand node 5 before we can copy its outgoing links. Note that we only expand node 5 up to the time step 17 and the node 5 at time step 18 will be ignored since the links added at time step 18 do not exist when node 9 copies at time step 17.

The example in Figure 4.1 suggests a recursive algorithm which expands a node in the random graph by only expanding the nodes which serve as prototype nodes for the requested node and their prototype nodes. We can even do better than this.

Figure 4.2 illustrates the recursive process suggested in the above example. As shown in Figure 4.2, for determining k outgoing links (in this case, $k = 2$) of an occurrence of a node, all we need is to find k paths (colored in red) to the leaf nodes and copy k links from them. Therefore, instead of expanding all the non-leaf nodes in the tree, we only need to partially expand those occurrences of nodes which are

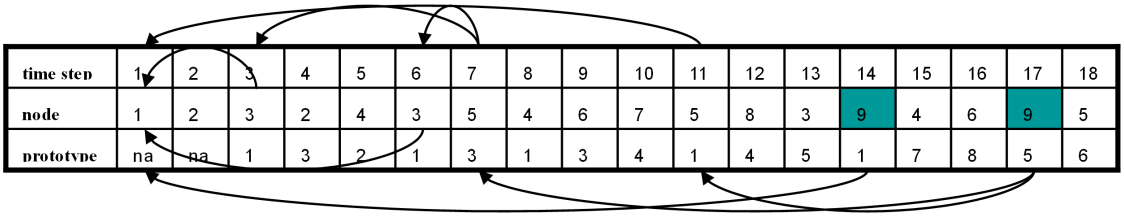

Figure 4.1: Node expanding based on hostgraph model.

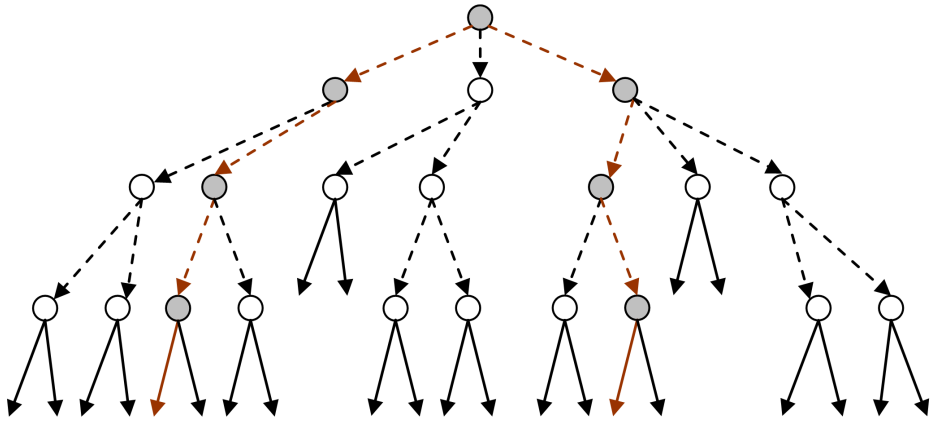

Figure 4.2: A tree representation of node expanding based on hostgraph model. Each node represents a node at a particular time step, each solid line represents an actual outgoing link and each dash line indicates that the child is an occurrence of the prototype node of the parent node

on the paths to the corresponding leaf nodes. Following this idea, we introduce our NEH algorithm in the next section.

4.2 NEH algorithm

Definitions: consider the table in Figure 4.1. For any positive integer t, let X_t be the set of distinct nodes in the table appearing in time steps $1, ..., t$. The random graph grows by adding new nodes at certain time steps. Let x_r be the r^{th} distinct node appear in the random graph. Since the same node may appear at more than one position in the table, once for each time it is chosen to acquire new edges, we use $x_r^{(i)}$ to denote the i^{th} occurrence of node x_r and $T(x_r^{(i)})$ to denote the time step when $x_r^{(i)}$ occurs. We use the notation $link(x_r^{(i)}, w)$ to denote the w^{th} outgoing link added to $x_r^{(i)}$. We say *an occurrence of a node is determined* if we know the time step when it occurs, and we say *a link is determined* if the destination of the link has been fixed. In our algorithm, we store all the occurrences of nodes that have been determined along with the time steps when they occur in a linklist A. For each expanded or partially expanded occurrence of node, we maintain a linklist to store all the determined links. In addition, we use $LP(x_p)$ to denote the last time step in A when the node x_p is selected as the prototype node and if node x_p has not been chosen as the prototype node yet, then $LP(x_p) = T(x_p^{(1)})$. For calculating the probability that a node x_p occurs at certain time step, we use $C(i)$ to denote the number of coin flips that have been conducted on time step i. Note that both $LP(x_p)$ and $C(i)$ can be easily obtained from A.

There is one issue that needs to be addressed before we get into the details of the NEH algorithm. When we determine the selected node and its prototype at a particular time step, we need to know the size of the random graph in terms of number of distinct nodes at that time step. This information is well implied in the

random graphs based on the linear copying model since the i^{th} distinct node must appear at the i^{th} time step. In the hostgraph model, the size of the random graphs in terms of number of distinct nodes can be different at the same time step unless we determine the first occurrences of all the distinct nodes. To do this, we initially determine the first occurrences of all the distinct nodes and store the time steps in an array I. If there are n distinct nodes in the random graph, then the array I uses n words.

The NEH algorithm determines a particular link on a particular occurrence of a node, for example $link(x_r^{(i)}, w)$. In the NEH algorithm, we first determine the prototype node x_p for $x_r^{(i)}$ by choosing one of the existing nodes uniformly at random. We then determine and store the occurrences of x_p by flipping a biased coin with probability $\frac{1}{|X_{t-1}|-C(t)}$ for all time steps t where $LP(x_p) < t < T(x_r^{(i)})$. Then we choose one of the occurrences of x_p, $x_p^{(m)}$, uniformly at random from all the occurrences of x_p between time step $T(x_p^{(1)})$ and $T(x_r^{(i)})$, and we randomly select one of its outgoing links, $link(x_p^{(m)}, j)$. If $link(x_p^{(m)}, j)$ is not determined, the algorithm recurses from $link(x_p^{(m)}, j)$ until a determined link is selected. The NEH algorithm is described in Algorithm 2.

Theorem 4.2.1 *Let $S(t)$ be the number of occurrences of nodes that need to be partially expanded by the NEH to establish one outgoing link of an occurrence of a given node at time step t. The expectation $E(S(t)) < 4 \ln t$.*

Proof We assume there is only one node in the initial graph, appearing at time step 1. Then this can be again considered the moving particle problem: time step t is the starting position of the particle. The particle can go back u time steps, where u is a random variable representing the number of time steps between the time step t and the time step $T(x_p^{(m)})$.

Algorithm 2 The NEH algorithm

Require: k, α, β needed by hostgraph model, linklist A, array I, the requested link $link(x_r^i, w)$

Ensure: $link(x_r^i, w)$ is determined

Toss a coin X with probability α
if $X = 1$ **then**
  choose destination uniformly at random from $X_{T(x_r^{(i)})-1}$ for $link(x_r^i, w)$
else
  if node x_r^i's prototype node has not yet been determined **then**
    choose x_p uniformly at random from $X_{T(x_r^{(i)})-1}$ as the prototype node of x_r^i
  else
    $x_p \leftarrow$ the prototype node of x_r^i
  if $T(x_r^{(i)}) > LP(x_p)$ **then**
    for all time step t between $LP(x_p)$ and $T(x_r^{(i)})$ at which the selected node has not yet been determined **do**
      Toss a coin X with probability $\frac{1}{|X_{t-1}|-C(t)}$
      if $X = 1$ **then**
        determine the selected node to be x_p at time step t and store it in A
  repeat
    choose a $x_p^{(m)}$ uniformly at random from all the occurrences of x_p up to $T(x_r^{(i)})$
    choose an integer j uniformly at random from 1 to k
  until $link(x_p^{(m)}, j)$ has not been copied by $x_r^{(i)}$
  if $link(x_p^{(m)}, j)$ has not been determined **then**
    apply NEH on $link(x_p^{(m)}, j)$
  $link(x_r^{(i)}, w) \leftarrow link(x_p^{(m)}, j)$ and store it in A.

Claim 1: *Let P_f be the event that $T(x_p^{(m)}) = f$. Then $Pr(P_f) \geq Pr(P_{f'})$ for all $T(x_p^{(1)}) < f < f' < t$.*

Proof of Claim 1 Since $x_p^{(m)}$ is chosen uniformly at random from all the occurrence of the prototype node x_p from $T(x_p^{(1)})$ to t, we have

$$Pr(P_f | x_p \; occurs \; at \; time \; step \; f) = Pr(P_{f'} | x_p \; occurs \; at \; time \; step \; f')$$

for all $T(x_p^{(1)}) < f, f' < t$. It is clear from the hostgraph model that

$$Pr(x_p \; occurs \; at \; time \; step \; f) \geq Pr(x_p \; occurs \; at \; time \; step \; f')$$

for all $T(x_p^{(1)}) < f < f' < t$. Then Since

$$Pr(P_f) = Pr(P_f | x_p \; occurs \; at \; time \; step \; f) * Pr(x_p \; occurs \; at \; time \; step \; f)$$

, we have $Pr(P_f) \geq Pr(P_{f'})$ for all $T(x_p^{(1)}) < f < f' < t$. ∎

Claim 2: *$E(u|$node x_p is chosen as the prototype node$) > (t - T(x_p^{(1)}) + 1)/2$.*

Proof of Claim 2

$$E(u|\text{node } x_p \text{ is chosen as the prototype node})$$

$$= \sum_{f=T(x_p^{(1)})}^{t-1} Pr(P_f) * (t-f) \;\; [by\ definition]$$

Reordering the terms

$$= \sum_{i=0}^{(t-2-T(x_p^{(1)}))/2} Pr(P_{T(x_p^{(1)})+i}) * (t - T(x_p^{(1)}) - i) + Pr(P_{t-1-i}) * (1+i)$$

Let $a = Pr(P_{T(x_p^{(1)})+i}),\ b = Pr(P_{t-1-i}),\ x = (t - T(x_p^{(1)}) - i)$ *and* $y = (1+i)$

$$= \sum_{i=0}^{(t-2-T(x_p^{(1)}))/2} ax + by$$

$$= \sum_{i=0}^{(t-2-T(x_p^{(1)}))/2} \frac{ax+by}{a+b} * (a+b)$$

$$= \sum_{i=0}^{(t-2-T(x_p^{(1)}))/2} \frac{\frac{(a+b)(x+y)}{2} + \frac{(a-b)(x-y)}{2}}{a+b} * (a+b)$$

by Claim 1, $a > b$ *and by observation* $x > y$

$$> \sum_{i=0}^{(t-2-T(x_p^{(1)}))/2} (x+y) * (a+b)/2$$

$$> \sum_{i=0}^{(t-2-T(x_p^{(1)}))/2} (t - T(x_p^{(1)}) + 1) * (Pr(P_{T(x_p^{(1)})+i}) + Pr(P_{t-1-i}))/2$$

$$= ((t - T(x_p^{(1)}) + 1)/2) \sum_{f=T(x_p^{(1)})}^{t-1} Pr(P_f)$$

$$= (t - T(x_p^{(1)}) + 1)/2$$

∎

Claim 3: *The expected time step of the first occurrence of the chosen prototype node* $E(T(x^{(1)}_{i\in_R[1,|X_{t-1}|]})) = t/2$.

Proof of Claim 3 Let $Pr(j)$ represent the event that a new node occurs at time step j and it is chosen. Since a new node is equally likely to appear at all time steps and each time step is also equally likely to be chosen given a new node occurs on it, we have $Pr(j) = Pr(j')$ for all $1 \leq j, j' \leq t-1$. We therefore have $Pr(j) = 1/(t-1)$ for all the value of j. Then we have: $E(T(x^{(1)}_{i\in_R[1,|X_{t-1}|]})) = \sum_{j=1}^{t-1} j * Pr(j) = t/2$. ∎

Claim 4: $E(u) > t/4$.

Proof of Claim 4 Since the inequality in Claim 2

$$E(u|node\ x_p\ is\ chosen\ as\ the\ prototype\ node) > (t - T(x^{(1)}_p) + 1)/2$$

holds for all the choices of node x_p, if we take expectation on both side, the inequality still holds. Then we have

$$E(u) > E((t - T(x^{(1)}_{i\in_R[1,|X_{t-1}|]}) + 1)/2) > (t/2) - E(T(x^{(1)}_{i\in_R[1,|X_{t-1}|]}))/2$$

By Claim 3, we get $E(u) > t/4$. ∎

Since $E(u) > t/4$, if we associate this with the moving particle problem, we get $E(S(t)) < \int_1^t \frac{4}{x} dx < 4 \ln t$, concluding the proof of the theorem. ∎

Note that the NEH algorithm only determines one outgoing link on an occurrence of a node. To fully expand a node, we need to first determine all the occurrences of the node and then expand each occurrence of this node. For expanding a particular occurrence of the node, we can keep calling the NEH algorithm on the same occurrence of the node until we get its entire set of outgoing links.

Theorem 4.2.2 *Consider a hostgraph model with parameter k and β. Let t be the total number of time steps to generate the random graph based on the hostgraph model. Let $N(x_r)$ be the total number of occurrences of nodes that need to be partially expanded for expanding all the occurrences of a node x_r in the random graph. The expectation $E(N(x_r)) = O(((1-\beta)k \ln^2 t)/\beta)$.*

Proof According to Theorem 4.2.1, $E(S(t)) < 4 \ln t$. Let $G(t')$ be the number of occurrences of nodes which need to be partially expanded for obtaining all k outgoing links of an occurrence of a node at time t'. Then $E(G(T(x^{(i)}_r))) \leq E(G(t)) = E(\sum_{j=1}^{k} S(t)) < 4k \ln t$

Claim 5: *Let $H(x_r)$ be the number of occurrences of node x_r in all time steps. The expectation $E(H(x_r)) = O((1-\beta) \ln t/\beta)$.*

Proof of Claim 5 Assume there are v distinct nodes in the random graph in total. It is clear from the hostgraph model that the first node in the random graph is more likely to be chosen to acquire links than any other node. So we have $E(H(x_r)) \leq$

$E(H(x_1))$. Let Y_i be an indicator variable such that $Y_i = 1$ if there is an occurrence of node x_1 at time step i and otherwise $Y_i = 0$. By definition,

$$E(Y_i) = Pr(x_1 \; occurs \; at \; time \; step \; i) = (1-\beta)/|X_i|$$

We can then write:

$$E(H(x_1)) = E(\sum_{i=1}^{t} Y_i) = \sum_{i=1}^{t} (1-\beta)/|X_i|$$

Let random variable c_i represent the number of time steps in which there are i distinct nodes, where i ranges over 1 and v. It is easy to see that $E(c_i) = 1/\beta$ for all i. Then we have:

$$E(H(x_1)|c_1, c_2, ..., c_v) = \sum_{i=1}^{v} (1-\beta)c_i/i$$

Take expectation on both side of the equation we get:

$$\begin{aligned} E(H(x_1)) &= E(\sum_{i=1}^{v} (1-\beta)c_i/i) \\ &= (1-\beta)\sum_{i=1}^{v} E(c_i/i) \\ &= ((1-\beta)/\beta)(\ln v + o(1)) \\ &= O(((1-\beta)/\beta)\ln t) \end{aligned}$$

∎

Since $N(x_r) = \sum_{i=1}^{H(x_r)} G(T(x_r^{(i)}))$, we have:

$$E(N(x_r)) = E(\sum_{i=1}^{H(x_r)} G(T(x_r^{(i)}))) = E(E(\sum_{i=1}^{H(x_r)} G(T(x_r^{(i)}))|H(x_r))) \tag{4.1}$$

$$= \sum_{h=1}^{t} E(\sum_{i=1}^{H(x_r)} G(T(x_r^{(i)}))|H(x_r) = h) * Pr(H(x_r) = h) \tag{4.2}$$

$$= \sum_{h=1}^{t} Pr(H(x_r) = h) * \sum_{i=1}^{H(x_r)} E(G(T(x_r^{(i)}))|H(x_r) = h) \tag{4.3}$$

$$\leq \sum_{h=1}^{t} Pr(H(x_r) = h) * E(G(t)|H(x_r) = h) \tag{4.4}$$

$$= E(H(x_r)) * E(G(t)) = O(((1-\beta)k\ln^2 t)/\beta) \tag{4.5}$$

(4.4) → (4.5) is based on the observation that $G(t)$ is independent of the value of $H(x_r)$. ∎

4.3 Sampling with NEH Algorithm

The NEH can be applied to RN the same way as NEC. We fully expand each sample node in the sample set and then expand all their outgoing neighbors. In RW and RJ, since we only need one random outgoing link from the requested node, there is no need to determine all the outgoing links of the requested node. Instead, we can only determine one random outgoing by applying NEH once on a random occurrence of each requested node.

Theorem 4.3.1 *Consider a hostgraph model with parameter k and β. Let n be the size of the random graph in terms of total number of distinct nodes and s be the sample size. Assume a word is large enough to store the name of a node. If $k, 1/\beta, s \ll n$, then the expected number of words required by the RNN, the RW and the RJ is $n + o(n)$.*

Proof Let t be the total number of time steps to generate the random graph. It is easy to observe that $E(t) = n/\beta$ and $t \geq n \gg k, 1/\beta, s$.
Claim 6: *Let $D(x_r)$ be the outdegree of the node x_r. The expectation $E(D(x_r)) = O(k(1-\beta)\ln t/\beta)$*

Proof of Claim 6 The proof follows Claim 5. ∎

Claim 7: *Let Z be the total number of occurrences of nodes need to be partially expanded to construct the sample subgraph. In RNN, $E(Z) = O((sk^2 \ln^2 t)(1-\beta)^2/\beta^2)$ and in the RW and the RJ, $E(Z) = O(s \ln t)$*

Proof of Claim 7 The proof follows the Claim 6 and Theorem 4.2.2 and using fact that the upper bound of $E(N(x_r))$ is independent of $D(x_r)$. ∎

The NEH needs to store an expected $O((1-\beta)\ln t/\beta)$ occurrences of the prototype node and one determined link for each time it is called to partially expand an occurrence of a node. Let W be the number of words needed for partially expanding one occurrence of a node. Then $E(W) = O((1-\beta)\ln t/\beta)$. Based on the observation that this upper bound of $E(W)$ is independent of Z (since $E(W|Z) = O((1-\beta)\ln t/\beta) = E(W)$), we can see that the RNN consumes an expected $O((sk^2 \ln^3 t)(1-\beta)^3/\beta^3) = o(t)$ words in the node expanding process, while the RW and the RJ consume an expected $O(s(1-\beta)\ln^2 t/\beta) = o(t)$. Due to the fact that all these three sampling methods need a pre-generated array I which consumes exactly n words, the expected number of words required by the RNN, RW and RJ is then $n + o(n/\beta) = n + o(n)$, since β is a constant. ∎

Chapter 5

Experiments and Performance

5.1 Experiment Setup

We implemented the NEC and the NEH algorithm in Java respectively. In each experiment, we expand the very last node (occurrence of node) of a random graph of certain size and record the total number of expanded nodes (occurrence of node). We run the experiments 100 times on different random graph of same size to obtain the expected number of expanded nodes (occurrence of node) for expanding the last node (occurrence of node) of random graph of certain size. All the experiments were conducted on an Intel dual core machine with 1 GB of RAM.

5.2 Results

The results are shown in Figure 5.1. The experiments show that for expanding the last node of a random graph of size n, the NEC algorithm only needs to expand an expected $O(\ln n)$ nodes. Similar result is also observed for the NEH algorithm, both of which agree with the previous analytical results.

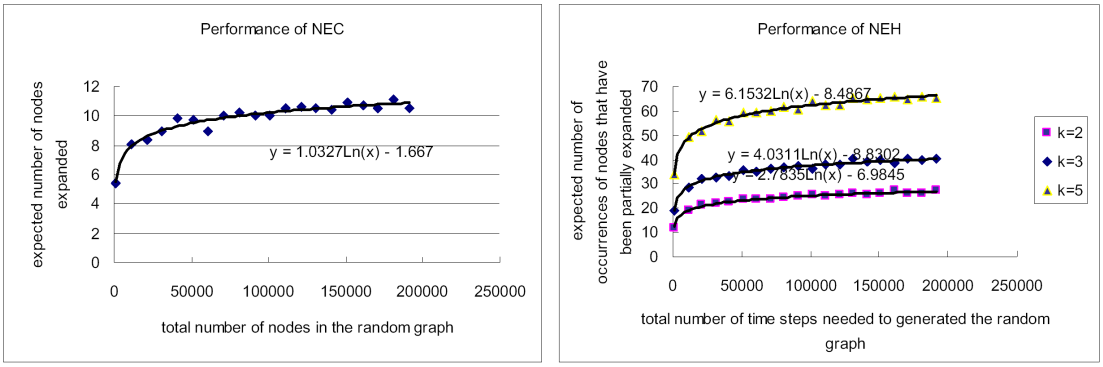

Figure 5.1: Performance of NEC and NEH

Chapter 6

Application

6.1 Guanxi in the Chinese Web

The Chinese web is notable for a large number of mutually linking web sites and other edges which we believe are associated with these links. We will attempt to show that this phenomenon is a manifestation of a complex dyadic social construct in China known as *guanxi*.[1] We hypothesis three type of structures resemble *guanxi* in the Chinese web: mutual links, Type 1 triangles which composed of two mutual links and one uni-directional link and Type 2 triangles in which all three sides are mutual links. In this Chapter, we examine the local linking structure of the Chinese web and the general web respectively to determine if there is structural evidence of *guanxi*. We compare them against local linking structure of a sample syntactic graph generated by the hostgraph model using the proposed sampling method to show that the phenomenon can not be explained by the hostgraph model. We then demonstrate our hypothesis further by examining the PageRank correlations.

6.2 Mutual links, Type 1 and Type 2 triangles

We randomly selected 10,000 web sites from a crawl of the Chinese web conducted by Peking University Sky Net search engine in 2007. We refer to these 10,000 web sites as the sample Chinese web. We also randomly selected 20,000 web sites from the Open Directory Project (www.dmoz.com) as the sample general web. We crawled all sites in both samples and all their neighbors reachable by their out-going links (RNN sampling method).

We also took 10,000[2] samples using our proposed NEH algorithm based on the RNN sampling method from a random graphs generated by hostgraph model with properties similar to the Chinese web. That is, by tuning the parameters of the hostgraph model, the target random graph is comparable in size and density to that of the Chinese site graph. The set of parameters is given in Table 6.2. We refer the generated graph as sample random graph.

Although the all these subgraphs obtained using RNN do not match the indegree distribution of the original graphs (LF06), they are sufficient to be used to measure

[1] This is a co-work with Valerie King and Louis Yu. A short version of this work is presented at WWW2008.

[2] Using the Chernoff bound, it is not hard to show that with such a sample size, we can obtain a good estimation with high probability.

size of the target random graph	800,000
size of the initial sample set	10,000
k	10
α	0.85
β	0.25

Table 6.1: settings of simulation using NEH based on RNN sampling method

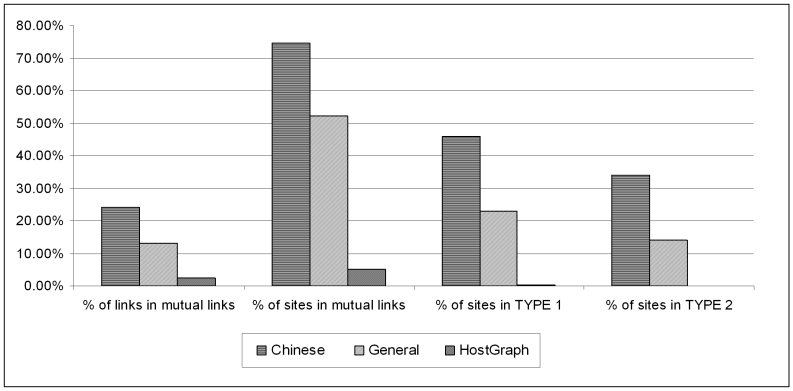

Figure 6.1: Mutual link and triangles in sample Chinese web, sample general web and sample random graph.

mutual links and Type 2 triangles.

We examined the percentages of the sites which are involved in mutual links and in Type 2 triangles, and estimated the percentage of sites which are involved in Type 1 triangles, in the sample Chinese web , the sample general web and sample random graph respectively. We calculated the total number of mutual links as a percentage of the total number of outgoing links in all these samples.

Figure 6.1 shows the experimental results from the sample Chinese web and the sample general web. Since our crawl does not capture all the in-coming links of the sample sites, the number of Type 1 triangles is undercounted. The overall results indicate that mutual linking widely exists in the World Wide Web, and it is especially significant in the Chinese web and it is not predicted by the hostgraph model.

6.3 PageRank Correlation

In order to demonstrate that mutual linking is a possible indication of *guanxi* in the web, and not the result of referencing under the generalized preferential attachment model or the hostgraph model, we also examine correlations between the sites' PageRank (PBMW98).

We look at two types of correlations: PageRank correlation between the sites that are connected by single links (C_S), and PageRank correlation between the sites that are connected by mutual links (C_M). We define C_S and C_M as follows: let E be a set of pairs (x, y) such that $(x, y) \in E$ iff site x points to site y. We define $S_i = \{(x, y) | (x, y) \in E \; and \; (y, x) \notin E \; and \; PageRank(x) = i\}$ and $M_i = \{(x, y) | (x, y) \in E \; and \; (y, x) \in E \; and \; PageRank(x) = i\}$. Then we define

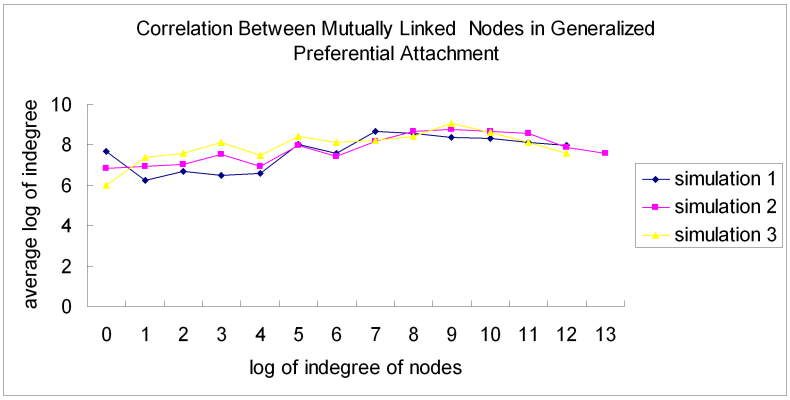

Figure 6.2: Correlation Between Mutually Linked Nodes in the Generalized Preferential Attachment Model.

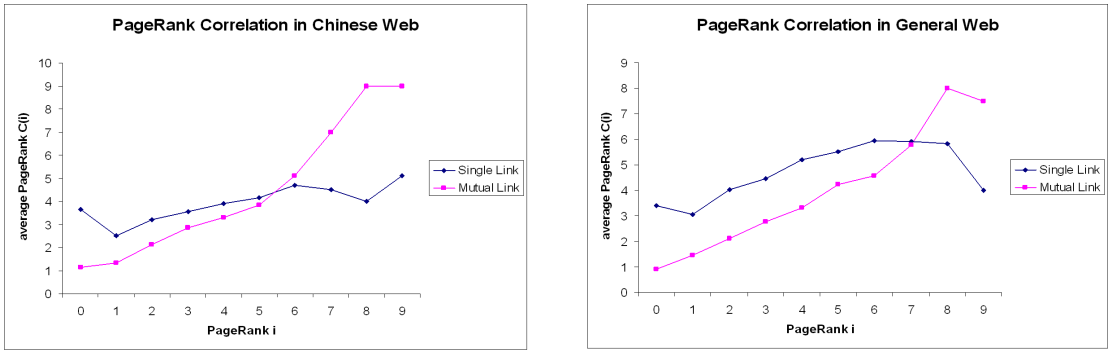

Figure 6.3: PageRank Correlation in Chinese Web and General Web.

$C_S(i) = \frac{\sum_{(x,y)\in S_i} PageRank(y)}{|S_i|}$ and $C_M(i) = \frac{\sum_{(x,y)\in M_i} PageRank(y)}{|M_i|}$.

Note that in a random graph, since the log of in-degree approximates the PageRank (Ups03), C_S (C_M) is then the expected log of in-degree of nodes with single (mutual) links from nodes with log of in-degrees equal to i. Under both the generalized preferential attachment model and the hostgraph model; the destinations of all links are determined independently of the in-degrees of the source nodes. Therefore, in the graphs generated by such models, the plot of $C_S(i)$ as a function of i will be flat.

The correlation between mutually linked nodes, $C_M(i)$, in such graphs is non-trivial, and we run simulations to determine this correlation. Figure 6.2 shows the log of in-degree correlation between mutually referenced nodes in a graph of 500,000 nodes and 5,000,000 links according to the generalized preferential attachment model. We see that fluctuates in the range of 6 to 9 except for very small value of i. This indicates that the log of in-degree i of the source nodes has little impact on.

Figure 6.3 shows the two types of PageRank correlations in the sample Chinese web and the sample general web, respectively. In the case of single links, the plot of $C_S(i)$ as a function of PageRank is almost flat, which agrees with the prediction of the generalized preferential attachment model and the hostgraph model.

In both the sample Chinese web and the sample general web, the PageRank values between mutually linked sites exhibit a very strong positive correlation, i.e. sites with high PageRank are more likely to be mutually linked to sites with high PageRank, and sites with low PageRank are more likely to be mutually linked to sites with low PageRank. A similar phenomena had been called assortative mixing (New02), and has been discovered in many undirected collaboration networks such as movie co-starring network and paper co-authorship network (New02). This positive correlation gives strong evidence that mutual linking in the web represents a type of interaction different than those in the hostgraph model or the generalized preferential attachment model.

Chapter 7

Conclusion

7.1 Contribution

This thesis presents two novel algorithms to sample from random web graphs and demonstrates the efficiencies of these algorithms both analytically and experimentally. Though the author only investigates the efficient methods for two particular random graph models under three different graph sampling schemes, the idea can be certainly applied to other random graph models based on the concept of copying and other graph sampling schemes, e.g. forest fire(LF06). The author believes that this work can help those who want to simulate the web using the copying mechanism but have only limited resources. The methods can also be used to validate and explore the properties of the random graph models based on the concept of copying.

In addition, the author also believes that the question asked in this thesis is itself very interesting and it can be generalized as follows: how can one efficiently sample from the outcome of a stochastic model? The question can be asked in all the areas where stochastic modeling is involved and this could possibly open a new research area.

7.2 Future Work

The NEH algorithm presented in this thesis only consumes $o(n)$ spaces. But in order to maintain a consistent view of the size of a random graph at each time step, the NEH algorithm initially takes an array I as input which itself consumes n words. Further research needs to be conducted to reduce this requirement.

The author will also look at other possible models in areas like economics and finance to see if the similar ideas could apply and hopefully this can result in the discovery of related techniques in the future.

Bibliography

[ACL02] William Aiello, Fan Chung, and Linyuan Lu, *Random evolution in massive graphs*, 97–122.

[AJB99] Reka Albert, Hawoong Jeong, and Albert-Laszlo Barabasi, *The diameter of the world wide web*, Nature **401** (1999), 130.

[BA99] Albert-Laszlo Barabasi and Reka Albert, *Emergence of scaling in random networks*, Science **286** (1999), 509.

[BCHR01] Krishna Bharat, Bay-Wei Chang, Monika Rauch Henzinger, and Matthias Ruhl, *Who links to whom: Mining linkage between web sites*, ICDM, 2001, pp. 51–58.

[BKM+00] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener, *Graph structure in the web*, Comput. Netw. **33** (2000), no. 1-6, 309–320.

[Bon04] Anthony Bonato, *A survey of models of the web graph*, First Workshop on Combinatorial and Algorithmic Aspects of Networking, Springer Berlin / Heidelberg, August 2004, pp. 159–172.

[DMS00] S.N. Dorogovtsev, J.F.F. Mendes, and A.N. Samukhin, *Structure of growing networks: Exact solution of the barabasi–albert's model*, Physical Review Letters **85** (2000), 4633.

[FLG00] Gary William Flake, Steve Lawrence, and C. Lee Giles, *Efficient identification of web communities*, KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining (New York, NY, USA), ACM, 2000, pp. 150–160.

[IK03] Noriko Imafuji and Masaru Kitsuregawa, *Finding a web community by maximum flow algorithm with hits score based capacity*, dasfaa **00** (2003), 101.

[KKR+99] Jon M. Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew S. Tomkins, *The web as a graph: Measurements, models, and methods*, 1999, pp. 1+.

[KRR+] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew S. Tomkins, and Eli Upfal, *The web as a graph.*

[KRR+00] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal, *Stochastic models for the web graph*, FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science (Washington, DC, USA), IEEE Computer Society, 2000, p. 57.

[KRRT99] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins, *Trawling the web for emerging cyber-communities*, Comput. Netw. **31** (1999), no. 11-16, 1481–1493.

[KYZ08] Valerie King, Louis Lei Yu, and Yan Zhuang, *Guanxi in the chinese web - a study of mutual linking*, WWW, 2008, pp. 1161–1162.

[LF06] Jure Leskovec and Christos Faloutsos, *Sampling from large graphs*, KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (New York, NY, USA), ACM, 2006, pp. 631–636.

[MR95] R. Motwani and P. Raghavan, *Randomized algorithms*, Cambridge University Press, New York (NY), 1995.

[New02] M.E.J. Newman, *Assortative mixing in networks*, Physical Review Letters **89** (2002), 208701.

[PBMW98] L. Page, S. Brin, R. Motwani, and T. Winograd, *The pagerank citation ranking: Bringing order to the web*, 1998.

[SWM05] M. P. Stumpf, C. Wiuf, and R. M. May, *Subnets of scale-free networks are not scale-free: sampling properties of networks.*, Proc Natl Acad Sci U S A **102** (2005), no. 12, 4221–4224.

[Ups03] Trystan Upstill, *Predicting fame and fortune: Pagerank or indegree*, In Proceedings of the Australasian Document Computing Symposium, ADCS2003, 2003, pp. 31–40.