2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

# Predictive Random Graph Ranking on the Web

Haixuan Yang, Irwin King, *Member, IEEE*, and Michael R. Lyu, *Fellow, IEEE*

*Abstract*— **The incomplete information about the Web structure causes inaccurate results of various ranking algorithms. In this paper, we propose a solution to this problem by formulating a new framework called, Predictive Random Graph Ranking, in which we generate a random graph based on the known information about the Web structure. The random graph can be considered as the predicted Web structure, on which ranking algorithm are expected to be improved in accuracy. For this purpose, we extend some current ranking algorithms from a static graph to a random graph. Experimental results show that the Predictive Random Graph Ranking framework can improve the accuracy of the ranking algorithms such as PageRank, Common Neighbor, and Jaccard's Coefficient.**

## I. Introduction

While the *PageRank* algorithm [1] has proven to be very effective for ranking Web pages, inaccurate *PageRank* results are induced because of the incomplete information about the Web structure. This problem is caused by the following phenomena:

1) *The Web is Dynamic (temporal dimension)*–The link structure evolves temporally. Some links are created and modified, while others are destroyed.
2) *The Observer is Partial (spatial dimension)*–For different observers (or crawlers), the Web structure may be different.
3) *Links are Different (local dimension)*–Not all out-links are created equal. Some out-links are more significant than others. For example, some people may tend to put the most important link on the top of their pages.

For the problem of the incompleteness and impreciseness of the Web structure, we have two contributions in this paper. Firstly, we provide a random graph perspective for the above phenomena. In temporal dimension, unknown links are modelled by random links; in spatial dimension, in order to generate a more accurate structure, different perspectives can be combined by means of a random graph; in local dimension, the different orders of links are seen to have random importance. Throughout this paper, we employ the random graph model.

Secondly, by the random graph perspective, we establish *Predictive Random Graph Ranking* framework. As illustrated in Figure I, the framework consists of two stages:

- **Random Graph Generation Stage**–The first stage engages the temporal, spatial and local link information to construct a random graph that can better model the Web. Statistical and other methods can be applied to generate this random graph that can better approximate the incomplete Web.
- **Random Graph Ranking Stage**–The second stage takes the random graph output and then calculates the ranking result based on a candidate ranking algorithm, such as, *PageRank*, *Common Neighbors, Jaccard's Coefficient, SimRank,* etc.

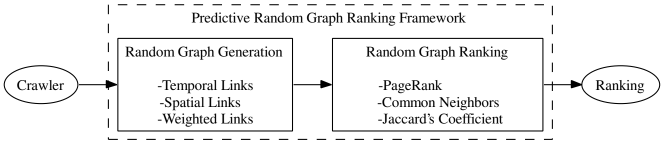


Fig. 1. The Predictive Random Graph Ranking Framework.

The intuition in the *Predictive Random Graph Ranking* framework is that: the more accurately we know the structure of the Web, the more accurately we can infer about the Web.

The rest materials are organized as follows. In the next section, we give a brief literature review on various ranking techniques. In Section III, we describe our predictive strategy. In Section IV, we describe the data sets that we worked on and the experimental results. In Section V, we draw a conclusion and present possible future work.

## II. Literature Review

We classify ranking techniques into two types: *Absolute Ranking* and *Relative Ranking*. *Absolute Ranking* assigns a real number to each page, and thus gives a total order for all pages. *PageRank* [1] belongs to *Absolute Ranking*. *Relative Ranking* assigns a real number to each pair of pages, and thus, for each one given page, determines a total order relative to the given page. *Common Neighbors* [2], *Jaccard's Coefficient* [3], and *SimRank* [4] belong to *Relative Ranking*.

All the mentioned ranking algorithms are established on a graph, and will be established on a random graph. For our convenience, we first give some notations. Throughout the paper, all the graphs mentioned are directed graphs. We denote a static graph by $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$, $E = \{(v_i, v_j) \,|\, \text{there is an edge from } v_i \text{ to } v_j\}$ is the set of all edges. Let $I(v_i)$ and $I(v_j)$ denote the nodes that link to node $v_i$ and $v_j$ respectively, and $|I(v_i)|$, $|I(v_j)|$ means the in-degree of the $v_i$ and $v_j$ respectively. The definition of a random graph $RG = (V, P)$ is given below.

*Definition 1:* A random graph $RG = (V, P = (p_{ij}))$ is defined as a graph with a vertex set $V$ in which the edges are

Haixuan Yang, Irwin King, and Michael R. Lyu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong (phone: 00852-62014825; fax: 00852-26035024; email: {hxyang, king, lyu}@cse.cuhk.edu.hk).

0-7803-9490-9/06/$20.00/©2006 IEEE

chosen independently, and for $1 \leq i, j \leq |V|$ the probability of $(v_i, v_j)$ being an edge is exactly $p_{ij}$.
The original definition of random graphs in [5], is slightly changed to consider the situation of directed graphs.

### A. Absolute Ranking

As a kind of *Absolute Ranking*, *PageRank* [1] gives the importance rank of Web page based on the link structure of the Web. The intuition behind *PageRank* is that it uses information external to the Web pages themselves–their in-links, and that in-links from "important" pages are more significant than in-links from average pages. Formally presented in [6], the Web is modelled by a directed graph $G = (V, E)$ in the *PageRank* algorithms, and the rank or "importance" $x_i$ for page $v_i \in V$ is defined recursively in terms of pages which point to it:

$$x_i = \sum_{(j,i) \in E} a_{ij} x_j, \tag{1}$$

where $a_{ij}$ is assumed to be $1/d_j$, $d_j$ is the out-degree of page $j$. Or in matrix terms, $x = Ax$. When the concept of "random jump" is introduced, the matrix form in Eq. (1) is changed to

Model 1:

$$x = [(1-\alpha) g e^T + \alpha A] x, \tag{2}$$

where the parameter $\alpha$ is the probability of following the actual link from a page, $(1-\alpha)$ is the probability of taking a "random jump", and $g$ is a stochastic vector (i.e. $e^T g = 1$). Typically, $\alpha = 0.85$ and $e$ is the vector of all ones.

### B. Relative Ranking

In [3], the authors survey an array of methods for *Relative Ranking*, including *Common Neighbors* and *Jaccard's Coefficient*. All the methods assign a connection weigh $s(i, j)$ to pairs of nodes $v_i$ and $v_j$, based on the input graph. The development of similarity search algorithms is motivated by the "related pages" queries of Web search engines and Web document classification [7]. Both applications require a similarity measure, which is computed by either the textual content of pages or the hyperlink structure or both. As in previous work [7], [8], we focus on similarities solely determined by hyperlink structure of the Web graph.

*1) Common Neighbors: Common neighbor* model is based on the idea that two pages are more similar if they have more common neighbors. The common neighbors of $v_i$ and $v_j$ can be defined as $s(i, j) = |I(v_i) \cap I(v_j)|$. It means that if more nodes points to $v_i$ and $v_j$ at the same time, $v_i$ and $v_j$ are more similar. In [2], the author computes the probability of collaboration between scientists in the Los Alamos as a function of the times of their past collaboration. A pair of scientists with more previous collaborators are more likely to collaborate than those with less previous collaborators. In [3], the authors employ common neighbors to predict if any two authors will coauthor papers in the future.

*2) Jaccard's Coefficient:* Another commonly used similarity metric is the *Jaccard coefficient*, which is used to measure the probability that both $v_i$ and $v_j$ share a feature. In [3], the authors take features to be neighbors in graph, which corresponds to the measure $s(i, j) = |I(v_i) \cap I(v_j)| / |I(v_i) \cup I(v_j)|$. In this paper we utilize this approach as well to measure the similarity between two pages in the Web.

### C. Dangling Nodes

Pages that either have no out-link or have no known out-link are called dangling nodes [6]. In [1], the authors suggested simply removing the pages without out-link and the links pointing to them. After doing so, it is suggested that they can be "added back in" without significantly affecting the results. However the situation is changed now and the dangling nodes problem has to be handled more accurately and directly.

Dangling nodes problem has received relatively little attention in the past. In [9], an absorbing model was suggested. This model can handle dangling nodes by modifying the original graph. Specifically speaking, it adds additional nodes (called clones), adds links from all the original nodes to their clones on the Web, and adds links from all the clones to themselves. As a result, the modified graph has no dangling node and so it is robust against dangling nodes. However, since the structure of the modified graph is different from the original Web structure, there will be a great difference between the final ranking results based on the modified graph and the ones based on the original graph.

In [10], pages whose out-degree is zero are handled by adding jump to a randomly selected page with probability 1 from every dangling node, and then by adding teleportation. More formally, the model 1 is modified as

Model 2:

$$x = [(1-\alpha) E + \alpha P'] x, \tag{3}$$

where $E = f e^T$, $P' = A + f d^T$, $f = e/n$, and $d$ denotes the $n-$dimensional column vector identifying the dangling nodes:

$$d_i = \begin{cases} 1 & \text{if } i \text{ is a dangling node,} \\ 0 & \text{otherwise.} \end{cases}$$

$f$ is referred as the personalization vector, it models the behavior of users when they get bored in following the link and decide to jump randomly.

In [6], dangling pages are handled in similar way. We reinterpret the model formally as follows.

Model 3:

$$\begin{aligned} \begin{pmatrix} x \\ y \end{pmatrix} &= \begin{pmatrix} \alpha C + (1-\alpha)/m \cdot \mathbf{1} & 1/m\mathbf{1} \\ \alpha D & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ &= (\alpha A + (1-\alpha) B) \begin{pmatrix} x \\ y \end{pmatrix} \end{aligned} \tag{4}$$

where $A = \begin{pmatrix} C & 1/m\mathbf{1} \\ D & 0 \end{pmatrix}, B = \begin{pmatrix} 1/m \cdot \mathbf{1} & 1/m\mathbf{1} \\ 0 & 0 \end{pmatrix}$, $m$ is number of nodes that have been crawled successfully,

$n$ is the number of nodes that have been found by the crawler, $C = (c_{ij})$, $D = (d_{ij})$ and if $d_j$ is the out-degree of node $j$,

$$c_{ij}, d_{ij} = \begin{cases} d_j^{-1} & \text{if there is a link node } i \text{ node } j, \\ 0 & \text{otherwise.} \end{cases}$$

Respectively by $C$ and $D$ we also denote the set of all nodes that have been crawled successfully and the set of remaining nodes.

In this model, the matrix $A$ models the users' behavior in case of following the actual links and the unknown links from dangling nodes to visited nodes. The matrix $B$ models the users' teleportation. Then the linear convex combination of the matrix $A$ and the matrix $B$ models the total behaviors of the users.

From Eq. (4), we can see the problem. For our convenience, we denote the matrix $\begin{pmatrix} C & 1/m\mathbf{1} \\ D & 0 \end{pmatrix}$ as $\begin{pmatrix} X & M \\ Y & N \end{pmatrix}$, the link information about $X$ and $Y$ is already known because the crawler has visited all the nodes in $C$ and therefore all the link from the nodes in $C$ to nodes in $C$ and $D$ have been known by the crawler. But the information about links from the nodes in $D$ to nodes in $C$ and $D$ is unknown by the crawler because the nodes in $D$ have not been visited yet or have not been visited successfully. Hidden in the matrix $\begin{pmatrix} C & 1/m\mathbf{1} \\ D & 0 \end{pmatrix}$, there is an assumption, in which users will jump randomly and uniformly from every node in $D$ only to nodes in $C$ , and therefore $M = 1/m\mathbf{1}$. This assumption can be improved to be more accurate. In reality, users may jump from nodes in $D$ to nodes in $D$, and thus the assumption that all the elements in the right-bottom part of the matrix are zero is problematic, and the assumption about the right-top part of the matrix need to be adjusted accordingly. In our model, we assume that users will jump randomly but not uniformly from every node in $D$ to both nodes in $C$ and nodes in $D$.

Our model is different from model 2 in that we get the information about the unknown part of the matrix by prediction while the model 2 assume the uniform distribution about the unknown part of the matrix. The authors in [10] also suggest re-defining the vector $f$ as non-uniform distribution, however, they only consider the vector $f$ as the personalization factor, which is a subjective factor, and from which the *PageRank* vector can be biased to prefer certain kinds of pages.

Our model is different from model 3 in two folds: The users will not jump uniformly from every node in $D$ to other nodes; The users will jump from every node in $D$ not only to nodes in $C$ but also to nodes in $D$.

The authors in [6] further discuss the "link rot" problem and suggest new methods of ranking motivated by the hierarchical structure of the Web. Although we can combine our model with the technique used in solve these kinds of problem, we do not focus them in this paper.

We consider a case in which dangling nodes are so significant that including them in the overall ranking may not only change the rank value of non-dangling nodes but also change the order of the non-dangling nodes. In the example of figure II-C, there are three pages, with one of them being a dangling node with a link from page 2. If we compute *PageRank* by the model 2, and let $\alpha = 0.85$, the matrix in the model 2 is

$$\begin{pmatrix} 0.05 & 0.475 & 1/3 \\ 0.9 & 0.05 & 1/3 \\ 0.05 & 0.475 & 1/3 \end{pmatrix}$$

By power iteration, the RageRank scores are $(x_1, x_2, x_3) = (0.3032, 0.3936, 0.3032)$. So in model 2, rank for node 2 is much higher than node 1. If we simply remove the dangling node 3, then by Eq. (1), the *PageRank* scores for nodes 1 and 2 are $(x_1, x_2) = (0.5, 0.5)$, in which the rank for node 1 is same as that for node 2. From this example, we can see that whether we handle dangling nodes will not only change the rank value of the non-dangling nodes but also change their order.

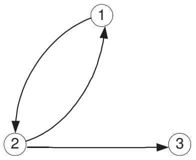


Fig. 2. A case in which considering dangling node will have significant effect on the ranks of non-dangling nodes

## III. Predictive Strategy

In this section, we first show the origin of the idea of the predictive strategy, then we show that the concept of a random graph is necessary, next we show how a random graph can be generated in various situations. This forms the first stage of the framework *Predictive Random Graph Ranking*, and can be found in Section 3.2 and 3.3. In Section 3.4 and Section 4, we extend several ranking models from static graphs to random graphs. These are the second stage of the *Predictive Random Graph Ranking* framework.

### A. Origin of Predictive Strategy

In [11], the authors propose a predictive ranking technique to improve the accuracy of *PageRank* through the estimation of the incomplete information caused by partial crawling on the Web. The more accurately estimated Web structure leads to a more accurate *PageRank* result. In this paper, we extend the basic idea in [11] from *PageRank* to a collection of ranking algorithms, from temporal incomplete information to spatial uncertainty and weighted links.

### B. From Static Graphs to Random Graphs

The concept of a random graph is necessary for *PageRank*. For example, the graph in Figure 3 may be encountered by a crawler in the early stage if all the unvisited nodes are ignored. If we employ Eq. (1) and use the power iterative method to solve the page rank problem, then we will suffer the problem of divergence unless the entire initial values of $x_i$ $(i = 1, 2, 3)$ take the value of $1/3$, which usually can not be found in practice. However, if we employ Eq. (2), the power iterative method will converge. This is because the modified matrix in Eq. (2) is a positive stochastic matrix, and so 1 is its largest absolute eigenvalue and no other eigenvalue whose absolute value is equal to 1, which is guaranteed by the Perron Theorem [12]. Behind Eq. (2), we can see that the Web graph has been modelled as a random graph, in which, the original link exists with a probability of $\alpha$, and there is a link that connects each pair of pages with a probability of $1 - \alpha$.

Furthermore, in the following, we discuss three situations: (1) *temporal links*, (2) *spatial links* and (3) *weighted links*, in which the concept of a random graph is also necessary.

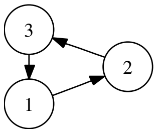


Fig. 3. A static graph.

*1) Random Graph Generated Temporal Links:* If we want to model the estimation about the temporal links, the concept of a random is necessary. In Figure 3, when the time continues, the crawler will visit more nodes, but at current time, the links (called temporal links) from the currently unvisited node are unknown. In general, it is difficult to estimate the temporal link structure accurately; however, some elementary estimation is possible. In this paper, we only estimate the indegree of each node in the set of nodes that have been found, and thus some information about the link structure can be inferred statistically. For more discussions, see Section III-C.

*2) Random Graph Generated by Several Graphs:* Several crawlers may visit some pages at different times and from different starting sites, and a link may exist for one crawler, but disappear for another. This causes the partial observer problem–the web graph is viewed differently from different points. Suppose that different Web graphs $G_i = (V_i, E_i), (i = 1, 2, \dots, N)$ are obtained by $N$ different observers (or crawlers). We can combine these different graphs and generate a random graph $RG = (V, P)$, where $V = \cup_{i=1}^{N} V_i, P = (p_{ij}), p_{ij} = n(i,j)/N, n(i,j)$ is the number of the graphs where the link $(i,j)$ appears. The intuition behind is that the more a link is reliable, the more times different observers will find it.

*3) Random Graph Generated by Weighted Links:* We have observed that some out-links are more significant than others. As an example, we may model the out-link significance by the exponential decay rule: $e^{1-k}$ where $k$ is the out-link order number from a particular page. Then a random graph generated by this rule will be $P = (p_{ij})$ where $p_{ij} = 0$ if there is no link from $i$ to $j$, and $p_{ij} = e^{1-k(i,j)}$ if $j$ is the $k(i,j)$-th out-link from $i$. By doing so, the significance of different out-links from a particular page is distinguished. The original static graph is changed to a random graph.

In next subsection, we emphasis on the problem of dangling nodes, which is caused by the nature of the dynamic Web. This problem is handled by predicting the link structure as a random graph. To sum up, it is necessary to extend the current ranking algorithms from a static graph to a random graph.

### *C. From Visited Nodes to Dangling Nodes*

*1) Why We Consider Dangling Nodes:* On the one hand, we can see that the *PageRank* algorithm depends on part of the Web structure, and that the visited fraction of the whole Web page by a crawler becomes smaller and smaller as the Web continues to grow. More and more dangling nodes appear because of the difficulty of sampling the entire Web. In [1], the authors reported that they have 51 million URLs not downloaded yet when they have 24 million pages downloaded. In [13], dynamic pages are estimated to be 100 times more than static pages, and in [6], the authors point out in their experiment that the number of uncrawled pages still far exceeds the number of crawled pages and that there are an essentially infinite number of URLs which is estimated to be at least $64^{2000}$. These experimental results and theoretical analysis mean that in reality, the huge number of unvisited pages tends to exceed the ability of a crawler.

On the other hand, some dangling pages are worthy of ranking because they contain important information. In such a situation, ranking those pages that only have been found may enrich the content of a search engine. As an example, a search engine may return the users the URLs of unvisited pages with high ranking scores. Moreover, including dangling nodes in the overall ranking may have significant effect not only on the rank value of non-dangling pages but also on the rank order. This will be shown in the Experiment section.

*2) How to Classify Dangling Nodes:* In the following, we follow the ideas in [6] in analyzing the reasons that cause the dangling nodes, and we classify dangling nodes into 3 classes according to these reasons.

Dangling nodes of class 1 ($DNC1$) are defined as nodes that have been found but have not been visited. One reason to produce such kind of dangling nodes is that the Web is so large that we cannot visit all the pages; another reason is that new Web pages are always being created.

Dangling nodes of class 2 ($DNC2$) are defined as nodes that have been tried but not visited successfully. The reason to produce dangling nodes of class 2 is that some pages may exist before, but now are damaged or are in maintenance, or they are protected by a robot.txt, or they are wrongly created.

Dangling nodes of class 3 ($DNC3$) are defined as nodes that have been visited successfully but from which no out-link is found. Dangling nodes of class 3 exist because there are many files on the Web with no hyperlink structure.

*3) How to Handle Dangling Nodes:* We first partition all the nodes $V$ of the graph $G$ $(|V| = n)$ into three subsets: $D^0$, $D^1$, and $D^2$, where $C^0$ $(|C^0| = m)$ denotes the subset of all nodes that have been crawled successfully and have at least one out-link; $D^1$ $(|D^1| = m_1)$ denotes the set of nodes of $DNC3$; $D^2$ ($|D^2| = n - m - m_1$) denotes the set of nodes of $DNC1$. Nodes of $DNC2$ are ignored here. The main idea of handling dangling nodes is to handle different nodes in different ways. In the following, we describe our method in detail.

1. We predict the real in-degree $d^-(v_i)$ by the number of found links $fd^-(v_i)$ from visited nodes to the node $v_i$. With the breadth-first crawling method, we assume that the real number of links from all nodes in $V$ to the node $v_i$ is proportional to the number of found links $fd^-(v_i)$ from visited nodes to the node $v_i$, and further we assume that

$$d^-(v_i) \approx \frac{n}{(m+m_1)} \cdot fd^-(v_i)(i = 1, 2, \ldots, n).$$

This assumption is based on the intuition that a crawler's ability of finding new links to a given node $v_i$ depends on the density of these links. The density of these links to the node $v_i$ is equal to $d^-(v_i)/n$. The crawler has found $fd^-(v_i)$ such kind of links when it has crawled $m$ nodes, and we consider $\frac{fd^-(v_i)}{(m+m_1)}$ as an approximate estimate of the density of these links. Following this, the above approximate equality holds.

2. With the approximate in-degree $d^-(v_i)$, we can rearrange the matrix. All the found links $fd^-(v_i)$ are from the nodes in $D^0$, and the remaining links $d^-(v_i) - fd^-(v_i)$ are from the nodes in $D^2$ (it is impossible that some of these links are from the nodes in $D^1$). Since we infer the number of the remaining links only out of $m + m_1$ visited nodes and the total number nodes is $n$, there is a risk of over-prediction. To prevent the over-prediction, we adopt a confidence index (or certainty) $(m + m_1)/n$ about this estimation, and so we expect $(d^-(v_i) - fd^-(v_i))(m + m_1)/n$ remaining links. Without any prior information about the distribution of these remaining links, we have to assume that they are distributed uniformly from the nodes in $D^2$ to the node $v_i$, i.e., these remaining links are shared by all the nodes in $D^2$. So matrix $A^T$ representing the random graph can be divided into six blocks shown below

$$A^T = \begin{pmatrix} C & X & M \\ D & Y & N \end{pmatrix},$$

where $(C, D)^T$ is used to model the known link structure from $D^0$ to $V$. Let $C = (c_{ij}), D = (d_{ij})$, then

$$c_{ij}, d_{i,j} = \begin{cases} 1, & \text{there is a link from } j \text{ to } i, \\ 0, & \text{otherwise.} \end{cases}$$

In $A^T$, $(X, Y)^T$ will be defined later, $(M, N)^T$ is used to model the link structure from $D^2$ to $V$, and is defined as follows:

$$\begin{pmatrix} M \\ \\ N \end{pmatrix} = \begin{pmatrix} l_1 & 0 & 0 & 0 \\ 0 & l_2 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & l_n \end{pmatrix} \mathbf{1}_{n\times(n-m-m_1)},$$

where $l_i = \frac{(d^-(v_i)-fd^-(v_i))(m+m_1)}{n(n-m-m_1)}$, $(i = 1, 2, \ldots, n)$, $n - m - m_1$ means that the expected remaining in-links $(d^-(v_i) - fd^-(v_i))(m + m_1)/n$ are shared uniformly by all nodes in $D^2$.

3. When we want to model the users' teleportation, we assume that the users will jump to node $v_i$ with a probability of $g_i$ when they get bored in following the actual links. So the matrix modelling the teleportation is $ge^T$. We denote here $(g_1\, g_2\, \ldots\, g_n)^T$ by $g$.

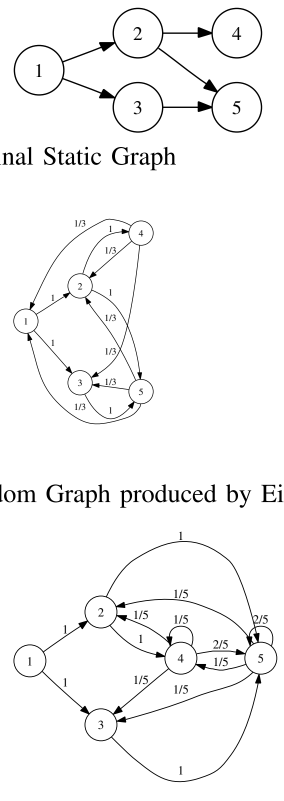
4. When the user encounters a node of $DNC3$, there is no out-link that the user can follow. In this case, we assume that the same kind of teleportation as in step 3 will happen, and so the matrix $(X, Y)^T$ in step 2 is used to model the link structure from $D^1$ to $V$ and it is assumed to be

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} g_1 & 0 & 0 & 0 \\ 0 & g_2 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g_n \end{pmatrix} \mathbf{1}_{n\times m_1}.$$

5. We further assume that $\alpha$ is the probability of following an actual out-link from a page, $1 - \alpha$ is the probability of taking a "random jump" rather than following a link. Then the random matrix $P$ is modelled as

$$P^T = (1 - \alpha)ge^T + \alpha A^T. \tag{5}$$

The matrix $P$ corresponds to a random graph, which models the temporal Web–to predict a future Web graph by an early Web graph. This is called *Temporal Web Prediction Model*. From the static graph in Figure 4(a), where node 4 and node 5 are assumed to be nodes of $DNC1$, a random graph in Figure 4(b) is generated by the above model ($\alpha = 1$).



(a) Original Static Graph

(b) Random Graph produced by Eiron's Model

(c) Random Graph produced by our Model

Fig. 4. Illustration on the random graph

### D. Random Graph Ranking

We need to extend the standard ranking technique to random graphs in order to handle the random graph outputs produced by the predictive strategy in three situations: (1) temporal links, (2) spatial links and (3) weighted links.

*1) PageRank on a Random Graph:* We extend the *PageRank* from the setting of a static graph to the setting of a random graph. Similar to *PageRank*, the page rank vector $x$ on a random graph can be defined recursively in terms of random graphs:

$$x_i = \sum_j q_{ij} x_j,$$

where $q_{ij} = p_{ji} / \sum_k p_{jk}$. Or in matrix form, $x = Qx$, where $Q = (q_{ij})$. In a static graph, if there is a link from $v_j$ to $v_i$, then the probability of a random surfer will follow the link is $1/d_j$, where $d_j$ is the out-degree of $v_j$. In a random graph, since the sum $\sum_k p_{jk}$ is the expected out-degree of $v_j$ and the link from $v_j$ to $v_i$ exists with a probability of $p_{ji}$, the expected probability of a random surfer will follow the link $(v_j, v_i)$ is $p_{ji} / \sum_k p_{jk}$. Consequently, the above equation is established.

*2) Common Neighbor on a Random Graph:* We extend the *Common Neighbor* approach from the setting of a static graph to the setting of a random graph.

First, the random neighbor set $RI(v_i)$ of $v_i$ is defined as

$$RI(v_i) = \{(v_k, p_{ki}) | v_k \in V\},$$

where $p_{ki}$ is the probability of $v_k$ as a neighbor of $v_i$. In the setting of a random graph, each node $v_k$ is linked to node $v_i$ with a probability $p_{ki}$, so $v_k$ is the neighbor of $v_i$ with a probability $p_{ki}$. This extends the definition of the set of neighbors of node $v_i$ in the setting of a static graph.

Second, the set of the common random neighbors of $v_i$ and $v_j$ is defined as

$$RI(v_i) \cap RI(v_i) = \{(v_k, p_{ki} p_{kj}) | v_k \in V\}.$$

The sets of random neighbors of $v_i$ and $v_j$ are $RI(v_i)$ and $RI(v_j)$ respectively. $v_k$ is the neighbor of $v_i$ with a probability $p_{ki}$, and $v_k$ is the neighbor of $v_j$ with a probability $p_{kj}$, then we can say $v_k$ is the common neighbor of $v_i$ and $v_j$ with a probability $p_{ki} p_{kj}$ since the random edges are drawn independently. This extends the meaning of the common neighbor.

Third, the expected number of nodes in $RI(v_i) \cap RI(v_i)$ is considered as the similarity measure $s(i,j)$, and is defined as

$$s(i,j) = \sum_k p_{ki} p_{kj}.$$

This extends the definition of number of common neighbors of $v_i$ and $v_j$ in the setting of a static graph.

*3) Jaccard's Coefficient on a Random Graph:* The *Jaccard's Coefficient* in a random graph is defined as

$$\begin{aligned} s(i,j) &= |RI(v_i) \cap RI(v_j)| / |RI(v_i) \cup RI(v_j)| \\ &= \textstyle\sum_k p_{ki} p_{kj} / \sum_k (p_{ki} + p_{kj} - p_{ki} p_{kj}), \end{aligned}$$

where $RI(v_i) \cup RI(v_j) = \{(v_k, p_{ki} + p_{kj} - p_{ki} p_{kj}) | v_k \in V\}$. The expected number elements in $RI(v_i) \cup RI(v_j)$ is equal to $\sum_k (p_{ki} + p_{kj} - p_{ki} p_{kj})$. Since $v_k$ is not the neighbor of $v_i$ with a probability $1 - p_{ki}$, and is not the neighbor of $v_j$ with a probability $1 - p_{kj}$, we assume that $v_k$ is not the neighbor of either $v_i$ or $v_j$ with a probability $(1 - p_{kj})(1 - p_{kj})$, and we have that $v_k$ is the neighbor of either $v_i$ or $v_j$ with a probability $1 - (1 - p_{kj})(1 - p_{kj}) = p_{ki} + p_{kj} - p_{ki} p_{kj}$. Therefore, $RI(v_i) \cup RI(v_j) = \{(v_k, p_{ki} + p_{kj} - p_{ki} p_{kj}) | v_k \in V\}$. The expected number elements is thus equal to $\sum_k (p_{ki} + p_{kj} - p_{ki} p_{kj})$.

Note that one can easily conclude that when the random graph becomes a static graph, the algorithms described in the above subsections degrade into the original algorithms. This means ranking algorithms on a random graph generalize the original ones.

## IV. Experiments

The temporal dimension of the *Predictive Random Graph Ranking* framework can actually be designed to be tested in experiments. For this, we design a comparison method by calculating the ranking difference and order difference between the early results (less accurate) and the final results (relatively accurate, and considered as a ground truth). For more details, see Section IV-B.

### A. Data

Our input data consists of a synthetic data set and a real-world data set. A detailed description follows.

*1) Synthetic Web Graph:* The degree sequences of the World Wide Web are shown to be well approximated by a power law distribution [14], [15], [16]. That is, the probability that a Web page has *k* outgoing (incoming) links follows a power law over many orders of magnitude $P_{out}(k) \sim k^{-\gamma_{out}}$ and $P_{in}(k) \sim k^{-\gamma_{in}}$.

The power law distribution of the degree sequence appears to be a very robust property of the Web despite its dynamic nature, therefore, we can generate synthetic Web-like random graphs to test the performance of our algorithms.

Several approaches to modelling power law graphs [14], [16] have been proposed. In our numerical experiment, we use the $(\alpha, \beta)$ model [16] to generate random graphs. By setting $\alpha = 0.52$ and $\beta = 0.58$, the model generates a random power law graph with $\gamma_{out} = 2.1$ and $\gamma_{in} = 2.38$, both of these values match the Web.

By simulating the procedure of crawling, we can obtain a series of growing incomplete graphs containing pages of $DNC1$. The number $V[t]$ of pages visited and the total number $T[t]$ of pages found at time $t$ are shown in Table I.

*2) Real Web Graph:* The data of a real Web graph were obtained from the domain *cuhk.edu.hk*. The graph series are snapshot during process of crawling pages restricted within this domain. The number $V[t]$ of pages visited and the total number $T[t]$ of pages found at time $t$ are shown in Table II.

TABLE I
DESCRIPTION OF THE SYNTHETIC GRAPH SERIES

| t | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| V[t] | 1000 | 1100 | 1200 | 1300 | 1400 | 1500 |
| T[t] | 1764 | 1778 | 1837 | 1920 | 1927 | 1936 |
| t | 7 | 8 | 9 | 10 | 11 | |
| V[t] | 1600 | 1700 | 1800 | 1900 | 2000 | |
| T[t] | 1952 | 1954 | 1964 | 1994 | 2000 | |

TABLE II
DESCRIPTION OF REAL DATA SETS WITHIN DOMAIN CUHK.EDU.HK

| t | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| V[t] | 7712 | 78662 | 109383 | 160019 | 252522 | 301707 |
| T[t] | 18542 | 120970 | 157196 | 234701 | 355720 | 404728 |
| t | 7 | 8 | 9 | 10 | 11 | |
| V[t] | 373579 | 411724 | 444974 | 471684 | 502610 | |
| T[t] | 476961 | 515534 | 549162 | 576139 | 607170 | |

### B. Methodology

The algorithms we run include *PageRank*, *Common Neighbors*, and *Jaccard's Coefficient*. For each algorithm $A$, we have two versions denoted by $A$ and $PreA$. $A$ is the version using the random graph generated by the traditional method [6], in which dangling nodes of $DNC1$ are considered to have random links uniformly to each node in $C$, and $PreA$ is the version using the *Temporal Web Prediction Model*. Both $PreA$ and $A$ are run on two data series–the synthetic data series and the real data series. Each data series contains 11 data sets, which are obtained by taking snapshots during the process of a crawler or a simulated crawler. Finally, for each data series and for each algorithm $A$, we obtained 22 ranking results, namely,

$$\begin{array}{llll} A_1, & A_2, & \ldots, & A_{11}, \\ PreA_1, & PreA_2, & \ldots, & PreA_{11}. \end{array}$$

The results on the first 10 data is not accurate because these data are incomplete, and the Web is dynamically changing. The result $A_{11}$ on the synthetic data should be the same as $PreA_{11}$ because *Temporal Web Prediction Model* will not have effect on complete information, but the result $A_{11}$ on the real data is not the same as $PreA_{11}$ because of the existence of dangling nodes of $DNC1$ in time 11.

If the difference between the results on time $t$ and the results on time 11 is smaller, we think it is more accurate. The value difference and order difference are described below.

*Value Difference.* The value difference between $A_t$ ($PreA_t$) and $A_{11}$ is measured as

$$||A_t/Max_t - Cut(t, A_{11})/CutMax_t||_2$$

($||PreA_t/Max_t - Cut(t, A_{11})/CutMax_t||_2$). Where $cut(t, A_{11})$ is the results cut from $A_{11}$ such that it has the same dimension as $A_t$, and $CutMax_t$ ($Max_t$) means the maximal value among results in $cut(t, A_{11})$ ($A_t$).

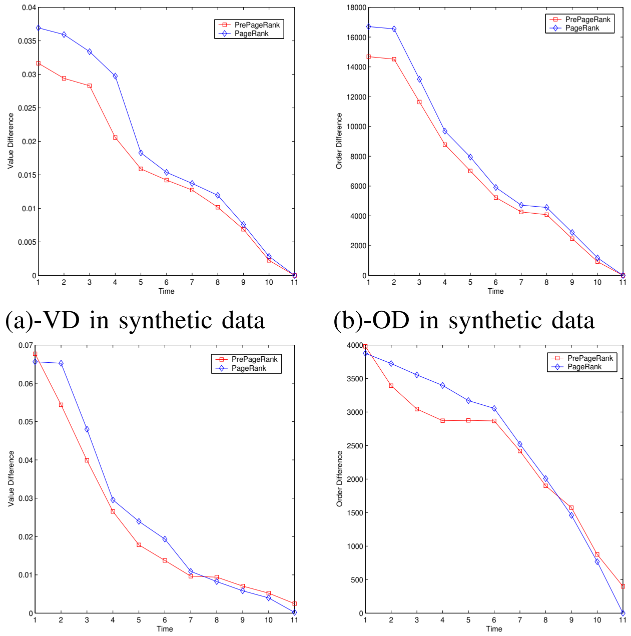
*Order Difference.* The order difference between $A_t$ ($PreA_t$) and $A_{11}$ is measured as the significant order difference between $A_t$ and $Cut(t, A_{11})$ ($PreA_t$ and $Cut(t, A_{11})$). The significant order difference between two similarity matrices $M$ and $N$ is calculated by the sum of the significant order difference for each row of $M$ and $N$, and for each row $M(i), N(i)$ of $M$ and $N$, the pair $(M(i,j), M(i,k))$ and $(N(i,j), N(i,k))$ is considered as a significant order difference if both $M(i,j) > M(i,k) + 0.005Max_M$ and $N(i,k) > N(i,j) + 0.005Max_N$, or both $M(i,k) > M(i,j) + 0.005Max_M$ and $N(i,j) > N(i,k) + 0.005Max_N$, where $Max_M$ ($Max_N$) is the maximum value of $M$ ($N$).

### C. Set Up

The experiments are conducted on the workstation whose hardware model is Nix Dual Intel Xeon 2.2GHz, whose RAM is 1GB, and whose OS is Linux Kernel 2.4.18-27smp (RedHat7.3). We set $\alpha = 0.85$ and set $g$ to be the uniform distribution in both *PageRank* and *PrePageRank*.

### D. Experimental Results

Figure 5 demonstrate the *PageRank* results on the synthetic data and the real data. On the synthetic data, in 100% early stages, *PreRageRank* is closer to the final result both in value difference and in significant order difference. Since the graph in time 11 is complete, there is no difference between the *PrePageRank* and *PageRank*, and the curves meet at time 11. On the real data, since the data at time 11 contains unvisited pages, *PrePageRank* and *PageRank* have a difference on this data, the employment of *PageRank* results on this data as a reference will cause a bias against *PrePageRank*. Even so, in 60% early stages, *PreRageRank* is closer to the final *PageRank* result in value difference; in 70% early stages, *PrePageRank* is closer to the final *PageRank* result in significant order difference.

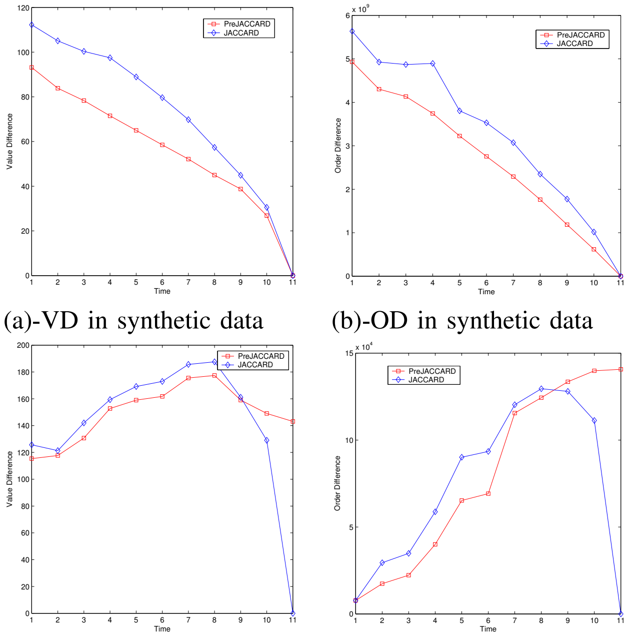


(a)-VD in synthetic data (b)-OD in synthetic data

(c)-VD in real data (d)-OD in real data

Fig. 5. PageRank Comparison Results

Figure 6 demonstrate the *Jaccard's Coefficient* results. On the synthetic data, in 100% early stages, *PreJaccard's Coefficient* is closer to the final result both in value difference and in significant order difference. On the real data, since the data at time 11 contains unvisited pages, *PreJaccard's Coefficient* and *Jaccard's Coefficient* have a difference on this data, the employment of *Jaccard's Coefficient* results on this data as a reference will cause a bias against *PreJaccard's Coefficient*. Even so, in 70% early stages, *PreJaccard's Coefficient* is closer to the final *Jaccard's Coefficient* result in value difference; in 70% early stages, *PreJaccard's Coefficient* is closer to the final *accard's Coefficient* result in significant order difference. For *Common Neighbor*, similar results are obtained.



(a)-VD in synthetic data (b)-OD in synthetic data

(c)-VD in real data (d)-OD in real data

Fig. 6. Jaccard's Coefficient Comparison Results on small synthetic data

## V. Conclusion and Future Work

We have shown that the *Temporal Web Prediction Model* is effective in *PageRank* and *Jaccard's Coefficient*. Because our model mines more information about the Web structure, the results of Predictive strategy on these algorithms are more accurate than those using the random graph produced by the method in [6]. We conclude that the random graph input indeed extends the scope of some original ranking techniques, and significantly improve some of them, admittedly computational cost of our method is increased a little compared to the traditional method of handling dangling nodes.

In our experiments, we only test the *Predictive Random Graph Ranking* framework in the viewpoint of dynamic Web. Besides the challenging work to consider the power law distribution in this viewpoint, it deserves further investigation in other two viewpoints of partial observers and weighted links. Such future work involves investigating pagemakers' preference on link orders and substantial users-based research. It is also interesting to extend other ranking algorithms such as *SimRank* in the framework of predictive random graph ranking. One more thing we concern is to reduce the computational cost of our method.

## VI. Acknowledgments

We thank Mr. Patrick Lau, Mr. Zhenjiang Lin and Mr. Zenglin Xu for their help. The work described in this paper is fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK4205/04E and Project No. CUHK4235/04E), and is affiliated with the VIEW Technologies Laboratory and the Microsoft-CUHK Joint Laboratory for Human-centric Computing & Interface Technologies.

## References

[1] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford Digital Library Technologies Project, Tech. Rep. Paper SIDL-WP-1999-0120 (version of 11/11/1999), 1999.

[2] M. E. J. Newman, "Scientific collaboration networks. I. Network construction and fundamental results," *Physical Review E*, vol. 64, no. 016131, pp. 1–8, 2001.

[3] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *Twelfth International Conference on Information and Knowledge Management*. ACM, November 2003, pp. 556–559.

[4] G. Jeh and J. Widom, "Simrank: A measure of structural-context similarity," *Proc. of SIGKDD*, 2002. [Online]. Available: citeseer.ist.psu.edu/jeh02simrank.html

[5] B. Bollobás, *Random Graphs*. Academic Press Inc. (London), 1985.

[6] N. Eiron, K. S. McCurley, and J. A. Tomlin, "Ranking the web frontier," in *Proceeding of the 13th World Wide Web Conference*, 2004, pp. 309–318.

[7] D. Fogaras and B. Rácz, "Scaling link-based similarity search." in *WWW*, A. Ellis and T. Hagino, Eds. ACM, 2005, pp. 641–650.

[8] H. Ino, M. Kudo, and A. Nakamura, "Partitioning of web graphs by community topology." in *WWW*, A. Ellis and T. Hagino, Eds. ACM, 2005, pp. 661–669.

[9] G. Amati, I. Ounis, and V. Plachouras, "The dynamic absorbing model for the web," University of Glasgow, Tech. Rep., 2003.

[10] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub, "Exploiting the block structure of the web for computing pagerank," Stanford University, Tech. Rep., 2003.

[11] H. Yang, I. King, and M. R. Lyu, "Predictive ranking: a novel page ranking approach by estimating the web structure." in *WWW (Special interest tracks and posters)*, A. Ellis and T. Hagino, Eds. ACM, 2005, pp. 944–945.

[12] C. R. MacCluer, "The many proofs and applications of perron's theorem," *SIAM Review*, vol. 42, no. 3, pp. 487–498, 2000.

[13] S. Handschuh, S. Staab, and R. Volz, "On deep annotation," in *Proceeding of the 12th World Wide Web Conference*, 2003, pp. 431–438.

[14] J. M. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, and A. S. Tomkins, "The Web as a graph: Measurements, models and methods," *Lecture Notes in Computer Science*, vol. 1627, pp. 1–18, 1999. [Online]. Available: citeseer.ist.psu.edu/kleinberg99web.html

[15] S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "Trawling the Web for emerging cyber-communities," *Computer Networks (Amsterdam, Netherlands)*, vol. 31, no. 11–16, pp. 1481–1493, 1999. [Online]. Available: citeseer.ist.psu.edu/article/kumar99trawling.html

[16] ——, "Extracting large-scale knowledge bases from the web," in *The VLDB Journal*, 1999, pp. 639–650. [Online]. Available: citeseer.ist.psu.edu/kumar99extracting.html

[17] A. Ellis and T. Hagino, Eds., *Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14*. ACM, 2005.