ARTICLE IN PRESS

Available online at www.sciencedirect.com
ScienceDirect

ELSEVIER

Information Processing and Management xxx (2007) xxx–xxx

INFORMATION PROCESSING & MANAGEMENT

www.elsevier.com/locate/infoproman

DistanceRank: An intelligent ranking algorithm for web pages

Ali Mohammad Zareh Bidoki *, Nasser Yazdani

Department of Electrical and Computer Engineering, University of Tehran, Tehran, Iran

Received 11 January 2007; received in revised form 27 June 2007; accepted 29 June 2007

Abstract

A fast and efficient page ranking mechanism for web crawling and retrieval remains as a challenging issue. Recently, several link based ranking algorithms like PageRank, HITS and OPIC have been proposed. In this paper, we propose a novel recursive method based on reinforcement learning which considers distance between pages as punishment, called "DistanceRank" to compute ranks of web pages. The distance is defined as the number of "average clicks" between two pages. The objective is to minimize punishment or distance so that a page with less distance to have a higher rank. Experimental results indicate that DistanceRank outperforms other ranking algorithms in page ranking and crawling scheduling. Furthermore, the complexity of DistanceRank is low. We have used University of California at Berkeley's web for our experiments.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Web ranking; Crawling; Web graph; Reinforcement learning

1. Introduction

The past decade has witnessed the birth and explosive growth of the World Wide Web. Exponential growth in the number of web servers has extended the web from a few dozen in 1981 up to over 400 millions today (ISC, 2007). Web servers can potentially host millions of pages which make the number of web pages extremely difficult to track. To track web pages, one has to download them, a process that can take weeks or months. Unfortunately, during this process, a considerable fraction of pages might have been modified, deleted or added. New pages are created at the rate of 8% per week. It is expected that only 20% of the current pages will be accessible after one year (Ntoulas, Cho, & Olston, 2004). As a rough (under) estimate of the total number of pages, Google claims that it has indexed 3 billion documents at the end of 2001, 4.28 billions on March 2004 (Google, 2004) and about 8 billions pages currently (Gulli & Signorini, 2005).

One of the most important challenging issues in any web search engine is finding high quality web pages. Quality of pages is defined based on the user preferences. Then, the problem of ranking is to sort web pages

* Corresponding author. Tel.: +98 21 66946927; fax: +98 21 8497642.
E-mail addresses: zare_b@ece.ut.ac.ir (A.M. Zareh Bidoki), yazdani@ut.ac.ir (N. Yazdani).

0306-4573/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2007.06.004

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

based on users' requests or preferences. Definitely, to make the web more interesting and productive, we need a good and efficient ranking algorithm for crawling and searching. Unfortunately, search engines do not index the entire web (Gulli & Signorini, 2005; Lawrence & Giles, 1999). Therefore, we have to focus on the most valuable and appealing pages. To do this, a better ranking criterion is required and a more efficient mechanism has to be devised. This will enable search engines to present the best related pages to the user in response to her queries. However, current ranking algorithms have low precision and high complexity. Meanwhile, they are sensitive to the "rich-get-richer" problem (Cho et al., 2005) meaning that the popular web pages receive even more popularity by time passing. Obviously, we need a solution to remedy these problems and make web ranking algorithms fairer.

In fact, web search is a subset of the general Information Retrieval (IR) problem. In Information Retrieval (Baeza-Yates & Ribeiro-Neto, 1999), the system tries to find documents related to the user query. IR Algorithms usually work based on matching words in documents. However, the web consists of huge unstructured documents linked together which create a massive graph. This poses new challenges to IR. New algorithms use links between web pages beside words relevancy in the documents with respect to a user query. Previous studies show that algorithms using hyperlinks for ranking yield good results (Henzinger, 2001). Their main powers come from using content of other pages to rank current pages. In other words, links carry information which can be used to evaluate relative importance of pages to the user query. Instances of the connectivity-based ranking algorithms are PageRank (Page, Brin, Motwani, & Winograd, 1998), HITS (Kleinberg, 1999) and OPIC (Abiteboul, Preda, & Cobena, 2003).

In this paper, we propose a ranking algorithm, called DistanceRank, based on reinforcement learning (Sutton & Barto, 1998) in which the distance between pages are considered as punishment. We define the distance as the number of "average clicks" between two pages or logarithm of a page's out degree (number of output links) as defined in Matsuo, Ohsawa, and Ishizuka (2003). In our method the main objective is to minimize the sum of received punishments (distance) by the user agent so that the page with the low distance will have a higher rank.

The distance d_j of page j is computed as $d_j = (1 - \alpha) * d_j + \alpha * \min_i(\log(\mathrm{O}(i)) + d_i)$ where i is a member of pages that point to j and $\mathrm{O}(i)$ shows out degree of i and α is the learning rate of the user.

One of major contribution of our method is modelling a user surfing the web randomly. Initially, a user browsing the web does not have any background about pages and she clicks based on the current status (page's content) of each page under consideration. As time goes on, the user selects a page (clicks on a link) based on both her background and the current content of each page. As she continues, she accumulates more knowledge from the environment and other web pages, and improves her link or page selection process.

We have used University of California at Berkeley's web to evaluate our algorithm. Our algorithm outperforms other ranking algorithms like PageRank and OPIC while removing some restricted conditions and problems. The time complexity of our solution is about $\mathrm{O}(p^*|E|)$ where $p \ll V$ and V and E are number of nodes and edges (links) in the web graph while p shows number of iterations for convergence. For instance in our experiments, we found that 5 iterations are sufficient for 5 million pages. In comparison with others, our solution has a higher throughput, i.e. it finds important pages faster than others. Furthermore, it sounds that our approach is less sensitive to the "rich-get-richer" problem and we will try to asses this problem with more experiments and analysis in the future.

Next section discusses our solution, DistanceRank. Experimental analysis and comparison with some of the well-known algorithms come in Section 3. Section 4 explains the convergence rate of DistanceRank. In Section 5, ranking problems is explained. Section 6 reviews related work. Finally, our conclusion and future work come in Section 7.

2. DistanceRank

To remedy ranking algorithms shortcomings, we propose an intelligent ranking algorithm based on reinforcement learning (Sutton & Barto, 1998) such that the distance between pages is considered as a punishment factor. Indeed, we use the minimum number of average clicks that a user requires to reach from page i to another page j. Since related pages have been usually linked to each other, the distanced based solution

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

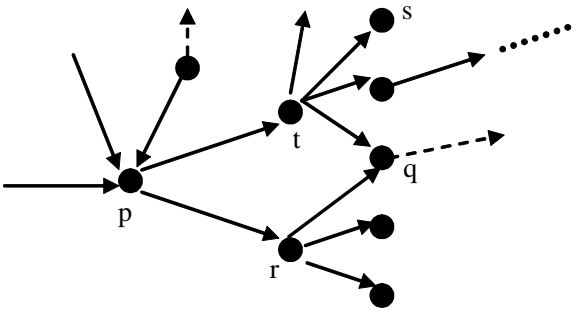

Fig. 1. A portion of a graph. The distance between p and q is $\log(2) + \log(3)$.

can find pages with high qualities more quickly. In other words, each page that has less average distance from others has a higher rank.

In ranking algorithms like PageRank, the rank of each page is defined as the weighted sum of ranks of all pages having links to the page. Then, a page has a high rank if it has more back links or pages having links to this page have higher ranks. Intuitively, these two properties should be true for the distance. A page having many input links should have low distance and if pages pointing to this page have low distance then this page should have a low distance. The following definitions clarify our idea:

Definition 1. If page i points to page j then the weight of link between i and j is equal to $\mathrm{Log}_{10}\mathrm{O}(i)$ where $\mathrm{O}(i)$ shows i's out degree (number of forward links).

Definition 2. The distance between two pages i and j is the weight of the shortest path (the path with the minimum value) from i to j. We call this *logarithmic distance* and denote it with d_{ij}.

For example, in Fig. 1, the weight of out-links in pages p, r and t is equal to log(2),log(3) and log(4) respectively and the distance between p and q is equal to $\log(2) + \log(3)$ if the path p–r–q was the shortest path between p and q. As figure shows, the distance between p and s is $\log(2) + \log(4)$. Thus, although both s and q are in the same link level from p (two clicks) but q is closer to p. In the following, we use distance and the logarithmic distance interchangeable unless stated otherwise.

Definition 3. If d_{ij} shows the distance between two page i and j as Definition 2, then d_j denotes the *average distance* of page j and is defined as the following where V shows number of web pages:

$$d_j = \frac{\sum_{i=1}^{V} d_{ij}}{V} \tag{1}$$

In this definition, instead of the classical distance definition, a new definition called the average click is used. In our method, the weight of each link is equal to $\log(\mathrm{O}(i))$. Naturally, if there is no path between i and j, d_{ij} will be set to a big value (we use $\log N$ here).

Now, the distance vector or logarithmic distances between all web pages can be computed like simple weighted all pair's shortest path solution (Cormen, Leiserson, & Rivest, 2001). After distance computation, pages are sorted in the ascending order and pages with smaller average distances will have high ranking.

Experimentally, we found that PageRank ordering and the average distance ordering are similar.[1] Clearly, since the distance of each page is related to its back links, the average distance rank has local property like PageRank.

Our method is dependent on the out degree of nodes in the web graph like other algorithms. Besides, it follows the web graph like the random-surfer model (Brin & Page, 1998) used in PageRank in that each output link of page i is selected with probability $1/\mathrm{O}(i)$. To illustrate this fact, we define rank's effect of i on page j as the inverse product of the out-degrees of pages in the logarithmic shortest path between i and j (this is similar

[1] Two ranking algorithms are equal if their page ordering is equal.

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

to branching in Baeza-Yates, Boldi, & Castillo (2006)). For example, if there is the logarithmic shortest path with single length 3 from i to j like $i \to k \to l \to j$, then i's effect on j is $(1/\mathrm{O}(i)) * (1/\mathrm{O}(k)) * (1/\mathrm{O}(l))$. In other words, the probability that a random surfer started from page i to reach to page j is $(1/\mathrm{O}(i)) * (1/\mathrm{O}(k)) * (1/\mathrm{O}(l))$. Lemma 1 illustrates the relation between the rank effect and the distance.

Lemma 1. *Suppose d_{ij} shows the logarithmic distance between page i and j and r_{ij} shows i's rank effect on j in the shortest path from i to j, then, if $d_{ij} < d_{ik}$ then $r_{ij} > r_{ik}$.*

Proof. We have

$$d_{ij} = \sum_{s \in path(i,j)} \log \mathrm{O}(S) = \sum_{s \in path(i,j)} -\log\left(\frac{1}{\mathrm{O}(s)}\right) = -\log \prod_{s \in path(i,j)} \frac{1}{\mathrm{O}(s)} = -\log r_{ij}$$

Since $d_{ik} = -\log r_{ik}$ then, when $d_{ij} < d_{ik}$, we will have $r_{ij} > r_{ik}$. □

Therefore, if the distance between i and j was less than the distance between i and k, then, i's rank effect on j is more than on k, in other words, the probability that a random surfer reach j from i is more than the probability to reach k.

In average distance, problems like sinking pages (Arasu, Cho, Garcia-Molina, Paepcke, & Raghavan, 2001), pages with no output and input, will not affect our distance measure. Moreover, it is not necessary that the web graph to be aperiodic. The good news is that the "Circular Contribution effect" (Wang, 2004) is also eliminated. Since we model both surfer and the page creator behavior, therefore, factors likes damping factor in PageRank are not required due to fact that we always work with the real web graph.

The *main problem* of average distance is its complexity, $\mathrm{O}(|V| * |E|)$ (Motwani & Raghavan, 1995) while PageRank complexity in the worst case is $\mathrm{O}(|V| * |E|)$ and practically is $\mathrm{O}(100 * |E|)$ (Haveliwala, 1999) i.e 100 iterations for an acceptable ranking is sufficient. Thus, implementation of average distance is not practical in the real web with 11.5 billions pages (Gulli & Signorini, 2005). To make average distance measure practical, we proposed a new definition of ranking called *DistanceRank*. Intuitively, to compute average distance of each page, there is a dependency between the distance of each page and its back links. For example, if page j has only one back link and it is from page i, to compute the average distance for page j, d_j, using Definition 1, 2 and 3, we will have the following relation:

$$d_j = \frac{\sum_{k=1}^{V} d_{kj}}{V} = \frac{\sum_{k \neq i}^{V} (d_{ki} + d_{ij}) + d_{ij}}{V} = \frac{\sum_{k \neq i}^{V} d_{ki}}{V} + d_{ij} = \frac{\sum_{k=1}^{V} d_{ki} - d_{ii}}{V} + d_{ij} \overset{\text{Eq (1)}}{\to} d_j = d_i - \frac{d_{ii}}{V} + d_{ij}$$
$$\approx d_i + d_{ij} = d_i + \log(\mathrm{O}(i)) \tag{2}$$

Since the size of the web is huge, then, V inclines to ∞ and, consequently, $\frac{d_{ii}}{V}$ is negligible.

In general, suppose $\mathrm{O}(i)$ denotes the number of forwarding (outgoing) links from page i and $B(j)$ denotes the set of pages pointing to page j. The DistanceRank of page j, denoted by d_j, is given by [2]

$$d_j = \min_i (d_i + \log \mathrm{O}(i)), \quad i \in B(j) \tag{3}$$

For example in Fig. 1, the distance d_q is computed as the following:

$$d_q = \min\{d_t + \log 4, d_r + \log 3\} = \min\{d_p + \log 2 + \log 4, d_p + \log 2 + \log 3\} = \{d_p + \log 2 + \log 3\}$$
$$= d_p + 0.77.$$

Considering this equation, we experimentally found that DistanceRank is similar to PageRank in ranking pages. Using Eq. (3), we propose the following formula based on the Q-learning that is a type of reinforcement learning algorithm (Sutton & Barto, 1998) to compute the distance of page j (i links to j).

$$d_{j_{t+1}} = (1-\alpha) * d_{j_t} + \alpha * \min_i (\log(\mathrm{O}(i)) + \gamma^* d_{i_t}), \quad i \in B(j),\ 0 < \alpha \leqslant 1,\ 0 \leqslant \gamma \leqslant 1 \tag{4}$$

[2] $\min_i(f(i))$ is equal to $f(i)$ with minimum value.

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

where α is learning rate and $\log(O(i))$ is the instantaneous punishment it receives in transition state from i to j. d_{j_t} and d_{i_t} show distance of page j and i in time t respectively and $d_{j_{t+1}}$ is distance of page j at time $t+1$. In other words, the distance of page j at time $t+1$ depends on its previous distance, its father distance (d_i) and $\log(i)$, the instantaneous punishment from selection page j by the user. The discount factor γ is used to regulate the effects of the distance of pages in the path leading to page j on the distance of page j. For example, if we have a path $s \to t \to u \to v$, then the effect of the distance of s on u is regulated with a γ^3 factor. In this environment, we are going to decrease sum of received punishments. Since the above equation is based on the reinforcement learning algorithm, it will converge finally and reach to the global optimum state (Sutton & Barto, 1998).

The value of the learning rate α comes from Eq. (5) where t shows time or iteration number and β is a static value to control regularity of the learning rate. Experimentally, we found that if the learning rate α is properly adjusted the system will convergence and reach to the stability state very fast with a high throughput. In the initial state of the algorithm, distances of pages are not known, so initially, we set α to one and, then, decrease it exponentially to zero

$$\alpha = e^{-\beta * t} \tag{5}$$

From the learning point of view, the user is an agent surfing the web randomly and in each step it receives some punishments from the environment. The system goal is to minimize sum of punishments. In each state, the agent has some selections, next pages for click, and the page with the minimum received punishment (distance) will be selected as the next page for visiting. Obviously, we can write Eq. (4) as follows:

$$d_j = \alpha * (\text{previous punishment of selecting } j) + (1-\alpha) * (\text{current punishment} + \text{instantaneous punishment that user will receive from selection } j)$$

So d_j is the total punishment that the agent receives from selecting page j.

One of the main contributions of our method is modelling the real user surfing web. Intuitively, when a user start browsing from a random page, she does not have any background about the web. Then, by surfing and visiting web pages, she clicks links based on both her pervious experiences and the current status (content) of web pages. By the time goes, she continuously accumulates knowledge which help the user to reach her goal and favourite pages faster. Our solution follows this step like a real user. First, its knowledge is little about web pages (environment), $\alpha = 1$, and, then, by visiting more pages, the system slowly learns more information (α decreases) and effectively selects next pages. Clearly, the learning rate is high at the beginning and as the time goes on the learning rate decreases.

As Eq. (4) shows DistanceRank is computed recursively like PageRank. The process iterates to converge. We show in the next section that it is possible to compute distances with $O(p * |E|)$ time complexity when $p \ll V$ which is very close to an ideal state. For instance, p is 5 for 5 millions pages implying that 5 iterations for an acceptable ranking is sufficient.

After convergence, we will have the DistanceRank vector. Pages are sorted in the ascending order and those with low DistanceRank will have high ranking. DistanceRank is computed using power method (Watkins, 2002) as the following pseudo code illustrates. First, all members have very large value and ε shows the error:

```
/*D is DistanceRank vector */
D_0 ← {∞, ∞, ∞, ...}
itr = 0; /* iteration number */
While δ > ε
  α ← e^(−β * iter)
  itr ← itr + 1
  For every page j ∈ V
    D_n[j] ← (1 − α) * D_{n−1}[j] + α * min_i(γ * D_{n−1}[i] + log(O[i]))
      i ∈ B(j), 0 < α ⩽ 1, 0 ⩽ γ ⩽ 1
  δ ← ‖D_{n−1} − D_n‖
End while
```

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

3. Experimental results

We used University of California at Berkeley's Web site with five millions web pages to evaluate DistanceRank. Two scenarios were used: crawling scheduling and rank ordering. In the crawling scheduling the goal was to find more important pages faster in the crawling process. But in the rank ordering, we compared the ordering of DistanceRank with PageRank and Google's rank with and without with and without respect to a user query.

First, the crawling scheduling scenario is used for comparison between ranking algorithms. We compared our results with the following ranking algorithms:

Breadth-first: In this strategy, the crawling process is done in the breadth-first ordering. Initially, the algorithms start with some starting URLs as the root of crawling tree (Najork & Wiener, 2001).

Back-link count: In this algorithm, first, we crawl the pages with more input links (Cho, Garcia-Molina, & Page, 1998). Pages with more input links have higher ranks.

Partial PageRank: This algorithm runs the PageRank algorithm on web pages seen so far and crawl the web pages with a high PageRank first. The PageRank algorithm is now an old algorithm that is no longer representative of the results returned by Google search engine.

Partial DistanceRank: The distance vector of the current web graph seen so far is computed and the web pages with a small DistanceRank are crawled first.

OPIC: In this algorithm, all pages start with the same amount of cash (Abiteboul et al., 2003). Every time a page is crawled, its cash is distributed to its forward links. In each step the next page for crawling is the one with highest cash it has received till now (the priority is cash). This method is online and fast.

First, the algorithm starts crawling the web with some starting URLs. Every time K (K is set to 250,000) new web pages are crawled; we run one of the ranking algorithms and sort the web pages in the queue considering their ranking. This process continues until a specified portion of the web is crawled. The global crawling and ranking algorithm is as the following. All methods run this procedure with their own ranking criteria (Cho et al., 1998):

Algorithm 1. The crawling algorithm

```
Input: starting_url, K = 250,000
enqueue(url_queue, starting_url)
while (not empty(url_queue))
    url = dequeue(url_queue)
    crawl_page(url)
    for each child u of url
      if (u ∉ url_queue and (u) ∉ crawled_pages)
            enqueue(url_queue, u)
    if (crawled_pages.count() %k == 0)
        reorder_queue(url_queue)End while
```

The enqueue(queue, element) function appends an element at the end of queue, dequeue(queue) removes the element at the beginning of queue and returns it and reorder_queue(queue) reorders queue using a ranking algorithm as the following:

1. **breadth first**
   do nothing
2. **backlink count**
   For each u in url queue
   backlink count[*u*]=*number of u input links*
   sort url_queue by backlink count[*u*]
3. **PageRank**
   Solve the following set of equations:

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

$PR[u] = (1 - 0.85)/n + 0.85\sum_{p \in B[u]} \frac{PR[p]}{O[p]}$ *where* $B[u]$ *shows list of page that link to u and* $O[p]$ *is the number of links in the page p*

Sort url_queue by $PR(u)$

4. **DistanceRank**

$Distance_n[j] \leftarrow (1 - \alpha) * Distance_{n-1}[j] + \alpha * \min_i(\gamma * Distance_{n-1}[i] + \log(O[i]))$,

$i \in B(j)$

B(j) shows list of pages that link to j and $O(i)$ *is the number of out links in page i*

Sort url_queue by distance vector in ascending order

For comparison, we chose PageRank as an ideal ranking mechanism. First, ranks of all web pages are computed using PageRank algorithm on the entire graph. The aim of crawling is to find hot pages, pages with high ranking in PageRank. In a set of k pages, gathered from running a crawling algorithm, a page is hot if it exists in the first k hot pages of the ideal ranking. Clearly, the algorithm that retrieves more hot pages will be better. We define throughput as the fraction of the crawled hot pages to all hot pages that can be discovered.

We compared previously discussed algorithms with DistanceRank in Fig. 2 for five millions web pages. In this experiment, damping factor of PageRank and β are set to 0.85 and 0.1 respectively. As the figure shows, DistanceRank outperforms other algorithms in terms of throughput. For instance, when 65% of pages are crawled, DistanceRank finds around 81% of hot pages whereas batch PageRank and OPIC find 74% and 73% respectively. In comparison to PageRank and OPIC, DistanceRank has 5% more throughput. This result is achieved compared to the ideal PageRank. In other words; DistanceRank finds high important pages faster than PageRank.

Modelling a real user browsing on the web is the main advantage of DistanceRank. A user is going to find her favourite pages faster and by clicking and visiting a new page, she accumulates more knowledge about the environment. Thus, the next clicks are done with more information and a better selection.

In reinforcement learning algorithms, the learning rate (α) plays an important role in convergence of the system. In Fig. 3, we compared throughput of DistanceRank for different learning functions such as exponential (Eq. (5)), x^{-1}, power law ($x^{-2.1}$) (Broder et al., 2000) and linear (x) where in all functions x was $\beta * t$. As the figure shows exponential and linear functions are better than others. In other words, the best function for learning rate is the one with the soft ramp. These functions model user behavior better than others. Relatively, a user at first learns more information from the system and by the time passes and the search proceeds, the rate of learning deceases slowly.

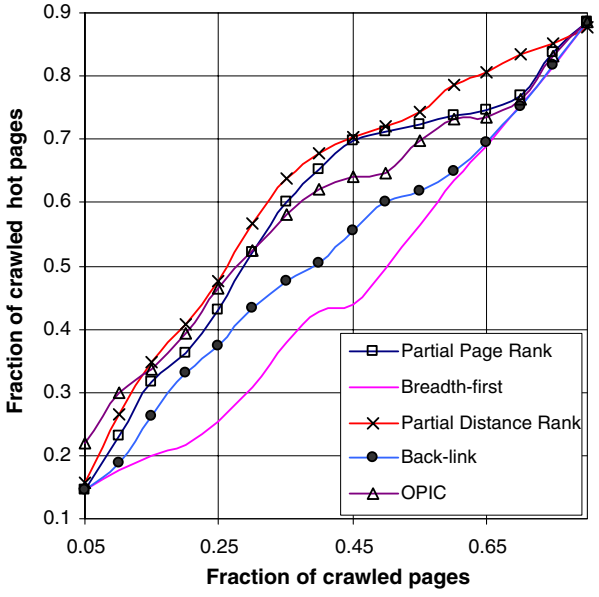

Fig. 2. Throughout of ranking algorithms in that ideal ranking is PageRank.

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

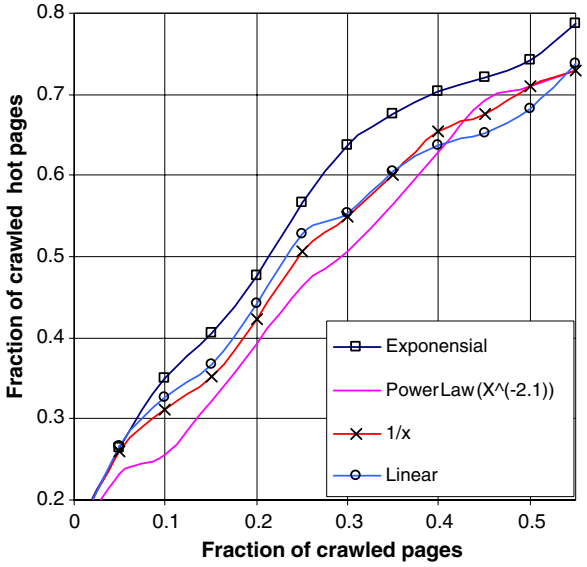

Fig. 3. Throughout of DistanceRank algorithm for different leaning rate functions ($x = \beta * t$).

We consider throughput of DistanceRank for different β in the exponential learning rate. Fig. 4 shows the results for three values of β. As the figure shows $\beta = 0.1$ sounds the best selection. This value causes the learning rate to decrease more slowly and simulates better the user behavior.

The second approach for comparison is rank ordering. In the rank ordering, we measure similarity between two ordered lists of PageRank and DistanceRank.

We used Kendall's τ metric (Kendall, 1970) for correlation between two rank lists. In Kendall's τ metric, two identical lists have $\tau = 1$, while two totally uncorrelated lists have $\tau = 0$ and reversed lists have $\tau = -1$. In Table 1, we calculated this coefficient for 30,000 samples of randomly selected pages ordering with each strategy against a list of pages ordered with PageRank on whole graph.

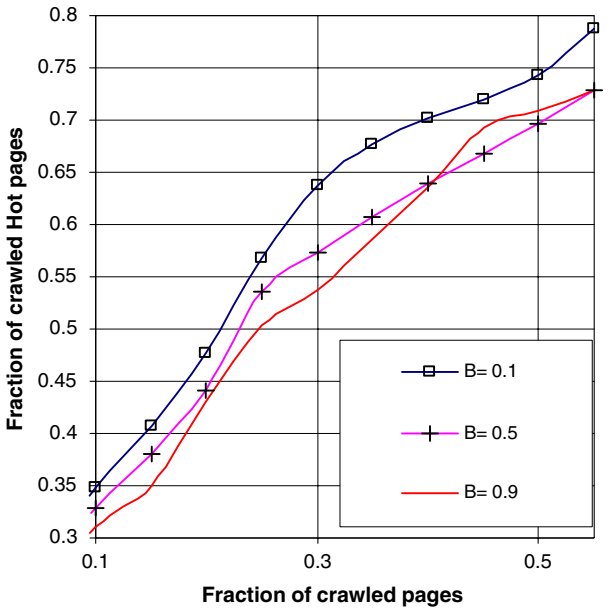

Fig. 4. Throughout of DistanceRank algorithm for different Beta in exponential learning.

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

Table 1
Kendall factor of ranking algorithms against PageRank

Algorithm	Kendall's Tau
Breadth-first	0.11
Back Link	0.40
OPIC	0.62
DistanceRank	0.75

DistanceRank has highest Kendall, i.e. 0.75. Naturally, this factor depends on α and β. In our experiments, Kendall's τ between DistanceRank and PageRank was between 0.55 and 0.8.

To evaluate quality of results, we compared top 20 results from DistanceRank, PageRank and Google's ranking (results from Google search engine on 20 April 2007) for some queries in a subjective manner. First, all of five millions pages are indexed. Then, after query processing, results are ordered by DistanceRank and PageRank separately. We also forced Google engine to search query only at University of Berkeley's web site (by site: .berkeley.edu instruction). Table 2 shows list (index) of all titles and URLs of these three ranking algorithms for "**web AND crawling**" query. Table 3 Shows list of top 20 results ordered by each algorithm. For each web page, we specify degree of relevance to the given query in two categories: high relevant (1 value) and low relevant (0 value). High relevant pages are specified by p_i^* and low ones are specified by p_i. Then, precision of each algorithm was computed as the proportion of relevant pages to all the pages retrieved (20 pages). Table 3 shows the result. In this experiment, precision of DistanceRank was better than both PageRank and Google's rank. Intersection of results between Google and DistanceRank is 9, while intersection between Google and PageRank is 7. In other words, results of DistanceRank are more close to Google than PageRank. Also the intersection between DistanceRank and PageRank is 10. Kendall factor between these 10 results from PageRank and DistanceRank is about 0.60. Furthermore, this experiment has been done for **"CORBA AND DCOM"** query that have been shown in Tables 4 and 5.

Above experiments show that DistanceRank is a suitable algorithm for ranking of web pages. Exact qualification of DistanceRank needs more experiments that remain as future work.

4. DistanceRank convergence

DistanceRank, like PageRank in the ideal state, needs V iterations to converge. However, in practice we can get the same results with very less iterations. Several measures can be used to analyze the convergence speed. One is the norm of difference between DistanceRank vectors from successive iterations. A more useful measure is looking at the ordering of pages produced by DistanceRank vector. In the rank ordering, we measure similarity between two ordered lists of DistanceRank. In many scenarios, we are only concerned with top pages and not necessarily to their exact ordering. We define similarity of two sets A and B as $\left|\frac{A\cap B}{A\cup B}\right|$ (Haveliwala, 1999). To visualize how closely two ranking match on identifying top pages, we successively computed similarity among top n pages in each ordering.

Therefore, we use a solution like the rank ordering to find the similarity between sets produced by different number of iterations. Fig. 5 shows similarity for 20,000 till 600,000 top pages. As the figure shows, the ordering obtained by only 5 iterations agree closely with ordering of 20 iterations for 5 million pages. In other words, a few number of iterations is sufficient to find satisfactory results. Considering this, the time complexity of algorithm can be reduced to $O(p * |E|)$ where $p \ll V$ and p shows number of iterations for convergence.

5. DistanceRank and ranking problem

One of the main problems in the current search engines is "rich-get-richer" that causes young high quality pages receive less popularity. In other words, popular high rank pages have more chances to be browsed by users and, consequently, young high quality pages can be the victims.

To sense effect of this problem, in Cho and Roy (2004), two models on how users discover new pages, Random-Surfer and Search-Dominant model have been introduced. In Random-Surfer, a user finds new pages

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

Table 2
List of all titles and URLS from top 20 results from Google, PageRank and DistanceRank algorithms (Query: web AND crawling)

Index	URL and title	Index	URL and title
P1*	Distributed **Web Crawling** over DHTs | EECS at UC Berkeley http://www.eecs.berkeley.edu/Pubs/TechRpts/2004/5370.html	P2*	CS294-4 Project Presentations http://www.cs.berkeley.edu/~kubitron/courses/cs294-4-F03/...
P3*	Modern Information Retrieval - Chapter 13 http://www.ischool.berkeley.edu/~hearst/irbook/chapters/chap13.html	P4*	SIMS 202 Fall '00 Schedule and Assigments http://www2.sims.berkeley.edu/courses/is202/f00/Assignments.html
P5*	Web Crawling http://db.cs.berkeley.edu/postmodern/lecs/jrvb/sld012.htm	P6*	SIMS 202 Fall '01 Homepage http://www2.sims.berkeley.edu/courses/is202/f01/Assignments.html
P7	Micronet Mail Archive: Re: [Micronet] concerns regarding ... http://ls.berkeley.edu/mail/micronet/2001/0357.html	P8	10.25.00 - Election 2000**Web** site at UC Berkeley... http://www.berkeley.edu/news/media/releases/2000/10/25_web.html
P9*	SIMS 202 Reader, Fall 00 http://www2.sims.berkeley.edu/courses/is202/f00/Reader.html	P10*	SIMS 202 Fall'04 Assignments http://www2.sims.berkeley.edu/courses/is202/f04/Assignments.html
P11	Secure Content and Private Areas Guidelines ... http://www.lib.berkeley.edu/digicoll/libraryweb/secure_content.html	P12*	Cha-Cha: A System for Organizing Intranet Search Results http://www.ischool.berkeley.edu/~hearst/papers/usits99/index.html
P13*	SIMS 141: Search Engines: Technology, Society, and ... http://www2.sims.berkeley.edu/courses/is141/f05/resources.html	P14*	The PIER Project http://pier.cs.berkeley.edu/papers.html
P15	Ninja Demos http://ninja.cs.berkeley.edu/demos/demos.html	P16	Computers Don't Teach – People Teach http://ninja.cs.berkeley.edu/demos/demos.html
P17*	EE122 Fall 2006 - Project 2 http://inst.eecs.berkeley.edu/~ee122/fa06/\/project2.htm	P18	11.02.00 - Election **Web** site shows power of new Internet... http://www.berkeley.edu/news/berkeleyan/2000/11/01/deepweb.html
P19	internet growth data http://www2.sims.berkeley.edu/research/projects/how-much-info/...	P20	296a1 Summary – Seminar Information Access Fall 2001 http://www2.sims.berkeley.edu/courses/is296a-1/f01/summary.html
P21*	SIMS 202 Fall '00 Schedule and Assigments http://www2.sims.berkeley.edu/courses/is202/f00/Lectures.html	P22*	Mining the Web http://www.cs.berkeley.edu/~soumen/mining-the-web/
P23*	Supplemental Readings for IS202, Part II http://www2.sims.berkeley.edu/academics/courses/is202/f97/reader2.htm	P24*	SIMS 290-2: Search Engines: Technology, Society... http://www2.sims.berkeley.edu/courses/is290-2/f05/resources.html
P25*	Information Retrieval on the World Wide Web http://db.cs.berkeley.edu/postmodern/lecs/jrvb/	P26*	Soumen Chakrabarti http://www.cs.berkeley.edu/~soumen/
P27*	SIMS 202 Fall '02 Assignments http://www2.sims.berkeley.edu/academics/courses/is202/f02/...	P28*	Boon Thau Loo http://www.eecs.berkeley.edu/~boonloo/courses.html
P29	Yoda's List of Jungle Cruise Jokes http://www.csua.berkeley.edu/~yoda/disneyland/jungle.htm	P30	Berkeley Parents Network: Postpartum Tummy http://parents.berkeley.edu/advice/pregnancy/ab_exercise.html
P31	Berkeley Parents Network: Advice about Car Trips http://parents.berkeley.edu/advice/going/longcartrips.html	P32	Internet Privacy Protection http://sis.berkeley.edu/SIS/sis-training/computing/internet-privacy.htm
P33	hardboiled 5.3 - mailbox overlows! http://hardboiled.berkeley.edu/issue5.3/letters.html	P34	Deployment http://calagenda-test.berkeley.edu/software/docs/9.0.4/install/....
P35	Cal Band Alumni Association:History Book, Ch 7 http://www.calband.berkeley.edu/calband/cbaa/historybook1993/07/	P36	Animus http://www.ocf.berkeley.edu/~tbischel/Blog.htm
P37	Oracle Ultra Search http://calagenda-test.berkeley.edu/software/docs/9.0.4/relnotes_904/...	P38*	SIMS 202 schedule http://www2.sims.berkeley.edu/academics/courses/is202/f97/Schedule...

only by surfing randomly (without search engine) while in Search-Dominant model user finds new pages only by search engine. The authors found that it takes 60 times longer for a new page to become popular under

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

Table 3
Top 20 results from each ranking algorithm and their precisions (Query: web AND crawling)

	Google	DistanceRank	PageRank
	P1*	P21*	P30
	P2*	P22*	P12*
	P3*	P23*	P31
	P4*	P13*	P29
	P5*	P24*	P21*
	P6*	P9*	P3*
	P7	P6*	P10*
	P8	P10*	P26*
	P9*	P25*	P13*
	P10*	P26*	P32
	P11	P4*	P25*
	P12*	P12*	P33
	P13*	P27*	P34
	P14*	P28*	P4*
	P15	P29	P20
	P16	P1*	P14*
	P17*	P3*	P28*
	P18	P5*	P35
	P19	P37	P27*
	P20	P38*	P36
Precision	12/20 = 0.6	18/20 = 0.9	11/20 = 0.55

Search-Dominant than Random-Surfer model. In Cho et al. (2005), a solution has been proposed which is based on the PageRank formula. The formula applies the difference between PageRank of two snapshots to PageRank algorithm.

Obviously, if page p be a young high quality page, it will receive more input links than other young low quality pages in a period. A ranking algorithm is less sensitive to "rich-get-richer" if it can find high quality pages and increase their popularity earlier (Cho et al., 2005). In other words, it should predict popularity that pages will gather in the future. Thus, our goal is a ranking algorithm with good prediction of future ranking.

Fairly, we have found that the "rich-get-richer" problem is less important in DistanceRank in comparison with PageRank. To illustrate this, we computed PageRank for whole graph as the base rank or future ranking that we are going to predict. To produce web graphs in the past, we changed the base graph by removing some edges randomly and computed both DistanceRank and PageRank algorithms for sub graphs as partial ranking. Finally, τ between the base rank and partial rankings was computed. So the algorithm with suitable prediction will have more Kendall factor. Fig. 6 shows the results. Naturally, for complete evaluation we need some snapshots for web graph that remains as future work.

As the figure shows, when there are more changes in the graph, DistanceRank predict their future ranks closer to the base rank in comparison to PageRank. For instance, when 70% of edges are removed, τ is 0.36 and 0.32 for DistanceRank and PageRank respectively. Furthermore, we might able to conclude that DistanceRank is more robust than PageRank in the rich-get-richer problem. Of course, it is not a fully convinced experiment and we need more experiments.

This gain is achieved since DistanceRank is based on an intelligent random-surfer model. We proved this property in lemma 2. Intuitively, young high quality pages have more relative rank increasing[3] (Cho & Roy, 2004) in two snapshots of web or increasing rate of their ranks is more than others. Because DistanceRank follow Q-learning, and Q-learning considers the difference between current and previous punishment, so it will predict high quality pages faster and its sensitivity to "rich-get-richer" is less than PageBank.

Lemma 2. *DistanceRank is more robust than PageRank in rich-get-richer problem.*

[3] Relative increase in ranking = (current rank − previous rank) current rank.

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

Table 4
List of all titles and URLS from top 20 results from Google, PageRank and DistanceRank algorithms (Query: CORBA AND DCOM)

Index	URL and Title	Index	URL and Title
P1*	Group G's Homepage https://www2.sims.berkeley.edu/academics/courses/is206/f97/GroupG/...	P2*	CORBA vs DCOM https://www2.sims.berkeley.edu/courses/is206/f97/GroupG/corbavdcom..
P3*	Object Web Paper https://www2.sims.berkeley.edu/courses/is206/f97/GroupF/objectweb...	P4*	A CAMPUS DIGITAL IMAGE REPOSITORY https://sunsite.berkeley.edu/moa2/papers/braindump2-03.html
P5*	The Making of American Project Proposal https://sunsite.berkeley.edu/moa2/moaproposal.html	P6*	CS 194 Lecture Schedule & Notes, Fall 2002 https://www.cs.berkeley.edu/~istoica/classes/cs194/05/class.html
P7*	Streaming Media Middleware is more than Streaming Media https://bmrc.berkeley.edu/papers/2001/159/SMWorkshop01.html	P8	The START Application Milestone 5: Distributed Application https://www2.sims.berkeley.edu/courses/is206/f98/GroupF/milestone5...
P9*	Sept. 10th, 1999: SEC Study Group: CORBA – Part II https://sec.eecs.berkeley.edu/studygroup/1999Fall/study-sept10-1999/...	P10	Impact of Network-Centric Computing on the Software ... https://groups.haas.berkeley.edu/citm/conferences/cec/papers/iyer/Iyer...
P11	Changing the Game https://www-inst.eecs.berkeley.edu/~eecsba1/sp98/reports/eecsba1g/...	P12*	IS 206 Projects Fall 1997 https://www2.sims.berkeley.edu/courses/is206/f97/Project.html
P13	[Gimp-developer] Re: Layers, dialogs and other bits of love on https://lists.xcf.berkeley.edu/lists/gimp-developer/2001-February/...	P14*	International Computer Science Institute Talks https://www.icsi.berkeley.edu/talks/previous/1997/kraemer2.html
P15*	e-Ptolemy: A Distributed Framework for Software Systems https://ptolemy.berkeley.edu/projects/summaries/02/e_Ptolemy.html	P16*	IS 206 Assignments https://www2.sims.berkeley.edu/courses/is206/f98/hmwk_11.html
P17	Using COM facilities in R https://www.stat.berkeley.edu/~nolan/stat133/Fall05/lectures/DCOM. ..	P18*	Implementation Technologies https://Sunsite.berkeley.edu/moa2/csg/tsld010.htm
P19*	CORBA www2.sims.berkeley.edu/courses/is206/f97/GroupG/corbavdcom/tsld001	P20	Diff between Abstract and Interface(rephrase) https://lists.xcf.berkeley.edu/lists/advanced-java/1999-September/...
P21	Status Report for Component Based Logic Digital Simulation http://embedded.eecs.berkeley.edu/Alumni/zhengyu/arpa_report/status...	P22	Intranet Server Suites http://www2.sims.berkeley.edu/courses/is206/f97/GroupA/...
P23*	java http://www2.sims.berkeley.edu/courses/is206/f97/GroupC/java.html	P24	Computer Aided Design for VLSI for 2003 http://www.eecs.berkeley.edu/IPRO/Summary/03abstracts/chapter2.html
P25*	Comparing ActiveX and CORBA/IIOP http://embedded.eecs.berkeley.edu/Alumni/zhengyu/arpa_report/activex..	P26*	Distributed Hierarchical Digital Simulation http://embedded.eecs.berkeley.edu/Alumni/zhengyu/arpa_report/project..
P27*	Brown/Pasetti/Pree/Henzinger/Kirsch: A Reusable and Platfo.. http://www.eecs.berkeley.edu/~tah/Publications/a_reusable_and_platfor...	P28*	Some Potential Key Technologies http://www.eecs.berkeley.edu/~newton/Presentations/WebArchTuto...
P29	An Infrastructure Approach to Context-Aware Computing http://guir.berkeley.edu/projects/confab/pubs/context-essay-final.htm	P30*	An Actor Oriented Framework for Distributed Software ... http://ptolemy.berkeley.edu/projects/summaries/05/ePtolemy05.html
P31	Distributed Design of Electronic Systems - March 1998 ... http://embedded.eecs.berkeley.edu/Respep/Research/weld/arpa/98q2/	P32	Interface and Repository Considerations for a System http://sunsite3.berkeley.edu/moa2/papers/dump2.html

Proof. It has been proved in Cho et al. (2005) that the rich-get-richer is improved by changing the rank computation as the following:

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

Table 5
Top 20 results from each ranking algorithm and their precisions (Query: CORBA AND DCOM)

	Google	DistanceRank	PageRank
	P1*	P12*	P12*
	P2*	P21	P1*
	P3*	P22	P7*
	P4*	P1*	P18*
	P5*	P23*	P23*
	P6*	P7*	P29
	P7*	P5*	P22
	P8	P4*	P21
	P9*	P18*	P31
	P10	P24	P2*
	P11	P16*	P5*
	P12*	P25*	P28*
	P13	P26*	P4*
	P14*	P27*	P32
	P15*	P28*	P24
	P16*	P29	P26*
	P17	P3*	P25*
	P18*	P15*	P15*
	P19*	P30*	P16*
	P20	P2*	P30*
Precision	14/20 = 0.70	16/20 = 0.80	14/20 = 0.70

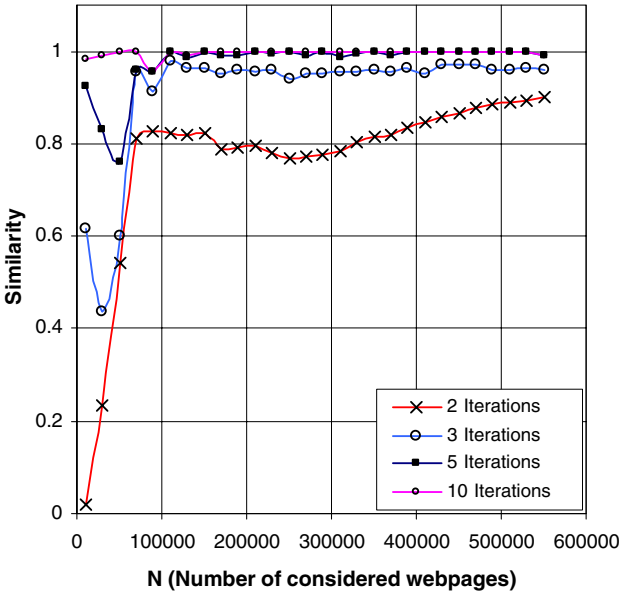

Fig. 5. The similarity between 2, 3, 5 and 10 with 20 iterations for 5 millions pages.

$$Q(p,t) = C^* \frac{\Delta PR_t/\Delta t}{PR(p,t)} + PR(p,t)$$

where $PR(p,t)$ and $Q(p,t)$ show PageRank and quality value of page p at time t respectively and $\Delta PR_t = PR(p,t) - PR(p,t-1)$. We can write this formula as

$$Q(p,t) = \beta * (\text{difference between current and previous PageRank}) + \text{current PageRank} \quad (6)$$

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

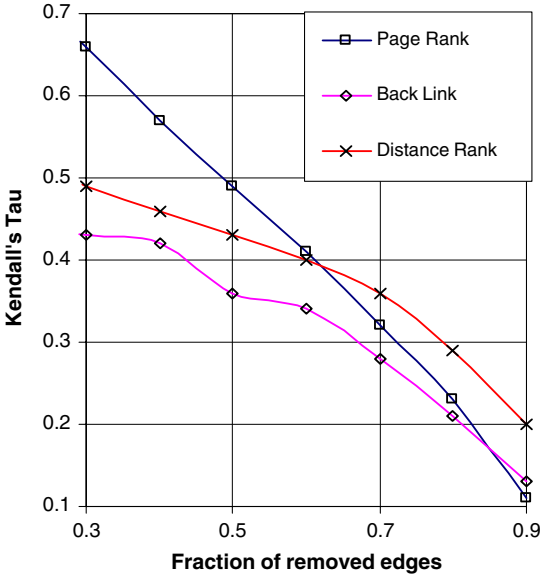

Fig. 6. Kendall factor between predicted PageRank (using PageRank, DistanceRank and Back-Link count) and real future PageRank.

From Eqs. (3 and 4) we have

$$d_c = \text{current distance} = \min_i(\log(O(i)) + \gamma * d_{i_t})$$

$$d_p = \text{previous distance} = d_{j_t}$$

Thus we have

$$\begin{aligned} d_{j_{t+1}} &= \alpha * d_{p+}(1-\alpha) * d_c = \alpha * (d_p - d_c) + d_c \\ &= -\alpha * (\text{difference between current and previous distance}) + \text{current distance} \end{aligned} \quad (7)$$

Since in DistanceRank we try to minimize distance (punishment), but in PageRank to maximize the rank value, then, we will have $-\alpha$ factor. From Eqs. (6) and (7) we conclude that DistanceRank is less susceptible to rich-get-richer. □

6. Background and related work

Roughly speaking, there are two categories of connectivity-based ranking algorithms (Henzinger, 2001), Query-independent and Query-dependent ranking. In the Query-independent ranking, a fixed score is assigned to each page in the collection. PageRank (Page et al., 1998), OPIC (Abiteboul et al., 2003) and (Castillo, Marin, Rodr'ýguez, & Baeza-Yates, 2004) are instances of the Query-independent scheme. In the Query-dependent or topic-sensitive ranking, we assign a score to each page in the collection of a specific query context. One of the well-known query-dependent solutions is HITS (Kleinberg, 1999).

The PageRank, an instance of the query-independent ranking, has been designed taking into account the known-relation between web pages. For example, if page p_1 has a link to page p_2, then, p_2's subject should be interesting for the p_1's creator. Therefore, the number of input links to a web page shows the interest degree of the page to others pages. Clearly, the interest degree of a page increases with the growing number of input links. Furthermore, when a web page receives links from an important page, then, this page should have a high rank. Therefore, PageRank of a web page corresponds to the weighting sum of input links.

Let pages on the web be denoted by $1, 2, \ldots, n$, and O(i) denotes the number of forwarding (outgoing) links from page i and $B(i)$ denotes the set of pages that point to page i. We assume that web pages form a strongly

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

connected graph, SCG, meaning that every page can be reached from any other page. The PageRank of page i, denoted by $r(i)$, is given by

$$r(j) = (1-d)/n + d * \sum_{i \in B(j)} r(i)/\mathrm{O}(i) \tag{8}$$

Thus, PageRank of page i is equal to sum of the input page ranks divided by their out degree. Dividing input pages ranks by their outputs degree, $N(i)$ has two effects. First, it distributes a PageRank to all outputs fairly and, secondly, it normalizes sum of each page effects and rank vector to one. Parameter d, damping factor, is used to remove effects of sink pages, pages with no outputs. Thus, in this way, each web page will have outgoing links to every single web page.

PageRank can be written as a linear relation $r = A^{\mathrm{T}} * r$ where r is an n dimensional vector $[r(1), r(2), \ldots, r(n)]$ and elements a_{ij} of matrix A are given by $a_{ij} = 1/N(i)$ if page i points to page j and $a_{ij} = 0$ otherwise. Our goal is to compute r that is the eigen vector of A^{T} for eigen value one (Watkins, 2002).

This mechanism is equivalent to the *Random-Surfer* behavior, a person who surfs the web by randomly clicking links on the visited pages. When the user reaches a page with no output links she will jump to a random page. Therefore, when a user is in a web page, with probability of d she will select one output link randomly or will a jump to other web pages with the probability of $1 - d$.

Abiteboul et al. (2003) proposed an algorithm called OPIC (On-line Page Importance Computation). In their method, each page has cash values that are distributed to all output links. This is similar to PageRank and is done in one step for all paths. In every state, the crawler will download the web page with higher cashes and when a page is found its cash will be distributed to its output links.

HITS (Hypertext-Induced Topic Search) proposed by Kleinberg is a query-dependent method that partitions the web pages as ''authority pages'' or ''hub pages''. An authority page is a ''topic core'' that includes relevant content for that topic. A hub page is a gate page linked to authority pages. So a page with a high authority is pointed by pages with a high hub score and vice versa. Each page is also assigned two values, authority and hub. Thus, hub and authority vectors will have reinforcing effects which iterate to converge to a final value.

Fortunately, these algorithms work well and are parts of the current search engines. However, there are many issues which have to be resolved. First, the rank model uses damping factors for all pages from the surfer model view to remove sinking effect that will change quality of pages since the rank value is computed for a different graph. Secondly, most of them suffer from rich-get-richer problem.

7. Conclusion and future work

We proposed a new iterative ranking algorithm for web pages called DistanceRank in this paper., This algorithm is based on reinforcement learning with the distance between pages as punishment. In other words; we are going to minimize the distance values of pages. We define the distance between two pages i and j as the logarithm of the number of i's output links when i points to j. Our algorithm models a real user surfing the web. When a user randomly browses the web, she selects the next pages based her background from the last pages and the current status of the web page. The learning rate, α, is used to model the user behavior in each state properly.

Naturally, the distance of each page can be computed from its inputs links. DistanceRank algorithm recursively iterates to converge to a static value. Finally, we have a vector as DistanceRank vector. Then, we sort the vector in the descending order and the pages with low distance will have high ranking.

The convergence speed of our algorithm is fast with a little number of iterations. We found experimentally that DistanceRank outperforms other ranking algorithm in page ranking and crawling scheduling.

In DistanceRank, it is not necessary to change the web graph for computation. Therefore, some parameters like the damping factor can be removed and we can work on the real graph. We also found partially and intuitively that the effects of problems like ''rich-get-richer'' in DistanceRank are less important than PageRank. The complete evaluation of this property will be done in the future.

We used University of California at Berkeley's Web to evaluate the proposed algorithm. We will use a bigger web graph and also a ground truth data set to better evaluate our algorithm. Also its susceptibility to rank

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004

spamming (Henzinger, Motwani, & Silverstein, 2002) remains as future work. We are also going to combine the logarithmic distance in PageRank, HITS and OPIC algorithms which believe will yield better results.

References

Abiteboul, S., Preda, M., & Cobena, G. (2003). Adaptive on-line page importance computation. In *Proceedings of the twelfth international conference on World Wide Web* (pp. 280–290).

Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A., & Raghavan, S. (2001). Searching the web. *ACM Transactions on Internet Technology (TOIT), 1*(1), 2–43.

Baeza-Yates, D., Boldi, P., & Castillo, C. (2006) Generalizing PageRank: Damping functions for link-based ranking algorithms. In *SIGIR* (pp. 308–315).

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. ACM Press/Addison-Wesley.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *Proceedings of 7th World Wide Web conference*.

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, et al. (2000). Graph structure in the web: Experiments and models. In *Proceedings of the ninth conference on World Wide Web* (pp. 309–320).

Castillo, C., Marin, M., Rodr'ýguez, A., & Baeza-Yates, R. (2004). Scheduling algorithms for web crawling. In *Latin American web conference (WebMedia/LA-WEB)* (pp. 10–17). IEEE CS Press.

Cho, J., & Roy, S. (2004). Impact of search engines on page popularity. In *Proceedings of the international World-Wide Web conference*.

Cho, J., Garcia-Molina, H., & Page, L. (1998). Efficient crawling through URL ordering. In *Proceedings of the seventh conference on World Wide Web, Brisbane, Australia*.

Cho, J., Roy, S., & E. Adams, R. (2005). Page quality: In search of an unbiased web ranking. In *Proceedings of ACM international conference on management of data (SIGMOD)*.

Cormen, T., Leiserson, C. E., & Rivest, R. L. (2001). *Introduction to algorithms*. Cambridge, MA: The MIT Press.

Google (2004). http://www.google.com/googlefriends/mar2004.html.

Gulli, A., & Signorini, A. (2005). The indexable web is more than 11.5 billion pages. *Special interest tracks and posters of the 14th international conference on World Wide Web, Chiba, Japan*.

Haveliwala, T. (1999). Efficient computation of pagerank. Technical Report, Database Group, Computer Science Department, Stanford University. Available from http://dbpubs.stanford.edu/pub/1999-31.

Henzinger, M. (2001). Hyperlink analysis for the web. *IEEE Internet Computing, 5*(1), 45–50.

Henzinger, M., Motwani, R., & Silverstein, C. (2002). Challenges in web search engines. *SIGIR Forum, 36*(2).

ISC (2007). http://www.isc.org/index.pl?/ops/ds/host-count-history.php.

Kendall, M. G. (1970). *Rank correlation methods*. London, England: Griffin.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM, 46*(5), 604–632.

Lawrence, S., & Giles, L. (1999). Accessibility of information on the web. *Nature, 400*, 107–109.

Matsuo, Y., Ohsawa, Y., & Ishizuka, M. (2003). Average-clicks: A new measure of distance on the World Wide Web. *Journal of Intelligent Information Systems, 20*(1), 51–62.

Motwani, R., & Raghavan, P. (1995). *Randomized algorithms*. Cambridge University Press.

Najork, M., & Wiener, J. L. (2001). Breadth-first crawling yields high-quality pages. In *Proceedings of the tenth conference on World Wide Web* (pp. 114–118). Elsevier Science.

Ntoulas, A., Cho, J., & Olston, C. (2004). What's new on the web? The evolution of the web from a search engine perspective. In *Proceedings of the 13th conference on World Wide Web* (pp. 1–12). ACM Press.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. Technical report, Computer Science Department, Stanford University.

Sutton, R. S., & Barto, A. G. (1998). *Learning: An introduction*. Cambridge, MA: MIT Press.

Wang, Z. (2004). Improved link-based algorithms for ranking web pages. In *WAIM 2004* (pp. 291–302).

Watkins, D. S. (2002). *Fundamentals of matrix computations*. John Wiley & Sons.

Please cite this article in press as: Zareh Bidoki, A. M., & Yazdani, N. , DistanceRank: An intelligent ranking algorithm ..., *Information Processing and Management* (2007), doi:10.1016/j.ipm.2007.06.004