Applied Soft Computing 13 (2013) 1686–1692

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Applying reinforcement learning for web pages ranking algorithms

Vali Derhami*, Elahe Khodadadian, Mohammad Ghasemzadeh, Ali Mohammad Zareh Bidoki

Electrical and Computer Engineering Department, Yazd University, Yazd, Iran

ARTICLE INFO

Article history:
Received 28 June 2012
Received in revised form
16 December 2012
Accepted 30 December 2012
Available online 11 January 2013

Keywords:
Ranking
Search engine
Reinforcement Learning
Artificial intelligence
Value function
Agent

ABSTRACT

Ranking web pages for presenting the most relevant web pages to user's queries is one of the main issues in any search engine. In this paper, two new ranking algorithms are offered, using Reinforcement Learning (RL) concepts. RL is a powerful technique of modern artificial intelligence that tunes agent's parameters, interactively. In the first step, with formulation of ranking as an RL problem, a new connectivity-based ranking algorithm, called RL_Rank, is proposed. In RL_Rank, agent is considered as a surfer who travels between web pages by clicking randomly on a link in the current page. Each web page is considered as a state and value function of state is used to determine the score of that state (page). Reward is corresponded to number of out links from the current page. Rank scores in RL_Rank are computed in a recursive way. Convergence of these scores is proved. In the next step, we introduce a new hybrid approach using combination of BM25 as a content-based algorithm and RL_Rank. Both proposed algorithms are evaluated by well known benchmark datasets and analyzed according to concerning criteria. Experimental results show using RL concepts leads significant improvements in raking algorithms.

1. Introduction

Nowadays World Wide Web (WWW) is considered to be the best source of information. Its importance mainly is due to easy access, low-cost and being responsive to users' requests in the shortest time [1]. Search engines are the predominant tools for finding and getting access to the contents on the web. Whenever users seek information, enter their query in search engine. The search engine searches through web pages and return a list of relevant ones.

Generally, search engines involve three processing stages. The first stage is called crawling. A crawler visits a web page, and follows all the links provided in that page. This operation leads to constructing a web graph (a web graph consists of nodes and edges, where nodes stand for web pages and edges show the links which are available from each page to other pages). After collecting web pages, content of each page is analyzed to determine how it should be indexed (e.g. words are extracted from the titles, headings, or special fields). Indexing allows information to be found as quickly as possible. Ranking is the final stage. In this stage millions of web pages were recorded in the previous stage are sifted to find matching cases for a specified query and sorting them based on the users' requests or preferences. Due to the huge size of the web, it is very common that a large number of relevant results are returned for a given query. Moreover, studies have shown that users do not have the time and the patience to go through all of them to find the ones which they are interested in. They often consider the top 10 or 20 results [2]. Therefore, an efficient ranking algorithm is required. This algorithm enables search engines to present the best related pages to users in response to their queries.

In this paper, we propose a new algorithm for ranking web pages based on web graph. The objective is determining the score of each web page based on paths which can be reached to that web page from other web pages as well as the out-degree (number of out links) of pages in the traverse paths. Consider a random surfer who transfers between pages randomly. After visiting a web page; she selects next page by clicking randomly on one of the links in that page. This process can be considered as a Markov Decision Process (MDP). Therefore, we formulate it as a Reinforcement Learning (RL) [3] problem where the objective is "policy evaluation". Elements of RL in this problem are defined as follows: 1 – *states*: web pages, 2 – *Actions*: out links on each page, 3 – *Policy*: agent (surfer) selects the next page by clicking randomly on one of the out links in current page. 4 – *Reward*: inverse of the out-degree of the source page. 5 – *Value function*: value function of each state (page) is the total amount of rewards that surfer can expect to accumulate during traveling through pages to reach that page. The proposed approach is called RL_Rank. Based on the above definitions, value function of each page is considered as the score of the page.

Since RL_Rank is a connectivity-based algorithm; in the next step, we combine it with BM25 which is a content-based algorithm and propose a hybrid ranking algorithm.

* Corresponding author.
E-mail addresses: vderhami@yazduni.ac.ir (V. Derhami), khodadadi@stu.yazduni.ac.ir (E. Khodadadian), m.ghasemzadeh@yazduni.ac.ir (M. Ghasemzadeh), alizareh@yazduni.ac.ir (A.M. Zareh Bidoki).

1568-4946/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.12.023

The remainder of this paper is organized as follow: Ranking algorithms are discussed in Section 2. In Section 3 we illustrate our proposed ranking algorithms. Experimental analysis and their results are presented in Section 4. Finally, in Section 5 we summarize our main contributions and discuss some possible further improvements on our proposed method.

2. Ranking algorithms

Web pages ranking algorithms divided into two categories namely content-based and connectivity-based algorithms.

Content-based algorithms usually work based on matching words in documents. In other words, for each query the documents with the most similar content to the query will be selected. Vector space [4], TF-IDF [5] and BM25 [6] are examples of these algorithms. These algorithms are suitable for well structured environments such as digital libraries, rather than the web pages which usually include large number of unstructured contents. Connectivity-based algorithms use links between web pages. Links carry information which can be used to evaluate the importance of pages and the relevancy of pages to the user query. These algorithms are divided into two major classes "query-independent" and "query-dependent". Instances of query-independent algorithms are PageRank [7], HostRank [8] and DistanceRank [9]. These algorithms use the entire web graph and compute the score of web pages offline, whereas query-dependent algorithms such as HITS [10] involve the construction of a query-specific graph, in other words these algorithms are online.

In the first step of this research, we concentrate on connectivity-based algorithms which are offline. Among these algorithms, PageRank as a well known and mostly used algorithm is at the center of our attention. In the second step, we present a hybrid approach with combination of a connectivity algorithm and a content algorithm. BM25 is used as a content-based algorithm in the proposed hybrid approach.

2.1. PageRank algorithm

PageRank is a popular ranking algorithm used by Google search engine. PageRank models the users' browsing behaviors as a random surfer model. In this model, a user surfs the web by randomly clicking links on the visited pages and sometimes jumps to another page at random. In this algorithm, fraction of time the surfer spends on a page is defined as the score of that page [11]. PageRank measures the importance of web pages as follows: the score of a page such as i, based on PageRank method, can be approximated by the following recursive formula [7]:

$$R(i) = \sum_{j \in B(i)} \frac{R(j)}{O(j)} \tag{1}$$

where $R(i)$ and $R(j)$ are rank scores of pages i and j, respectively. $O(j)$ is the number of out links in page j which is called out-degree of page j. $B(i)$ is the set of pages that point to page i.

In fact, PageRank supposes that a link from page p_1 to p_2 indicates that the author of p_1 is interested in page p_2. If a page has many links in other pages, it can be concluded that many people are interested in that page and the page should be considered an important one. PageRank takes the backlinks (incoming links to a web page) into account and propagates the ranking through links: a page has a high rank if the sum of the ranks of its backlinks is high. Fig. 1 is an example to show how score of a page is computed in each step with PageRank algorithm that the score of page p in each step is updated with:

$$R(p) = \frac{R(A)}{4} + \frac{R(B)}{3} \tag{2}$$

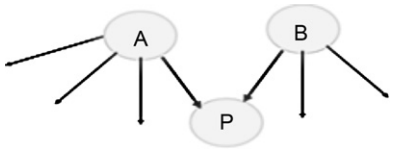

Fig. 1. A portion of a web graph: pages A and B point to page p and the out-degree for A and B is 4 and 3, respectively.

The PageRank formula in Eq. (1) is not suitable for disconnected web graphs, because it will not converge. Hence, the score of a page such as i ($R(i)$) can be approximated by the following recursive formula [7]:

$$R(i) = \left(d \times \sum_{j \in B(i)} R(j)/O(j)) + (1 - d/n \right) \tag{3}$$

where $R(i)$ and $R(j)$ show score of pages i and j, respectively. d is the damping factor, n is the total number of pages and $B(i)$ and $O(j)$ are the set of pages pointed to page i and the out-degree of the page j, respectively.

The presence of the damping factor is necessary, because the web graph is not a strongly connected graph (SCG), so damping factor used to guarantee the convergence of PageRank and remove the effects of sink pages (pages with no out-link).

2.2. BM25 algorithm

The BM25 formula was proposed by Robertson et al. [6]. In BM25, documents are ordered by decreasing probability of their relevance to the query. The formulation takes into account the number of times a query term appears in a document (tf), the proportion of other documents which contain the query term (idf), and the relative length of the document. A score for each document is calculated by summing the match weights for each query term [12]. Given a query Q, containing keywords $q_1, \ldots, q_n$; BM25 score of a document D is [6]:

$$S(D, Q) = \sum_{i=1}^{n} \frac{IDF(q_i)(f(q_i, D)(k_1 + 1))}{f(q_i, D) + k_1(1 - b + b(\left|D\right| / avgdl))} \tag{4}$$

where $f(q_i,D)$ is frequency of term q_i in the document D, $|D|$ is the length of the document in words, and $avgdl$ is the average of document's length in the text collection from which documents are drawn. k_1 and b are free parameters, usually chosen, in absence of an advanced optimization, as $k_1 \in [1.2, 2.0]$ and b is 0.75. $IDF(q_i)$ is the IDF (inverse document frequency) weight of the query term q_i. It is usually computed as

$$IDF(q_i) = \log \frac{(N - n(q_i) + 0.5)}{(n(q_i) + 0.5)} \tag{5}$$

where N is the total number of documents in the collection and $n(q_i)$ is the number of documents containing q_i.

3. The proposed algorithm

RL_Rank algorithm inspired from reinforcement learning concepts. So in this section, we first review reinforcement learning concepts. Afterwards, two proposed algorithms: RL_Rank and hybrid algorithm are introduced.

3.1. Reinforcement learning (RL)

Reinforcement learning, one of the machine learning techniques, learns by interactive in dynamic environment. Also, it is

a powerful tool in determining effective states in states space. In an RL problem, the learner is called the agent who learns through its interaction with the environment and it acquires knowledge through reward or punishments of an action undertaken [3].

In an agent-based system with reinforcement learning, at each time step t, the agent is involved with a state called current state and selects an action from a set of possible actions. The policy, denoted by $\pi(s, a)$, is the probability of selecting action a when agent is concerned with states. Afterwards, the environment goes to next state (s_{t+1}), and the agent receives reinforcement signal $r_{t+1} = (r(s_t, a_t))$ that is called a reward [3]. Reinforcement signal is a scalar signal and it indicates the intrinsic desirability of the action. Then, agent updates value function of the state. The state-value function under policy π is expected value of the sum of received discounted rewards, defined as follows [3]:

$$V^{\pi} = E_{\pi}\left\{\sum_{k=0}^{\infty}\gamma^{k}\, r_{t+k+1}\,\middle|S_t = s\right\} \quad 0 \le \gamma \le 1 \tag{6}$$

where k is time step and γ is a discount factor that determines the present value of the future rewards that can be achieved over time. $E_{\pi}\{0\}$ denotes the expected value and r_{t+k+1} is a reward that agent receives during transition between state.

3.2. RL_Rank algorithm

In our algorithm, we use link structure of web pages and define ranking in form of reinforcement learning problem. The proposed approach is named RL_Rank. In RL_Rank algorithm, an agent is considered as a surfer and each web page as a state. In each page (state), the surfer (agent) clicks on one of the available links in that page with a uniform probability, and goes to the next state. Therefore, an agent's action is to click on one of the links randomly with a uniform probability. In other words, when surfer selects next page by clicking randomly on one of the links in the current page, the policy π is equal to $1/O(current\ \ state)$, where $O(current\ \ state)$ is the out-degree of the current page. The reward is given when a transition occurs from a current state (j) to another state (i) defined by

$$r_{ji} = \frac{1}{O(j)} \tag{7}$$

where $O(j)$ is the out-degree of page j. Hence, page with less out-degree gives more reward to its children.

We define the score of page i to be the expected value of sum of discounted rewards that agent accumulates during traveling through pages to reach page i. Then agent adds the received reward r_{ji} to the discounted accumulated rewards. Therefore, score of page i is probability of reaching it from other pages multiplied by sum of the transition reward and discounted accumulated rewards. The score of page is defined as follows:

$$R_{t+1}(i) = \sum_{j \in B(i)} \Big((prob(j)/O(j)) \times (r_{ji} + \gamma R_t(j))\Big) \tag{8}$$

where $R_{t+1}(i)$ is rank of page i in time $t+1$ and $R_t(j)$ shows the rank page j in time t, $B(i)$ is the set of pages that point to page i, $prob(j)$ is the presence probability of the agent at page j. $O(j)$is the out-degree of page j and r_{ji} the reward for transition from page j to i defined by Eq. (7). Therefore, the rank of page p depends on the out-degree and rank of the pages pointing to page p.

The value of $prob(j)/O(j)$ is the probability of reaching page i from page j. It is equal to presence probability of the agent at state j multiplied by selection probability of page i when agent is in state j. Since the agent selects one of the links by uniform probability distribution, the selection probability of page i from j is equal to one divided by out-degree of page j. $R(j)$ is the rank of page j that presents accumulated discounted rewards the agent has received until getting to page j. Therefore, rank of page i based on Eq. (8) depends on the out-degree and rank of the pages pointing to i. Using the policy evaluation idea [3] in the RL algorithm, we propose a practical approach to estimate the rank of each page. As Eq. (8) shows RL_Rank is computed recursively like PageRank. The following pseudo code illustrates our RL_Rank procedure. Finally, we will have the RL_Rank vector and pages sorted in the descent order. With respect to the pseudo code, it is obvious that the time complexity of RL_Rank is linear.

Algorithm: RL_Rank
//V: all web pages
//prob: presence probability of the agent at page j
//R: RL_Rank vector
//ε: A small positive number
Initialize R, prob vectors
$\delta \leftarrow 0$
while($\delta > \varepsilon$)
  For every page i ∈ V
    $prob_{new}(i) = \left(d \times \sum_{j \in B(i)} prob(j)/O(j)) + (1 - d/n\right)$
  End for
  $\delta \leftarrow \left|\left|prob_{new} - prob\right|\right|$
  $prob \leftarrow prob_{new}$
End while
$\delta \leftarrow 0$
while($\delta > \varepsilon$)
  For every page p ∈ V
    $r_{ji} = 1/O(j)$
    $R_{new}(i) = \sum_{j \in B(i)} \Big((prob(j)/O(j)) \times (r_{ji} + \gamma R(j))\Big)$
  End for
$\delta \leftarrow ||R_{new} - R||$
$R \leftarrow R_{new}$
End while

3.2.1. RL_Rank convergence

In this section, we prove convergence of RL_Rank algorithm.

Lemma 1. In RL_Rank algorithm (Eq. (8)), $R(i)$ converges.

Proof. The rank scores in RL_Rank are computed recursive by Eq. (8). It has to be noticed that rank score of pages with zero in-degree (in-degree of a page is equal the number of links from other pages to the page) are not changed in iterations and their final amounts are equal their initial values. However, some pages with zero in-degree have out-links to other pages; Hence, rank score of other pages are affected by the rank score of these pages. Therefore, Eq. (8) is rewritten as follows:

$$R_{t+1}(i) = \sum_{j \in B(i)} \Big((prob(j)/O(j)) \times (r_{ji} + \gamma R_t(j))\Big) + \sum_{k \in B'(i)} \Big((prob(k)/O(k)) \times (r_{ki} + \gamma R(k))\Big) \tag{9}$$

where $B(i)$ is the set of pages with non-zero in-degree that point to page i and $B'(i)$ is the set of pages with zero in-degree that point to page i. Amount of the second term of the left side in Eq. (9) is constant. In other words, it is not updated during iterations. This amount for i-th page is denoted as $k(i)$:

$$R_{t+1}(i) = \sum_{j \in B(i)} \Big((prob(j)/O(j)) \times (r_{ji} + \gamma R_t(j))\Big) + k(i) \tag{10}$$

Here, we define the matrix P and the vectors Z, R, K as follows: P is a $n' \times n'$ matrix that each element is defined as

$$p(ij) = \begin{cases} (prob(j)/O(j)) & j \in B(i) \\ 0 \quad \text{otherwise} \end{cases} \tag{11}$$

where $prob(j)$ is the presence probability of agent at page j. This probability is constant during RL_Rank computations, since it is independently computed sooner. $O(j)$ is out-degree of page j. n' is the total number of pages with non zero in-degree.

Z is a $n' \times 1$ vector that its element is as

$$z(j) = \frac{1}{O(j)} \tag{12}$$

R is vector containing the score of pages.

K is $n' \times 1$ vector that that i-th its elements is $k(i)$.

γ is the discount factor that $0 < \gamma < 1$.

Based on the above definitions, we can rewrite Eq. (10) as a matrix form:

$$R = \gamma PR + PZ + K \tag{13}$$

As seen γP is the coefficient of vector R, and PZ and K are to constant vector. With respect to Eq. (11) all elements on main diagonal of matrix P is 0 as well another elements are less than 1. Therefore, $\|\gamma P\|_\infty < 1$ according to the convergence theorem of iterative methods [13], It can be concluded $R(i)$ in Eq. (8) converges.

3.3. Hybrid algorithm

Generally, search engines use a combination of connectivity-based and content-based algorithms. Therefore, in this section, a hybrid approach is introduced in order to show the impact of RL_Rank on performance of a hybrid algorithm. In the proposed hybrid algorithm, RL_Rank as a connectivity-based algorithm is combined with BM25 as a content-based algorithm. Content-based algorithms usually act based on matching words in documents. In other words, the documents with the more similar content to each query will be selected as the more relevant ones. It has to be noticed that queries are generally short (2.4 terms in average [14]) and vocabulary is huge while in classical Information Retrieval (IR) usually the number of documents is not huge and queries are long. These differences pose new challenges to IR. In addition, contents available on the web are often inconsistent and include a lot of misinformation. Therefore, application of content-based (classical IR) algorithms to the web content may result in problems such as low precision [2]. To remedy these issues, connectivity-based algorithms have been proposed that use links between web pages. Previous studies indicate that algorithms using hyperlinks for ranking yield satisfactory results [15]. The main strength of these algorithms comes from using the votes of other pages to rank current pages. In other words, links carry information which can be used to evaluate the importance of pages and the relevancy of them to the user's query. Although these algorithms are appropriate in some situations, on average their precision is low compared to content-based algorithms [16]. Also, web spam is a challenge to information retrieval (IR). Web spam is a phenomenon where web pages are manipulated for the purpose of obtaining some kind of benefits by illicitly gaining web traffic. Self and Mutual promotion are two basic forms of web spam. Self promotion tries to create a web page that gains high relevance for a search engine, mainly based on its content. Mutual promotion is based on the cooperation of various sites in order to benefit each other. Hence, content-based algorithms consist of self web spam, while connectivity algorithms suffer from Mutual web spam [17].

It seems that we can obtain better performance and overcome the mentioned drawbacks by using a hybrid approach. It is obvious that the properties of the page play an important role in the quality of the page. Therefore, weighted combination of different ranking algorithms (such as PageRank, BM25, TF_IDF, HITS, etc.) as properties of page can be effective [18]. In hybrid algorithm, weights for all participating algorithms are assigned. Therefore, by using this operator all of the algorithms will have chance to affect the final aggregation value. The score of the page by hybrid algorithm (weighted combination of some algorithms) is calculated as follows:

$$S(i) = \sum_{m=1}^{n} w_m S_m(i) \tag{14}$$

where $S(i)$ and $S_m(i)$ are scores of page i by hybrid algorithm and m-th algorithm, respectively. w_m is weight of m-th algorithm. m varies from 1 to n (n depicts the number of participating ranking algorithms).

Here, we combine RL_Rank and BM25 algorithms. This hybrid approach is named CRLBM. Also, for demonstration superiority of RL_Rank, we combine PageRank and BM25 algorithms and it called CPRBM. A fixed weight to any of them is assigned by Borda method [19,20]. Moreover, the score obtained from each algorithm is normalized for using Eq. (14).

4. Evaluation and experimental results

To assess the proposed methods, they are evaluated experimentally on well known benchmark based on standard criteria.

4.1. Benchmark datasets

We conducted some experiments on LETOR [21] and dotIR [22] benchmark datasets which have recently provided for research on information retrieval (IR) and they are publicly available.

1. LETOR data collection: It released by Microsoft Research Asia, derived from exiting English test collections. It is constructed based on the existing datasets and query sets, namely, the "Gov" and "OHSUMED" corpora. We used 50 TREC 2003 queries on the "Gov" corpus. Generally, the LETOR package contains 50 queries, relevance judgments, the extracted features and some tools to compare the accuracy of the newly proposed ranking algorithms. In TREC 2003, there are 1,053,110 web pages and 50 queries [21].
2. dotIR data collection: It is a Persian benchmark on Iran web which is recently released by Iran Telecommunication Research Center (ITRC) [23]. The dataset consists of the contents of web pages, queries, and human judgments on the retrieved documents with respect to the queries. Also, there are 997,462 web pages and 50 queries [22].

4.2. Evaluation measures

In order to assess the proposed algorithms, we use two related and well known benchmark datasets and use three common evaluation measures which are widely used in IR, namely *Precision at n* (*P@n*) [24], *Mean Average Precision* (*MAP*) [24] and *Normalized Discount Cumulative Gain* (*NDCG*) [25]. Their definitions are briefed as follows:

a) *Precision at n* (*P@n*): This criterion indicates the ratio of top relevant documents to total number of documents (n) in presented results. In fact, it indicates system accuracy [24]:

$$P@n = \#\ of\ relevant\ in\ top\ n\ results/n \tag{15}$$

b) *Mean average precision (MAP)*: Average Precision (AP) corresponds to the average of P@n values for all relevant documents of a given query and is computed by following equation [24]:

$$AP = \sum_{i=1}^{n} (P@i.rel(i))\ /\#total\ relevant\ docs\ for\ one\ query \tag{16}$$

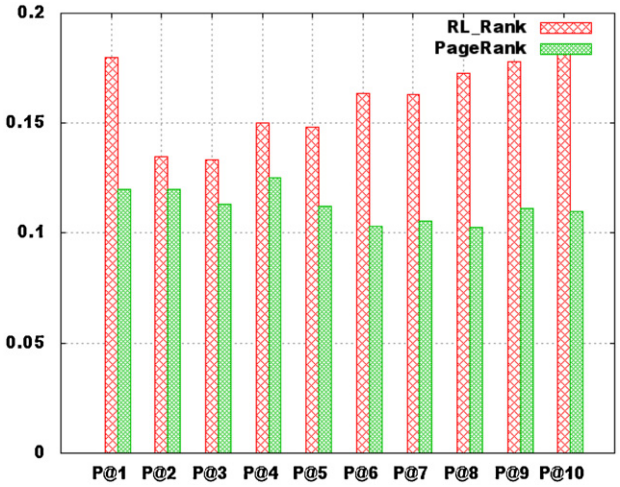

Fig. 2. Comparison of RL_Rank with PageRank in the P@n measure on dotIR benchmark.

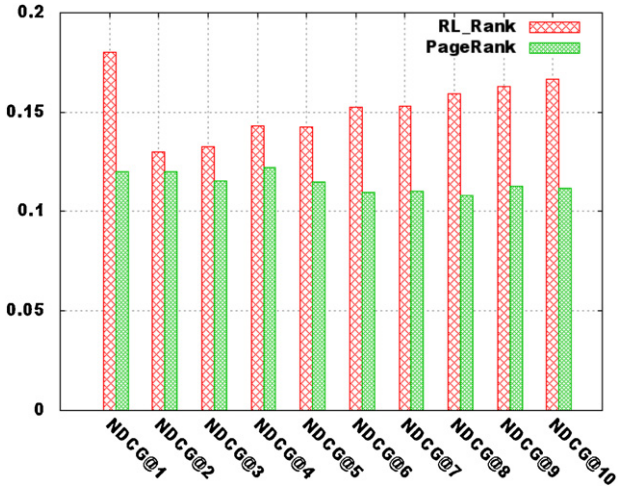

Fig. 3. Comparison of RL_Rank with PageRank in the NDCG@n measure on dotIR benchmark.

where n is the number of retrieved documents, and $rel(i)$ is a binary function on the relevance of the i-th document. If i-th document is a relevant page, $rel(i)$ will be equal to 1, otherwise it is 0. Finally, MAP is obtained by computing the average of AP values over the set of queries.

c) *Normalized Discount Cumulative Gain* (*NDCG*): The above mentioned criteria (P@n and MAP) can only provide binary judgment: "relevant" or "irrelevant". Therefore a new criterion has been proposed, which is known as Normalized Discount Cumulative Gain (NDCG) [25]. It can perform multiple levels of relevance judgments by considering the following two observations:
- A document with lower ranking position is less valuable for users.
- Highly relevant documents are more valuable than other relevant documents.

According to the above points, the NDCG value of a ranking list at position n is computed as

$$NDCG@n = \sum_{i=1}^{n} 2^{r_j} / \log(1+i) \tag{17}$$

where r_j is the rating of the j-th document in the ranking list. For comparisons, this paper reports P@1, ..., P@10, NDCG@1, ..., NDCG@10, as well as MAP.

4.3. Results

The first experiments compare RL_Rank with PageRank as a well-known connectivity-based ranking algorithm. In the experiments, the factor γ in RL_Rank was set to 0.9 and the damping factor in PageRank was set to 0.85. The results of evaluation on dotIR benchmark dataset are shown in Figs. 2–4.

Figs. 2 and 3 show the obtained P@n and NDCG@n, respectively. As shown, the obtained values for RL_Rank are higher than those for PageRank, especially P@1 and NDCG@1. Fig. 4 shows that RL_Rank obtains improvement about 26% over the PageRank in terms of MAP measure.

Graphical analyses of Results on TREC 2003 benchmark dataset are depicted in Figs. 5–7 in terms of P@n, NDCG@n and MAP measures, respectively. The values obtained for RL_Rank are higher than those for PageRank except for P@2 and NDCG@2. Fig. 7 shows that RL_Rank exceed PageRank by 7% in performance.

A close look at the results indicates that RL_Rank is a suitable algorithm for ranking of the web pages. The results signify that

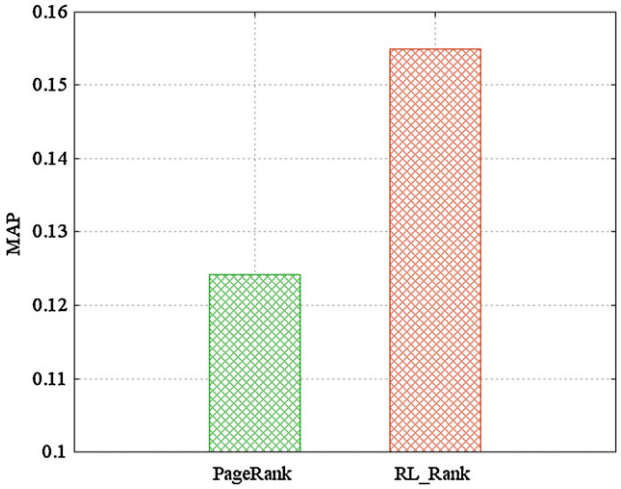

Fig. 4. Comparison of RL_Rank with PageRank in the MAP measure on dotIR benchmark.

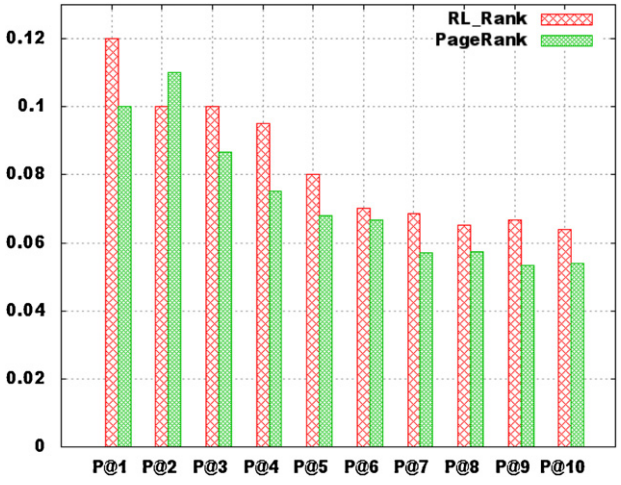

Fig. 5. Comparison of RL_Rank with PageRank in the P@n measure on TREC 2003 benchmark.

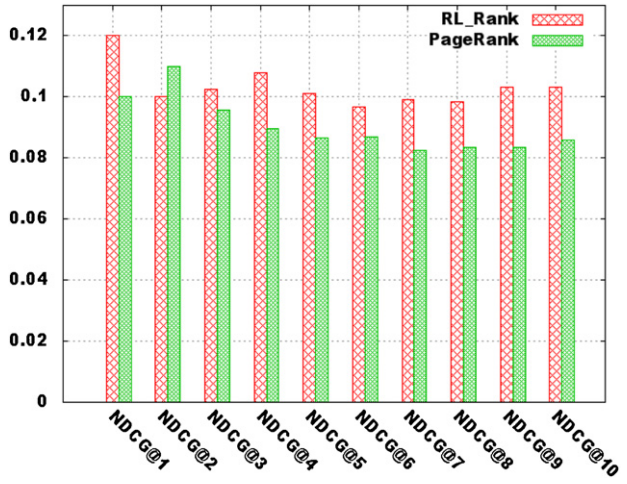

Fig. 6. Comparison of RL_Rank with PageRank in the NDCG@n measure on TREC 2003 benchmark.

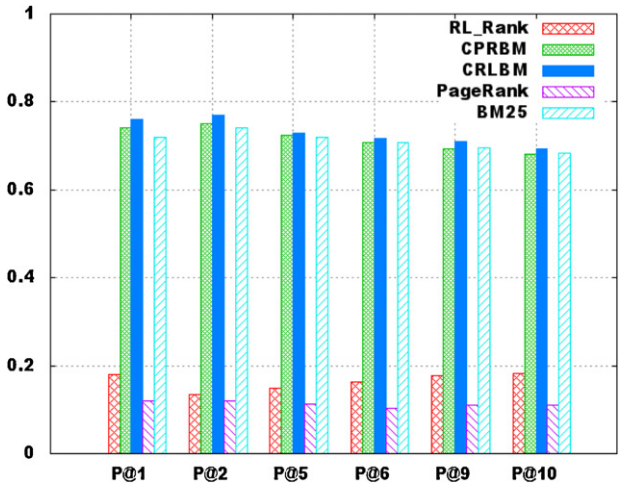

Fig. 8. Evaluation of hybrid algorithms in the P@n measure ondotIR benchmark.

RL_Rank algorithm makes larger improvements on dotIR dataset in compared to TREC 2003 dataset. It should be noticed that RL_Rank and PageRank are two connectivity-based ranking algorithms and they are influenced by connectivity features of web pages; especially out-degree of pages. Studying on statistical characteristics of two graphs (dotIR and TREC 2003) shows that these two graphs of datasets have great different, for example mean of out-degree in dotIR is about 40 whereas this feature in TREC 2003 is about 10. This difference reflects dotIR graph is denser than TREC 2003 graph. Therefore, it can be concluded that RL_Rank has higher performance in dense web graphs.

The second part of experiments is about evaluation of hybrid algorithm. In our experiments, RL_Rank and PageRank weight in Eq. (14) were set to 0.15479 and 0.143, respectively. BM25 weights in CRLBM and CPRBM were set to 0.84521 and 0.857, respectively.

Figs. 8 and 9 summarized results of evaluation both hybrid algorithms on dotIR benchmark dataset in terms of P@n and NDCG@n measures. This results show that both hybrid algorithms are much better than all other basis ranking algorithms. Also, the proposed hybrid algorithm (combination of BM25 with RL_Rank) outperforms CPRBM approach. Therefore, we can conclude RL_Rank algorithm generally is superior to PageRank.

4.4. Demonstration of RL_Rank convergence

The convergence speed of our algorithm is fast with a little number of iterations. 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 RL_Rank vectors from successive iterations. A more useful measure is the order of pages produced by RL_Rank vector. In the rank ordering, we measure similarity between two ordered lists of RL_Rank [26]. 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 Bas $(A \cap B)/(A \cup B)$. To evaluate how closely two ranking methods match on identifying top pages, we successively computed similarity among top n pages in each ordering.

Fig. 10 shows similarity for 20,000 till 570,000 top pages. As the figure shows, the ordering obtained by only 10 iterations agrees closely with ordering of 15 iterations for 1,000,000 million pages.

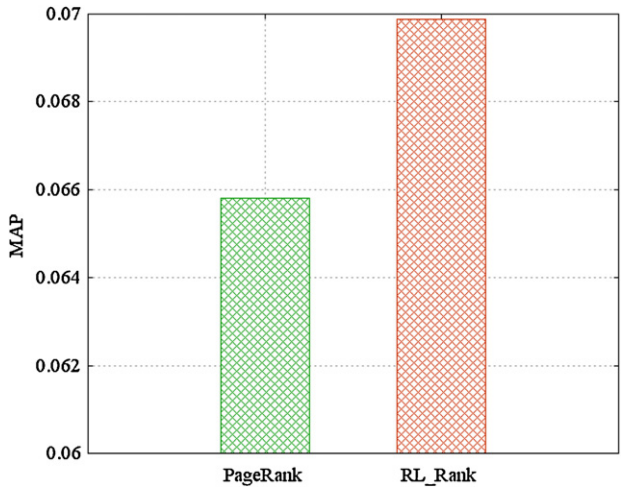

Fig. 7. Comparison of RL_Rank with PageRank in the MAP measure on TREC 2003 benchmark.

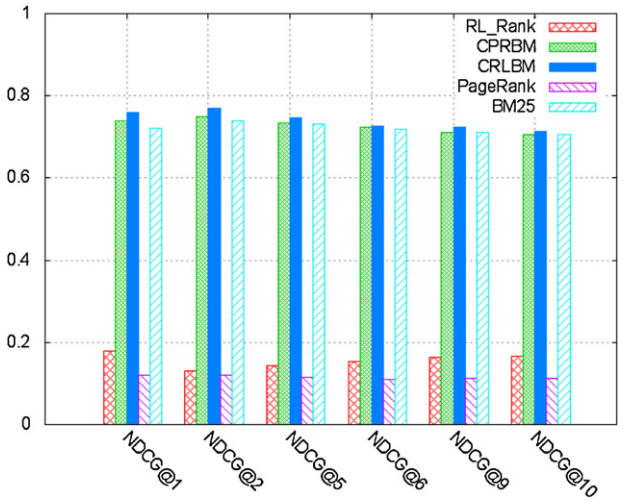

Fig. 9. Evaluation of hybrid algorithms in the NDCG@n measure on dotIR benchmark.

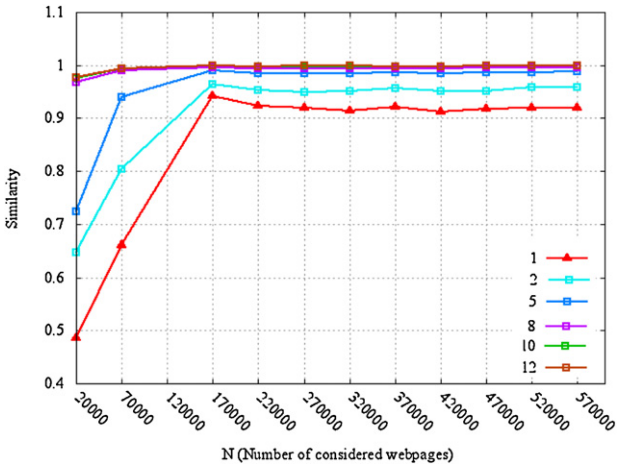

Fig. 10. The similarity between 1, 2, 5, 8, 10 and 12 with 15 iterations.

In other words, a few numbers of iterations is sufficient to find satisfactory results.

5. Conclusion

In this paper, using the reinforcement learning concepts, we first proposed RL_Rank algorithm which is a novel connectivity-based algorithm for ranking web pages. This algorithm considers rank determination of a page as an RL problem where the reward for transition from current page to the next page is proportional to the inverse of the out-degree of the current page. In fact, RL_Rank models the user who surfs the web by accumulating transition rewards to obtain rank of each page. The convergence of RL_Rank was proved in Lemma 1. Moreover, a hybrid algorithm with combination of RL_Rank and BM25 was offered. Experimental results showed that RL_Rank can achieve much better results than PageRank in standard criteria. The linear complexity of the RL_Rank signifies the scalability of this algorithm on large datasets. Therefore, RL_Rank can be used either as a connectivity-based ranking algorithm in search engines like Google or as a graph based problems like Word Sense Disambiguation [27]. Also we saw that RL_Rank behaves differently on different datasets (it makes larger improvements on dotIR dataset in comparison to TREC 2003 dataset). Hence, it can be concluded that RL_Rank has high performance in dense web graphs. Therefore, it can be strongly suggested for dense graphs for example Slovakia graph [28] (Slovakia web graph, the mean of out-degree = 50) or Italy graph [28] (Italy web graph, the mean of out-degree = 27.87). Experiment results for hybrid algorithms showed that these algorithms are better than the basic algorithms (BM25, PageRank, and RL_Rank) in quality of rankings and overcome drawbacks of content and connectivity-based algorithms. As future works, we are going to define another type of reward signals in RL_Rank, and adjust weights in hybrid algorithm by learning methods.

Acknowledgments

This research was supported by Iran National Science Foundation (INSF). The authors would like to thank INSF for its support.

References

[1] A.M. Zareh Bidoki, Efficient ranking and crawling in the web, Ph.D. Dissertation, Electrical and Computer Engineering Faculty, University of Tehran, Iran, 2008.
[2] A.M. Zareh Bidoki, P. Ghodsnia, N. Yazdani, F. Oroumchian, A3crank. An adaptive ranking method based on connectivity, content and click-through data, Information Processing and Management 46 (2010) 159–169.
[3] R.S. Sutton, A.G. Barto, Reinforcement Learning. An Introduction, MIT Press, Cambridge, MA, 1998.
[4] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Journal of Information Processing and Management 24 (1998) 513–523.
[5] G. Salton, The sMART Retrieval System – Experiments in Automatic Document Processing, Prentice-Hall, USA, 1971.
[6] S.E. Robertson, S. Walker, Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval, in: Proceedings of SIGIR Conference on Research and Development in Information Retrieval, 1994, pp. 232–241.
[7] L. Page, S. Brin, R. Motavni, T. Winogard, The PageRank citation algorithm: bringing order to the web, Technical Report, Stanford Digital Library Technologies Project, 1998.
[8] A.M. Zareh Bidoki, N. Yazdani, DistanceRank: an intelligent ranking algorithm for web pages, Journal of Information Processing and Management 44 (2008) 877–892.
[9] G.R. Xue, Q. Yang, H.J. Zeng, Y. Yu, Z. Chen, Exploiting the hierarchical structure for link analysis, in: Proceedings of SIGIR Conference on Research and Development in Information Retrieval, 2005, pp. 186–193.
[10] J.M. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM 46 (1999) 668–677.
[11] P. Boldi, M. Santini, S. Vigna, PageRank as a function of the damping factor, in: Proceedings of World Wide Web Conference, 2005, pp. 557–566.
[12] T.G. Upstill, Document ranking using web evidence, Ph.D. Dissertation, The Australian National University, 2005.
[13] S. Berger Henengouwen, Engineering Numerical Analysis, Cybered Incorporated, 1998.
[14] Y. Zhang, A. Moffat, Some observations on user search behavior, in: 11th Australasian Document Computing Symposium, 2006, pp. 1–8.
[15] M. Henzinger, Hyperlink analysis for the web, IEEE Internet Computing 5 (2001) 45–50.
[16] T.Y. Liu, J. Xu, T. Qin, W. Xiong, H. Li, LETOR. Benchmarking learning to rank for information retrieval, in: Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval, Amsterdam, Netherlands, 2007.
[17] F. Javier Ortega, J.A. Troyano, F.L. Cruz, C.G. Vallejo, Polarityspam propagating content -based information through a web-graph to detect web-spam, International Journal of Innovative Computing Information and Control 8 (2012) 1–11.
[18] A.M. Zareh Bidoki, M. Azadnia, N. Yazdani, A.M. Keihanipour, Adaptive combinational web ranking algorithm using coarse-grained and fine-grained, in: 13th Conference on Iran Computer Society, 2009.
[19] J.C. Borda, M'emoire sur les 'elections au scrutin, InHistoire de l'Academie Royale des Sciences, 1781.
[20] D. Coppersmith, L. Fleischer, A. Rudra, Ordering by weighted number of wins gives a good ranking for weighted tournaments, in: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, 2006, pp. 776–782.
[21] T. Qin, T.Y. Liu, J. Xu, H. Li, LETOR. A benchmark collection for research on learning to rank for information retrieval, Journal of Information Retrieval 13 (2010) 346–374.
[22] E. Darrudi, H. BaradaranHashemi, A. Aleahmad, A. Habibian, A.M. ZarehBidoki, A. Shakery, M. Rahgozar, A standard web test collection for ir domain, Technical.Report, Iran Telecommunication Research Center, 2009.
[23] E. Darrudi, H. BaradaranHashemi, A. Aleahmad, A. Habibian, A.M. Zareh Bidoki, A. Shakery, M. Rahgozar, dotIR benchmark collection, http://ece.ut.ac.ir/dbrg/webir
[24] A. Moffat, J. Zobel, Rank-biased precision for measurement of retrieval, Journal of ACM Transactions on Information Systems 27 (2008) 1–27.
[25] M. Gordon, P. Pathak, Finding information on the world wide web: the retrieval effectiveness of search engines, Journal of Information Processing and Management 5 (1999) 141–218.
[26] T. Haveliwala, Efficient computation of pagerank, Technical Report, Database Group, Computer Science Department, Stanford University, 1999.
[27] R. Mihalcea, P. Tarau, E. Figa, PageRank on semantic networks, with application to word sense disambiguation, in: Proceedings of the 20th International Conference on Computation Linguistics, 2004.
[28] G. Melancon, Just how dense are dense graphs in real world? in: Proceedings of the 2006 AVI Workshop on Beyond Time and Errors: Novel Evaluation Methods for Information Visulization, 2006, pp. 1–7.