International Journal of Physical Sciences Vol. 7(13), pp. 2066 - 2075, 23 March, 2012
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.1708
ISSN 1992 - 1950 ©2012 Academic Journals

Full Length Research Paper

A new method based on distributed learning automata for page ranking in web

Tayebeh Yarahmadi[1]*, Javad Akbari Torkestani[2] and Fatemeh Zandevakili[1]

[1]Department of Computer Engineering, Arak Branch, Islamic Azad University, Arak, Iran.
[2]Young Researchers Club, Arak Branch, Islamic Azad University, Arak, Iran.

Accepted 23 January, 2012

With the growing information in the web, ranking algorithms is very important in searching information. Currently, there are two categories of the ranking algorithm based on content and connectivity. Ranking algorithms which are based on content have low accuracy and recall and also contain the rank spamming problem. Ranking algorithms which are based on connectivity contain the rich get richer problem too. Therefore, in this paper, ranking algorithm based on distributed learning automata was presented, which use pages content's information, hyperlinks between pages and web usage data to present better results. In this paper, at first, two algorithms for determining the structure of web documents based on DLA was proposed, and then was used for ranking. The obtained results of the simulation proposed algorithm was evaluated with RankCorrelation and P@n measures.

Key words: Ranking algorithms, PageRank, structures of web documents, distributed learning automata.

INTRODUCTION

Finding web pages with high quality is one of the important purposes in search engines. The quality of pages is defined based on request and preferences of user. Usually, there are millions of relative pages with each query. Nevertheless, users specially consider only 10 to 20 of those pages. However, ranking is to arrange pages based on their quality. Ranking algorithms divide in two categories (Zareh Bidoki et al., 2010; Ghodsnia et al., 2008):

1. Ranking algorithms based on content (traditional IR methods).
2. Ranking algorithms based on connection (today methods).

In retrieval of traditional information (Baeza and Ribeiro, 1999), system tries to find pages that are corresponding to the user query, and algorithms of this category are based on matching query words with content of web documents. There are four main models for IR (Liu, 2007), Boolean model, vector space model, language model and probabilistic model. Being or not being of word in document is considered in Boolean model (Liu, 2007). In TF-IDF vector- space model, which is the best and the most useable model, documents and queries are shown like a weigh vector (Singhal, 2001; Salton and Buckley, 1988). Statistical language models are based on probability and its root is in statistical theory, some of its instance are developed in Zhai (2006) and Chen and Goodman (1998). LSI is another instance of algorithms based on content, which was presented by Deerwester to solve the problem of synonym and polysemy. These algorithms are suitable for structured environments such as digital libraries (Deerwester et al., 1990). In these environments, queries are long and well specified and vocabulary is small. But web consist of unstructured documents and short queries usually. Besides, since these methods are based on content and the contents of pages may be inconsistent and includes a lot of misinformation, so it will have results with low precision and recall, and there would be rank spamming problem (Henzinger et al., 2002). It means the retrieved pages by these algorithms may be related to query but they may not be popular or reliable. Ranking algorithms based on connection were proposed to solve these problems. These algorithms use information of hyperlinks between

*Corresponding author. E-mail: yarahmadi10@yahoo.com.

pages to ranking. Hyperlinks between the pages consist of information that can be used to calculate page rank. The most important algorithms proposed in this category are PageRank (Page et al., 1998; Brin and Page, 1998), Hits rank (Kleinberg, 1997) and distance rank (Zareh and Yazdani, 2008). PageRank algorithm acts independently of query and it is just based on link between pages. Optimization of this algorithm is presented in Ghodsnia et al. (2008), Bressan and Peserico (2010), Cicone and Serra (2010), Ma et al. (2008), Wu (2008), Kerchove et al. (2008) and Sun and Wei (2006).

Unlike PageRank, Hits algorithm is depended on query. This algorithm at first retrieves a small set of relative pages with query, then, it continues ranking regarding to linking between these pages and the pages out of the set repeatedly until converge to a fixed point. Ranking algorithms based on connection are compared in Sidiropoulos and Manolopoulos (2006). Although, these algorithms are appropriate in some situations, their accuracy is low in compared to algorithms based on content (Najork et al., 2007) and are encountered with some problems like "rich get richer" that causes popular pages to be on top of the results list and young high quality pages have low chance to be selected by user (Cho et al., 2005). In retrieval of web information, the user plays the most important role, and basic objective of ranking is to satisfy them. Therefore, we can use web usage data for ranking to get results that are more reliable. In continuation, we will review tasks which are a combination of three categories.

Keyhanipouet al. (2007) proposed a solution for aggregation of results in metasearch engines that uses click-through data and also the OWA operator for merging, that combine the results of nine search engines. Similarly, the researches developed by Shakery and Zha (2006) and Qin et al. (2005) showed that both content and connectivity have been used without considering click-through data. Therefore, its results are more desirable than algorithms which use content only. In the same way, the researches developed by Richardson and Domingos (2002) used a model called "intelligent surfer" instead of a simple random surfer, while in PageRank, a probabilistic combination of content and link has been proposed. Agichtein et al. (2006) by using click–through data features as user feedback in the ranking process both directly or indirectly, they found interesting results. They used 3000 query for evaluation and found a 31% increase in ranking quality. Golub and Van Loan (1983) proposed an algorithm based on OWA operator, that combines the results of some algorithms such as TF–IDF, PageRank and Bm25. An algorithm is proposed in Zareh Bidoki et al. (2010) that uses content, connectivity and click-through data triple called A3Crank; as well as reinforcement learning. Almpanidis et al. (2007) combined LSI and PageRank, and obtained an optimized ranking algorithm. Anari et al. (2008) and Saati and Meybodi (2006) used distributed learning automata for ranking.

Mojtahedi and Meybodi (2009), Saati and Meybodi (2005), Hashemi and Meybodi (2005) and Anari and Meybodi (2007) proposed some methods to identify structure of web document. In this paper, at first two algorithms are proposed to identify structure of web document, and then the proposed structures was used for ranking. In the first algorithm, to identify structure of web document, similarity between pages and existence of relevant link for determining amount of reward to pages was used. But in the second algorithm the time which user stay in a page to determine amount of reward was used. In the case of existence of cycles, the pages that are in cycles were penalized. None of the previous task showed that the existence of relevant link and the time which the user stayed on a page was used to determine reward. In proposed method in Saati and Meybodi (2005), Hashemi and Meybodi (2005) and Anari and Meybodi (2007), the connectivity between pages is not considered to identify structure of web document, and it was rewarded to user movement from one page to another one, even if it not being related to user query.

In the proposed method by Mojtahedi and Meybodi (2009), an action was chosen randomly. In a log file, if the movement corresponds to other movements, it will get a reward but if not, it will be penalized and another action will be chosen randomly. Moreover, the wrong movement of user is not considered and it is supposed that all of user movements from one page to another are right. Besides, sometimes randomly choice of an action causes all actions which are corresponding to a page to be chosen, and finally, the last action is done by user and this is time consuming. Also, in this method, the existence of cycle that shows the user is not satisfied and a comeback to start page of traversal is not considered. In ranking algorithm of PageRank, all of the input links are considered equally in calculating page rank and there is no difference between relevant and irrelevant links, and this is one of this algorithm’s weak points.

In proposed algorithm for ranking based on two proposed algorithm, a weight to every link is specified, and also relevant links are identified from irrelevant ones and they are not considered in calculating rank. The proposed algorithms have very little repetitions and high convergence.

LEARNING AUTOMATA

Learning automata are adaptive decision-making devices operating on unknown random environments. The automata approach to learning involves the determination of an optimal action from a set of allowable actions. An automaton can be regarded as an abstract object which has finite number of possible actions. In each decision process, the automata select an action from its finite set of actions. This action is applied to a random environment. The random environment evaluates the selected action and gives a grade to applied action of automata. The random response of environment (that is, grade of action) is used by automata in further action selection. By continuing this process, the automat learns to select an action with best grade. The learning algorithm uses by automata to determine the selection of next action from the response of the environment. An automaton acting on unknown random environment and improves its performance is some specified manner, is referred to as learning automata (LA). Learning automata can be classified into main categories: fixed structure learning automata and variable structure learning automata (Narendra and Thathachar, 1989). In the following, the variable structure learning automata which will be used in this paper is described.

Variable structure learning automata is represented by quintuple <α ,β , *p*, *T*(α ,β , *p*) > , where $\alpha \equiv \{\alpha_1,\alpha_2,\ldots,\alpha_r\}$, $\beta \equiv \{\beta_1,\beta_2,\ldots,\beta_r\}$, and $p \equiv \{p_1,p_2,\ldots,p_r\}$ are an action set with *r* actions, an environment response set, and the probability set *p* containing *r* probabilities, each being the probability of performing every action in the current internal automaton state, respectively. The function of *T* is the reinforcement algorithm, which modifies the action probability vector p with respect to the performed action and received response. If the response of the environment takes binary values, the P-model is used as the learning automata model, but if it takes a finite output set with more than two elements in the interval of [0, 1], such a model is referred to as Q-model, and when the output of the environment is a continuous variable in the interval [0, 1], it is referred to as S-model.

It is evident that the crucial factor affecting the performance of the variable structure learning automata is learning algorithm for updating the action probabilities. Various learning algorithms have been reported in the literature. Let α be the action chosen at step n as a sample realization from probability distribution p. The linear reward-inaction algorithm is one of the learning schemas and its recurrence equation for updating action probability vector p is defined as Equations 1 and 2.

Reward: $p_i(n+1) = p_i(n) + a[1 - p_i(n)]$

$$p_j(n+1) = (1-a)p_j(n) \quad \forall j, j \neq i \tag{1}$$

Penalty: $p_i(n+1) = (1-b).p_i(n)$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j, j \neq i \tag{2}$$

Where 0<α <1 is called step length and determines the amount of increases (decreases) of the action probabilities. The aforementioned learning automata have a fixed number of actions. In some applications, like our first proposed algorithm, we need that LA has a changing number of actions (Thathachar and Bhaskar, 1987). A LA with changing number of actions, at any time instance n can selects its action from a set of active actions *V* (*n*) and behaves like this. For selecting an action, the learning automata first computes the sum of its actions' probability *K*(*n*) and then the vector *p*ˆ (*n*) is computed according to Equation 3. The automaton selects one of its active actions randomly based on actions probabilities, that is, *p*ˆ (*n*). The automaton applies the selected action *i* α to the environment and gets the response. For desirable responses, the *p*ˆ (*n*) vector is updated based on Equation 5 and for undesirable actions is updated based on Equation 6. Finally, the automaton updates the actions' probability vector *p* (*n*) based on vector *p*ˆ (*n* +1) as shown in Equation 7.

$$k(n) = \sum_{\alpha_i \in V(n)} p_i(n) \tag{3}$$

$$\hat{p}_i(n) = prob[\alpha(n) = \alpha_i \mid V(n) \text{ is set of active actions}, \alpha_i \in V(n)] = \frac{p_i(n)}{K(n)} \tag{4}$$

$$\hat{p}_i(n+1) = \hat{p}_i(n) + a\left(1 - \hat{p}_i(n)\right) \quad \alpha(n) = \alpha_i$$

$$\hat{p}_j(n+1) = \hat{p}_j(n) - a.\hat{p}_j(n) \quad \alpha(n) = \alpha_j \ \forall j, j \neq i \tag{5}$$

$$\hat{p}_i(n+1) = (1-b).\hat{p}_i(n) \quad \alpha(n) = \alpha_i$$

$$\hat{p}_j(n+1) = \frac{b}{r-1} + (1-b)\hat{p}_j(n) \quad \alpha(n) = \alpha_j \ \forall j, j \neq i \tag{6}$$

$$p_j(n+1) = p_j(n+1).K(n) \quad \text{for all } j, \alpha_j \in V(n) \tag{7}$$

$$p_j(n+1) = p_j(n) \quad \text{for all } j, \alpha_j \notin V(n)$$

Distributed learning automata

A distributed learning automaton (DLA) is a network of LA which collectively cooperates to solve a particular problem. The number of actions for a particular LA in DLA is equal to the number of LA's that are connected to this LA. Selection of an action by a LA in the network activates one LA corresponding to the action. Formally, a distributed learning automata can be defined by a graph *DLA* = (*A*, *E*), where the set A= $\{A_1,A_2,\ldots,A_n\}$ is the set of n learning automata and $E \subset A \times A$ is the set of edges in the graph. The edge (*i*, *j*) represents the action j of automata LA_i. In other words, LA_j is activated when action j of automata LA_i is selected. The number of actions for particular automata LA_k (K=1, 2,…, n) is equal to the out-degree of that node. If p_j corresponds to the probability distribution of actions of LA_j, then p^j_m shows the probability of selecting action α_m by automata A_j. In other words, we can assign a weight to each edge (*i*, *j*) in graph which is equal to the probability of selection of action i by automata j (Meybodi and Beigy, 2001; Beigy and Meybodi, 2002; Beigy and Meybodi, 2006).

For example, in Figure 1, every automaton has two actions. Selection of action α3 by A_1 will activate automata A_3. Activated automata choose one of its action which results in activation of the LA corresponding to the selected action. At any given time, only one of the automata in the network could be active.

PROPOSED ALGORITHMS BASED ON DISTRIBUTED LEARNING AUTOMATA TO DETERMINING INFORMATIONAL WEB DOCUMENT STRUCTURE

In proposed methods, users play the role of a random environment for present automata in DLA. The output of DLA is a continuation of reviewed documents by a user which shows direction of movement toward his or her intended document. The environment will prepare an answer for DLA by using this continuation, the regarding present answer, the inside structure of automata available in DLA is updated according to learning algorithm. In both proposed algorithms, to determine structure of web documents, a DLA with n learning automata, which shows n web pages, is used. Each learning automata has n-1 action. For each learning automata, at each time, one subset of its actions is active.

First proposed algorithm

In the first proposed method, for every $page_j$ we consider learning automata j. Selection of action j by learning automata i means activating learning automata j correspond to $page_j$. In this case, the selected action action kth of LAi, (a^i_k =j) will be considered as corresponding probability with aforementioned action. At first all of the learning automata are inactive. Beginning with the user

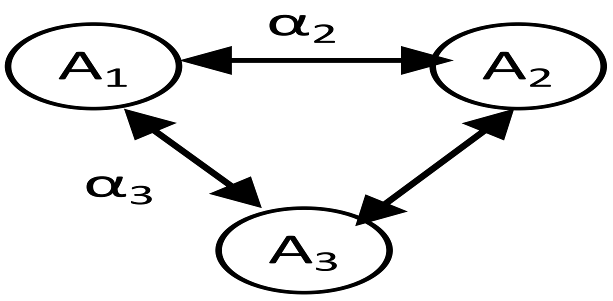

Figure 1. Distributed learning automata.

movement from page$_i$, LAi will be active and the corresponding action will be rewarded or penalized by that environment. Rewarding the selected actions in the aforementioned algorithm depend on three factors:

1- The traversed paths by users
2- The existence of link between pages
3- The similarity between pages

Penalty of selected actions depends on two factors too:

1- The existence of cycle in the path of users movement
2- The lack of similarity between two pages in every user's movement

Reward or penalty to actions of learning automata is done according to a learning algorithm. So by use of penalty the actions probability of automata will be update. $L_{R\varepsilon P}$ is used in this paper. As regards the movement of the user from page$_i$ to page$_j$, while the cosine similarity between page$_i$ and page$_j$ is more than 0.45, the corresponding action would be rewarded. Amount of reward is based on amount of cosine similarity and the link which is between two pages. If two pages have little similarity, little reward is given to them and if they have high similarity, more reward is given to them. The coefficient of reward will be calculated according to Equation 8.

$$a = \text{cosin}(\text{pagei}, \text{pagej}) * \omega + \lambda \quad (8)$$

Where cosine (pages i, page j) is amount of similarity between page$_i$ and page$_j$. ω= 0.01 is a fixed parameter and λ =0.02 is fixed too, and it is equal to zero if there was not a link between page$_i$ and page$_j$, otherwise it is equal to a fixed quantity.

Existence of cycle in used paths by user shows wrong movement of user, wandering in web or his/her unsatisfying of traversed path and to come back to beginning page. Therefore, existence pages in cycle are penalized according to Equation 9.

b= (distance between page$_i$ and page$_j$ in cycle)*β (9)

b is penalty parameter and β=0.002. Second state of penalty will transpire when cosine similarity of two pages is less than 0.45, so the corresponding action will be counted according to Equation 10.

$$b = \beta \quad (10)$$

First proposed algorithm is shown in Algorithm 1.

Second proposed algorithm

In the second proposed algorithm, such as the first one, the movement of user from page$_i$ to page$_j$ will result to reward to corresponding action and if there is a cycle, existence pages will be penalized. But the way of penalty and rewarding is different from the first proposed algorithm. Amount of reward is depending on following factors:

1- The traversed paths by users
2- Stay time of user in the page

If a user enters a page and discovers it is his/her intended page and it is also related to his/her subject that being searched, he/she would stay in that page for a long time, otherwise she/he would not stay in that page very much and would move to another page. As a result, if the user stay a lot of time in a page, it means that the page has much things that are related to what is been searched for and more reward will be achieved. On the contrary, if user stays in a page for a little time, it means that the page has little relation with his/her searching subject. Therefore, it will get little reward. As a result, we use user timestamp factor in page$_j$, which is registered in log file to determine amount of reward according to Equation 11.

$$a = \text{timestamp}(\text{pagej}) * \omega + \lambda \quad (11)$$

Timestamp (page$_j$) is the stay time of user in the page$_j$. ω=0.001 is a fixed parameter and λ=0.02 is fixed too, and it is equal to zero if there was not a link between page$_i$ and page$_j$, otherwise it is equal to a fixed quantity.

In proposed algorithm, actions corresponding to user's movement will be penalized just when it is a part of a cycle. Existence of cycle in traveled paths by user shows his/her wrong movements, wandering in web or his/her unsatisfying of traversed path and to come back to beginning page. Therefore, existence pages in cycle are penalized according to Equation 12.

b = (distance between page$_i$ and page$_j$ in cycle)*β (12)

b is penalty parameter and β= 0.002. Second proposed algorithm is shown in Algorithm 2.

PROPOSED ALGORITHM FOR RANKING WEB DOCUMENTS BY USE OF DISTRIBUTED LEARNING AUTOMATA

Page Rank algorithms specify rank of page based on rank of pages which link to it. In page rank, rank of page$_i$ will be calculated according to Equation 13.

$$PR(T_i) = \frac{1-d}{N} + d.\left\{\frac{PR(T_1)}{out(T_1)} + \cdots + \frac{PR(Tn)}{out(Tn)}\right\} \quad (13)$$

d is damping factor and it is between zero and one. N is the number of pages and out (T_i) is the number of output links of page T_i.

One of the ranking faults of PageRank, is that, it considers all the links as the same. In the proposed algorithm, to determine weight of links between pages, the proposed algorithms aforementioned was used, and therefore, links with high weight are more relevant and have more effect in ranking. Rank of page in the proposed ranking is calculated according to Equation 14.

$$PR(T_i) = \frac{1-d}{N} + d.\left\{\frac{PR(T_1)*DLA(T_1,T_i)}{out(T_1)} + \cdots + \frac{PR(Tn)*DLA(T_n,T_i)}{out(Tn)}\right\} \quad (14)$$

Given that DLA (T_1, T_i) is the specified weight to link between two pages 1 (i), this weight is accounted by two proposed algorithms. Another PageRank's fault is that, it considers all of the input links of a page to be even irrelevant links.

To solve this problem in the proposed method, irrelevant links are identified and they are not to be considered for ranking. To identify them, we use the similarity between pages. If cosine similarity between two pages is less than 0.45, the link between those pages

```
Procedure DLA-GP-1
   n: number of pages of the web
   G: hyperlink graph of the web
   User log: list of user's navigation graph
   p: is an n×n matrix
  $\alpha^i_j$: is the action j of learning automaton i.
 Begin
      DLA← Create a distributed learning automata with n learning automata
      For i=1 to n do
         For j=1 to n do
             If   (i=j)
                 P (i, j) =0
             Else
                 p (i, j) =1 / n-1
            End/if
         End/for
      End/for
  For each user$_u$ in user Log do
     If navigation graph of user$_u$ contains a cycle then
       For each cycle$_c$ found in the navigation graph of user$_u$ do
         For each (page$_i$, page$_j$) in cycle$_c$ which page$_i$ is visited before page$_j$ do
             Penalize action $\alpha^i_j$ according to b= (distance between page$_i$ and page$_j$ in cycle)*β
         End/for
       End/for
     Else
     For each (page$_i$, page$_j$) in navigation graph of user$_u$ which page$_i$ is visited before page$_j$ do
        If page$_i$ and page$_j$ have cosine (page$_i$, page$_j$) > 0.45 then
          Reward action $\alpha^i_j$ according to a= $\cos$in(pagei, pagej) $* \omega + \lambda$
        Else
          Penalize action $\alpha^i_j$ according to b=β
        End/if
     End/for
   End/if
 End/for
 End.
```

Algorithm 1. First proposed algorithm.

is irrelevant and will not be considered in ranking. The proposed ranking algorithm is shown in Algorithm 3.

SIMULATION RESULTS

To evaluating the proposed algorithms, we used the model presented by Liu et al. (2004), to show web structure and how users use the web. Validity of this model is confirmed by Liu and collaborators by using the information obtained from several large websites such as Microsoft. Therefore, in this paper, user's interest profile was considered as a power law distribution, the distribution of web pages content was considered as normal distribution and user's behavior pattern was considered normal. Other parameters by Liu et al. (2004) used for simulation in this paper are shown in Table 1. In this model, a content vector (cn) shows every web document, length of this vector is equal to the number of subject in system. Each member of this vector shows the degree of relationship of corresponding document to that vector with one of subjects.

$$C_n = \{CW^1_n, CW^2_n, \ldots, CW^m_n\} \qquad (15)$$

Correlation metric is used to evaluating algorithms for discovering the informational structure of web documents, which are presented previously.

$$\mathrm{Corr}(p, p') = \frac{\sum pp' - (\sum p \sum p')/n}{\sqrt{(\sum p^2 - (\sum p)^2/n)(\sum p'^2 - (\sum p')^2/n)}}$$

$p=\{p_{ij} \mid i, j=1,2,3,\ldots,n, i\neq j\}$

$$p_{ij} = \frac{d_{ij}^{-1}}{\sum_{k=1}^{n} d_{ik}^{-1}} \qquad (16)$$

```
Procedure DLA-GP-1
  n: number of pages of the web
  G: hyperlink graph of the web
  User log: list of user's navigation graph
  p: is an n×n matrix
  $\alpha^{i}_{j}$: is the action j of learning automaton i.
Begin
    DLA← Create a distributed learning automata with n learning automata
    For i=1 to n do
        For j=1 to n do
            If (i=j)
                P (i, j) =0
            Else
                p (i, j) =1 / n-1
            End/if
        End/for
    End/for
  For each $user_u$ in user Log do
      If navigation graph of $user_u$ contains a cycle then
          For each $cycle_c$ found in the navigation graph of $user_u$ do
              For each ($page_i$, $page_j$) in $cycle_c$ which $page_i$ is visited before $page_j$ do
                  Penalize action $\alpha^{i}_{j}$ according to b= (distance between $page_i$ and $page_j$ in cycle)*β
              End/for
          End/for
      Else
      For each ($page_i$, $page_j$) in navigation graph of $user_u$ which $page_i$ is visited before $page_j$ do
          If $page_i$ and $page_j$ have cosine ($page_i$, $page_j$) > 0.45 then
              Reward action $\alpha^{i}_{j}$ according to $a= cosin(pagei, pagej) * \omega + \lambda$
          Else
              Penalize action $\alpha^{i}_{j}$ according to b=β
          End/if
      End/for
    End/if
  End/for
  End.
```

Algorithm 2. Second proposed algorithm.

```
Procedure DLA- ranking
    N: number of pages of the web
    DLA: a matrix N*N from document structure (from last section)
    $R_1$: A matrix N*1 initialized by 1/N
    $R_2$: A matrix N*1 initialized by zero
    Out ($page_i$): number of forward links $page_i$
    In ($page_i$): number of citation links $page_i$
    d: damping factor
    Threshold: 1e- 8
Begin
    While ($|R_1- R_2|$ > threshold)
        $R_2= R_1$
        For i=1 to N
            For j=1 to in ($page_i$)
                If cosine ($page_i$, $page_j$) > 0.45 then
                    $temp = temp + \frac{R1(page\ j)}{out\ (page\ j)} * DLA(pagei, pagej)$
                End if
            End for
            $R_1$ ($page_i$) = (1-d)/N + d*temp
        End for
    End while
End.
```

Algorithm 3. Proposed ranking algorithm.

Table 1. Parameters in the model for experiment.

Degree of coupling	0.7
Number of agents	20000
Number of nodes	20
Number of topics	5
T_c	0.2
ΔM_t^c	-
ΔM_t^v	-
α_u	1
φ	1.2
λ	0.5
μ_m	5.97
μ_t	-
σ_m	0.25
α_p	3
σ	0.25
θ	1

$$d_{ij}= \sqrt{\sum_{k=1}^{m}\left(cw_i^k - cw_j^k\right)}$$

$$p' = \{p'_{ij} \mid i, j=1, 2, 3,\ldots, n, i \neq j\}$$

Set P is made structure in Liu et al. (2004) and set p' is obtained structure by proposed algorithm. D_{ij} is Euclidean distance between two documents of i and j. In Figure 5, the results of evaluation of first and second proposed algorithm were compared with the proposed algorithm in Mojtahedi and Meybodi (2009) (Mojtahedi algorithm). As it is shown in Figure 5, the proposed Algorithm-1 have high Correlation than proposed Algorithm-2, and this shows that, reward according to similarity between pages have better results than stay time of user in the page; the proposed algorithm contain high correlation than Mojtahedi algorithm, Because, the factors of similarity between pages and stay time of user in the page for rewarding is used in proposed algorithms.

To evaluating the proposed ranking algorithm, we use two metrics of "Precision at position n" (P@n) and RankCorrelation.

$$\text{RankCorrelation} = 1 - \left[\frac{6*\sum_{i=1}^{n} d_i^2}{(n*(n^2-1))}\right] \qquad (17)$$

Where n is the number of web pages and d_i is difference between ranks of i^{th} member of ranks set that is created by two different algorithms. To account for d_i of the proposed ranking algorithm, besides the presented proposed algorithms previously mentioned, an ideal connection matrix presented in the study of Liu et al. (2004) are applied and d_i is difference between these two ranking sets. The RankCorrelation is a number between 0 and 1. Figure 6 shows the result of simulation. Horizontal axis is learning stages to 200 repetitions. The graph of Rank-Algorithm1 is obtained from first proposed algorithm, the graph Rank-Algorithm2 is the result of operating the proposed ranking algorithm on matrix, and it was obtained from the second proposed algorithm. The graph PageRank is the result of operating PageRank ranking algorithm and the graph Anari algorithm is the result of operating ranking algorithm proposed in Anari et al. (2008). Rank-Algorithm1 and Rank-Algorithm2 have almost the same results and they have high performance than PageRank and Anari algorithm.

One of other evaluating metric is P@n which measures the relevancy of the top n results of the ranking list with respect to a given query (Equation 18). Figure 7 shows the obtained results from the evaluated P@n metric. In this paper, the amount of n is considered 10 and 60 query and was used for the result's test. For each query, the metric of P@n was for accounted, and then in each dimension, n is taken as their average.

$$P@n = \frac{\#\text{ of relevent docs in top n results}}{n} \qquad (18)$$

As it is shown in the Figure 7, the proposed ranking algorithm with two proposed structure gave better results than Anari algorithm and PageRank algorithm. Especially for the first result which is the most important one and it is a page that user clicks on.

CONCLUSION AND FURTHER WORK

The available ranking algorithms, containing algorithms based on connection and algorithms based on content have low recall and accuracy. In this paper, an algorithm was proposed for the ranking of web pages which uses content, connection and click-through data triple. At first,

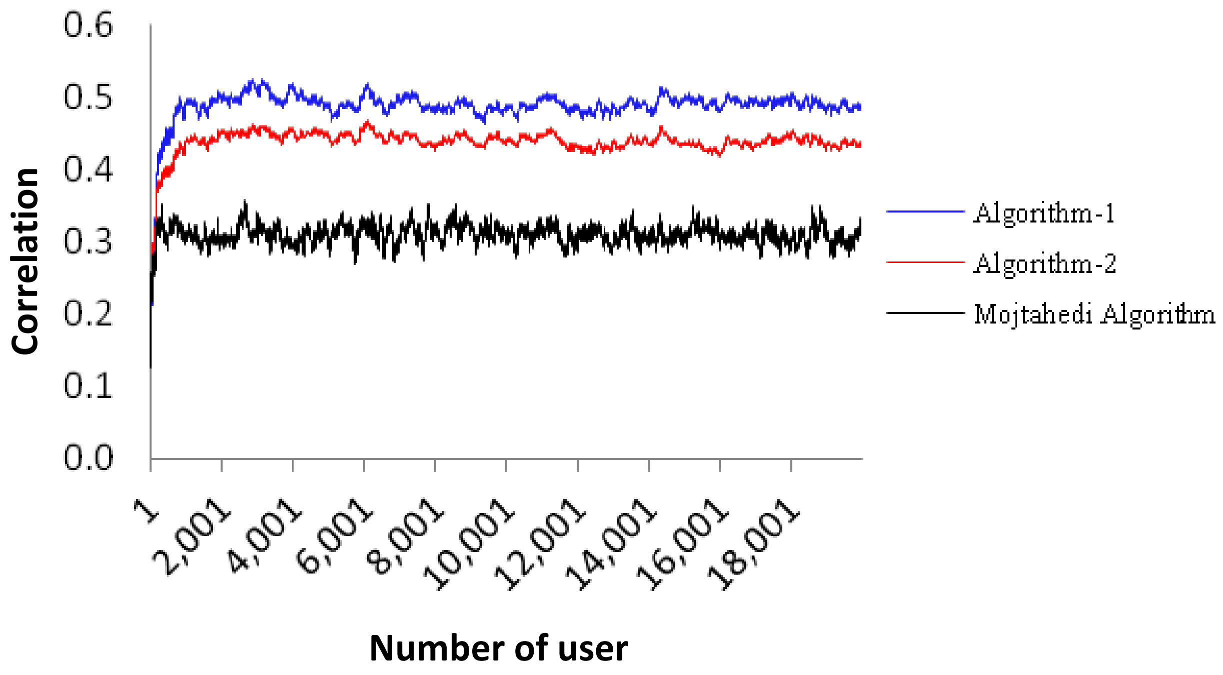

Figure 5. Comparison between the proposal methods with Mojtahedi algorithm.

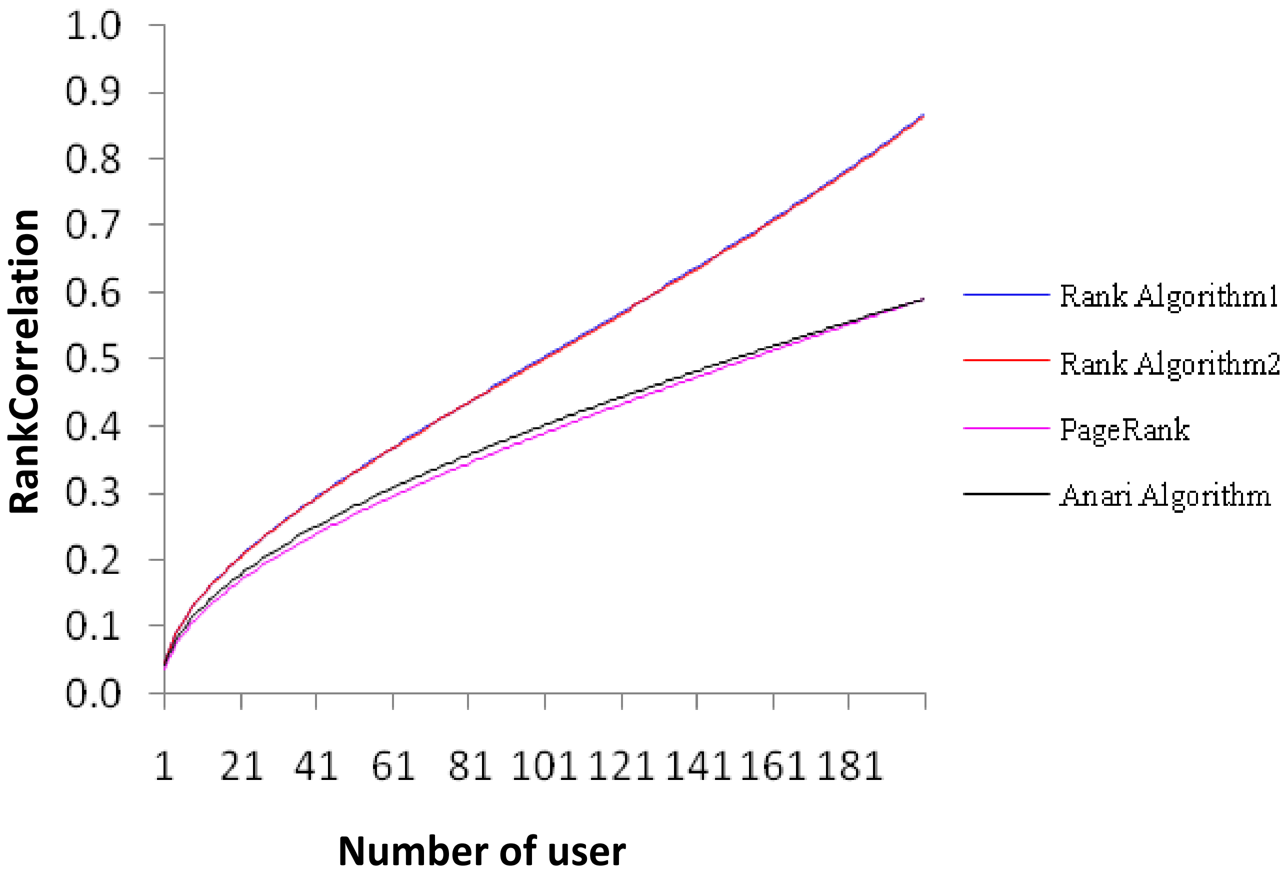

Figure 6. Comparison between the proposal methods with PageRank and Anari algorithm.

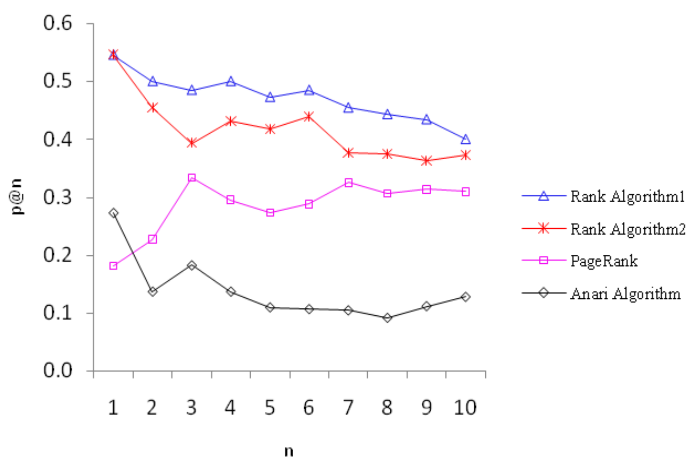

Figure 7. Comparison between the proposal methods with PageRank and Anari algorithm.

two algorithms based on DLA were proposed to determine structure of web document. Then, to determine rank of pages, the obtained results from these algorithms were used. First, the proposed algorithm specifies a weight to each link that is between two pages. Calculating weight of each link was based on similarity of pages content, existence of relevant link and web usage data. Second, the proposed algorithm calculates this weight based on other factors like timestamp that user stay in a page. After that, in the next stage, rank of each page was determined based on weight of citation relevant links. Proposed ranking algorithm considers just relevant links against PageRank algorithm that affects all citation links in calculating rank. Besides, PageRank considers all the links to be the same, but the proposed ranking algorithm specifies a weight for each link and that link have effect in ranking based on weight amount. To simulate the proposed algorithm, the model introduced by Liu et al. (2004) was used, while to evaluate the proposed ranking algorithm, P@n and RankCorrelation measures were used. Other algorithms like Hits can be used for proposing a ranking algorithm based on DLA, and also, DLA with changing number of actions for ranking algorithms can be used.

REFERENCES

Agichtein E, Brill E, Dumais S (2006). Improving Web Search Ranking by Incorporating User Behavior Information. In Proceedings of the ACM Conf. Res. Dev. Inf. Retrieval, pp. 1-8.
Almpanidis G, Kotropoulos C, Pitas I (2007). Combining text and link analysis for focused crawling - An application for vertical search engine. Inf. Syst., 32: 886-908.
Anari B, Meybodi MR (2007). A Method Based on Distributed Learning Automata for Determining Web Documents Structure. Proc. 12th Annual CSI Comput. Conf. Iran, pp. 2276-2282.
Anari B, Meybodi MR, Anari Z (2008). A New Method based on Distributed Learning Automata for Page Ranking in Web. Proc. 13th Annu. CSI Comput. Conf. Iran, pp. 9-11.
Baeza YR, Ribeiro NB (1999). Modern Information Retrieval. ACM Press/ Addison-Wesley. 1st edition.
Beigy H, Meybodi MR (2002). A New Distributed Learning Automata Based Algorithm For Solving Stochastic Shortest Path Problem. Proc. Sixth Int. Joint Conf. Inf. Sci., Durham, USA., pp. 339-343.
Beigy H, Meybodi MR (2006). Utilizing Distributed Learning Automata to Solve Stochastic Shortest Path Problem. Int. J. Uncert. Fuzziness Knowledge-based Syst., 14(5): 591-617.
Bressan M, Peserico E (2010). Choose the damping, choose the ranking. J. Discrete Algorithms, 8: 199-213.
Brin S, Page L (1998). The anatomy of a large-scale hypertextual web search engine. WWW7/Computer Networks, 30: 107-117.
Chen SF, Goodman J (1998). An Empirical Study of Smoothing Techniques for Language Modeling. Tech. Rep. TR-10-98. Harvard University, pp. 310-318.
Cho J, Roy S, Adams E, Quality P (2005). In Search of an Unbiased Web Ranking. In Proceedings of ACM International Conference on Management of Data, pp. 1-12.
Cicone A, Serra S (2010). Google Page Ranking problem: The model and the analysis. J. Comput. Appl. Math., 234: 3140-3169.
Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R (1990). Indexing by Latent Semantic Analysis. J. Am. Soc. Inf. Sci., 41: 391-407.
Ghodsnia P, Zareh Bidoki AM, Yazdani N (2008). WPR: A Weighted Approach to PageRank. Int. Rev. Comput. Softw., 3(1): 99-109.
Golub GH, Van Loan CF (1983). Matrix Computations. The Johns Hopkins University Press. 1st edition.
Hashemi AB, Meybodi MR (2005). Web Usage Mining Using Distributed Learning Automata. Computer Engineering Department, Technical Report, pp. 553-560.
Henzinger M, Motwani R, Silverstein C (2002). Challenges in web search engines. SIGIR Forum, 36(2).
Kerchove CD, Ninove L, Dooren PV (2008). Maximizing PageRank via out links. Lin. Algebra Appl., 429: 1254-1276.
Keyhanipour H, Moshiri B, Kazemian M, Piroozmand M, Lucas C (2007). Aggregation of Web Search Engines Based on Users' Preferences in Web fusion. Knowledge- Based Syst., 20(4): 321-328.
Kleinberg J (1997). Authoritative sources in a hyperlinked environment. J. ACM., 46(5): 604-632.
Liu B (2007). Web Data Mining-Exploring Hyperlinks, Contents, and Usage Data. Springer Ser. Data Centric Syst. Appl., p. 493.
Liu J, Zhang S, Yang J (2004). Characterizing web usage Regularities with information Foraging Agents. IEEE, 4: 566-584.
Ma N, Guan J, Zhao Y (2008). Bringing PageRank to the citation analysis. Inf. Process. Manage., 44: 800-810.
Meybodi MR, Beigy H (2001). Solving Stochastic Path Problem Using Distributed Learning Automata. Proc. Sixth Annu. Int. CSI Comput. Conf., pp. 70-86.
Mojtahedi SR, Meybodi MR (2009). A New Method based on Distributed Learning Automata for Discovering Documents Structures in Web. Proceedings of Third National Conference of Iran's Scientific Society of Command, Control, Communications, Comput. Intell., pp. 489-496.
Najork M, Zaragoza H, Taylor MJ (2007). Hits on the web: How does it compare? Proc. SIGIR'07, pp. 471-478.
Narendra K, Thathachar MAL (1989). Learning Automata: An Introduction. Prentice Hall, Englewood Cliffs, New Jersey, p. 476.
Page L, Brin S, Motwani R, Winograd T (1998). The PageRank citation ranking: Bringing order to the web. Available from http://dbpubs .stanford.edu /pub / 1999-66.
Qin T, Liu TY, Zhang XD, Chen Z, Ma WY (2005). A study of relevance propagation for web search. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval ACM Press., pp. 408-415.
Richardson M, Domingos P (2002). The intelligent surfer: Probabilistic combination of link and content information in PageRank. Adv. Neural Inf. Process. Syst., 14: 1441-1448.
Saati S, Meybodi MR (2005). A Self Organizing Model for Document Structure Using Distributed Learning Automata. Proceedings of the Second Int. Conf. Inf. Knowl. Technol., pp. 24-26.
Saati S, Meybodi MR (2006). Document Ranking Using Distributed Learning Automata. Proc. 11th Annual CSI Computer Conference of

Iran, Fundam. Sci. Res. Center, pp. 467-473.
Salton G, Buckley C (1988). Term-Weighting Approaches in Automatic Retrieval. Inf. Process. Manage., 24(5): 513–525.
Shakery A, Zha C (2006). A probabilistic relevance propagation model for hypertext retrieval. In Proceedings of the 15th ACM International Conference on Information and Knowledge Management, pp. 550-558.
Sidiropoulos A, Manolopoulos Y (2006). Generalized comparison of graph-based ranking algorithms for publications and authors. J. Syst. Softw., 79: 1679-1700.
Singhal A (2001). Modern Information Retrieval: A Brief Overview. IEEE Data Eng. Bull., 24(4): 35-43.
Sun H, Wei Y (2006). A note on the PageRank algorithm, Appl. Maths. Comput., 179: 799-806.
Thathachar MAL, Bhaskar RH (1987). Learning automata with changing number of actions. IEEE Trans. System Man Cybern., 17(6): 1095-1100.
Wu G (2008). Eigenvalues and Jordan canonical form of a successively rank-one updated complex matrix with applications to Google's PageRank problem. J. Comput. Appl. Math., 216: 364-370.
Zareh Bidoki AM, Ghodsinia P, Yazdani N, Oroumchian F (2010). A3CRank: An adaptive ranking method based on connectivity, content and click-through data. Inf. Process. Manage., 46(2): 159-169.
Zareh Bidoki AM, Yazdani N (2008). DistanceRank: An Intelligent Ranking Algorithm for Web Pages. Inf. Proc. Manage., 44(2): 877-892.
Zhai H (2006). Statistical Language Model for Information Retrieval. Tutorial Notes Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retrieval, 14: 1441-1448.