The 2011 CSI International Symposium on
Artificial Intelligence and Signal Processing
Department of Computer Engineering, Sharif University of Technology

Determining Web Pages Similarity Using Distributed Learning Automata and Graph Partitioning

Shahrzad Motamedi Mehr Majid Taran
Department of Electrical and Computer Engineering
Islamic Azad University, Qazvin Branch
Qazvin, Iran
motamedi@tmu.ac.ir, m_taran@isc.iranet.net

Ali B. Hashemi M. R. Meybodi
Computer Engineering and Information Technology
Department, Amirkabir University of Technology
Tehran, Iran
{a_hashemi, mmeybodi}@aut.ac.ir

Abstract— **Determining similarity between web pages is a key factor for the success of many web mining applications such as recommendation systems and adaptive web sites. In this paper, we propose a new hybrid method of distributed learning automata and graph partitioning to determine similarity between web pages using the web usage data. The idea of the proposed method is that if different users request a couple of pages together, then these pages are likely to correspond to the same information needs therefore can be considered similar. In the proposed method, a learning automaton is assigned to each web page and tries to find the similarities between that page and other pages of a web site utilizing the results of a graph partitioning algorithm performed on the graph of the web site. Computer experiments show that the proposed method outperforms Hebbian algorithm and the only learning automata based method reported in the literature.**

Keywords-web usage mining, distributed learning automata, web page similarity

I. INTRODUCTION

World Wide Web has been growing rapidly in recent years and this has resulted in a huge volume of hyperlinked documents which contain no logical organization. Currently, Google indexes more than 3 billions web pages in the world which this number increases with the rate of 7. 3 million pages per day. The massive influx of information onto World Wide Web has facilitated user, not only information retrieval, but also knowledge discovery. However, users are provided with more information and service options, it has become more difficult for them to find the "right" or "interesting" information, the problem commonly known as information overload due to the fact of significantly increasing and rapidly expanding growth in amount of information on the web.

Most researches in the data mining field based on document content analysis (data mining content) or graph structure and relation of documents (data mining structure). In addition information obtained from these two methods can be used information about the behavior of users (using Log files on Web servers or applications on the user side) to determine the relationship between the documents [4], the proposal pages [5][12] change the structure of Web site [11], personalized services like [13][14][15], search engine optimization [9] can be used. In [10] application information system is presented in detail.

In paper [22] model the problem as Q-Learning while employing concepts and techniques commonly applied in the web usage mining domain. It provides a system which is constantly in the learning process; Does not need periodic updates; can easily adapt itself to changes in website structure and content and new trends in user behavior.

In [4] new approach has been presented for solving web usage mining problem. Based on this idea is that if two documents respond to an information necessity, then the two documents are similar. This method assumes that users want to choose next step are aware of the relative of content document. In fact the users create virtual data communication between documents and will surf them. These relationships are user's mental model and may have no physical links.

a system that emulates human intuition, so that it can anticipate the desires of its users and provide them with the information they would find most interesting, even when these users cannot explicitly formulate what they are looking for. This is particularly useful for multimedia documents, which do not contain any searchable keywords, and for queries that are as yet ill defined.

Therefore a user moves from document i to document j, the only connection between these two documents (a(i,j)) is intensified. Strengthen the connection of two documents i and j corresponding increase in the similarity of these two documents have been considered. The proposed algorithm, the user travel from document i to document j, are not only strengthen the connection between these two documents, But with regard to transitive relation, connections the document i to other documents which the user will surf after j in the path, will be strengthened with a reduction coefficient.

Using the ideas discussed in the previous paragraph, in [1][2][3] a method is presented based on self-organizing Distributed Learning Automata to determine the similarity of documents in a digital library. Paper [2] uses a distributed learning automata corresponding to communication graph documents in digital libraries. In this method, the only

978-1-4244-9832-1/11/$26.00 ©2011 IEEE

documents are similar that user are traveling directly in the surfing path.

Algorithm [1] is based on distributed learning Automata, which the amount of similarity between the documents are calculated simultaneous with surf from one document to another document, the amount of similarity between the documents are calculated simultaneous. The advantage of paper [1] is used online but paper discussed in [2] doesn't use it. Transitive relation and the use of the Web graph structure of the proposed algorithm has been added to improve the correlation. Transitive relation means that if a user observe page A at first, then b and c at the end therefore the reward will be given in Automata a to b and c path with a reduction coefficient. Penalty is the same this but difference in factor, is increasing. Other features of proposed algorithm, the use of the Web graph structure. Web pages that are connected in Graph structures have high priority for surfing user.

In previous methods has been used distributed learning Automata, correlation decreased with large number of pages and can lead to undesirable results. Other features of the proposed algorithm are that the correlation value is not deceased with the large number of pages. The graph partitioning has been used for enhance the algorithm with large number of pages. The proposed algorithm is based on the idea that Internet users entering the network looking for a specific purpose in specific range. Web graph use partition algorithm to decrease the area of the Web and try to resolve users need have the range. When the user outside of its partition, surfing the incorrect path, therefore penalty with reduce the effect of inaccurate information. Rate calculated similarity decreases for pages outside the path, the specified relationship. Other penalties that the proposed method for the user is considered, there around the path is user survey.

The proposed method is based on distributed learning automata, the amount of similarity between documents is calculated when the user surf from one document to another. This feature causes the algorithm is used online. The proposed algorithm does not use any information about the content of documents and only using user behavior will calculate the amount of similar documents with each other. Using this method is useful to search documents that simply by Keyword (such as multimedia documents) can not be searched. The user can use the similar topic and other users surfing similar information to improve own searching

To compare the proposed algorithm with other algorithms, rather than using the actual Web pages and Web user's real data, used the model introduced in [6]. Also for graph partitioning are used multilevel algorithms [27][28]. This type of partitioning algorithms can be implemented in 3 phases. In the first phase of the Web graph size is small. in the second phase obtained graph in first Phase partitioning based on traditional methods, and the third phase the graph will become the primary mode, which is called the refinement phase.

Simulation results show that the proposed method efficiency for similarity detection is higher to compare the reports based on distributed learning automata with greater number of documents.

The rest of this paper is organized as follows. Section 2 provides Learning Automata and distributed learning. In section 3 we briefly introduced graph partitioning. In Section 4, after introducing the model used for simulation, simulation results are presented and review. Section 5 is conclusion.

II. 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 (i. e. grade of action) is used by automata in further action selection. By continuing this process, the automata learn to select an action with best grade. The learning algorithm used 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 [16]. In the following, the variable structure learning automata which will be used in this paper is described.

Variable structure learning automata is represented by quadruple $<\alpha,\beta,P,T>$, where β is the environment response set and $\alpha=\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ and $P=\{p_1, p_2, \ldots, p_r\}$ are the action set and corresponding probability set and the probability set, each being the probability of performing every action in the current internal automaton state, respectively. The function of T is the learning algorithm, which modifies the action probability vector P with respect to the performed action and the received response from the environment.

The above mentioned learning automata have a fixed number of actions. In some applications, it is required that learning automata has a changing number of actions [17]. 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 Equation(1).

$$p_i(n+1) = p_i(n) + a.(1-\beta(n)).(1-p_i(n)) - b.\beta(n).p_i(n)$$
$$p_j(n+1) = p_j(n) + a(1-\beta(n)).p_j(n) + \frac{b.\beta(n)}{r-1} - b.\beta(n).p_j(n) \qquad if\ j \neq i \tag{1}$$

where parameters a and b represent reward and penalty step length respectively.[24][25][26]

978-1-4244-9832-1/11/$26.00 ©2011 IEEE

A. *Distributed Learning Automata*

Distributed learning automata [26] is a network of automata which collectively cooperate to solve a particular problem. A DLA can be modeled by a directed graph in which the set of nodes of graph constitute the set of automata and the set of outgoing edges for each node constitutes the set of actions for corresponding automaton. When an automaton selects one of its actions, another automaton on the other end of edge corresponding to the selected action will be activated. An example of DLA is given in figure 1, in which every automaton has two actions. If automaton A_1 selects action α_3 as, then automaton A_3 will be activated. Activated automaton A_3 chooses one of its actions which in turn activates one of the automata connected to A_3. At any time only one automaton in the network will be activated. Formally, a DLA with n learning automata can be defined by a graph (A, E), where $A=\{A_1, A_2, ..., A_n\}$ is the set of automata and $E \subset A \times A$ is the set of edges in the graph in which an edge (i, j) corresponds to action j of automaton A_i denoted by α_j^i . [18]

III. THE PROPOSED ALGORITHM

In the proposed algorithm, for a web site a distributed learning automata, which is initially an isomorphic of the hyperlink graph of the web site, tries to determine the similarity between each pair of pages using the user interactions and hyperlink graph of the web site. The proposed algorithm can be described as follows.

First, the hyperlink graph of the web site is partitioned to K partitions[19][20][21]. Then, for a web site with n pages, a distributed learning automata, which has n learning automata with changing number of actions, is created. Each automaton corresponds to a particular page and has n-1 actions. Each action j of automata i (α_j^i) corresponds to a possible link between page i and page j. Initially, corresponding action to any existing hyperlink in the hyperlink graph of the web site is enabled and other actions in DLA, which has no corresponding hyperlink, remain disabled. Then, DLA will be updated using each user session as follows.

If a session contains a cycle, for each cycle, corresponding action for any page i that was visited before any page j (α_j^i) will be penalized if this action has already been enabled, i.e. there is a hyperlink from page i to page j. The rationale behind this update is that it is considered the pages in the cycle may not be similar enough to each other than what user have had in mind, which caused the user to go back to a page that has already visited and started the browsing from that page again. In addition, it is considered that the farther two pages are visited from each other, the less they may be similar. Hence, the penalty step length is set proportional to the distance of page i and page j in the cycle (eq.2).

$$b_{i,j}^{cycle} = (\text{distance between page } i \text{ and page } j \text{ in the cycle}).\ b_0^{cycle} \qquad (2)$$

where b_0^{cycle} is a constant value for the base cycle penalty step length.

Procedure DLA-GP
  n: number of pages of the web site
  HG: hyperlink graph of the website
  K: number of partitions of the hyperlink graph of the web site
  userLog: list of users navigation graph
  s: is an n×n matrix which will hold the similarity of every two pages i and j.
  α_j^i : is the action j of learning automaton i.
begin
  Partition the HG into K partitions
  $DLA \leftarrow$ Create a distributed learning automata with n learning automata with changing number of actions; each has n-1 actions which all are disabled by default.
  for each hyperlink from $page_i$ to $page_j$ in HG **do**
    enable action α_j^i of the DLA
  end-for
  for each $user_u$ in userLog **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**
          if action α_j^i is enabled **do**
            penalize action α_j^i according to eq. (2) and eq. (1)
          end-if
        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 α_j^i is disabled **do**
          enable action α_j^i
        end-if
        if $page_i$ and $page_j$ are in the same partition **then**
          reward action α_j^i according to eq. (3) and eq. (1)
        else
          penalize action α_j^i according to eq. (2) and eq. (1)
        end-if
      end-for
    end-if
  end-for
  for each ($page_i$,$page_j$) **do**

$$s(i,j) = \begin{cases} 0 & \text{if action } \alpha_j^i \text{ is disabled} \\ \dfrac{\hat{p}_j^i}{\sum_{j=1}^{n} \hat{p}_j^i} & \text{if action } \alpha_j^i \text{ is enabled} \end{cases}$$

  end-for
end
end

Figure 1. Pseudo code of proposed algorithm

Otherwise, if the user session does not contain any cycle, for any page i visited before any page j, corresponding action in DLA (α_j^i) first will be enabled, if it is not enabled yet, then will be updated. The idea is that, after partitioning of the hyperlink graph of the web site, two pages are partitioned into the same partition are more likely to be similar that two pages which are put into different partitions. Therefore, if page i and page j are in the same partition, action α_j^i will be rewarded with a reward step length inversely proportional to the distance between page i and page j in the session (eq.3). Otherwise, α_j^i will be penalized with a constant penalty step length b_0 .

978-1-4244-9832-1/11/$26.00 ©2011 IEEE

$$a = \frac{1}{\text{distance between page } i \text{ and page } j \text{ in the session}}\omega + \lambda \quad (3)$$

where ω is a constant parameters. Parameter λ will be zero, if there is no link from page i to page j. Otherwise it will be set to a constant value λ_0.

The above process continues for all sessions. At any time, similarity between page i and page j is proportional to the probability of the corresponding action, i.e. action j of automaton i (α_j^i), if it is enabled. Otherwise, two pages are considered completely different. Pseudocode of proposed algorithm is presented in figure 1.

IV. EXPERIMENTATIONS

In this section we explain the model used for generating web usage data then presents experimental results of the proposed algorithm and compare the result with previous algorithms in the literature.

A. Web Site and Users Model

In order to measure the performance of the proposed algorithm, we utilized the user log data generated by the web surfing model proposed by Liu et al [6]. In this model, which has been validated against some empirical web log data, users of a web site are considered as information foraging agents and the web page as the information resources. Each web page contains certain information contents, and each hyperlink between two web pages signifies certain content similarity between them.

The contents contained in the page i can be characterized using a M-dimensional content vector C_i, where each component cw_i^t corresponds to the relative content weight of page i on topic t. Similarity between two pages i and j is considered as the inverse of the distance between content vector of two pages i and j (d_{ij}), which is determined by the Euclidean distance of their content vectors (eq. 4).

$$d_{ij} = \sqrt{\sum_{k=1}^{M}\left(cw_i^k - cw_j^k\right)^2}$$
$$D = \left\{d_{ij} \mid i, j = 1,2,\cdots,n\right\} \quad (4)$$

We considered user of the website as "rational" users who have specific interested topics in mind and forage in order to locate the pages that contain information on those topics. When a rational user reaches a new page, he will try to decide whether or not the content sufficiently matches his information needs (denoted by his interest profile) and, if not, predicts which page at the next level will be likely to become a more interesting one. In predicting the next page after page i, the user will select a page j, that has a link in the page i, with a probability proportional to the similarity of his interest profile vector and content vector of page j. When a user does not find any interesting information after some foraging steps or has found enough contents satisfying his information needs, he will stop foraging and leave the web site. The values of the parameters of this model used in the following experiments are listed in table 1.

TABLE I. PARAMETERS OF THE WEB SITE AND USERS' MODEL

Degree-of-coupling, r	0.7
Number of users	50000
Number of pages	501
Number of topics	5
T_c the content increment offset on the topic	0.2
ΔM_t^c The constant decrement in ΔM_t at each time step,	-
ΔM_t^v The variable factor that dynamically changes at each time step	-
α_u the shape parameter of a power-law distribution.	1
ϕ Weights of reward	1.2
λ Absorbing factor in [0,1]	0.5
μ_m mean of the log-normal distribution of ΔM_t^v	5.97
σ_m variance of the log-normal distribution of ΔM_t^v	0.25
μ_t Mean of normally distributed offset T	-
α_p the shape parameter of a power-law distribution	3
σ_t Variance of normally distributed offset T	0.25
θ Weights of motivation	1
Minimum of motivation for the continued search	0.2

B. Performance measure

In the proposed algorithm, similarity of two pages i and j is denoted by s'_{ij}. If action$_j$ of LA$_i$ is active then s'_{ij} will be set to p_j^i (probability of action$_j$ of LA$_i$) otherwise s'_{ij} will be set to zero (eq.5).

$$s'_{ij} = \begin{cases} p_j^i & \text{if } \alpha_j^i \text{ is enabled} \\ 0 & otherwise \end{cases}$$
$$S' = \left\{s'_{ij} \mid i, j = 1,2,\cdots,n \;,\quad i \neq j\right\} \quad (5)$$

Performance of the proposed algorithms is measured by the correlation between the matrix of the distance between content vector of every two pages (D) and the similarity matrix calculated by the proposed algorithm. Since similarity between two pages is inverse proportion to the distance between their content vectors, the better the algorithm performs the closer to -1 the correlation value will be.

$$Correlation(D, D') = \frac{Cov(D, D')}{\sigma_D \sigma_{D'}} = \frac{\sum DD' - (\sum D \sum D')/N}{\sqrt{(\sum D^2 - (\sum D)^2/N)(\sum D'^2 - (\sum D')^2/N)}} \quad (6)$$

C. Experimental Settings

In the experiments, an implementation [21] of multilevel graph partitioning method [23] is used to create partitions of 25 web pages from the graph of the web site.

D. Experimental Results

The simulation results of this algorithm are compared with the algorithm present in [1] to evaluate the proposed algorithm. Also, a simple statistical method is used to show a lower bound for the performance of this algorithm. In this statistical method is calculated similarities between document i and j is : the number of surfing document i to j divided by the number of

978-1-4244-9832-1/11/$26.00 ©2011 IEEE

surfing document *i* to any other document such as document k (Equation (7)).

$$similarity(i,j) = \frac{s(i,j)}{\sum_{k=1}^{n} s(i,k)} \qquad (7)$$

In figure 2 is shown efficacy of statistical method. Since the rate of use of connection between two documents *i* and *j* has an inverse ration with the distance vector of their content we expect a negative correlation for these two values. This value is negative as is seen in figure 2.

At simple and advanced Hebbian method increases the amount of a_{ij} if the user surf link (i,j). When there is little Euclidean distance between two nodes *i* and *j* (small amount d_{ij}) therefore we expected a_{ij} is much. Assuming that users can select connections with weight (distance) less for their next move hence are expected that amount of correlation a_{ij} with real value of distance nodes is negative. As you can see in Figure 3, correlation value of matrix A and D is negative.

As shown in figure 3, not only proposed algorithm performs better than two previously introduced DLA based methods [1, 2]after processing all user sessions, but also it is more stable as soon as the its correlation value reaches the minimum, it does not increases as previous DLA-based method. Moreover, the correlation of the similarity matrix in the proposed algorithm is significantly better than the naïve method introduced above and extended Hebbian methods [4] (figure 2).

The effect of different parts of the proposed algorithm on its performance is also studied in figure 4. In figure 4, the correlation between similarity matrix obtained from the proposed algorithm and the matrix of the distance between content vectors of every two pages the when different modules of the proposed algorithm is disabled, are presented. As it is shown in this figure, when the proposed algorithm stop penalizing either the links in the user navigation path cycles or when two consecutive page in the user navigation path do not belong to one partition, it results the worse correlation value. In addition, when the algorithm, only utilizes the effect of consecutive visited pages, the correlation value go down to -0.55. In addition, it is shown that using the graph partitioning results helps improving the accuracy of the similarity matrix as shown 0.05 improvements in the correlation value.

As shown in figure 5, when number of pages increases, if the proposed algorithm does not use the graph partitioning, the correlation value deteriorates significantly. However, by adding graph partitioning module (figure 6), correlation value does not change significantly for a web site with 501 pages compare to a web site with 26 pages (figure 4).

Figs. 7 shows the correlation of proposed algorithm and DLA Method [1] with 501 pages and 50000 users. In figure 10 the proposed algorithm with graph partitioning is significantly better than DLA method[1] and proposed algorithm without graph partitioning.

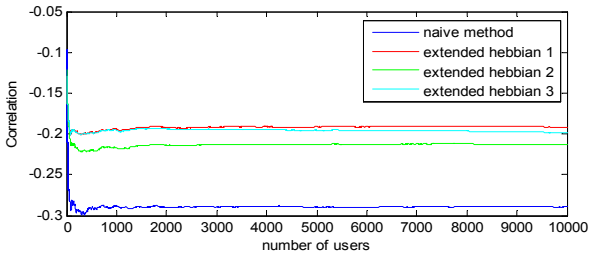

Figure 2. Correlation for simple and advanced Hebbian algorithm

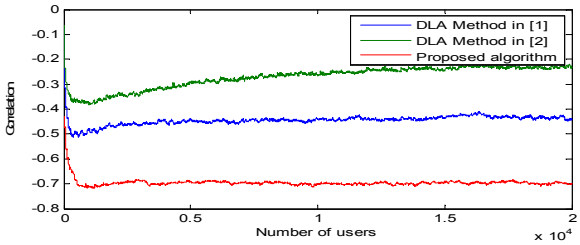

Figure 3. Correlation of algorithm based on distributed learning automata and proposed combination algorithm in a web site with 26 pages

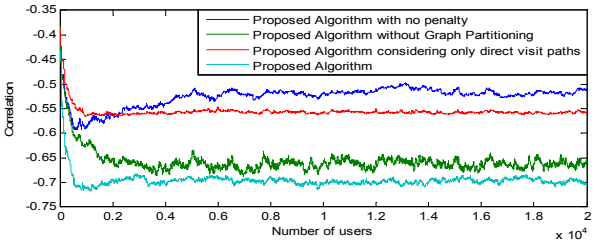

Figure 4. Correlation of proposed algorithm with different conditions in a web site with 26 pages

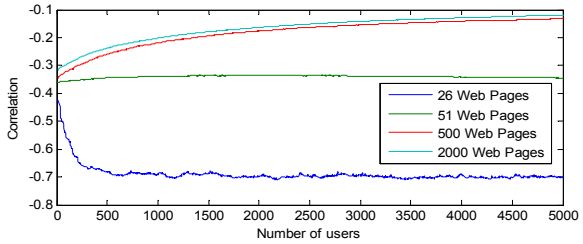

Figure 5. Correlation of proposed algorithm without graph partitioning module when number of pages of a website varies

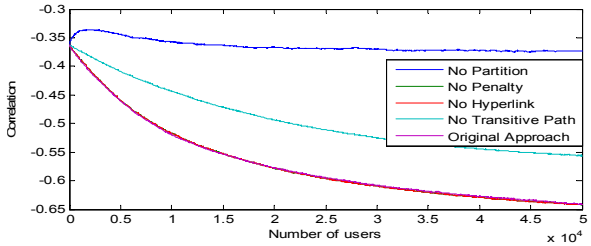

Figure 6. Correlation of proposed algorithm with different conditions in a web site with 501 pages

The proposed method will produce better results for larger documents.

Moreover, the obtained result could be used to find out web pages similarities and identify web communities.

978-1-4244-9832-1/11/$26.00 ©2011 IEEE

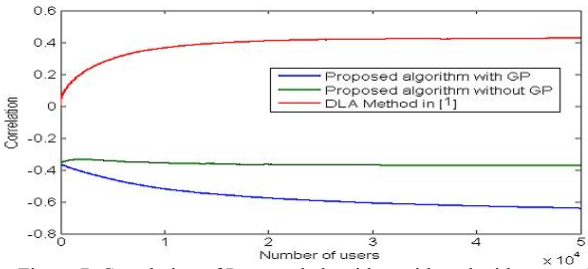

Figure 7. Correlation of Proposed algorithm with and without graph partitioning and DLA Method [1] with 501 web pages and 50000 users

V. CONCLUSION

In this paper we proposed new method based on distributed learning automata and graph partitioning. We improved method based on learning automata. In this paper web graph and transitive relation and graph partitioning to improve web mining performance are used. In this paper we have expanded the proposed algorithm for more pages and the result have improved it is shown that the proposed method performs better than the Hebbian algorithm and the only learning automata based method reported in the literature. The proposed method's correlation is more than distributed learning automata based method reported in the literature. Partitioning improves the quality of recommended pages significantly.

REFERENCES

[1] A. B. Hashemi, M. R. Meybodi, "Web Usage Mining Using Distributed Learning Automata", Proceedings of 12th Annual CSI Computer Conference of Iran, Shahid Beheshti University, Tehran, Iran, pp. 553-560, Feb. 20-22, 2007.

[2] S. Saati, M. R. Meybodi, "A Self Organizing Model for Document Structure Using Distributed Learning Automata", Proceedings of the Second International Conference on Information and Knowledge Technology (IKT2005), Tehran, Iran, May 24-26, 2005.

[3] B. Anari, M. R. Meybodi, "The Method for Document Web Structure Using Distributed Learning Automata", Proceedings of 12th Annual CSI Computer Conference of Iran, Shahid Beheshti University, Tehran, Iran, Feb. 20-22, 2007.

[4] F. Heylighen and J. Bollen, "Hebbian Algorithms for a Digital Library Recommendation System," Proceedings of the International Conference on Parallel Processing Workshops (ICPPW'02), 2002, pp. 439-446.

[5] M. Pazzani, J. Muramatsu and D. Billsus, "Syskill & Webert: Identifying Interesting Web Sites," Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96), AAAI Press, 1996, pp. 54–61.

[6] J. Liu, S. Zhang, and J. Yang, "Characterizing Web Usage Regularities with Information Foraging Agents," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 4, April 2004, pp. 566-584.

[7] T. Joachims, "Optimizing Search Engines Using Click Through Data," Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-02), 2002, pp. 133–142.

[8] J. Srivastava, R. Cooley, M. Deshpande, and P. N. Tan, "Web Usage Mining: Discovery and Applications of Usage Ppatterns from Web Data," SIGKDD explorations, vol. 1, no. 2, 2000, pp. 12–23.

[9] Mike Perkowitz and Oren Etzioni, "Adaptive Web Sites," Communications of ACM, vol. 43, no. 8, 2000, pp. 152-158.

[10] R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell, "WebWatcher: A Learning Apprentice for the World Wide Web," Proceedings of AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, AAAI Press, 1995, pp 6–12.

[11] B. Mobasher, R. Cooley, and J. Srivastava, "Automatic Personalization Based on Web Usage Mining," Communications of the ACM, vol. 43, no. 8, 2000, pp. 142–151.

[12] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa, "Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization," Data Mining and Knowledge Discovery, vol. 6, no. 1, 2002, pp. 61-82.

[13] Pierrakos, G. Paliouras, C. Papatheodorou, and C. D. Spyropoulos, "Web Usage Mining as a Tool for Personalization: A Survey," User Modeling and User-Adapted Interaction, vol. 13, no. 4, 2003, pp. 311-372.

[14] K. Narendra, M. A. L. Thathachar, "Learning Automata: An Introduction", Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[15] M. A. L. Thathachar, R. Harita Bhaskar, "Learning Automata with Changing Number of Actions", IEEE Transactions on Systems Man and Cybernetics, vol. 17, no. 6, Nov. 1987, pp. 1095-1100.

[16] M. R. Meybodi, H. Beigy, "Solving Stochastic Path Problem Using Distributed Learning Automata", Proceedings of The Sixth Annual International CSI Computer Conference, CSICC2001, Isfahan, Iran, 2001, pp. 70-86.

[17] M. R. Meybodi, H. Beigy, "Solving Stochastic Shortest Path Problem Using Monte Carlo Sampling Method: A Distributed Learning Automata Approach", Springer-Verlag Lecture Notes in Advances in Soft Computing: Neural Networks and Soft Computing, 2003, pp. 626-632.

[18] H. Beigy, M. R. Meybodi, "A New Distributed Learning Automata Based Algorithm For Solving Stochastic Shortest Path Problem", Proceedings of the Sixth International Joint Conference on Information Science, Durham, USA, 2002, pp. 339-343.

[19] S. T. Barnard and H. D. Simon, "A Fast Multilevel Implementation of Recursive Spectral Bisection for Partitioning Unstructured Problems". in Proceedings of the Sixth SIAM Conference on Parallel Processing for Scientific Computing. 1993. Norfolk, Virginia, USA.

[20] C. -K. Cheng, and Y. -C. A. Wei, "An improved two-way partitioning algorithm with stable performance". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 1991. 10(12): p. 1502-1511.

[21] B. Hendrickson and R. Leland, "A multilevel algorithm for partitioning graphs". Technical Report SAND93-1301, Sandia National Laboratories, 1993.

[22] N. Taghipour, A. Kardan, S. S. Ghidary. "Usage-based web recommendations: a reinforcement learning approach". ACM Conference on Recommender Systems. Minneapolis, MN, USA 2007.

[23] G. Karypis and V. Kumar, "METIS: A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices". *Version 5. 0pre2*. 2007: Minneapolis.

[24] N. Taghipour, A. Kardan, S. S. Ghidary. "Usage-based web recommendations: a reinforcement learning approach". ACM Conference on Recommender Systems. Minneapolis, MN, USA 2007.

[25] M. Thathachar, P. Sastry, Varieties of learning automata: an overview, IEEE Transactions on Systems, Man and Cybernetics, 32 (2002) 711-722.

[26] H. Beigy, and M. R. Meybodi, "Utilizing Distributed Learning Automata to Solve Stochastic Shortest Path Problem," International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, vol. 14, no. 5, pp. 591-617, 2006.

[27] G. Karypis and V. Kumar. A fast and highly quality multilevel scheme for partitioning irregular graphs. SIAM Journal on Scientific Computing, 1998 (to appear). A short version appears in Intl. Conf. on Parallel Processing 1995.

[28] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. Journal of Parallel and Distributed Computing, 48(1):96–129, 1998.

978-1-4244-9832-1/11/$26.00 ©2011 IEEE