A Dynamic Web Recommender System Based on Cellular Learning Automata

Mojdeh Talabeigi
Department of Computer Engineering
Islamic Azad University, Qazvin Branch
Qazvin, Iran
Mojde.talabeigi@gmail.com

Rana Forsati
Department of Computer Engineering,
Islamic Azad University, Karaj Branch
Karaj, Iran
forsati@kiau.ac.ir

Mohammad Reza Meybodi
Department of Computer Engineering,
Amirkabir University of Technology
Tehran, Iran
mmeybodi@aut.ac.ir

Abstract- Different web recommendation systems have been proposed to address the problem of information overload on the Internet. They attempt to guide users toward interesting and useful items in a large information space. They anticipate the information needs of on-line users and provide them with recommendations to facilitate and personalize their navigation. There are many approaches to building such systems, but most of these systems are static which analyze information resources, discover patterns from this data and make recommendations based on the extracted knowledge. So they need to periodically update extracted pattern and rules in order to make sure they still reflect the trends of users or the changes of the site structure or content. In this paper we propose a dynamic web page recommender system based on asynchronous cellular learning automata (ACLA) which continuously interacts with the users and learns from his behavior. Furthermore, we try to use all factors which have influence on the quality of recommendation and might help system to be able to make more successful recommendations. The proposed system use web usage data and structure of the web site to learn user navigation patterns and predicting user's future requests. We evaluate our method under different settings. Our experiments on real data set show that our proposed system performs better than the other algorithm we compared to and show how this method can improve the overall quality of web recommendations.

Keywords- Asynchronous cellular learning automata, Web recommendation, web mining.

I. INTRODUCTION

In recent years, with the explosive growth of knowledge available on the World Wide Web, which lacks an integrated structure or schema, it becomes much more difficult for users to access relevant information efficiently. Web users usually suffer from the information overload problem due to the fact of significantly increasing and rapidly expanding growth in amount of information on the web. Recommender systems have been introduced to solve these problems. They attempt to predict user preferences and needs from different resources such as user's access behaviors or structure or content of website and then guide users toward their interesting and useful items.

One research area that has recently contributed greatly to this problem is web mining. Recently, there has been an increasing interest in applying web usage mining techniques to build web recommender systems [1,2,3,4,5]. Web usage recommender systems take web server access logs as input, and make use of data mining techniques such as association rule and clustering to extract navigational patterns, which are then used to provide recommendations. Web server access logs record user browsing history, which contains plenty of hidden information regarding users and their navigation. These systems are mainly concerned with analyzing web usage logs, discovering patterns from this data and making recommendations based on the extracted knowledge [1,2,4]. They could, therefore, be a good alternative to the explicit user rating or feedback in deriving user models. In web usage recommender systems, navigational patterns are generally derived as an off-line process.

In general, a web recommender system is composed of two modules: an off-line module and an on-line module. The off-line module pre-processes data to generate user models and extract pattern, while the on-line module uses and updates the models on-the-fly to recognize user goals and predict recommendation lists [6]. One issue commonly faced in such systems is the need to periodically update patterns extracted from access logs in order to make sure they still reflect the trends residing in user behavior or the changes of the site structure or content. Taghipour [7] modeled the recommendation process as a Q-learning problem and proposed the first dynamic web recommender system which is trained with common web usage logs. Although the experiments achieved promising results, but there is some disadvantages in using Q-learning for prediction process. The system is complex because we were faced with a rather large number of states and there were cases where the state resulted from the sequence of pages visited by the user had actually never occurred in the training phase.

In this paper, we propose another machine learning approach toward the problem, which we believe is more suitable to the nature of web page recommendation problem and has some intrinsic advantages over previous method. For this purpose, we use an asynchronous cellular learning automaton (ACLA) for learning user navigation behavior and predicting useful and interesting pages. The idea of using ACLA is that in proposed ACLA we have small number of cell which each of them represents a web page. Each cell can hold the traversed page and recommended page because of the characteristic of CLA and use them for rewarding or penalizing process. The experimental results show that our system performance is relatively well. Our system has less complexity because it has not lots of state and there are not cases where the state resulted from the sequence of pages visited by the current user had actually never occurred in the training phase, because each page has a corresponding cell and when the current user visit a page, it is not necessary to find an exact match between current user's visited page and V list. Instead, the cell corresponds to the last page visited by user, will be activated.

The organization of the paper is as follows: in section 2, learning automaton, cellular learning automata and asynchronous cellular learning automata are briefly introduced. The factors which influence the recommendation process discussed in section 3. Section 4 describes our proposed system for recommendation process. The overall performance of the system is evaluated in section 5. Finally, section 6 summarizes the paper and introduces future work.

II. BACKGROUND

In this section, we will briefly introduce cellular learning automata and asynchronous cellular learning automata.

Cellular learning automata, which is a combination of cellular automata (CA) and learning automata (LA), is a powerful mathematical model for many decentralized problems and phenomena. CLA is a mathematical model for dynamical complex systems that consist of a large number of simple components. The simple components, which have learning capability, act together to produce complex emergent global behavior. The basic idea of CLA, which is a subclass of stochastic CA, is to use learning automata to adjust the state transition probability of stochastic CA. A CLA is a CA in which a learning automaton is assigned to every cell. The learning automaton residing in a particular cell determines its action (state) on the basis of its action probability vector. Like CA, there is a rule that the CLA operates under. The rule of the CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to the LA residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is non stationary because the action probability vectors of the neighboring LAs vary during evolution of the CLA. The basic idea of CLA, which is a subclass of stochastic CA, is to use learning automata to adjust the state transition probability of stochastic CA. A d-dimensional CLA is formally defined below.

A *d* dimensional cellular learning automata can be defined as *CLA=(*Z^d*,φ,A,N,F)* where

- Z^d is a lattice of d−tuples of integer numbers. This network can be finite, infinite or semi-finite.
- *φ* is a finite set of states.
- *A* is a set of learning automata each of which is assigned to one of the cells of cellular automata.
- *N={*x_1*,…,*x_m*}* is a finite subset of Z^d and is named neighbourhood vector.
- *F:* $\varphi_m \rightarrow \beta$ is the local rule of CLA in where β is the set of values that the reinforcement signal can take. Function F computes the reinforcement signal for each LA based on the actions selected by the neighboring LA. It computes the new state for each cell from the current states of its neighbors.

A CLA is called synchronous if all LAs are activated at the same time in parallel. A CLA is called asynchronous (ACLA) if at a given time only some LAs are activated independently from each other, rather than all together in parallel. For more information about CLA the reader may refer to [12,13,14].

In synchronous CLA, all cells are synchronized with a global clock and executed at the same time. In some applications such as call admission control in cellular networks, a type of CLA in which LAs in different cells are activated asynchronously (ACLA) is needed. LAs may be activated in either *time-driven* or *step-driven* manner. In time-driven ACLA, each cell is as summed to have an internal clock which wakes up the LA associated to that cell. The internal clock possibly has different time step durations and are asynchronous, i.e., each going at its own speed and does not change time simultaneously. In step-driven ACLA, a cell is selected in fixed or random order. Formally, a *d*-dimensional step-driven ACLA with *n* cells is a structure _*Zd,*Φ*,A,N, F, ρ*_, where *Zd* is a lattice of *d*-tuples of integer numbers, Φ is a finite set of states, *A* is the set of LAs assigned to cells, *N = {¯x1, ¯x2, . . . , ¯x ¯m215 }* is the neighborhood vector, *F* : Φ*m¯* → *β* is the local rule, and *ρ* is an *n*-dimensional vector called activation probability vector, where *ρi* is the probability that the LA in cell *i* (for *i* = 1, . . . , *n*) is to be activated in each stage [15].

The operation of ACLA takes place as the following iterations. At iteration *k*, each LA *Ai* is activated with probability *ρi*, and the activated LAs choose one of their actions. The activated automata use their current actions to execute the rule (computing the reinforcement signal). The actions of neighboring cells of the activated cell are their most recently selected actions. Let *αi* ∈ *αi* and *βi* ∈ *β* be the action chosen by the activated and the reinforcement signal received by LA *Ai*, respectively. The reinforcement signal is produced by the application of local rule *Fi*. Finally, activated LAs update their action probability vectors, and the process repeats [22].

Let *n*∈*N* be the set of nonnegative integers. A general linear schema for updating action probabilities can be represented as follows. Let action *i* be performed then if the action *i* must recieve reward, the probability vector will be updated according to (1). Otherwise, the probabilities will be updated according to (2).

$$\begin{aligned} p_i(n+1) &= p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) &= (1-a)p_j(n) \qquad \forall j \quad j \neq i \end{aligned} \tag{1}$$

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n) \\ p_j(n+1) &= (b/s-1) + (1-b)p_j(n) \quad \forall j \quad j \neq i \end{aligned} \tag{2}$$

Where *a* and *b* are reward and penalty parameters. When *a=b*, the automaton is called L_{RP}. If *b=0* the automaton is called L_{RI} and if $0<b<<a<1$, the automaton is called $L_{R\varepsilon P}$. For more Information about learning automata the reader may refer to [11].

III. EFFECTIVE FACTORS on QUALITY OF RECOMMENDATIONS

To improve quality of our system we take into account all the metrics which might influence the quality of recommendations. These metrics used in other web recommender systems, we choose more effective one from other literature [18, 19].

One issue which we must consider to improve quality of recommendation is the importance of a page. In general, all accessed page can be considered interesting to various degrees because users visited them. It is quite probable that not all the pages visited by the user are of interest to him. A user might request a page only to find a better page or he has been lost, so the page has no value to him, causing irrelevant page accesses to be recorded into the log file and may mislead future recommendations. Therefore, it is not efficient to use all the visited pages equally to make recommendation. So we try to approximate the degree of importance and interest of a web page for users. Some factors which have influence on importance of a page are time user spends on a page, frequency of a page in access log [2,23], position of page in graph of web.

We conjecture that the longer time a user spends on a page, the likelier the user is interested in the page. If a web page is not interesting, a user usually jumps to another page quickly [20]. However, a quick jump might also occur due to the short length of a web page so the size of a page has influence on the time a user spends on a page. Hence, it is more appropriate to accordingly normalize "Duration" by the length of the web page, that is, the total bytes of the page. We use (3) to measure the "Duration" of a web page.

$$Duration(p) = \frac{\frac{Total\ Duration(p)}{Size(p)}}{\max_{Q\in T}\frac{Total\ Duration(p)}{Size(p)}} \tag{3}$$

"Frequency" is the number of visits of a web page. It seems natural to assume that web pages with a higher frequency are more interesting and usefulness for users. A parameter that is must be considered in the calculating the frequency of a page is the number of incoming links to the page. It is obvious that a page with large in-degree has more probability to be visited by a user. Specially, in comparing two pages with same visiting rate, the page with small in-degree is more interesting. The formula of ''Frequency'' is given in (4), which is normalized by the total number of visits of web pages in the session:

$$Frequency(p) = \frac{Number\ of\ \ visit(p)}{\sum_{Q\in T} Number\ of\ \ visit(Q)} * \frac{1}{\text{Indegree}(p)} \tag{4}$$

"Frequency" and "Duration" are considered two strong indicators of users' interest [2, 23]. Therefore, we devised, "Frequency" and "Duration" are valued equally. We use the harmonic mean of "Frequency" and "Duration" to represent the importance degree of a web page to a user in the session, shown as below:

$$\text{Importance}(p) = \frac{2\times Frequency(p)\times Duration(p)}{Frequency(p)+Duration(p)} \tag{5}$$

Equation (5) guarantees that "Importance" of a page is high only when "Frequency" and "Duration" are both high. Meanwhile, the value of "Importance" is normalized to be between 0 and 1.

IV. WEB RECOMMENDER SYSTEM BASED ON ACLA

In this section we present a web page recommender system which can easily adapt itself to change in website structure and content and new trend in user behavior. An asynchronous cellular learning automaton (ACLA) is used to predict user preferences and interests and make useful recommendations. The proposed algorithm employs the web usage data and underlying site structure to recommend pages to the current user.

In the proposed algorithm, we represent each web page by a cell and equip each cell with a learning automaton. Each cell in propose ACLA has n-1 neighbours and each learning automaton has n-1 actions where n is the number of web pages. The task of each learning automaton (LA) is to learn how making better recommendation. Obviously the goal of the system would be to make the most successful recommendations.

Considering the above observations, the system would only learn actions that predict the immediate next page which is not the purpose of our system. Maybe a page which was recommended in a cell will be visited in continuance of session. For this purpose we adopted the notion of N-Grams which is commonly applied in similar personalization systems based on web usage mining [4, 16, 17]. knowing only the last w page visits of the user, gives us enough information to predict his future page requests we use a sliding window of size w on user's visited pages called "visit window", resulting in considering only the last w pages requested by the user. Also we put a sliding window of size w' on recommended pages called "recommend window", allows only the last w' pages recommended by the system, have influence on rewarding or penalizing in each step. In fact, we put two arrays V and R in each cell of ACLA. V with length w is the sequence of the pages that user really visited and R with length w' is sequence of the pages that learning automaton of the corresponding cell recommended to user.

The proposed system is consisting of two phases, training and testing phase. In training phase, system will learn how to make recommendations; this is somehow different from most of the methods in which the purpose was to find explicit and static patterns or rules from the data.

In test phase, In addition to learning, the system will be evaluated. So this system is constantly in learning process. At beginning of training phase, V and R are empty.

For every user session in the log file, system begins with the first page and activates the cell corresponding to the page and its residing LA. Then learning automaton must choose one of its actions. In other word LA must recommend a page to user by choosing one of its actions with respect to its probability vector. At beginning of training phase, the probabilities of all actions are equal, so LA chooses an action randomly. But after that the probability vector change and LA chooses the action with highest probability. After choosing an action, then the page corresponds to the next page of the session and its residing LA will be activated. The recommended and visited list of previous cell is copied in the activated cell. Then the lists are shifted to left and the page which has been actually visited in user session is added to visited list of this cell. Subsequently, the page which has been recommended to user is added to recommended list.

In each activated cell, the visited and recommended list will be compared and if they have similar page, the LA of the cell which has been chosen that page will be rewarded. In other word, an action should be rewarded if it recommends a page that will be visited in one of the w' next step, of course not necessarily the immediate next step, otherwise it should be penalized.

One thickly issue worth considering is that we must reward LA for a particular action just one time because the page remain in recommended page until w' next step and if user immediately visit the page it will cause extra credit for a single move. To avoid this, we simply consider only the occurrence of the last page visited in the recommended pages list [7].

To complete our rewarding function we take into account the effective factors which were discussed in section 3. Also the rewarding and penalizing function will be affected if one of the following cases occurs:

- System recommends two pages consecutively which these two pages were visited consecutively by user. In other word the system recommend two pages which have been appeared in the path(s) followed by a web user in the session. For example, the visited page is <a,b,c> and the recommended page is <b,c, d>. In this case system had two correct recommendations consecutively and we must reward these two actions more than common case.
- Another issue is considering when the page was predicted by the system and when the user actually visited that page. According to the goal of the system this might affect on system efficiency. If we consider shortening user navigation as a sign of successful guidance of user to his required information, we should consider a greater reward for pages predicted sooner in the user's navigation path. Greater distance of page p from the end of the recommended pages list, shows that system has been predict this page sooner [7].
- If the residing LA of a cell chooses an action or in other word recommends a page which has link with the cell's corresponding page, then we should consider more reward for that action. For example, LA of the cell corresponds to page "a", choose action "b". If page a and page b have link then the action "b" in LA of cell which correspond to page "a" should receive higher reward.
- If there were a cycle in the session, we assume that the user got lost and we penalize the actions which caused the cycle more.

LA is rewarded or penalized and the probability of actions of LA is updated. This reinforcement signal help system to make better recommendations. The action probability vector of the learning automaton is updated according to $L_{R\varepsilon P}$ learning algorithm with time varying learning parameters *a* and *b* defined according to (6) and (7).

$$a = c_1 \frac{dist(p)}{w'} + c_2 \times impor\tan ce(p) \tag{6}$$

$$c_3 \frac{length(sharedpath)}{\sum length(path)} + c_4 \frac{link(p' \rightarrow p)}{In\deg ree(p)}$$

Where dist (p) is the distance of page p from the end of list R. Importance (p) is defined in section 3. Length (sharedpath) is the length of accordance in visited and recommended list (|V∩R|). $\sum length(path)$ is the sum of paths which were traversed by user. $link(p' \rightarrow p)$ is 1 if there is a link between p and p' else 0. In above equation, we consider C_1, C_2, C_3, C_4 are valued equally and sum of them is equal to 1.

$$b = \frac{1}{w'+1} * \frac{length(cycle)}{\sum length(path)} \tag{7}$$

The numerator is the sum of the lengths of the cycles and the denominator is the sum of the lengths of the paths which are navigated by the web users.

At the end of the access log, algorithm terminates its work.

V. Experimental evaluation

In this section we present a set of experiments that we performed for evaluating the impact of our proposed technique on the prediction process. Overall our experiments have verified the effective of our proposed techniques in web page recommendation.

As our evaluation data set we used the web logs of the DePaul University CTI Web server [21], based on a random sample of users visiting the site for a 2 week period during April 2002 (DePaul Web Server Data). This dataset contains 13745 distinct user sessions of length more than1and 683 distinct pages. The entries in the usage data correspond to the amount of time (in seconds) spent on pages during a given session. We split the data sets in two non-overlapping time windows to form training and a test data set. 70% of the data set (9745 sessions) was used as the training set and the remaining was used to test the system. For our evaluation we presented each user session to the system, and recorded the recommendations it made after seeing each page the user had visited.

A. Evaluation metrics

In order to evaluate the recommendation effectiveness for our method, we measured the performance of proposed method using two different standard measures, namely, Accuracy, Coverage. Recommendation accuracy and coverage are two metrics quite similar to the precision and recall metrics commonly used in information retrieval literature. Recommendation accuracy measures the ratio of correct recommendations (i.e., the proportion of relevant recommendations to the total number of recommendations), where correct recommendations are the ones that appear in the remaining of the user session. For each visit session after considering each page p the system generates a set of recommendations $R(p)$. To compute the accuracy, $R(p)$ is compared with the rest of the session $T(p)$ as follows:

$$Accuracy = \frac{T(p) \cap R(p)}{R(p)} \tag{8}$$

Recommendation coverage on the other hand shows the ratio of the pages in the user session that the system is able to predict (i.e., the proportion of relevant recommendations to all page views that should be recommended) before the user visits them:

$$Coverage = \frac{T(p) \cap R(p)}{T(p)} \tag{9}$$

B. Experimental results

We split the data sets in two non-overlapping time windows to form training and a test data set. As noted before, this system is designed so that it can learn in both training and testing phase. We evaluate our method under different settings. In all experiments we measured accuracy against coverage. In this paper, we put a sliding window called visit window with size w on user's traversed pages. In fact, we use the previous navigation path for predicting the future behavior of the current user. Also we put a window with size w' on recommended pages for considering the recommendation to w' step ahead. Size of these windows affects the efficiency of our system.

In first experiment, to consider the impact of visit window's size (w), we vary visit window's size from 1 to 4 while the size of recommendation window is constant and equal to 3. Fig. 1 shows the accuracy of the system against its coverage while the w is changed.

The results show that precision increases as a larger portion of user's history is used to generate recommendations. As our experiments show the best results are achieved when using a window of size 3. It can be inferred from this diagram that a window of size 1 (w=1) which considers only the user's last page visit does not hold enough information in memory to make the recommendation, the accuracy of recommendations improve with increasing the window size and the best results are achieved with a window size of 3 (w=3). As shown in Fig.1 using a window size larger than 3 results in weaker performance, it seems to be due to the fact that, the needs and trends of the user will be changed over time.

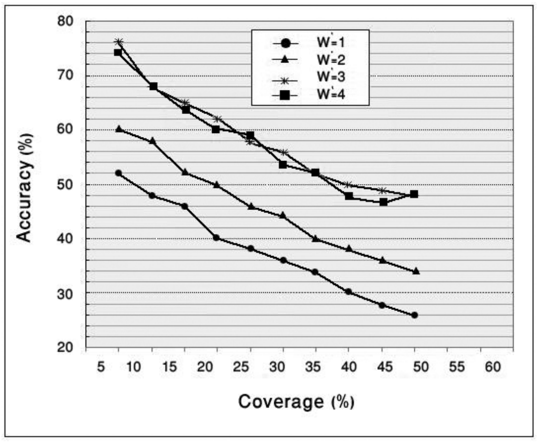

Figure 1. System performance with various visited window size

In the next experiment, similar experiment has done on recommendation window. In other word, this time we vary recommendation window' size and consider visit window's size is constant and equal to 3. Fig.2 shows the result of this experiment. The results show that accuracy increases when we consider the system recommendations of neighbor cells. But when we used recommendation window size larger than 3 results in lower accuracy, it probably occurs because the recommendations provided by the system can be useful for the user only few next step and then the information needs of the user will be changed.

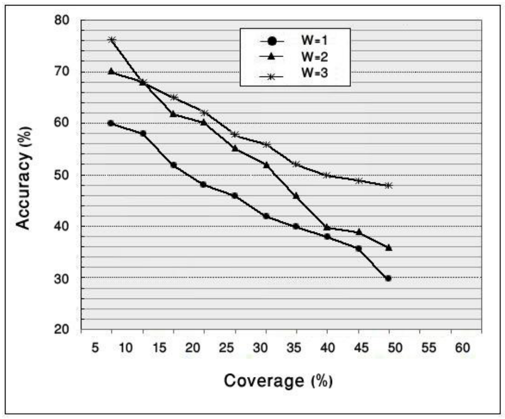

Figure 2. System performance with various recommendation window sizes

In the last part of our experiment, we consider performance of our system in comparison with association rules which is commonly known as one of the most successful approaches in web mining based recommender systems [4]. Fig.3 has shown the comparison of our system's performance with AR method in the sense of their accuracy against coverage on CTI dataset. In this experiment, we consider two window size are fix and equal to 3. As the coverage increases, naturally accuracy decreases in all systems, but our system gains much better results than the association rule algorithm. It can be seen the rate in which accuracy decreases in our algorithm is lower than traditional association rule algorithm. It seems due to the fact that the system is designed so that it can learn during the testing phase.

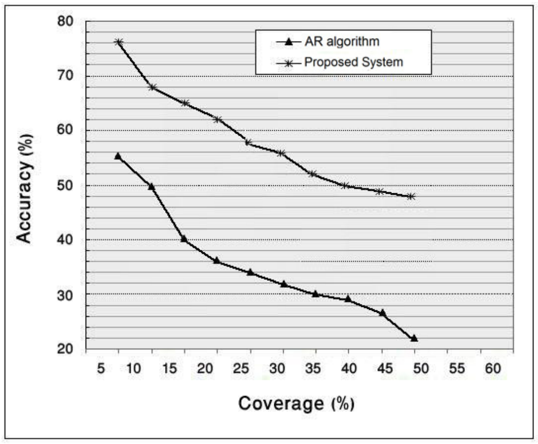

Figure 3. Comparing system performance with AR algorithm

Experimental results show that the proposed dynamic web recommender system increases coverage and accuracy significantly and our system gains much better results than the association rule algorithm. The system is dynamic which constantly interacts with the user and learns from his behavior. By using all factors which might help system to be able to make more successful recommendations and due to nature of cellular learning automata which consider the action that the neighbor cells selected, it can be concluded that our approach is capable of making web recommendation more accurately and effectively against the conventional method.

VI. Conclusion

In this paper we proposed a web recommender system based on cellular learning automata which use web usage data and structure of the web site to learn user navigation patterns and predicting user's future requests. The proposed system is a dynamic system which can easily adapt itself to change in website structure and content and new trend in user behavior. This system constantly learns the user navigation behavior, so accuracy of the system will remain high over time. Proposed system has less complexity in compared with Q-learning approach, so it is more useful for recommendation. Results of experiment show that the efficiency of this system is relatively well and Knowledge obtained from the above system, has been significantly improved the quality of recommendations.

In future work, we aim to use other data sources such as content of website and providing a combination method based on ACLA for solving new page's problems and improving the quality recommendations.

References

[1] X. Fu, J. Budzik, and K.J. Hammond, "Mining navigation history for recommendation," In *Intelligent User Interfaces*, pp.106–112, 2000.

[2] R. Forsati, and M. R. Meybodi, "Web Page Personalization using Weighted Associative Rules", *Proceedings of International Conference on Electronic Computer Conference (ICECT 2009)*, pp. 13-135, Macao, Feb. 20-22, 2009.

[3] A.L.C. Yi-Hung Wu, Y. Chen, "Enabling personalized recommendation on the web based on user interests and behaviors," In *11th International Workshop on research Issues in Data Engineering*, 2001.

[4] B. Mobasher, R. Cooley, J. Srivastava, "Automatic personalization based on web usage mining," *Communications of the ACM*. 43 (8), pp. 142-151, 2000.

[5] B. Mobasher, H. Dai, T. Luo, M. Nakagawa, "Effective personalization based on association rule discovery from web usage data," *3rd ACM Workshop on Web Information and Data Management*, 2001.

[6] R.Zaiane, J.Li, R.Hayward, "Mission-based navigational behavior modeling for web recommender system," 5th International Conference on Electronic Commerce and Web Technologies, 2006.

[7] N. Taghipour, A. Kardan, S.Shiry, "Usage-based web recommendations: a reinforcement learning approach," *Proceedings of the 1st ACM Recommender Systems Conference*. 2007.

[8] M. Mitchell, "Computation in cellular automata: A selected review," *Technical report,* Santa Fe Institute, Santa Fe, NM, USA, 1996.

[9] N.H. Packard, S. Wolfram, "Two-dimensional cellular automata," *Journal of Stat. Phys. 38*, pp. 901-946, 1985.

[10] J. Kari, "Reversibility of 2D cellular automata is undecidable," Physica D45, pp. 379-385, 1990.

[11] K. S. Narendra, M. A. L. Thathachar, "Learning automata: an introduction," *Prentice Hall*, 1989.

[12] H. Beigy, M.R. Meybodi, "A Mathematical Framework for Cellular Learning Automata," *Advances on Complex Systems*, Vol.7, Nos.3-4, pp.295-320, 2004.

[13] M.R. Meybodi, H. Beigy, M. Taherkhani, "Cellular learning automata and its applications," *Journal of Science and Technology*, No. 25, pp.54-77, 2004.

[14] H. Beigy, M.R. Meybodi, "Open synchronous cellular learning automata," *Advances in Complex Systems*, Vol. 10, No. 4, pp. 1-30, 2007.

[15] H. Beigy, M.R. Meybodi, "Asynchronous cellular learning automata", *Automatica*, Vol. 44, No. 5, 2008.

[16] B. Mobasher, H. Dai, T. Luo, Y. Sun, J. Zhu, "Integrating web usage and content mining for more effective personalization," *In EC-Web*, pp. 165–176, 2000.

[17] Z. Su, Q. Yang, Y. Lu, H. Zhang, " What next: A prediction System for Web Requests Using N-gram Sequence Models," *In Proceedings of the First International Conference on Web Information Systems and Engineering Conference*, 2000.

[18] P.K. Chan, "A non-invasive learning approach to building web user profiles", *Workshop on Web usage analysis and user profiling, 5th*

International Conference on Knowledge Discovery and Data Mining, San Diego, 1999.
[19] S. Dumais, T. Joachims, K. Bharat, A. Weigend, "Sigir 2003 workshop report: implicit measures of user interests and preferences", *ACM SIGIR Forum* ,2003.
[20] M. Morita, Y. Shinoda, " Information filtering based on user behavior analysis and best match text retrieval", *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieva*l, Springer-Verlag New York, Inc., Dublin, Ireland, pp. 272–281, 1994.
[21] CTI DePaul web server data.
[22] H. Beigy, M.R. Meybodi, "Cellular Learning Automata With Multiple Learning Automata in Each Cell and Its Applications", *IEEE Transaction on Systems, Man, and Cybernetics*, Part B: Cybernetics, Vol. 40, No. 1, pp. 54-66, 2010.
[23] R. Forsati, M.R. Meybodi, "Effective Web Page Recommendation Algorithms Based on Distributed Learning Automata and Weighted Association Rules", *Journal of Expert Systems with Applications*, Vol. 37, No. 2, pp. 1316-1330, 2010.