Physica A 482 (2017) 422–432

Contents lists available at ScienceDirect

Physica A

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

# A novel time series link prediction method: Learning automata approach

Behnaz Moradabadi, Mohammad Reza Meybodi *

*Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran*

## HIGHLIGHTS

- This paper proposes a new time series link prediction for social networks.
- The proposed method is based on learning automata.
- The proposed method uses time series information of the social network and different similarity metrics to predict future links.

## ARTICLE INFO

*Article history:*
Received 18 August 2016
Received in revised form 1 March 2017
Available online 28 April 2017

*Keywords:*
Social network
Link prediction
Time series
Learning automata

## ABSTRACT

Link prediction is a main social network challenge that uses the network structure to predict future links. The common link prediction approaches to predict hidden links use a static graph representation where a snapshot of the network is analyzed to find hidden or future links. For example, similarity metric based link predictions are a common traditional approach that calculates the similarity metric for each non-connected link and sort the links based on their similarity metrics and label the links with higher similarity scores as the future links. Because people activities in social networks are dynamic and uncertainty, and the structure of the networks changes over time, using deterministic graphs for modeling and analysis of the social network may not be appropriate. In the time-series link prediction problem, the time series link occurrences are used to predict the future links In this paper, we propose a new time series link prediction based on learning automata. In the proposed algorithm for each link that must be predicted there is one learning automaton and each learning automaton tries to predict the existence or non-existence of the corresponding link. To predict the link occurrence in time $T$, there is a chain consists of stages 1 through $T-1$ and the learning automaton passes from these stages to learn the existence or non-existence of the corresponding link. Our preliminary link prediction experiments with co-authorship and email networks have provided satisfactory results when time series link occurrences are considered.



## 1. Introduction

The predicting linkage between data objects is an interesting task in the data mining research area. For examples predicting web hyperlink creation, genetic prediction, protein–protein interactions, and the record linkage problem. In link prediction problem the data are represented with a network/graph representation. These data can be visualized as graphs, where a vertex corresponds to a person and a link represents some form of association between the corresponding persons [1,2]. The

* Corresponding author.
*E-mail addresses:* moradabadi@aut.ac.ir (B. Moradabadi), mmeybodi@aut.ac.ir (M.R. Meybodi).

http://dx.doi.org/10.1016/j.physa.2017.04.019
0378-4371/

concept of a link in a social network usually is a common interest of the corresponding social network. All link prediction methods address the following question: "Given a pair of nodes $u$ and $v$ in the current social network, how likely it is that $u$ will interact with $v$ in the future?" [1]. Link prediction can be applied in very areas. For example, in the Internet and web science applications, it can be used to find automatic web hyperlink creation [3] and website hyperlink prediction [4]. In e-commerce, link prediction can be used to produce recommender systems [5,6]. It also has various applications in other scientific disciplines: in the bibliography, for deduplication [7], record linkage [8]; in Bioinformatics, for protein–protein interaction (PPI) prediction [9], and in security, applications to identify hidden groups of terrorists and criminals [2].

Most of the previous link prediction methods have been proposed based on a static network representation, where a snapshot of the network structure is available and the goal is to predict the future links. In such a static network, each link occurrence is represented by a one-time event and all interest is only the existence of the link. For example, one may be interested to know whether a customer will purchase a product in the future or whether an author will ever collaborate with another author in the future. But in many applications, the social networks are really online, non-deterministic and unpredictable, and the structure of the network and its parameters change over time; so using deterministic social network models with fixed values for links are restrictive in solving real social network problems. In other words, link prediction methods based on the static graph representation fail when the social network has online and non-deterministic behavior. In these applications, much richer information could be extracted from the time series information of the link occurrences, such as the periodic patterns and temporal trends of the communication intensities. One of the solutions to overcome this problem is link prediction using time series information. This problem is known as the time series link prediction that is the problem of "Given link data between $x$ and $y$ for times 1 through $T$, how much is the likelihood of link occurrence between $x$ and $y$ at time $T + 1$?" [10]. In this paper, we propose a Learning Automata based Time Series Link Prediction, called LA-TSLP, to predict the link occurrences using time series information. In the proposed method each test link is modeled as a time series and predicted using an optimization tool called Learning Automaton.

Learning automaton (LA) is an adaptive decision-making unit that tries to learn the optimal action from a set of allowable actions by interacting with a random environment [11]. In each step, it selects an action from its action-set. The action selection in the LA is based on a probability distribution over the action set. The selected action is applied to the environment and then a reinforcement signal is produced by the environment. The LA updates the probability distribution of its actions according to both reinforcement signal and a learning algorithm and selects an action again. These steps are repeated until some termination criteria are reached.

In this paper, we propose a new time series link prediction method that uses a set of LAs to predict future links. This method takes advantage of using a team of LAs to predict future links using time series information. It uses one LA for each test link and each LA determines either the corresponding link should be appeared or not in time $T$ by using previous link occurrences in the network. In the proposed method, to predict a link occurrence in time $T$, the corresponding LA passes from stage 1 through stage $T - 1$ and in each stage, the action probability distribution of the LA is updated according to the reinforcement signal that is generated from the environment of the stage $t$. The environment of the stage $t$ is the set of predictions for time $t + 1$ that is calculated based on different similarity metrics using the adjacent matrix of time $t$. Because of the nature of LA, it tries to learn the optimal behavior and so the existence or non-existence of the corresponding link. The main reason for using LA in this paper is that the LA can operate and adapts to unknown environments. So, we can use multi similarity metrics as well as time information to predict future links. These steps are repeated until some termination criteria are reached. The experimental results show that the LA-TSLP is superior to some static link prediction methods such as Common Neighborhood (CN), Jacaard Index, Preferential attachment (PA) and Adamic–Adar Index (AA) in term of accuracy and performance.

The rest of the paper is organized as follows. Section 2 reviews the relevant literature on time-series link prediction problem. The learning automata are described in Section 3. Section 4 introduces the proposed time series link prediction method based on learning automata. Section 5 presents the experimental study on predicting co-authorship and email data sets. Finally, Section 6 summarizes the main conclusion of the paper.

## 2. Background and related work

### 2.1. Problem formulation

As it defined in [12], the time series link prediction problem is formally introduced as follows: Let $V$ be the list of nodes, $V = \{1, 2, \ldots, N\}$. A graph series is a list of graphs $\{G_1, G_2, \ldots, G_T\}$ corresponding to a list of adjacency matrices $(M_1, M_2, \ldots, M_T)$. Each $M_t$ is a $N \times N$ matrix with elements $M_{T(i,j)}$ corresponding to the edges in $E_{T(i,j)}$. The value of $M_{T(i,j)}$ is from the set $\{0, 1\}$ and it is the indicator of existence or non-existence of the edge $(i, j)$ during the period $t$. Then in the time series link prediction, we try to predict the existence or non-existence of the links in time $T + 1$ using previous times $M_1, M_2, \ldots, M_T$.

### 2.2. Related works

This section reviews the recent time series link prediction methods: *Tylenda et al.* proposed a new method for time-aware link prediction [13]. Their proposed method is an extension of the local probabilistic model that is introduced in [14]

by using temporal information. An empirical evaluation of the technique was performed over two collaboration networks. They showed that the link time occurrence can be considered as a main feature in the prediction result.

*Dunlavy et al.* in [10] formulated the link prediction problem as a periodic temporal link prediction and study that if the data has underlying periodic pattern, given link data for $T$ time steps, can they predict the links at time $T+1$, $T+2$, $T+3$, etc.? In the proposed method they introduced two matrix and tensor-based methods for predicting future links and aggregate the data of multi-time periods into a single matrix using a weight-based method. Then they used a CANDECOMP/PARAFAC tensor decomposition to illustrate the usefulness of using the natural three-dimensional structure of temporal link data and showed the superiority of their method on some bibliometric social networks.

*Oyama et al.* also have proposed a method called it cross-temporal link prediction which tries to predict the links in different time frames [15]. Their method is an extension of the dimension reduction method that is proposed by *Vert et al.* [16] by including a time stamp in the prediction task. They first tried to study the repeated link prediction problem. Second, to predict unobserved links, the authors proposed a cross-temporal locality preserving projection (CT-LPP) method in which data in different time frames is modeled by using low-dimensional latent feature space. Finally, they evaluated the cross-temporal link prediction to show the accuracy improvement of the proposed method.

*Soares et al.* proposed a new time series link prediction in which the topological matrices in different times are modeled as a time series data and used by a forecasting model to predict the future links [17]. So, for each link that must be predicted, the past similarity metrics is calculated as a time series data and the next value of the similarity metric is predicted using some forecasting models.

*Huang et al.* proposed a new link prediction approach using both the time-series patterns and similarity methods [12]. They built a time series data from the link occurrences in different periods and used the autoregressive integrated moving average (ARIMA) model to predict the next value of the time series data. The final prediction is calculated based on a combination of forecasting result in addition to a chosen similarity based method. Their result showed that the proposed method achieves a good performance comparing to methods that use time series models or similarity methods alone.

*Huang et al.* in [18] try to propose a weighted approach for modeling the occurrence of time and finally, in our previous research, we have proposed a new link prediction method based on temporal similarity metrics and Continuous Action set Learning Automata (CALA) [19]. The proposed method takes advantage of using different similarity metrics as well as different time periods. In the proposed algorithm, we had modeled the link prediction problem as a noisy optimization problem and used a team of CALAs to solve the noisy optimization problem. The obtained link prediction results showed satisfactory of the proposed method for some social network data sets.

## 3. Learning automata

A learning automaton [20] is an agent that can make a decision and learn how to choose the true decision by interacting with a random environment iteratively. Each LA has a probability distribution over its finite action-set and at each iteration, it selects an action based on the corresponding probability and sends the action to the random environment. The environment sends a reinforcement signal to the LA based on the evaluation of the input action. The action probability vector of the LA is updated based on the given reinforcement signal from the random environment. The goal of a learning automaton is to find the optimal action such that the average penalty received from the environment is minimized as much as possible. The environment can be described by a triple $E \equiv \{\alpha, \beta, c\}$, where $\alpha \equiv \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ represents the finite set of the inputs, $\beta \equiv \{\beta_1, \beta_2, \ldots, \beta_m\}$ denotes the set of the values that can be taken by the reinforcement signal, and $c \equiv \{c_1, c_2, \ldots, c_m\}$ denotes the set of the penalty probabilities, where the element $c_i$ is associated with the given action $\alpha_i$. If the penalty probabilities are constant, the random environment is said to be a stationary random environment, and if they vary with the time, the environment is called a non-stationary environment. The random environments based on their reinforcement signal $\beta$ can be classified into P-model, Q-model and S-model. The reinforcement signal can only take two binary values 0 and 1 in P-model environments, take one of a finite number of values in the interval [0, 1] in Q-model environment, and take a value in the interval $[a, b]$ in S-model. The relationship between the learning automaton and its random environment has been shown in Fig. 1 [20]. Learning automata can be classified into two main families [20]: fixed structure learning automata and variable structure learning automata. Variable structure learning automata are represented by a triple $\langle \beta, \alpha, T \rangle$, where $\beta$ is the set of inputs, $\alpha$ is the set of actions, and $T$ is the learning algorithm. The learning algorithm is a recurrence relation which is used to modify the action probability vector. Various learning algorithms have been proposed. But because we use P-model learning automata in the proposed algorithm we use the following learning algorithm: Let $\alpha_i(k) \in \alpha$ be the action that is selected by a learning automaton from its action-set $\alpha$ and $p(k)$ is the action probability vector at instant $k$. At each instant $k$, in linear learning algorithm the action probability vector $p(k)$ is updated by the following equation:

$$p_j(k+1) = \begin{cases} p_j(k) + a\left(1 - p_j(k)\right) & \text{if } i = j \text{ and selected action } \alpha_i(k) \text{ is rewarded} \\ (1-a)\, p_j(k) & \text{if } i \neq j \text{ and selected action } \alpha_i(k) \text{ is rewarded} \\ 1 - b p_j(k) & \text{if } i = j \text{ and selected action } \alpha_i(k) \text{ is penalized} \\ \dfrac{b}{r-1} + (1-b)\left(1 - p_j(k)\right) & \text{if } i = j \text{ and selected action } \alpha_i(k) \text{ is penalized} \end{cases} \tag{1}$$

where $a$ and $b$ are reward parameter and penalty parameter, respectively. And also $r$ denotes the number of actions that can be taken by LA. The parameters $a$ and $b$ indicates the amount of increase and decrease of the action probability values,

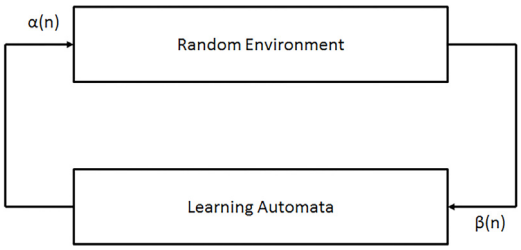


**Fig. 1.** The relationship between a learning automata and its random environment.

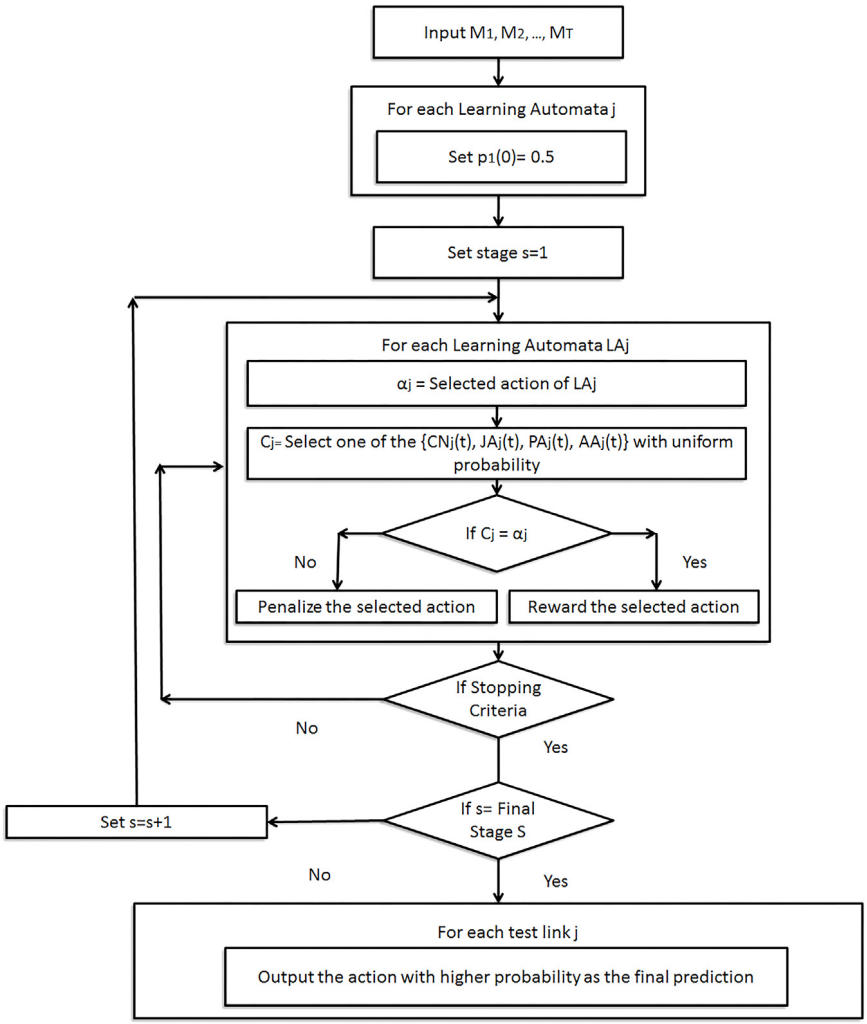


**Fig. 2.** The pseudo code of the proposed time series link prediction.

respectively. If $a = b$, the above recurrence equation is called linear reward–penalty ($LR$-$P$) algorithm; if $a \gg b$ the presented equation is called linear reward-$\varepsilon$penalty ($LR$-$\varepsilon P$) algorithm; and finally if $b = 0$, it is called linear reward-Inaction ($LR$-$I$) algorithm. In the proposed link prediction we use the $LR$-$P$ learning automata as the learning model of the method.

Learning automata have been found to be useful in different areas such as optimization tasks [21,22], capacity assignment problems [23–25], graph problems [26–30] artificial intelligent [31–34], social area [35–39] and other applications [40–43].

## 4. The proposed time series link prediction approach

In this section, a new Learning Automata based Time Series Link Prediction (LA-TSLP) that takes advantage of using time information in predicting future links is proposed. In the proposed method there is one learning automaton for each test

link in time $T + 1$ and the LA-TSLP uses the network structures of time 1 through $T$ sequentially to predict future links in time $T + 1$. The network structure of time $t$ is called one stage. In each stage $t$, each learning automaton in LA-TSLP tries to predict the link occurrence of the corresponding link for time $t + 1$ using the environment $t$. After finishing the prediction task in stage $t$, the LA-TSLP goes to the next stage. In the next stage, each learning automaton tries to update and improve its estimation using the new environment $t + 1$ to predict the link occurrence of the corresponding link for the time $t + 2$ and so on. Generally, LA-TSLP has two main phases: (1) Learning Phase: an iterative phase to learn future links and (2) Prediction Phase: a phase to generate the final prediction results. The learning phase in each stage $t$ also has two steps: (1) Select the action of each learning automaton and use it as the indicator of existence or non-existence of the corresponding link, (2) update the action probability distribution of each learning automaton based on a reinforcement signal the is produced from the environment $t$. In stage $t$, for action selection phase, each learning automaton determines whether the corresponding link appears in time $t + 1$ or not. This action selection is done based on the internal action probability distribution of the learning automaton. Because there is no prior knowledge about the existence of the link, for each learning automaton we set the initial probability of choosing action 1 (link existence) to 0.5, $p_1 = 0.5$. After choosing the actions of LAs, a reinforcement signal is produced for each learning automaton based on the effectiveness of the selected action in the environment $t$. The environment of stage $t$ is a set of predictions for time $t + 1$ that is calculated based on different similarity metrics using the adjacent matrix of time $t$. So each learning automaton uses time series information of the time $t$ to generate prediction for the next time $t + 1$. Finally each learning automaton updates its action probability distribution based on the received reinforcement signal and a learning algorithm. In the next iteration, each learning automaton selects a new action again based on the new action probability distribution and this procedure repeats until some stop criteria is satisfied. Then the LP-TSLP goes to the stage $t + 1$ and start to learn the links existence using the current action probability distributions using the new environment $t + 1$.

In the prediction phase of LA-TSLP, for each test link the action with higher probability of the corresponding learning automaton is used as the prediction result. The overall process of the LA-TSLP can be seen in Fig. 2. The proposed method takes advantage of using time series information to predict future links. Also, the pseudo code of the LA-TSLP can be seen in the Algorithm 1. In the following sections, we describe the two main phases of the proposed method in detail.

Algorithm 1: Pseudo code of the proposed link prediction (LA-TSLP)

**Let** $M_1, M_2, \ldots, M_T$ be the adjacent matrix for time 1 through T.
**Let** $K$ and $T_{min}$ be the maximum number of iterations and the entropy threshold, respectively.
**Let** $k$, $s$ be the iteration counter and stage counter and initially set to 0 and **let** S be the total number total number of stages.
**Let** $E_t$={$CN_t$, $JA_t$, $PA_t$, $AA_t$} be the environment for stage t.
**Let** $a, b$ be the reward and punishment rate to update learning automaton in the learning algorithm.
**Set** LAs be the set of learning automata indexes with 1 through J, one learning automaton for each link that must be predicted.
**Set** the initial probability distribution of choosing action 1 for each learning automaton j to be $p_1(j) = 0.5$.
**While** $s<S$ **do**
  **While** $k<K$ **or** $Entropy<T_{min}$ **do**
    Select action j of each learning automaton $LA_j$ based on the action probability distribution of the learning automaton, p(j).
    Calculate the reinforcement signal $\beta_k(j)$ based on the environment of the stage $s$ using equation (2).
    Update the probability distribution function of each learning automata $LA_j$, $p(j)$, based on the reinforcement signal $\beta_k(j)$ and the learning algorithm in equation 1.
    **Set** $k=k+1$
  **End While**
  **Set** $s=s+1$
**End While**
**For** each test link j
  **Set** $O_j$ as the output of prediction result based on equation (4)

### 4.1. Learning phase

This subsection presents the learning phase of the LA-TSLP. In this phase, there is one learning automaton for each test link that must be predicted. On the other hand, there is a set of stages $\{S_1, S_2, \ldots, S_T\}$, one stage for each time $t$. To do this, each learning automaton starts its learning in stage 1. In this stage each learning automaton $LA_j$ tries to learn the existence or non-existence of the corresponding link for time 2 with the reinforcement signal that is produced from the environment $E_1$. This prediction repeats until some termination criteria are satisfied. Then the learning automaton $LA_j$ goes to stage 2. In

stage 2, each learning automaton $LA_j$ tries to update and improve its actions using the new environment $E_2$ to predict the occurrence or non-occurrence of the corresponding link for time 3. The environment of stage $S_t$, $E_t$, is the set of predictions for time $t+1$ that is calculated based on different similarity metrics using the adjacent matrix of time $t$. So in each stage $t$, each learning automaton $LA_j$ uses different similarity metrics of time $t$ for link $j$ to learn the existence or non-existence of the corresponding link. These steps are repeated until the learning automaton goes to time $T$ and tries to learn the prediction for time $T+1$.

To do this, the action set of each learning automaton $LA_j$ is a set of actions $\{\alpha_0, \alpha_1\}$ which determines the existence or non-existence of the corresponding link, respectively and each learning automaton $LA_j$ has a probability distribution $p_0, p_1$ that determines the probability of choosing action 0 and 1, respectively. Because there is no prior knowledge about the existence of the link, for each learning automaton we set the initial probability of choosing action 1 (link existence) to 0.5, $p_1 = 0.5$. So, in the action selection step of the LA-TSLP, each learning automaton chooses action 1 and action 0 based on $p_1$ and $1 - p_1$ as $\alpha_i$, respectively. The action $\alpha_i$ is used as the prediction of corresponding link in time $t+1$. After all LAs choose their actions, these actions are evaluated based on environment $t$ and each learning automaton updates its probability distribution based on the reinforcement signal.

Now, to generate a reinforcement signal in stage $t$, we define the environment of stage $t$ as a set of predictions for time $t+1$ that is calculated based on different similarity metrics using the adjacent matrix of time $t$. In this paper, we consider as Common Neighborhood (CN), Jacaard Index (JA), Preferential attachment (PA) and Adamic–Adar Index (AA) as similarity based methods. So the environment of the stage $t$ is defined as $E_t = \{CN_t, JA_t, PA_t, AA_t\}$ that is the Common Neighborhood, Jacaard Index, Preferential attachment, and Adamic–Adar prediction results for time $t$. Then we select one element from $E_t(j)$ as $C_t(j)$ using the uniform probability and generate the reinforcement signal $\beta_t(j)$ based on the following equation:

$$\beta_t(j) = \begin{cases} 1 & C_t(j) \neq \alpha_1(j) \\ 0 & \text{else.} \end{cases} \tag{2}$$

Then each learning automaton $LA_j$ updates its action probability distribution using the reinforcement signal $\beta_t(j)$ and the $L_{R-P}$ learning algorithm using Eq. (1).

From Eq. (1) it follows that if action 1 is attempted in iteration $k$ the probability distribution $p_1(k)$ is increased in iteration $k$ for a favorable response and decreased for an unfavorable response. Also, if action 0 is attempted in iteration $k$ the probability distribution $p_1(k)$ is decreased in iteration $k$ for a favorable response and increased for an unfavorable response.

In the next iteration, each learning automaton selects a new action again based on the new action probability distribution and this procedure repeats until some stop criteria are satisfied. In the proposed algorithm the learning phase in each stage is repeated until the average of entropy of probability vector of learning automata reaches a predefined value $T_{\min}$ or the maximum number of iteration, $K$, is reached to a threshold $k$. The information entropy of a learning automaton with $r$ actions can be defined as follows [44]:

$$H = -\sum_{i}^{r} p_i \cdot \log(p_i) \tag{3}$$

where $p$, is the probability of choosing $i$th action of a learning automaton. The entropy for a learning automaton has maximum value of one when all the actions have equal probabilities of choosing and has minimum value of zero when the action probability vector is a unit vector. After the stopping criteria is reached, then the learning automaton goes to the next stage and these procedures repeated for $t$ next stages. After stopping in stage $T$, to predict the links for time $T+1$ we use the following prediction phase.

### 4.2. Prediction phase

After running learning phase, the prediction phase generates the final link prediction for the proposed method. To do this, for each test link $j$ in time $T+1$, the final prediction is defined as $O_j$, and calculated based on the following rule:

$$O_j = \begin{cases} \alpha_1 & p_1 > p_0 \\ \alpha_0 & \text{else.} \end{cases} \tag{4}$$

That means for each LA the action with higher probability is chosen as the final prediction result for the corresponding link. Finally, the LA-TSLP outputs the set $O$ values as the output of proposed link prediction.

## 5. Experiment results

In this section, in order to evaluate the performance of the proposed algorithm, some computer experiments have been conducted and the proposed algorithm has been evaluated in term of performance and accuracy. In the proposed method for stopping criteria we used the following parameters: $T_{\min} = 0.3$, $k = 2000$. Also for the learning automata parameters we used $a = 0.05$, $b = 0.01$. It also should be noted that these parameters are obtained based on empirical experiments and all the results reported in following experiments are based on the averages taken over 30 runs in a PC, which has a

**Table 1**
Network size in terms of nodes and edges.

| Data set | Nodes | Edges | Description |
|---|---|---|---|
| Hep-th | 9877 | 51,971 | Collaboration network of Arxiv High Energy Physics Theory |
| Hep-ph | 12,008 | 237,010 | Collaboration network of Arxiv High Energy Physics |
| Astro-ph | 18,772 | 396,160 | Collaboration network of Arxiv Astro Physics |
| Email-Enron | 36,692 | 0.7328 | Email communication network from Enron |
| Email-EuAll | 26,521 | 42,004 | Email network from the EU research institution |

single CPU of Intel(R) Core(TM)2 Duo 3.33 GHz and a 6 GB memory. In the rest of this section, we first give the data sets and the evaluation metric we used in our experiments and then give a set of two experiments. In the first experiment, the performance of the proposed algorithm compared with the performance of some static link prediction methods and in the second experiment, the accuracy and convergence rate of the LP-TSLA is computed and reported.

### 5.1. Data sets

In this section, the social network's data used in our experiments are described. For the experiments developed in this work, we consider the following two groups of networks:

(1) Co-authorship networks: A type of social network where the nodes represent the authors and two authors are connected if they have collaborated in a paper. Collaboration network is widely used to understand the topology and dynamics of complex networks. In this paper, we have adopted three co-authorship networks from three sections of Arxiv[1] and extracted data from the years 1993 to 2003 for all these data sets. The first network is composed by authors that collaborated in theoretical high energy physics[2] (hep-th). The second one is formed by authors who published papers in the high energy physics[3] (hep-ph) and the third one is sampled from collaboration in the Astro Physics[4] (Astro-ph). In these data sets, if an author $i$ co-authored a paper with author $j$, the graph contains an undirected edge from $i$ to $j$. If the paper is co-authored by $k$ authors this generates a completely connected (sub) graph on $k$ nodes.
(2) Email communication networks: A type of social network where the nodes of the network are email addresses and if an address $i$ sent at least one email to address $j$, the graph contains an undirected edge from $i$ to $j$. In our experiment, we use two email communication data sets: Enron email communication network[5] and Eu-All email communication network.[6] The Enron email communication network includes a data set around half million emails that are public by the Federal Energy Regulatory Commission and we extract data from May 1999 through May 2002 (36 months). Also, the Eu-All email communication network was extracted using email data from a large European research institution and we extracted data from October 2003 to May 2005 (18 months). The network specification of each data set is presented in Table 1. Since these networks are highly sparse, to make computation feasible, we reduce the number of candidate pairs by choosing only the ones that have at least two connections on the network.

In order to do experiments for collaboration networks (Hep-th, Hep-ph, and Astro-ph), we consider the data from 1993 to 2002 as the training data (each year as a time period) and year 2003 as test data. Also for email networks (Enron and EuAll), we consider the first 70% available months as the training data (each month as a time period) and the 30% remaining months as the test data.

### 5.2. Static link prediction

The baseline methods we use them as the environment are briefly described in this sub-section. Let $\Gamma(x)$ denote neighbors of the node $x$:

1. Common Neighborhood (CN) [45,46]: In this measure, two nodes, $x$ and $y$, are more likely to have a link if they have many common neighbors. This score is defined as

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)| \tag{5}$$

where $\Gamma(x)$ is the neighbors of the node $x$.

1 http://www.arxiv.org.
2 http://arxiv.org/archive/hep-th.
3 http://arxiv.org/archive/hep-ph.
4 http://arxiv.org/archive/Astro-ph.
5 http://www.cs.cmu.edu/~enron/.
6 http://snap.stanford.edu/data/email-EuAll.html.

**Table 2**
Comparison of the LA-TSLP with similarity-based methods on Hep-th, Hep-ph, and Astro-ph data sets.

| Method | CN | | | JI | | | PA | | | AA | | | LA-TSLP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Year | Hep-th | Hep-ph | Astro-ph | Hep-th | Hep-ph | Astro-ph | Hep-th | Hep-ph | Astro-ph | Hep-th | Hep-ph | Astro-ph | Hep-th | Hep-ph | Astro-ph |
| 1996 | **0.7982** | 0.7441 | **0.7189** | 0.6091 | 0.5941 | 0.5198 | 0.6140 | 0.5841 | 0.5252 | 0.7042 | 0.6895 | 0.6246 | 0.7212 | **0.7591** | 0.7046 |
| 1997 | 0.8000 | 0.7955 | **0.7325** | 0.6101 | 0.6033 | 0.5247 | 0.6199 | 0.6041 | 0.5299 | 0.7284 | 0.7151 | 0.6478 | **0.8298** | **0.8023** | 0.7294 |
| 1998 | 0.7981 | 0.7904 | 0.7412 | 0.6210 | 0.6074 | 0.5395 | 0.6158 | 0.5999 | 0.5341 | 0.7594 | 0.7395 | 0.6501 | **0.8341** | **0.8012** | **0.7586** |
| 1999 | 0.8123 | 0.7876 | 0.7402 | 0.6390 | 0.6147 | 0.5498 | 0.6269 | 0.6014 | 0.5417 | 0.7710 | 0.7541 | 0.6516 | **0.8547** | **0.8194** | **0.7716** |
| 2000 | 0.7811 | 0.7512 | 0.7495 | 0.6401 | 0.6218 | 0.5510 | 0.6317 | 0.6118 | 0.5591 | 0.7794 | 0.7612 | 0.6593 | **0.8641** | **0.8231** | **0.7850** |
| 2001 | 0.7790 | 0.7458 | 0.7514 | 0.6501 | 0.6301 | 0.5546 | 0.6388 | 0.6128 | 0.5603 | 0.7901 | 0.7717 | 0.6712 | **0.8514** | **0.8274** | **0.7892** |
| 2002 | 0.8041 | 0.7798 | 0.7617 | 0.6445 | 0.6357 | 0.5573 | 0.6321 | 0.6202 | 0.5651 | 0.7935 | 0.7810 | 0.6679 | **0.8759** | **0.8492** | **0.7924** |
| 2003 | 0.7945 | 0.7652 | 0.7794 | 0.6438 | 0.6398 | 0.5612 | 0.6400 | 0.6299 | 0.5678 | 0.7962 | 0.7864 | 0.6699 | **0.8947** | **0.8513** | **0.7973** |

2. Jaccard Index (JI) [47]: This index was proposed by Jaccard over a hundred year ago, and is defined as:

$$Jaccard\,(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}. \tag{6}$$

3. Preferential Attachment (PA) [48,49]: The preferential attachment (PA) algorithm is motivated by the preferential attachment phenomena [50] discovered in a variety of realworld complex systems. Under this algorithm, the link occurrence score is set to be the product of the degrees of the involved nodes and is defined as follows:

$$PA\,(x, y) = |\Gamma\,(x)\,| \times |\Gamma\,(y)\,|. \tag{7}$$

4. Adamic–Adar Index (AA) [51]: This index refines the simple counting of common neighbors by assigning the less-connected neighbors more weight and is defined as:

$$AA\,(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log(\Gamma\,(z))}. \tag{8}$$

### 5.3. Evaluation metric

This section introduces the AUC metric that we use it in our experiments: If we rank the entire non-existent links according to their scores, the AUC metric can be interpreted as the probability that a random missing link has a higher score than a random non-existent link [52]. In the algorithmic implementation, at each time we usually pick a missing link and a nonexistent link in a random fashion and compare their scores. If among $n$ independent comparisons there are $n'$ times when missing links have a higher score and $n''$ times when they have the same score, the AUC value is:

$$\text{AUC} = \frac{n' + 0.5n''}{n}. \tag{9}$$

If the AUC has a value greater than 0.5, it is better than the random link prediction algorithm; and the farther from 0.5, the more accurate the algorithm.

### 5.4. Link prediction comparison

#### 5.4.1. Co-authorship networks

In this experiment, we compare the LA-TSLP with some similarity based methods using three co-authorship networks. To do this, we consider 10 years from 1993 to 2002, one stage for each year and we predict the test links for the years 1996 through 2003. To predict a test link in year $y$ we use a learning automaton consist of stages 1993 through $y - 1$ and the link prediction results of the proposed algorithm (LP-TSLA), for years 1996 through 2003 is compared to the some similarity based link predictions using AUC metric. The obtained results of the proposed algorithm and its comparison with other methods for Hep-th, Hep-ph and Astro-ph data sets are reported in Table 2. Also, it should be mentioned that for the following tables given in this section, the best results are highlighted. The results reported here demonstrate that the proposed time series link prediction method is able to achieve an average better AUC measure than the CN, Jaccard Index, PA, and AA. The results suggest that the prediction is better in term of accuracy that is due to that the proposed algorithm uses the time series information to predict future links.

#### 5.4.2. Email networks

In this experiment, we compare the LA-TSLP with some similarity based methods using two email social networks. To do this, we consider one stage of each training month and predict the links of the test months and compare it with some similarity based link predictions using AUC metric. The obtained results of the proposed algorithm and its comparison with

**Table 3**
Comparison of the LA-TSLP with similarity-based methods on Enron and EuAll data sets.

| Data set | Method | |
|---|---|---|
| | Enron | EuAll |
| CN | 0.8214 | 0.7518 |
| JI | 0.6791 | 0.6217 |
| PA | 0.6914 | 0.6571 |
| AA | 0.7724 | 0.7328 |
| LA-TSLP | **0.8320** | **0.8019** |

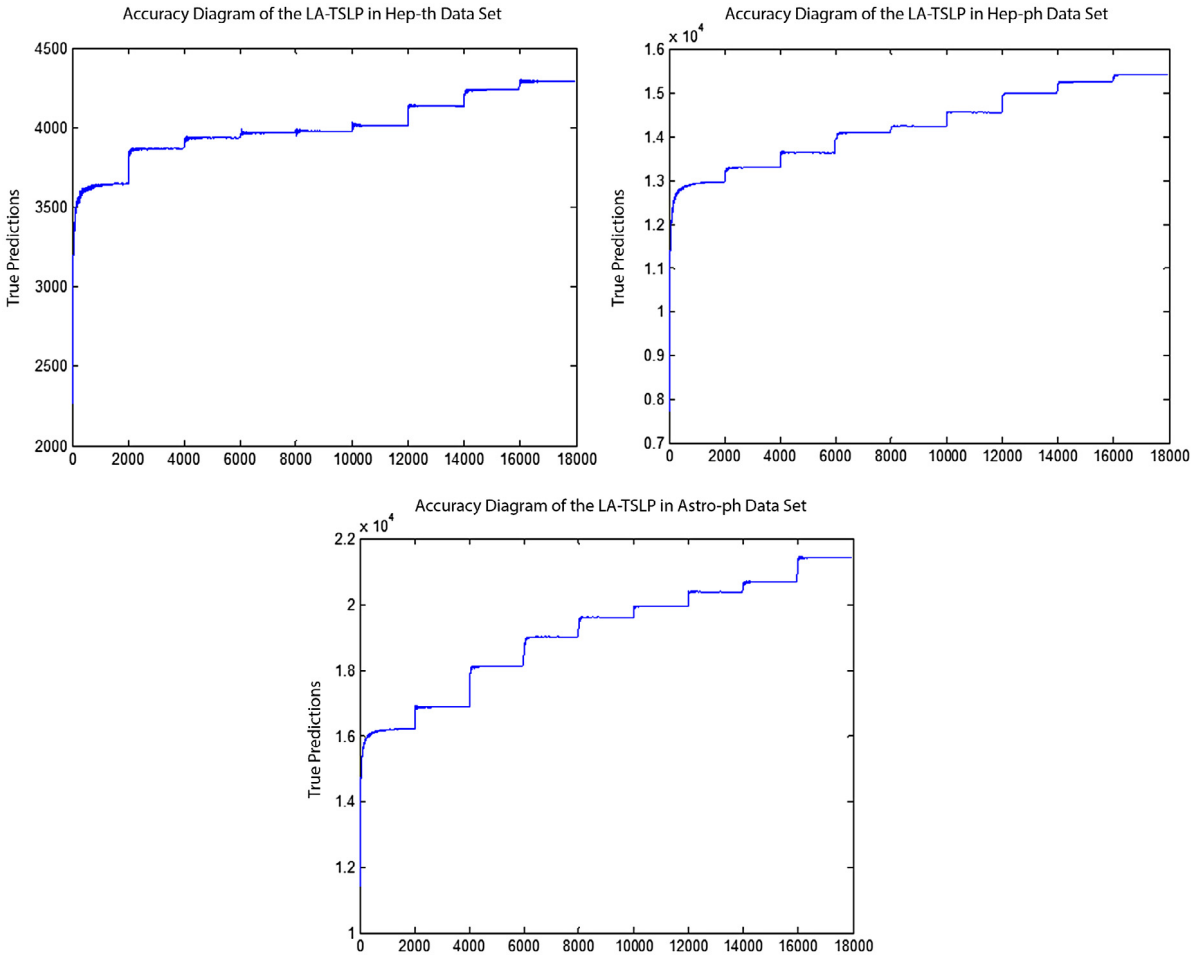


**Fig. 3.** The accuracy diagram of the LA-TSLP for Hep-th, Hep ph, and Astro-ph data sets.

other methods for Email-Enron and Email-EuAll data sets are reported in Table 3. The results reported here also show that the proposed time series link prediction method is able to achieve an average better AUC measure than the CN, Jaccard Index, PA, and AA.

### 5.5. Running diagram of the LA-TSLP

In this experiment to show the progressive behavior of the LA-TSLP, the accuracy diagram of the proposed algorithm for data set Hep-th, Hep-ph, and Astro-ph are presented in Fig. 3. As it shown in this figure, at the start of each stage the LA-TSLP operates more stochastic and worse that is due to the operating in a new environment, but after little iterations, it learns the new pattern that is the aggregation of the previously learned patterns in addition to the new environment pattern. Also in order to evaluate the convergence rate of the LAs, we have calculated the information entropy of the LAs and reported it in Table 4. From this report, we can conclude that the LA-TSLP is finished in each stage with a good convergence rate.

**Table 4**
The information entropy of the learning automata through the learning process.

| Year | Data set | | |
|---|---|---|---|
| | Hep-th | Hep-ph | Astro-ph |
| 1996 | 0.0311 | 0.0412 | 0.0591 |
| 1997 | 0.0342 | 0.0517 | 0.0541 |
| 1998 | 0.0418 | 0.0541 | 0.0593 |
| 1999 | 0.0484 | 0.0678 | 0.0571 |
| 2000 | 0.0396 | 0.0612 | 0.0695 |
| 2001 | 0.0437 | 0.0632 | 0.0612 |
| 2002 | 0.0417 | 0.0657 | 0.0713 |
| 2003 | 0.0455 | 0.0682 | 0.0712 |

## 6. Conclusion

This paper presents a new time series link prediction method which uses learning automata to predict the existence or non-existence of each link in time $T + 1$ by using the network structure from time 1 to $T$. In the proposed method, for each link that must be predicted, there is one learning automaton and each learning automaton tries to learn the existence or not existence of the corresponding link in time $T + 1$. The proposed algorithm has some stages and each stage corresponds to one time period. In each stage $t$ of the proposed algorithm, each learning automaton tries to predict the link occurrence of the next time $t+1$ by using similarity metrics of time $t$. After finishing the prediction task in time $t$, the estimated prediction goes to the next stage. In the next stage, each learning automaton tries to update and improve its estimation using a new environment $t + 1$ to predict the link occurrence in time $t + 2$ and so on. The experimental results reported here show that the proposed algorithm is superior to other static algorithms which consider only one snapshot of the network. The better result can be due to the learning capability of learning automata that shows the occurrence of the link in the network evolves through time and using different similarity metrics at different times can be used for future predictions.

## References

[1] Linyuan Lü, Tao Zhou, Link prediction in complex networks: A survey, Physica A 390 (6) (2011) 1150–1170.
[2] Mohammad Al Hasan, Mohammed J. Zaki, A survey of link prediction in social networks, in: Social Network Data Analytics, Springer US, 2011, pp. 243–275.
[3] Sisay Fissaha Adafre, Maarten de Rijke, Discovering missing links in Wikipedia, in: Proceedings of the 3rd International Workshop on Link Discovery, ACM, 2005.
[4] Jianhan Zhu, Jun Hong, John G. Hughes, Using Markov models for web site link prediction, in: Proceedings of the Thirteenth ACM Conference on Hypertext and Hypermedia, ACM, 2002.
[5] Xin Li, Hsinchun Chen, Recommendation as link prediction: a graph kernel-based machine learning approach, in: Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries, ACM, 2009.
[6] Zan Huang, Xin Li, Hsinchun Chen, Link prediction approach to collaborative filtering, in: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries, ACM, 2005.
[7] Bradley Malin, Edoardo Airoldi, Kathleen M. Carley, A network analysis model for disambiguation of names in lists, Comput. Math. Organ. Theory 11 (2) (2005) 119–139.
[8] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, Vassilios S. Verykios, Duplicate record detection: A survey, IEEE Trans. Knowl. Data Eng. 19 (1) (2007) 1–16.
[9] Valerio Freschi, A graph-based semi-supervised algorithm for protein function prediction from interaction maps, in: International Conference on Learning and Intelligent Optimization, Springer, Berlin, Heidelberg, 2009.
[10] Daniel M. Dunlavy, Tamara G. Kolda, Evrim Acar, Temporal link prediction using matrix and tensor factorizations, ACM Trans. Knowl. Discov. Data (TKDD) 5 (2) (2011) 10.
[11] Mandayam A.L. Thathachar, P. Shanti Sastry, Varieties of learning automata: an overview, IEEE Trans. Syst. Man Cybern. Part B Cybern. 32 (6) (2002) 711–722.
[12] Zan Huang, Dennis K.J. Lin, The time-series link prediction problem with applications in communication surveillance, INFORMS J. Comput. 21 (2) (2009) 286–303.
[13] Tomasz Tylenda, Ralitsa Angelova, Srikanta Bedathur, Towards time-aware link prediction in evolving social networks, in: Proceedings of the 3rd Workshop on Social Network Mining and Analysis, ACM, 2009.
[14] Chao Wang, Venu Satuluri, Srinivasan Parthasarathy, Local probabilistic models for link prediction, in: Seventh IEEE International Conference on Data Mining, 2007, ICDM 2007, IEEE, 2007.
[15] Satoshi Oyama, Kohei Hayashi, Hisashi Kashima, Cross-temporal link prediction, in: 2011 IEEE 11th International Conference on Data Mining, (ICDM), IEEE, 2011.
[16] Jean-Philippe Vert, Yoshihiro Yamanishi, Supervised graph inference, in: Advances in Neural Information Processing Systems, 2004.
[17] da Silva Soares, Ricardo Paulo, R. Bastos Cavalcante Prudencio, Time series based link prediction, in: The 2012 International Joint Conference on Neural Networks, (IJCNN), IEEE, 2012.
[18] S. Huang, Y. Tang, F. Tang, J. Li, Link prediction based on time-varied weight in co-authorship network, in: Computer Supported Cooperative Work in Design, CSCWD, Proceedings of the 2014 IEEE 18th International Conference on, 2014, pp. 706–709.
[19] B. Moradabadi, M.R. Meybodi, Link prediction based on temporal similarity metrics using continuous action set learning automata, Physica A 460 (2016) 361–373.
[20] Mandayam A.L. Thathachar, Pidaparty S. Sastry, Networks of Learning Automata: Techniques for Online Stochastic Optimization, Springer, 2003.
[21] G. Santharam, P.S. Sastry, M.A.L. Thathachar, Continuous action set learning automata for stochastic optimization, J. Franklin Inst. B 331 (5) (1994) 607–628.
[22] Hamid Beigy, M.R. Meybodi, A new continuous action-set learning automaton for function optimization, J. Franklin Inst. 343 (1) (2006) 27–47.
[23] B. John Oommen, T. Dale Roberts, Continuous learning automata solutions to the capacity assignment problem, IEEE Trans. Comput. 49 (6) (2000) 608–620.

[24] Mohsen Jahanshahi, Mehdi Dehghan, Mohammad Reza Meybodi, On channel assignment and multicast routing in multi–channel multi–radio wireless mesh networks, Int. J. Ad Hoc Ubiquitous Comput. 12 (4) (2013) 225–244.
[25] Mohsen Jahanshahi, Mehdi Dehghan, Mohammad Reza Meybodi, A mathematical formulation for joint channel assignment and multicast routing in multi-channel multi-radio wireless mesh networks, J. Netw. Comput. Appl. 34 (6) (2011) 18.
[26] S.M. Vahidipour, M.R. Meybodi, M. Esnaashari, Adaptive Petri net based on irregular cellular learning automata with an application to vertex coloring problem, Appl. Intell. (2016) 1–13.
[27] M. Vahidipour, M.R. Meybodi, M. Esnaashari, Finding the shortest path in stochastic graphs using learning automata and adaptive stochastic petri nets, Internat. J. Uncertain. Fuzziness Knowledge-Based Systems (2016).
[28] Javad Akbari Torkestani, Mohammad Reza Meybodi, Learning automata-based algorithms for finding minimum weakly connected dominating set in stochastic graphs, Internat. J. Uncertain. Fuzziness Knowledge-Based Systems 18 (06) (2010).
[29] A. Rezvanian, M.R. Meybodi, Stochastic graph as a model for social networks, Comput. Hum. Behav. 64 (2016) 621–640.
[30] A. Rezvanian, M.R. Meybodi, Finding minimum vertex covering in stochastic graphs: a learning automata approach, Cybern. Syst. 46 (8) (2015) 698–727.
[31] J. Kazemi Kordestani, A. Ahmadi, M.R. Meybodi, An improved differential evolution algorithm using learning automata and population topologies, Appl. Intell. (2014) http://dx.doi.org/10.1007/s10489-014-0577-2.
[32] D. Yazdani, B. Nasiri, A. Sepas-Moghadam, M.R. Meybodi, M.R. Akbarzadeh-Totonchi, mNAFSA: A novel approach for optimization in dynamic environments with global changes, in: Swarm and Evolutionary Computation, Elsevier, 2014, http://dx.doi.org/10.1016/j.swevo.2.
[33] A. Alizadegan, M.R. Meybodi, B. Asady, A novel hybrid artificial bee colony algorithm and differential evolution for unconstrained optimization problems, Adv. Comput. Sci. Eng. 8 (1) (2012).
[34] M. Mahdaviani, J.K. Kordestani, A. Rezvanian, M.R. Meybodi, LADE: Learning automata based differential evolution, Int. J. Artif. Intell. Tools 24 (06) (2015) 1550023.
[35] Mohammad Soleimani-Pouri, Alireza Rezvanian, Mohammad Reza Meybodi, An ant based particle swarm optimization algorithm for maximum clique problem in social networks, in: State of the Art Applications of Social Network Analysis, Springer. International Publishing, 2014, pp. 295–304. s.n..
[36] Mohammad Soleimani-Pouri, Alireza Rezvanian, Mohammad Reza Meybodi, Finding a maximum clique using ant colony optimization and particle swarm optimization in social networks, in: Proceedings of the 2012 International Conference on Advances in Social. Network Analysis and Mining, (ASONAM 2012), IEEE Computer Society, 2012, s.n..
[37] Alireza Rezvanian, Mohammad Rahmati, Mohammad Reza Meybodi, Sampling from complex networks using distributed learning automata, Physica A 396 (2014) 224–234.
[38] A. Rezvanian, M.R. Meybodi, Sampling algorithms for weighted networks, Soc. Netw. Anal. Min. 6 (1) (2016) 60.
[39] M.M.D. Khomami, A. Rezvanian, M.R. Meybodi, Distributed learning automata-based algorithm for community detection in complex networks, Internat. J. Modern Phys. B 30 (8) (2016) 1650042.
[40] S.M. Vahidipour, M.R. Meybodi, M. Esnaashari, Learning automata-based adaptive petri net and its application to priority assignment in queuing systems with unknown parameters, IEEE Trans. Syst. Man Cybern. Syst. 45 (1) (2015).
[41] H. Mostafaei, M. Esnaashari, M.R. Meybodi, A coverage monitoring algorithm based on learning automata for wireless sensor networks, Int. J. Appl. Math. Inf. Sci. (2014).
[42] S.M. Safavi, M.R. Meybodi, M. Esnaashari, Learning automata based face-aware mobicast, in: Wireless Personal Communications, Springer, 2014.
[43] A. Safari Mamaghani, K. Asghari, M.R. Meybodi, Designing a new structure based on learning automata to improve evolutionary algorithms (with considering some cased study problems), J. Adv. Comput. Res. 4 (3) (2013) 1–24.
[44] A. Mousavian, A. Rezvanian, M.R. Meybodi, Cellular learning automata based algorithm for solving minimum vertex cover problem, in: 2014 22nd Iranian Conference on Electrical Engineering, ICEE, 2014, pp. 996–1000.
[45] Jilin Chen, et al., Make new friends, but keep the old: recommending people on social networking sites, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 2009.
[46] David Liben-Nowell, Jon Kleinberg, The link-prediction problem for social networks, J. Am. Soc. Inf. Sci. Technol. 58 (7) (2007) 1019–1031.
[47] Gobinda Chowdhury, Introduction to Modern Information Retrieval, Facet Publishing, 2010.
[48] Albert-László Barabási, Réka Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[49] Yan-Bo Xie, Tao Zhou, Bing-Hong Wang, Scale-free networks without growth, Physica A 387 (7) (2008) 1683–1688.
[50] Mark E.J. Newman, Clustering and preferential attachment in growing networks, Phys. Rev. E 64 (2) (2001) 025102.
[51] Lada A. Adamic, Eytan Adar, Friends and neighbors on the web, Soc. Networks 25 (3) (2003) 211–230.
[52] L. Lü, T. Zhou, Link prediction in complex networks: A survey, Physica A 390 (6) (2011) 1150–1170.