Journal of Experimental & Theoretical Artificial Intelligence

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/teta20

New Cellular Learning Automata as a framework for online link prediction problem

Mozhdeh Khaksar Manshad, Mohammad Reza Meybodi & Afshin Salajegheh

To cite this article: Mozhdeh Khaksar Manshad, Mohammad Reza Meybodi & Afshin Salajegheh (2023): New Cellular Learning Automata as a framework for online link prediction problem, Journal of Experimental & Theoretical Artificial Intelligence, DOI: 10.1080/0952813X.2023.2188261

To link to this article: https://doi.org/10.1080/0952813X.2023.2188261

Published online: 11 Mar 2023.

JOURNAL OF EXPERIMENTAL & THEORETICAL ARTIFICIAL INTELLIGENCE
https://doi.org/10.1080/0952813X.2023.2188261

Taylor & Francis
Taylor & Francis Group

ARTICLE

New Cellular Learning Automata as a framework for online link prediction problem

Mozhdeh Khaksar Manshad[a], Mohammad Reza Meybodi[b] and Afshin Salajegheh[c]

[a]Department of Computer Engineering, Omidiyeh Branch, Islamic Azad University, Omidiyeh, Iran; [b]Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran; [c]Department of Computer Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran

ABSTRACT

One of the main areas of research in Social Network Analysis (SNA) is Link Prediction (LP). The LP problem is useful in understanding the evolution mechanism of social networks, as well as in different applications such as recommendation systems, bioinformatics and marketing. In LP algorithms, prior network information is used to predict future connections in social networks. In this paper, we introduce a multi-wave cellular learning automaton (MWCLA) and use it to solve the LP problem in social networks. This model is a new CLA with a connected structure and a module of LAs in each cell where the neighbours of the cell module are its successors. The MWCLA method uses multiple waves at the same time in the network in order to improve convergence speed as well as accuracy. For predicting links in the social network, multiple waves can be used to consider different aspects of the network. Here we show that the model converges upon a stable and compatible configuration. Compared to some state-of-the-art approaches, MWCLA produces significantly better results when applied to the LP problem.

ARTICLE HISTORY
Received 17 May 2021
Accepted 2 March 2023

KEYWORDS
Social networks; Cellular Learning Automata; link prediction problem

Introduction

A social network is officially represented by a graph in which the nodes represent the entities, and an edge between two entities represents the connection between them. LP is a social network analysis issue that examines the creation of new links in the future based upon the observed patterns between entities and their connections. It can be applied in different areas, such as forecasting the outbreak of a particular disease, suggesting an alternative route during the traffic and detecting a spam email (Daud et al., 2020). Due to the rapid growth of social networks and their high dynamics, we need adaptive algorithms to adapt to changes in the environment quickly and consider the various features of the environment simultaneously.

Several good features make learning automata suitable for use in many applications such as LP problem. These include:

1- It is possible to use learning automata without a priori knowledge of the underlying application. 2- LAs can be very useful for applications involving a great deal of uncertainty. 3- The environment is the only source of feedback required by learning automata. 4- These automata provide a great deal of utility to systems with limited intercommunication and incomplete information, such as multi-agent and distributed systems. 5- In software and hardware, learning automata have a straightforward structure and can be implemented very easily. 6- Learning automata use

CONTACT Mozhdeh Khaksar Manshad khaksar.mojdeh@yahoo.com Department of Computer Engineering, Omidiyeh Branch, Islamic Azad University, Omidiyeh, Iran

symbolic or numerical values as their action set. 7- Generally, optimisation algorithms based on learning automata do not require that the objective function be a mathematical function with adjustable parameters. 8- Learning automata can be studied using powerful mathematical methodologies. 9- It has been proven that learning automata work well in single, hierarchical, and distributed structures. 10- For real-time applications, learning automata require a few maths operations for each iteration. 11- For most applications, learning automata offer flexibility and analytical tractability

So, LA is a proper tool for LP problem because social networks problem is non-deterministic, dynamic and heterogeneous.

In LP problem in social networks, the learning process has to be propagated in parallel in several stages with a particular order to increase speed while preserving accuracy. The simple idea for such situations is to use a kind of wavefront CLA (WCLA) in which a multiplicity of waves run parallel to each other, instead of only one wave being active. Therefore, we introduce Multi-Wave CLA (MWCLA) as a new method.

A Cellular Automaton (CA) (Wolfram, 1983) is a model that consists of a high number of simple cells, and they pass through a collection of states based on local interaction. The collection of a cell's neighbourhood makes up its local environment, based on which the cell's local rule is defined. Each cell selects one state among a limited set of states. The state of each cell, along with the state of its adjacency cells, is called that cell's configuration. The local rule of CAs shows their evolution through time and indicates CAs' configuration changes in one stage. CAs are suitable for modelling systems made up of simple, connected building blocks with a local connection (Das, 2011).

Learning Automata (LAs) (Mandayam A. L. Thathachar & Sastry, 2002) are simple agents that, throughout their learning process, attempt to pick out the optimum action from a limited team of actions according to the environment's response to previous actions. They can be of use in unknown random environments. The capabilities of LAs are known when several LAs interact locally. This local interaction can be shown in various forms, such as trees, meshes, or arrays, based on its application.

Combining CAs with LAs, reference (Beigy & Meybodi, 2004) introduces a new model named Cellular Learning Automata (CLAs). CLAs are distributed computation models, which combine LAs' learning capabilities with CAs' computation power. This model of automaton outdoes CA for its ability to learn optimal actions. It also surpasses a single LA for in it, a set of LAs interacts with each other. CLAs come with many applications in various areas (Navid & Aghababa, 2013), such as computer networks (Esnaashari & Meybodi, 2008), social networks (Ghavipour & Meybodi, 2017; Mojdeh Khaksar Manshad et al., 2012 Manshad et al. 2021; Zhao et al., 2015), evolutionary computation (Mirsaleh & Meybodi, 2016), as well as optimisation (Vafashoar et al., 2012), and image processing (Patel & More, 2013).

In the social network analysis, the CLAs have been used in different problems that are explained as follows: In (Khomami et al., 2018), a novel community detection algorithm based on CLA is proposed in which several learning automata (LA) collaborate. In order to reduce the network size, the proposed algorithm uses irregular CLA to find a partial spanning tree and then create local communities on it at each step. Likewise, another CLA-based algorithm for detecting communities in complex networks has been introduced in (Zhao et al., 2015) (CLA-net algorithm). In the CLA-net algorithm, the entire network is modelled as an irregular CLA and the ideal community structure is determined through the evolution of the CLA.

The authors in (Daliri Khomami et al., 2018) concentrate on the issue of influence maximisation in social networks, which is the process of selecting a select group of target nodes in order to spread influence as widely as possible using a specific diffusion model. They first suggest a learning automaton-based algorithm to address the problem of the minimum positive influence dominating set (MPIDS), and they then employ the MPIDS to maximise influence in online social networks.

Lately, a class of asynchronous CLAs called Wavefront CLAs (WCLAs) (Moradabadi & Meybodi, 2018b) has been introduced. WCLAs are different from CLAs in two main ways. One is that they enjoy the connected neighbourhood; that is, LA neighbours are defined as successor LAs in the network.

The second is that they have the capacity to propagate information. Also, in (Moradabadi & Meybodi, 2018b), their steady-state behaviour through time has been analysed and assessed with a minimum spanning tree as well as graph colouring.

The proposed MWCLA in this paper is a generalisation of WCLA. The idea we tap is to use a module of LAs instead of one LA in each cell of a CLA and that each cell's neighbourhoods are characterised by the successor cells. Each of the n LAs in a cell can create a wave and dispatch it to neighbour cells if their selected action is different from their previous actions. These LAs create n waves running in parallel. The method we introduce in this paper paves the way for choosing more than one action at a time, helping boost convergence speed without losing accuracy. The main idea here is that since in the LP problem, the environment is stochastic, decision-making based upon different responses from the environment will produce fewer errors compared to decisions based on one response. This article will first introduce MWCLAs and then analyse their convergence. Likewise, we use MWCLA as a framework for solving the LP problem.

In the following, we summarise the contributions of the paper:

- A new CLA method is proposed, which can propagate multiple waves into the environment from a cell. MWCLA can be used in any problem where the learning process must propagate in multiple ways and successive order.
- The convergence analysis of the MWCLA is considered.
- The proposed method is used as a framework for improving LP problem accuracy (MWCLA-LP).
- The results of experiments for the MWCLA-LP method are compared with other LA and CLA-based algorithms for LP.

The paper's upcoming sections have been sorted as follows: Section 2 provides a brief overview of the LP problem and the studies that have been proposed so far. In Section 3, a short description is given about the LAs and CLAs. Section 4 gives full explanations about the MWCLA and its convergence analysis. The modelling phase of MWCLA for the social network has been described in Section 5. A new model based on MWCLA for the LP problem in a social network has been introduced in section 6. In Section 7, the experimental study of MWCLA and their comparison with other methods have been described. In Section 8, the research's main findings have been briefly discussed.

Related works

Recently, due to the increasing number of social networks and their growth in size, the introduction of new LP algorithms with higher speed and accuracy has attracted special attention. There are many basic and generic LP criteria, and we categorise available algorithms for the LP problem into two classes. First, similarity-based algorithms and second learning-based algorithms. The former one calculates the similarity between nodes by using different approaches such as node-based approaches (Akcora et al., 2013; Anderson et al., 2012; Bhattacharyya et al., 2011), topology-based methods (Liben-nowell & Kleinberg, 2007), social network criteria-based methods (Dong et al., 2012; Liu et al., 2013; Qiu et al., 2010; Valverde Rebaza & de Andrade Lopes, 2013; Yang et al., 2011). The latter one, learning-based algorithms, is created by using internal features, external information, and similarity-based algorithms. It contains several subcategories such as matrix factorisation algorithm (Menon & Elkan, 2011), feature-based classification algorithms (Bliss et al., 2014; Huang & Lin, 2009; Rossetti et al., 2015; Tan et al., 2014b), probabilistic-based algorithms (Clauset et al., 2008; Guimerà & Sales-Pardo, 2009), learning automata-based algorithms (Moradabadi & Meybodi, 2016, 2017a, 2017b).

In this part of the paper, we introduce some new notations that will be used next in this paper.

Definition (1):

• $N = |V|$ shows the total number of nodes in the social network.

- $M = |E|$ is the total number of connections.
- $\Gamma(v) = \{x \in V : (x, v) \in E\}$ denotes the neighbours of v.
- $d(v) = \Gamma(v)$ is the degree of the node.

Let us assume that the social network is represented as a graph; our aim is then to predict the connections created in time $t + 1$ regarding the information from the previous graph structure.

In the following, we have described some of the popular algorithms of each category.

Common neighbourhood (CN) (Al Hasan & Zaki, 2011): in this index, two nodes are more likely to be related in the future if they have many common neighbours compared to other neighbouring nodes.

$$CN(v_i, v_j) = \left|\Gamma(v_i) \cap^{\Gamma} (v_j)\right| \tag{1}$$

Jaccard index (JC) (Jaccard, 1901): the probability that a neighbour of v_i or v_j is a neighbour of both v_i and v_j has been measured.

$$JC(v_i, v_j) = \frac{\left|\Gamma(v_i) \cap^{\Gamma} (v_j)\right|}{\left|\Gamma(v_i) \cup^{\Gamma} (v_j)\right|} \tag{2}$$

Preferential attachment (PA) (Newman, 2001): if a pair of nodes have a higher degree, the corresponding connection will give a higher score.

$$PA(v_i, v_j) = d(v_i) \times d(v_j) \tag{3}$$

Adamic-Adar index (AA) (Adamic & Adar, 2003): it is counting common neighbours between two nodes by using the following Equation:

$$AA(v_i, v_j) = \sum_{z \in \Gamma(v_i) \cap^{\Gamma} (v_j)} \frac{1}{\log(|\Gamma(z)|)} \tag{4}$$

Katz index (Katz, 1953): it calculates the summation of the number of paths with various lengths.

$$Katz(v_i, v_j) = \sum_{l=1}^{\infty} \beta^l . \left|path(v_i, v_j)^{\langle l \rangle}\right| \tag{5}$$

Local path index (Zhou et al., 2009): this index is a limited version of the Katz criterion that calculates the sum of paths with lengths 1 and 2.

$$Local\ Path\ Index(v_i, v_j) = A^2 + \varepsilon A^3 \tag{6}$$

where A shows the proximity matrix, and the free parameter is indicated by ε.

By combining covariance matrix and evolutionary strategy, a new LP algorithm called CMA-ES has been introduced in (Bliss et al., 2014). The authors use a linear formula derived from a combination of similarity-based methods to evaluate the corresponding methods' weight. Evaluations of the CMA-ES have shown that the introduced algorithm is very accurate for the 20 first predicted links.

In (Sherkat et al., 2015), a new LP method using the ant colony algorithm has been introduced, which is called ACO-LP. ACO-LP firstly finds triangular subgraphs in the network, and then, based on the recognised subgraphs, new connections in the social network have been predicted. The main advantages of the ACO-LP method are that it has reasonable execution time in comparison with other algorithms. Also, for some networks, the ACO-LP's results are much better compared to other structural LP methods.

In (Tan et al., 2014a), a new method based on information theory and network structure's mutual information has been proposed, which is called MI-LP. The results of the experiments have shown that MI-LP has a better precision and execution time compared to six other methods. A new method, which is used time-series prediction, as well as feature selection, has been proposed in (Rossetti et al., 2015), which is called the interaction prediction (IP) algorithm.

A new concept of LP in (YounessZadeh & Meybodi, 2018) has been proposed that combines LP based on similarity scores and probabilistic LP, and it is categorised into a new category of LP methods. Probabilistic techniques are used to measure similarity between nodes in this idea. On standard datasets, this method produces better results than other criteria methods by using learning automata.

In (Chen et al., 2021) a new method to search for high-quality relationship paths is proposed which is called RLPath. RLPath alternately trains a reinforcement learning model with a trainable reward setting, and a translation-based model for realising link prediction. The model also comes with a novel reward setting that is shared with the translation-based model, so that the parameters are not only able to measure the contribution of relation paths, but also guide agents to search for connection paths that contribute highly to link prediction, resulting in mutual promotion.

In (Lim et al., 2019) a deep reinforcement learning (DRL) is proposed. To compare the time-evolving DRL model (TDRL) with the metadata fusion model (FDRL), they used a purely time-evolving DRL model (TDRL) without metadata fusion. According to the experimental results in (Lim et al., 2019), the FDRL model is more accurate in predicting new and recurrent relationships than TDRL, which does not factor in fusion of data from different sources.

In (Moradabadi & Meybodi, 2017b), a new method for time-series link prediction based on LA (LA-TSLP) has been introduced. LA-TSLP uses various local temporal similarity-based algorithms from time *1 to T* for predicting links in time $T + 1$. In (Moradabadi & Meybodi, 2016), the authors have proposed a linear formula of similarity-based methods and then consider the LP problem as an optimisation problem.

A new approach to link prediction for stochastic social networks based on LA has been proposed in (Moradabadi & Meybodi, 2018a). The links' weights in a stochastic social network are random variables. To achieve this, the authors first redefine some of the similarity metrics for link prediction in stochastic graphs and then propose a method based on LA to compute the distribution of the proposed similarity metrics assuming that the probability distributions of the link weights are unknown.

Continuous-action learning automata (CALA) have been used as a tool for estimating the weight of similarity-based methods based on the proposed formula. Furthermore, a Continuous Action-Set Cellular Learning Automaton (CACLA) has been introduced to consider the link prediction problem in a weighted multi-layer social network. In the proposed model, every cell has a Continuous Action-Set Learning Automata (CALA) module where its neighbours are its successors. It is shown that the model converges on a stable and compatible configuration (Khaksar Manshad et al., 2022). Likewise, in (Moradabadi & Meybodi, 2017a), a new fuzzy-based algorithm, which uses distributed LAs (DLAs) for estimating the strength of links in the social networks has been proposed (FLP-DLA). FLP-DLA estimates the strength of links by using the information of the social network, which are considered as the result of the LP problem. In (Mozhdeh Khaksar Manshad et al., 2020), a new time-series LP based on irregular CLAs (ICLAs) and evolutionary computing has been introduced (ICLA-EC-TSLP). The proposed method uses its previous experiences and its local neighbours' experiences for predicting future links. A new approach based on ICLA has been introduced in (Manshad et al., 2021) in which an ICLA has been allocated to each node in the network, and it considers the information of local communities for weighing the real links in the network and then calculates the scores of the unconnected connections by using the weight of real connections.

Background

In this Section, we briefly go over LAs and CLAs.

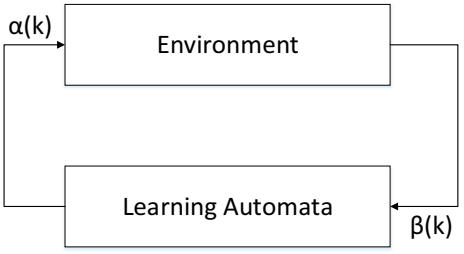


Figure 1. Schematic of the relationship between an LA and its environment.

LAs

LAs (Thathachar & Sastry, 2002) provide a technique for computational intelligence for solving complicated, uncertain problems or where plenty of information within the environment is lacking. LAs improve their performance by learning optimal actions from a limited collection of actions interacting with an environment. The chosen action is applied as input to the environment according to the LA's action probability distribution. Then, the environment sends the automata a reinforcement signal as an input action response. The reinforcement signal can be used for updating the automata actions' probability distribution. As this process continues, the LA learns to pick out the optimal action from its team of actions, thus receiving the environment's desired response.

Definition (2): An LA can be shown as the sextet $\langle \bar{\alpha}, \bar{\emptyset}, \bar{\beta}, \bar{p}, F, G \rangle$, so that $\bar{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ is a set of actions out of which it chooses its desired action, $\bar{\emptyset} = \{\emptyset_1, \emptyset_2, \ldots, \emptyset_s\}$ is the set of the automata's internal states, $\bar{\beta} = \{\beta_1, \beta_2, \ldots, \beta_q\}$ is the automata's set of inputs, $\bar{p} = \{p_1, p_2, \ldots, p_n\}$ is the automata's action probability distribution vector, $G : \bar{\emptyset} \to \bar{\alpha}$ is the output map and specifies the action chosen by the LA if its internal state is $\bar{\emptyset}$, and $F : \bar{\emptyset} \times \bar{\beta} \to \bar{\emptyset}$, specifies the LA's state transition map according to input from the environment.

As we have shown in Figure 1 at the time k, the action chosen based on p's action probability vector by LA such as $\alpha(k)$ is given as input to the environment, and random response $\beta(k)$, i.e., reinforcement signal, is received from the environment. Then, the $p(k)$, actions' probability is updated by a learning algorithm.

CLAs

A CLA (Navid & Aghababa, 2013) is a CA, which has been assigned an LA. The configuration of cells is defined as the state of all cells. A CLA works through its local rule. Based on the local rule and the cell's configuration, a reinforcement signal is produced, which penalises or rewards the LA in the cell. The operations of a CLA can be explained according to the following stages. In the first stage, each LA in a cell picks out one of its actions through the automata probability distribution vector, thus defining the state of the cell. A reinforcement signal is then obtained using the CLA's local rule and configuration, which penalises or rewards the LA in each cell, therefore updating the LA's probability distribution vector according to a learning algorithm. This process is repeated until the desired result is achieved. Officially, a CLA is explained as:

Definition (3): A CLA is shown with the quintet $(z^d, N, \bar{\emptyset}, \bar{A}, F)$ so that: z^d is a lattice, a d-tuple is an integer number, $N = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_m\}$ is a limited subset of z^d which is called a neighbourhood vector so that $\bar{x}_m \in z^d$, $\bar{\emptyset}$ shows a limited team of states. The state of the cell c_i is shown as $\bar{\emptyset}_i$. In each cell of CLA, there is a team of LAs, which are shown by $\bar{A}$. $F_i : \bar{\emptyset}_i \to \bar{\beta}$ shows the local rule of CLA in cell c_i. So that the probable values, which are used for showing the reinforcement signal, are indicated by $\bar{\beta}$.

CLAs can be categorised in different categories, as follows:

Closed vs. open CLAs (Beigy & Meybodi, 2007): In closed CLAs, the LA's action selection in each cell for the next stage is conducted according to its local environment. In open CLAs, by contrast, each LA's action selection for the next step depends not only on its local environment but on its exclusive and global environments as well.

Regular vs. irregular CLAs (ICLAS) (Mehdi Esnaashari & Meybodi, 2014): In the ICLAs, the presumption for having a regular structure has been dropped.

Synchronous vs. asynchronous CLAs (Beigy & Meybodi, 2008): The LAs residing in different cells are operated by a synchronous global clock in synchronous CLAs, while in asynchronous CLAs, only a limited number of the cells are active at a given time, while the state of others stays unchanged. Wavefront CLAs (WCLAs) (Moradabadi & Meybodi, 2018b) are a class of asynchronous CLAs, which are different from CLAs in two main ways. First, the neighbours of an LA are defined as successor LAs. Second, each LA in the network can forward a wave into its successors if its selected action is opposed to the prior action.

Static vs. dynamic CLAs (Saghiri & Meybodi, 2017): The former demonstrates that the cellular structure of CLAs remains the same through time, whereas the latter illustrates that one of CLAs' sights, such as a local rule, structure, or adjacency radius, changes over time. In the sources mentioned above, the models have been introduced, and it has been illustrated that for commutative rules (a class of rules), the mentioned models converge based on a stable general state.

Multi-wave CLA

A Multi-Wave Cellular Learning Automaton (MWCLA) is the Wavefront CLA (WCLA)'s generalised form. The main difference between the MWCLA and WCLA is that in the WCLA, in each instance, only one wave moves along the network. In contrast, in each instance, in the MWCLA, a multiplicity of waves can move along the network, as shown in Figure 2, and several LAs have generated the waves from a cell, which work parallel to each other.

The initial idea by parallel operation of waves is that since the response from the environment is quite random, a decision made on multiple responses compared to a decision made on one response will have less chance of a mistake. Updating the action probability vector will be more precise, also facilitating faster convergence.

In some applications, such as social networks, more than one wave must be active in the system at each given time. Each wave is considered for one decision-making variable, which can be matched

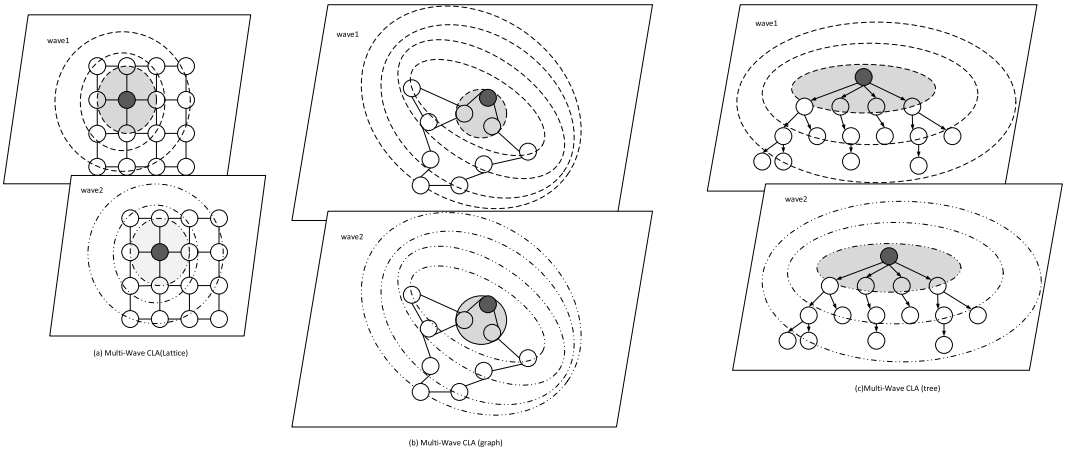


Figure 2. Multi-wave CLAs with different structures. As shown in each structure's figures, one cell is activated, and two parallel waves are propagated through the network. It means that each cell consists of a module of LAs, and each LA generates one wave. In this figure, there are two LAs in each cell, and they generate two waves.

by one LA. We have called this type of asynchronous CLA the Multi-Wave CLA (MWCLA). Likewise, in the MWCLA, as in WCLA, each cell's adjacent cells are described as its successors, and in each cell, each LA activated can dispatch a wave to its corresponding LAs in the neighbourhood cells. If the action chosen is distinct from the prior action, the neighbour cells are activated (the propagation feature).

Each LA in a cell receiving a wave is activated and should select a new action. In the starting phase, each cell's initial internal state is defined according to the action probability distribution of LAs residing in the cell. In the k^{th} run, the LAs in a cell are activated and choose one of their actions. If their chosen actions are different from their previous actions, they will send some waves over the network. Each cell obtaining the waves will be activated, and the corresponding LAs that have received the waves should choose their actions. Then, using the local rule, a reinforcement signal will be created by the MWCLA to update the probability distribution of the active LAs. The process is repeated until the desired outcome is achieved. The MWCLA model is asynchronous since, at each given time, one cell is active, and the LAs residing in the activated cell module select their actions, so more than one wave is active in a parallel manner (Figure 2). It is also close since it only uses the local environment for updates. It is also static since its structure does not change with time.

In the MWCLA, the path of propagation for waves depends only on the neighbourhood structure, and MWCLA can have different structures, as shown in Figure 2. Each wave propagated in the environment has two features: 1. Energy 2. Frequency. The energy of a wave is indicative of the wave's propagation intensity. The waves' energy diminishes as they move along the network until it vanishes, and then the wave disappears, and the movement stops. The frequency of waves is the length of time in which they are repeated. In the MWCLA, some waves are sent by a root cell over the network in each run. The waves are not repeated until the LAs of the root cell choose new actions.

The procedure of the MWCLA is demonstrated in Figure 3 in a simple viewpoint. In this Figure, first, the root cell 1 is activated and each LA residing in the module of LAs in cell 1 selects an action. As each LA selects a new different action in comparison with the previously selected action, each LA activates all of its children LAs in the next level (LAs residing in cell 2 and cell 3, LA_1^1 generate wave one and forward wave 1 to LA_2^1 and LA_3^1, likewise LA_1^2 generate wave two and forward wave 2 to LA_2^2 and LA_3^2 and so on). Then, each activated LA chooses a new action. If its selected action is the same as its previous action, then the chain is stopped at that vertex (coloured as red). If the new action is different from the previous action, it activates its children (for wave one is coloured as green, for wave two is coloured as blue, and for wave three is coloured as purple), and this procedure goes on.

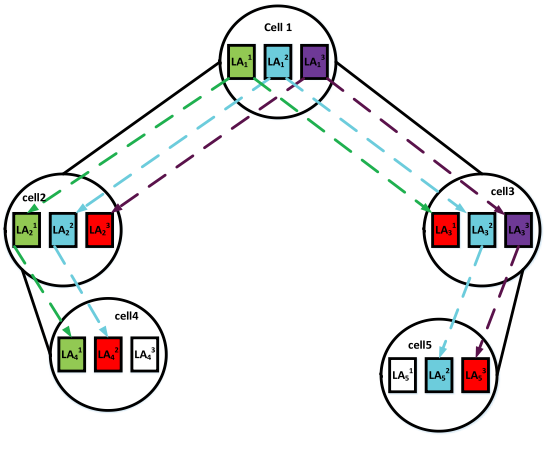


Figure 3. An MWCLA example.

A dotted arrow indicates the direction of propagation of each wave with a specific colour (wave one with green colour, wave two with blue colour and wave three with purple colour).

Definition (4): Granted that in each of the N cells there are n LAs, we demonstrate the MWCLA with $\langle S, \bar{\emptyset}, \bar{A}, F, \bar{W}, \bar{C} \rangle$, where S is the structure of MWCLA, which can be graphs, trees, or lattices (Figure 2). $\bar{\emptyset}$ is a limited collection of states. $\bar{A}$ is the team of LAs assigned to the MWCLA so that $\bar{A}_i$ is the team of resident LAs in the i^{th} cell. $F : \bar{\emptyset}^{n \times k} \rightarrow \bar{\beta}$ defines the MWCLA's local rule so that the set of possible values shown by $\bar{\beta}$ for the reinforcement signal is computed for each LA in a cell using the corresponding LAs in successor cells. $\bar{W} = \{w_1, w_2, \ldots, w_n\}$ defines the energy of waves and specifies when and where a wave is stalled. In an MWCLA $\bar{C} = \{c_1, c_2, \ldots, c_n\}$is illustrated whether each LA in the module of LAs $\{LA_1, LA_2, \ldots, LA_n\}$ has chosen a different action than the prior action or not.

Based on the concepts mentioned above, the main difference between MWCLA and WCLA is that a module of LAs in each cell and each LA can generate a wave. Suppose there are initially N LAs in the WCLA, so in MWCLA, there will be N cells, each possessing a module with n LAs. These LAs now generate n waves acting in parallel. The LAs of the i^{th} cell in each such MWCLA together create the i^{th} module and share a common actions probability vectorp_i.

Definition (5): In stage k, the MWCLA configuration is shown as $P(k)$, which is defined as a probability vector for the entire LAs:

$$P(k) = \left(p_1^T(k), p_2^T(k), \ldots, p_N^T(k)\right)^T \tag{7}$$

Let assume that each LA has r_i actions. Then,

$$p_i(k) = \left(p_{i1}(k), p_{i2}(k), \ldots, p_{ir_i}(k)\right)^T \tag{8}$$

where $p_{ij}(k)$ shows the probability that the i^{th} LA in each wave chooses the j^{th} action at time k.

Every one of the n parallel waves generated from a module interacts with its environment based on the selected actions of its LAs and receives a common reinforcement signal for whole resident LAs in the corresponding module. Supposed that $\beta^\tau(k)$ is the reinforcement signal by the τ^{th} wave at time k, the objective of the MWCLA will then be defined as:

$$maximize f(P) = \sum_{\tau=1}^{n} E[\beta^\tau | P] \tag{9}$$

subject to $\sum_{j=1}^{r_i} p_{ij} = 1, \quad i \in \{1, \ldots N\}$.

As the n waves share the common action probability vector $p_i, (i = 1, \ldots, N)$,

$$E\left[\beta^1 | P\right] = E\left[\beta^2 | P\right] = \ldots = E[\beta^n | P] \tag{10}$$

Therefore:

$$f(P) = nE[\beta^\tau | P] \ for any \tau \in \{1, 2, \ldots, n\} \tag{11}$$

Considering that $\beta^\tau(k)$ can vary with τ. Therefore, (9) equals to:

$$maximize f(P) = E[\beta^\tau | P] \ for any \tau \in \{1, 2, \ldots, n\} \tag{12}$$

Definition (6): At time k, let:

$\alpha_i^\tau(k)$= The action is selected by the i^{th} LA in the τ^{th} wave

$\beta^\tau(k)$= The reinforcement signal received by the τ^{th} wave

$q_{ij}(k) = \sum_{\tau=1}^{n} \beta^\tau(k) I\{\alpha_i^\tau = \alpha_{ij}\}$= The total reinforcement signals of the j^{th} action of the i^{th} module

$q(k) = \sum_{\tau=1}^{n} \beta^\tau(k)$= The total reinforcement signal to each module

According to the above definition, the j^{th} action's probability of the i^{th} LA in each wave will be updated according to the below algorithm:

$$p_{ij}(k+1) = p_{ij}(k) + \tilde{\lambda}\big(q_{ij}(k) - p_{ij}(k).q(k)\big) j = 1, 2, \ldots, r_i; i = 1, 2, \ldots, N \tag{13}$$

where $\tilde{\lambda} = \frac{\lambda}{n}$ is a normalised learning parameter. Using definitions related to $q_{ij}(k)$ and $q(k)$, algorithm (13) can be rewritten as:

$$p_{ij}(k+1) = p_{ij}(k) + \tilde{\lambda} \sum_{\tau=1}^{n} \beta^{\tau}(k)\big(I\{\alpha_i^{\tau} = \alpha_{ij}\} - p_{ij}(k)\big) \tag{14}$$

We must note that $p_{ij}(k)$ is updated once and not necessarily by each LA; the updated value $p_{ij}(k+1)$ is shared by all the LAs in the module. The convergence of the MWCLA is analysed in the Appendix 1.

MWCLA as a framework for social network problems

The LP issue's main challenge is that this problem revolves around the noticeable changes in the online network structure. We need to design an algorithm that can adapt itself during the online network changes. Since MWCLA has the propagation probability, it can be used for the online LP problem. Also, more than one wave can work in the environment each time, so MWCLA is suitable for considering more than one factor in the issue. First, we propose two models based on MWCLA as a social network framework (modelling phase).

The modelling phase aims to create a model based on MWCLA (MWCLA-LP) for solving the social network problem. First, because the structure of the MWCLA must be connected, we consider an MWCLA for each connected component in the social network. In this paper, to model social networks with MWCLA, the link-based social network solution has been used. In this kind of solution, the links have the main role in the solution, while in the node-based solutions; the nodes have the main role in solving the problem. Therefore, based on the aforementioned information, we present two models of MWCLA as follows:

- MWCLA model based on vertices:

This model can be used in vertex-based social network solutions. For this purpose, we assign an MWCLA with a module of LAs to a vertex in the social network, and the neighbours of each module of LAs are the resident modules of the corresponding neighbouring vertex in the graph structure.

- MWCLA model based on connections:

This model can be used in connection-based social network solutions. For this purpose, we assign an MWCLA with a module of LAs to a connection in the social network, and the neighbours of each module of LAs are connected to one incident node of the corresponding connection.

The proposed method for LP based on MWCLA

We first model the LP issue using MWCLA and then present the running phase for the MWCLA-LP model for considering this problem.

In the LP problem, we consider an MWCLA with a module of LAs for each test link (MWCLA model based on connections) in the social network that must be predicted. In the following, we describe the life cycle of one MWCLA because the set of MWCLAs operate similarly.

In each MWCLA, there is a module of LAs for each link that must be predicted; in other words, each module of LAs related to a specific test link and, as shown in Figure 4, the neighbours of each module of LAs are defined by the modules of LAs, which are connected to one incident node of the

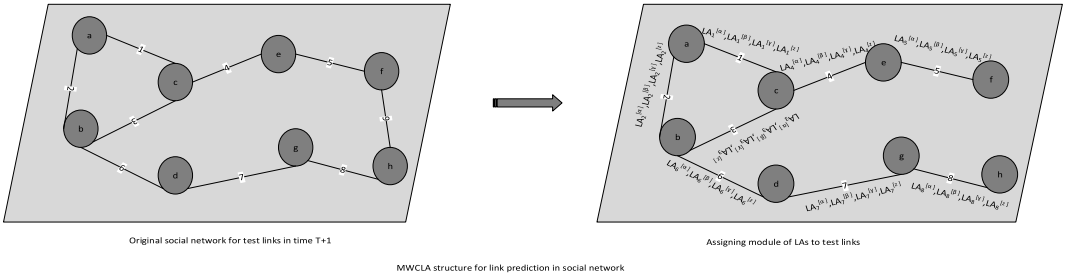


Figure 4. Graph G_{test} in which a module of LAs is assigned to each test link. It is shown that the neighbors of each link are defined by the links which are connected to one incident node of the corresponding link. For example, in the above figure, the neighbors of link 1 are links 2, 3, 4.

corresponding test link. Therefore, in this method, there is an original network G, and there is a test network G_{test} and we assign a module of LAs to each test link in G_{test}(Figure 4).

Each LA in the module is used for generating a wave in the social network for a specific environment. Until now, we have an MWCLA-based model for link prediction. In this paper, we have a module of LAs for each test connection and LAs in a module working in a parallel manner in the place of a single learning automaton; each LA chooses its action and obtains a corresponding reinforcement signal from the environment simultaneously. In parallel operation of LAs, different parameters will be tested simultaneously. After obtaining a reinforcement signal by each LA, the common action probability vector related to the module of the LAs will be updated based on the learning algorithm. In (Moradabadi & Meybodi, 2017b), authors for predicting links in the social network have just decided based on one environment, so the environment has been defined as some similarity metrics, then one environment has been selected, and the reinforcement signal for the selected action LA has been generated. Due to the fact that a decision based on various responses would have less error than a decision based on a single response, we have considered several environments in parallel by using several waves. In the next step, we should describe each LA's action-set and the logic of the running phase of the MWCLA.

We explain one LA's life cycle in a module of LAs because they work similarly to each other in a parallel manner. The τ^{th} LA in the i^{th} module (LA_i^τ) that makes the τ^{th} wave and corresponds to the i^{th} test link in the social network has an action-set with two actions: {0,1}. The initial probability for each of the two actions at the beginning of the algorithm is set to 0.5. Action 0 determines that the corresponding link must not appear in the prediction result, and action 1 determines that the corresponding link must appear as a predicted link. It should be noted that the module of LAs related to a cell uses a common action probability vector. We select a module of LAs (a link) randomly as a root module for running the algorithm. Each LA_i^τ in the module selects its action, and if their selected action is different from its prior action, it generates a wave and sends it to its neighbours (until a special depth) and activates them to select new actions. When all its neighbours select their action, we call the local rule to generate the reinforcement signal based on its environment for the correspondingLA_i^τ. The environment has been defined based on similarity metrics {CN, JC, PA, AA}. The local rule for each LA_i^τgets the selected action by itself and its neighbours' actions and produces a reinforcement signal as the following: it considers the original graph G and adds the links based on the action selected by the LA_i^τ neighbours and calls the new graph,G'^τ. In other words, for each test link in the neighbourhood of LA_i^τ if the corresponding LA action is 1, then the link is added to the network. Then one of the adjacent original links, link k, that has a common incident node with the test link that corresponds to LA_i^τ is selected randomly, and the similarity metric of the link k is calculated based on the G'^τ Structure with two situations: $G'^\tau$with link i and G'^τ without link i and then we consider the normalised differential of the two similarity metrics as the reinforcement signal forLA_i^τ.

In other words, we try to estimate the impact of the link *i* considering its neighbours' actions in calculating the similarity metrics for a random real adjacent link. If the link *i* has a good effect on raising the similarity metric of the link *k* and the selected action of LA_i^{τ} is 1, or if it does not have a good impact on raising the similarity metric of the link *k* and its action is 0, then it is rewarded; otherwise, it is penalised. We also calculate the reinforcement signal for other LAs in a module that is working in other environments. Then, based on the received reinforcement signals from different environments, we update the common action probability vector of the activated module by using the learning algorithm (13). The algorithm for the MWCLA-LP is presented in Figure 5. This procedure is repeated for every cell

Algorithm1. The proposed link prediction method based on MWCLA

Offline phase
Let G<V, E> be the social network
Create a graph G_{test} from the test link in the social network
Assign $\overline{A}$ as a module of LAs to each test link
Let *r* be the number of actions for each LA
Let $p_i = (p_{i1}, p_{i2}, \ldots, p_{ir})^{\tau}$ Where p_{ij} is the probability that each LA in the i^{th} module chooses the j^{th} action in which $p_{ij} = \{p_{i1}, p_{i2}\}$ and initialized to $p_{i1} = p_{i2} = \frac{1}{2}$
Let $\alpha_{ij}^{[\tau]} = \{\alpha_{i1}^{[\tau]} = 0, \alpha_{i2}^{[\tau]} = 1\}$ be the actin-set of τ^{th} LA in the i^{th} module (LA_i^{τ})
Let *s* be the iteration counter and initially set to *1* and let *MaxIteration* be the total number of iterations.
Let *Active_cells* be the set of cells for the next iteration and initially is set to {}.
Let W be the wave energy related to each wave.
Let *t* be the threshold of similarity difference.
Let E={CN, JC, PA, AA}be the set of environment.
Begin
  Set Candidate_cells={}
  Repeat
    s++
    If Candidate_cells={} **then**
      Active_cells= The set of j cells, each cell with one module of LAs for each MWCLA that is selected randomly.
      For each $cell_i$ in Active_cells do
        For each LA_i^{τ} in Active_cells do
          Learning automata LA_i^{τ} selects an action based on cell's action probability vector.
        End
        Add $cell_i$ to Candidate_cells
        Remove $cell_i$ from Active_cells
      End
    End
    For each $cell_i$ in Candidate_cells **do**
      For each LA_i^{τ} in $cell_i$ **do in parallel**
        If selected action of LA_i^{τ} <> previouse action of LA_i^{τ}**then**
          Active all $cell_i$ neighbors in depth W and the corresponding LA choose a new action and add cell to the Candidate_LAs of environment.
          Set configuration for each environment be the set of selected actions of LA_i^{τ} and all its neighbors' action until depth W.
          Extend G graph into graph G'^{τ} by adding the test links corresponds to LA_i^{τ}neighbors that have been activated in the previose loop and choose action 1 as their new action.
          Set *k* be a random link (not test links) from graph G'^{τ} which has a common incident node with LA_i^{τ}.
          Set similarity be the differential of similarity metric for link m using two graphs G'^{τ}+<$link_i$> and G'^{τ}.
          If (similarity>*t* and Action(LA_i^{τ})==1) or (similarity<=t and Action(LA_i^{τ})==0) then
            Set β=1 //**Reward the** LA_i^{τ}
          Else
            Set β=0 // **penalty the** LA_i^{τ}
          End
        End
        Update the action probability distribution based on generated β and the learning algorithm.
    End
    Remove $cell_i$ from Candidate_cells
  End
Until *s*< *MaxIteration*
End

Online phase
When a change is occurred on the network **do**
  If a test link *k* is appeared **then**
    Add a new module of LAs to the MWCLA
    LAs of the module choose their action based on their action probability vector.
    Add cell to the Candidate_cells
  Else if the k^{th} test link is removed **then**
    Set k^{th} module of LAs corresponds to the remove test link
    Active all neighbors of k^{th} cell and all LAs residing in $cell_k$'s neighbor choose a new action and add them to the Candidate_cells
    Remove k^{th} cell from MWCLA structure
  End
Run offline phase with current Candidate_cells
End

Figure 5. Algorithm of the proposed MWCLA-LP for LP problem.

that is activated. Finally, we use each cell's action probability vector for determining the results. For the i^{th} test link, if the probability of choosing action 1 is greater than the probability of selecting action 0, it will appear in the results list and vice versa.

For the online social network, when a link or vertex appears, we create a module of LAs for the new link or vertex and activate it to generate some actions and propagate its actions through the network. As a result, the corresponding module neighbours can be activated to adapt their action based on the new change in the social network, and this procedure can be repeated in multiple iterations for a predefined iteration number. Now we have a stable network with stable MWCLAs until another change occurs in the network. Also, if a link or a vertex is removed from the network, we remove its module of LAs from the MWCLA and activate the corresponding module's neighbours to choose their action again. In other words, when a change occurs in the network, first we change the structure of the MWCLA in order to adapt the MWCLA structure with the new social network structure, and then we activate the modules of LAs that correspond to the change that occurred in the network to adapt the MWCLA behaviour with the new social network structure. This procedure enables us to propagate the changes in social networks.

Results and experiments

In the present section, we explain the experiments designed to examine the MWCLA-LP, as well as the acquired results. In Section 7.1, the social network datasets used in the experiments have been described. Then, in section 7.2, the evaluation metric has been explained. In Section 7.3, the performance of MWCLA-LP is compared with several LP algorithms.

Datasets

In this Section, we have proposed two groups of datasets, which are used for experiments. The analysis of these datasets is given as follows:

(1) Astro-ph, Hep-th and Hep-ph are co-authorship networks, where a node indicates an author, and a connection between two vertices shows that two authors have written a paper (Barabâsi et al., 2002). The co-authorship datasets, which are utilised in this paper have been collected from authors in different fields Astro-ph, Hep-th, and Hep-ph. In this paper, the data has been extracted from 1993 to 2003 for the entire dataset. As these networks are extremely sparse, it is therefore necessary to reduce the number of node pairs to make computation much easier. In this way, the vertices that have at least two neighbouring nodes from the year 1993 to 2003 have been selected.

(2) Enron Email communication network (Shetty & Adibi, 2004) considers the set of emails from May 1999 through May 2002. In this dataset, a vertex shows an email address, and a connection represents that the i^{th} address has sent an email to the j^{th} address. Table 1 describes the topological features of each dataset, which are used in this paper. *M* and *N* represent the number of links and vertices of the networks in this Table, respectively.

Table 1. Datasets used in experiments and their statistical information.

Dataset	N	M	Description
Hep-th	9,877	51,971	The network of co-authors in ArXiv's High Energy Physics Theory
Hep-ph	12,008	237,010	The network of co-authors in ArXiv's High Energy Physics
Astro-ph	18,772	396,160	The network of co-authors in ArXiv's Astro Physics
Email-Enron	87,273	1,148,072	The Enron E-mail communication network

Metrics for evaluating the proposed LP algorithm

An important metric for assessing methods' accuracy is the Area Under the Receiver Operating Characteristic Curve (AUC) (Al Hasan & Zaki, 2011; Shang et al., 2019). We use the AUC metric to compare MWCLA-LP with other approaches as follows:

In each given network, E is a set of existing edges that is divided into two sets: E_T is the set of training edges and E_P is the set of test edges.

To calculate the AUC, we pick an edge from E_P and one from non-existing edges E_N, as well as the calculated scores for edges. After, this process is repeated n times, AUC can be calculated using Equation (15).

$$AUC = \frac{n' + 0.5n''}{n} \tag{15}$$

In this case, a value closer to 1 indicates that the link prediction method is more effective.

The number n' in Equation (15) indicates how many times the scores of the nodes connected by the edge picked from the set E_P are greater than the scores of the nodes connected by the edge picked from the set E_N, n indicates how often the two scores are equal. For every pair of links in E_P and E_N, we always compare their values, with respect to the value of n.

An example of an AUC schematic is presented in Figure 6. As shown in Figure 6(b,c) five edges in this network are randomly divided into four training edges and one test edge. Figure 6d shows the non-existing edges for this network. The likelihood of formation of the test edge e_{35} should be compared with the likelihood of formation of the non-existing edges e_{13}, e_{14}, e_{15}, e_{24}, and e_{45} to calculate AUC.

Based on the proposed algorithm, we assume that the score of the test and non-existing edges based on probability of choosing action 1 are equal to, $score_{35} = 0.75$, $score_{13} = 0.75$, $score_{14} = 0.59$, $score_{15} = 0.6$, $score_{24} = 0.85$ and $score_{45} = 0.59$. Thus, $n' = 3$ and $= 1$ and $AUC = \frac{3+0.5\times1}{5} = 0.7$.

Experimental study and results

For evaluating the MWCLA-LP method in comparison with other methods, we have designed some experiments. Various local similarity-based methods such as AA (Adamic & Adar, 2003), PA (Newman, 2001), CN (Al Hasan & Zaki, 2011), JC (Jaccard, 1901), a global similarity-based method such as Katz (Katz, 1953), a quasi-local similarity-based method such as local path (Zhou et al., 2009), and some recently proposed methods such as FLP-DLA (Moradabadi & Meybodi, 2017a), MI-LP (Tan et al., 2014a), LA-TSLP (Moradabadi & Meybodi, 2017b), CALA-LP (Moradabadi & Meybodi, 2016), ICLA-LP (Manshad et al. 2021), IP (Rossetti et al., 2015), ICLA-EC-TSLP (Mozhdeh Khaksar Manshad et al., 2020), CMA-ES (Bliss et al., 2014) and ACO-LP (Sherkat et al., 2015) have been chosen to compare with MWCLA.

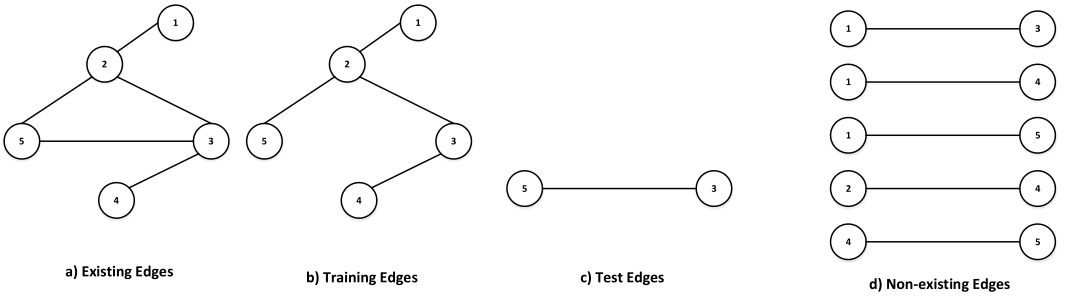


Figure 6. An example of both existing links and non-existing links as well as training and test sets.

In the MWCLA-LP for stopping metric, the following parameters are used: Maximum_Iteration = 100000, $\{w_1 = w_2 = w_3 = w_4 = 3\}$,$\widetilde{\lambda} = 0.005$, the aforesaid parameters are set based upon several experiments. The results that are reported in this paper are based on the averages taken over 20 runs. To perform experiments on co-authorship datasets, we consider data from 1993 to 2001 as our training data and then years 2002 through 2003 as our test data. We consider January 2000 through December 2001 as training data and May 2000 to April 2002 as test data for the Enron Email network. To evaluate the performance of the MWCLA-LP, we have used the various similarity algorithms introduced in Section 4. Thus, One-WCLA-LP-XX indicates that the introduced algorithm uses a similarity algorithm XX and predicts future links by propagating one wave in the environment. Similarly, Two-WCLA-LP-XX1-XX2 shows that the MWCLA uses two similarity metrics X×1 and X×2, for predicting future connections by propagating two waves in parallel, and so on.

The AUC values for MWCLA-LP and other methods for Hep-ph, Enron Email, Hep-th and Astro-ph datasets are shown in Table 2, respectively. The best results of the experiments in Table 2 are reported in bold.

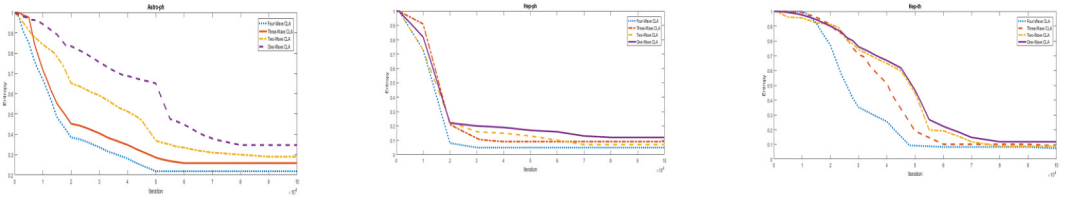
The gained results show that the Four-WCLA-XX1- X×2- X×3- X×4 algorithm gets the best AUC value for datasets in comparison with similarity-based algorithms. MWCLA-LP is better because it examines the probability of creating a connection between the two nodes in the future based on the different knowledge of similarity-based methods. MWCLA-LP is also superior to Katz and LP; therefore, it can be realised that the introduced MWCLA-LP has better performance than static methods. Similarly, MWCLA achieves better AUC than some supervised and LA-based algorithms, according to the results of Table 2. This is because the proposed algorithm has the capability of propagation and removes additional complexity in the network. The LA-TSLP method uses the local similarity metrics in time *1* to *T* to predict future connections. However, MWCLA-LP considers the local similarity criteria simultaneously. The information of selecting the test links is propagated in the network, so choosing the test links is adapted based on the whole network's structural information. As a result, MWCLA-LP achieves better results. Likewise, MWCLA-LP is superior to ICLA-EC-TSP and ICLA-LP. ICLA-EC uses the local information of itself and its neighbours at different times. ICLA-LP weighs the social network links based on the local subgraphs and then uses the weighted graph for predicting links in the

Table 2. Comparison of the MWCLA-LP with other methods on the co-authorship and email datasets based on AUC measure.

Methods/Datasets	Hep-th	Hep-ph	Astro-ph	Enron Email	Average
CN (Al Hasan & Zaki, 2011)	0.5577	0.5275	0.5350	0.5911	0.5528
JC (Jaccard, 1901)	0.5587	0.5385	0.5550	0.5916	0.5610
AA (Adamic & Adar, 2003)	0.5537	0.5420	0.5788	0.5841	0.5646
PA (Newman, 2001)	0.5212	0.5510	0.5188	0.4283	0.5048
Local Path (Zhou et al., 2009)	0.6370	0.6112	0.6322	0.6554	0.6340
Katz (Katz, 1953)	0.6478	0.6527	0.6602	0.6827	0.6609
ACO-LP (Sherkat et al., 2015)	0.5692	0.6253	0.5872	0.6122	0.5984
IP (Rossetti et al., 2015)	0.6894	0.6478	0.6634	0.6733	0.6685
MI-LP (Tan et al., 2014a)	0.7634	0.8034	0.7985	0.7078	0.7683
CMA-ES (Bliss et al., 2014)	0.6581	0.6311	0.6476	0.6702	0.6518
LA-TSLP (Moradabadi & Meybodi, 2017b)	0.7553	0.7947	0.7325	0.7022	0.7461
FLP-DLA (Moradabadi & Meybodi, 2017a)	0.7795	0.8201	0.8297	0.7212	0.7876
CALA-LP (Moradabadi & Meybodi, 2016)	0.7930	0.8527	0.8745	0.7395	0.8149
ICLA-EC-TSLP (Mozhdeh Khaksar Manshad et al., 2020)	0.7849	0.8515	0.8650	0.7212	0.8056
ICLA-LP (Manshad et al., 2021)	0.7905	0.8650	0.8558	0.7322	0.8108
One-WCLA-LP-CN	0.7623	0.8236	0.7635	0.7125	0.7723
One-WCLA-LP-JC	0.7615	0.8241	0.7627	0.7162	0.7714
One-WCLA-LP-AA	0.7661	0.8301	0.7745	0.7145	0.7736
One-WCLA-LP-PA	0.7601	0.8279	0.7688	0.7124	0.7714
Two-WCLA-LP-CN-PA	0.7824	0.8612	0.8135	0.7621	0.7913
Two-WCLA-LP-CN-AA	0.7846	0.8601	0.8325	0.7649	0.7824
Two-WCLA-LP-CN-JC	0.7815	0.8598	0.8412	0.7609	0.7995
Three-WCLA-LP-CN-JC-AA	0.8162	0.8861	0.8906	0.7831	0.8421
Four-WCLA-LP-CN-JC-AA-PA	0.8329	0.9114	0.9425	0.8035	0.8726

Table 3. Computation time comparison for the proposed algorithm (in seconds).

Methods/Datasets	Hep-th	Hep-ph	Astro-ph	Enron – Email
CN	118.35	127.22	261.12	2634.39
JC	124.69	129.68	253.57	2831.85
AA	119.62	123.96	241.39	2795.37
PA	73.29	86.33	169.95	1756.95
Local Path	130.58	146.28	286.34	3269.54
Katz	151.26	160.83	302.64	4519.54
FLP-DLA	5236.84	11342.96	18635.96	9635.47
CALA-LP	1263.28	1725.36	1735.66	5863.17
ICLA-EC-TSLP	2635.91	3012.58	3261.27	8647.93
ICLA-LP	4312.75	5863.91	6541.31	10893.57
MWCLA-LP	10543.25	12365.59	24856.59	30653.81



a) Astro-ph b) Hep-ph c) Hep-th

Figure 7. The convergence diagram of the proposed MWCLA using different module size.

future. The mentioned algorithms cannot propagate the information in the network, so MWCLA-LP has a better performance compared to ICLA-EC-TSLP and ICLA-LP.

Similarly, Tables 2 shows that as the number of LAs in the modules increases and as a result, the number of waves in the environment increases, the accuracy of the algorithm in link prediction increases. In other words, the algorithm's efficiency increases if more similarity criteria are examined simultaneously in the environment to predict the link in the network. Finally, we have used information entropy for evaluating the convergence behaviour of the MWCLA-LP.

We compare the computation time of the M-LP algorithm with other similarity-based algorithms for all four data sets to determine its performance in terms of usage time. Table 3 shows the computation time measured for the proposed algorithm. On the basis of its accuracy, one can see that the MWCLA-LP requires rational computation times. We calculate the average information entropy of the action probability of LAs for three data sets in Figure 7(a,b,c). During the proceedings of the algorithm, the average entropy of MWCLA-LP decreases. The results obtained in this study are shown in Figure 6 for module sizes of one (one-wave CLA), two (two-wave CLA), three (three-wave CLA), and four (four-wave CLA). Each wave has been propagated in a special environment {CN, PA, AA, JC} and calculated the probability of appearing a new link using the corresponding environments. The figures illustrate a faster speed of convergence for a more extensive module size.

Also, the best configuration for this algorithm based on the obtained results is $\tilde{\lambda} = 0.005$, as we mentioned earlier, this configuration has been used for all experiments in this paper.

Conclusion

In this paper, we propose MWCLA, which is an extension of CLA. The proposed method supports the idea of using a larger number of waves, and these waves operate in a parallel manner. The parallel operation requires a module of LAs instead of a single LA. The module's LAs share a common action probability distribution vector for all, which is updated once each time. Each LA in the module chooses an action independently and extracts reinforcement signal from the environment, which forms the parallelism in the algorithm. The learning algorithm for MWCLA has the ability to improve

the speed, while it maintains the required accuracy. This method is suitable for those problems where several actions can be simultaneously activated. The second feature of this approach is that each node's neighbours are its successors' nodes, and each LA in a module can generate a wave and forward the wave to the successors' nodes if its chosen action is opposed to its previous action (the propagation property). These properties described here are useful in many applications such as web mining, social network analysis and classification. Different types of CLAs can be extended to the parallelisation and propagation paradigm. We also evaluated the MWCLA by LP problem in social networks and analysed the behaviour of the MWCLA. The MWCLA has got better results in the LP problem because it considers different measures simultaneously as well as propagates the information through the social network.

Data availability statement

The data that support the findings of this study are available at Hep-th http://arxiv.org/archive/hep-th, Hep-ph http://arxiv.org/archive/hep-ph, Astro-ph http://arxiv.org/archive/astro-ph and Email-Enron http://www.cs.cmu.edu/~enron/ which are used in our experiments (Leskovec et al., 2007; Shetty & Adibi, 2004).

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Mozhdeh Khaksar Manshad http://orcid.org/0000-0002-7705-0183

References

Adamic, L. A., & Adar, E. (2003). Friends and neighbors on the web. *Social Networks*, *25*(3), 211–230. https://doi.org/10.1016/S0378-8733(03)00009-1

Akcora, C. G., Carminati, B., & Ferrari, E. (2013). User similarities on social networks. *Social Network Analysis and Mining*, *3*(3), 475–495. https://doi.org/10.1007/s13278-012-0090-8

Al Hasan, M., & Zaki, M. J. (2011). *A survey of link prediction in social networks (social network data analytics*. Springer.

Anderson, A., Huttenlocher, D., Kleinberg, J., & Leskovec, J. (2012). Effects of user similarity in social media. *Proceedings of the fifth ACM international conference on Web search and data mining*.

Barabâsi, A. -L., Jeong, H., Néda, Z., Ravasz, E., Schubert, A., & Vicsek, T. (2002). Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and Its Applications*, *311*(3–4), 590–614. https://doi.org/10.1016/S0378-4371(02)00736-7

Beigy, H., & Meybodi, M. R. (2004). A mathematical framework for cellular learning automata. *Advances in Complex Systems*, *7*(03n04), 295–319. https://doi.org/10.1142/S0219525904000202

Beigy, H., & Meybodi, M. R. (2007). Open synchronous cellular learning automata. *Advances in Complex Systems*, *10*(04), 527–556. https://doi.org/10.1142/S0219525907001264

Beigy, H., & Meybodi, M. R. (2008). Asynchronous cellular learning automata. *Automatica*, *44*(5), 1350–1357. https://doi.org/10.1016/j.automatica.2007.09.018

Bhattacharyya, P., Garg, A., & Wu, S. F. (2011). Analysis of user keyword similarity in online social networks. *Social Network Analysis and Mining*, *1*(3), 143–158. https://doi.org/10.1007/s13278-010-0006-4

Bliss, C. A., Frank, M. R., Danforth, C. M., & Dodds, P. S. (2014). An evolutionary algorithm approach to link prediction in dynamic social networks. *Journal of Computational Science*, *5*(5), 750–764. https://doi.org/10.1016/j.jocs.2014.01.003

Chen, L., Cui, J., Tang, X., Qian, Y., Li, Y., & Zhang, Y. (2021). Rlpath: A knowledge graph link prediction method using reinforcement learning based attentive relation path searching and representation learning. *Applied Intelligence*, *52*(4), 1–12. https://doi.org/10.1007/s10489-021-02672-0

Clauset, A., Moore, C., & Newman, M. E. J. (2008). Hierarchical structure and the prediction of missing links in networks. *Nature*, *453*(7191), 98–101. https://doi.org/10.1038/nature06830

Daliri Khomami, M. M., Rezvanian, A., Bagherpour, N., & Meybodi, M. R. (2018). Minimum positive influence dominating set and its application in influence maximization: A learning automata approach. *Applied Intelligence*, *48*(3), 570–593. https://doi.org/10.1007/s10489-017-0987-z

Das, D. (2011, December 9–11). A survey on cellular automata and its applications. *Global Trends in Computing and Communication Systems: 4th International Conference ObCom 2011*, Vellore, TN, India.

Daud, N. N., Ab Hamid, S. H., Saadoon, M., Sahran, F., & Anuar, N. B. (2020). Applications of link prediction in social networks: A review. *Journal of Network and Computer Applications*, *166*, 102716. https://doi.org/10.1016/j.jnca.2020.102716

Dong, Y., Tang, J., Wu, S., Tian, J., Chawla, N. V., Rao, J., & Cao, H. (2012). *IEEE 12th International conference on data mining*.

Esnaashari, M., & Meybodi, M. R. (2008). A cellular learning automata based clustering algorithm for wireless sensor networks. *Sensor Letters*, *6*(5), 723–735. https://doi.org/10.1166/sl.2008.m146

Esnaashari, M., & Meybodi, M. R. (2014). Irregular cellular learning automata. *IEEE Transactions on Cybernetics*, *45*(8), 1622–1632. https://doi.org/10.1109/TCYB.2014.2356591

Ghavipour, M., & Meybodi, M. R. (2017). Irregular cellular learning automata-based algorithm for sampling social networks. *Engineering Applications of Artificial Intelligence*, *59*, 244–259. https://doi.org/10.1016/j.engappai.2017.01.004

Guimerà, R., & Sales-Pardo, M. (2009). Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, *106*(52), 22073–22078.

Huang, Z., & Lin, D. K. J. (2009). The time-series link prediction problem with applications in communication surveillance. *INFORMS Journal on Computing*, *21*(2), 286–9856. https://doi.org/10.1287/ijoc.1080.0292

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull Soc Vaudoise Sci Nat*, *37*, 547–579. https://doi.org/10.5169/seals-266450

Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, *18*(1), 39–3123. https://doi.org/10.1007/BF02289026

Khaksar Manshad, M., Meybodi, M. R., & Salajegheh, A. (2022). A new multi-wave continuous action-set cellular learning automata for link prediction problem in weighted multi-layer social networks. *The Journal of Supercomputing*, *78*(17), 1–30. https://doi.org/10.1007/s11227-022-04615-z

Khomami, M. M. D., Rezvanian, A., & Meybodi, M. R. (2018). A new cellular learning automata-based algorithm for community detection in complex social networks. *Journal of Computational Science*, *24*, 413–426. https://doi.org/10.1016/j.jocs.2017.10.009

Leskovec, J., Kleinberg, J., & Faloutsos, C. (2007). Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *1*(1), 2-es. https://doi.org/10.1145/1217299.1217301.

Liben-nowell, D., & Kleinberg, J. (2007). The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, *58*(7), 1019–2882. https://doi.org/10.1002/asi.20591

Lim, M., Abdullah, A., Jhanjhi, N., & Khan, M. K. (2019). Situation-aware deep reinforcement learning link prediction model for evolving criminal networks. *IEEE Access*, *8*, 16550–16559. https://doi.org/10.1109/ACCESS.2019.2961805

Liu, H., Hu, Z., Haddadi, H., & Tian, H. (2013). Hidden link prediction based on node centrality and weak ties. *EPL (Europhysics Letters)*, *101*(1), 18004. https://doi.org/10.1209/0295-5075/101/18004

Manshad, M. K., Manshad, A. K., & Meybodi, M. R. (2012). *6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT)*.

Manshad, M. K., Meybodi, M. R., & Salajegheh, A. (2020). A new irregular cellular learning automata-based evolutionary computation for time series link prediction in social networks. *Applied Intelligence*, *51*, 1–14. https://doi.org/10.1209/s10489-020-01685-5

Manshad, M. K., Meybodi, M. R., & Salajegheh, A. (2021). A variable action set cellular learning automata-based algorithm for link prediction in online social networks. *The Journal of Supercomputing*, *77*, 1–29. ISBN: 0920–8542.

Menon, A. K., & Elkan, C. (2011). Link prediction via matrix factorization, *European Conference*, Athens, Greece, September 5–9, 2011.

Mirsaleh, M. R., & Meybodi, M. R. (2016). A Michigan memetic algorithm for solving the community detection problem in complex network. *Neurocomputing*, *214*, 535–2312. https://doi.org/10.1016/j.neucom.2016.06.030

Moradabadi, B., & Meybodi, M. R. (2016). Link prediction based on temporal similarity metrics using continuous action set learning automata. *Physica A: Statistical mechanics and its applications*. *Physica A: Statistical Mechanics and Its Applications*, *460*, 361–373. https://doi.org/10.1016/j.physa.2016.03.102

Moradabadi, B., & Meybodi, M. R. (2017a). Link prediction in fuzzy social networks using distributed learning automata. *Applied Intelligence*, *47*(3), 837–849. https://doi.org/10.1007/s10489-017-0933-0

Moradabadi, B., & Meybodi, M. R. (2017b). A novel time series link prediction method: Learning automata approach. *Physica A: Statistical Mechanics and Its Applications*, *482*, 422–432. https://doi.org/10.1016/j.physa.2017.04.019

Moradabadi, B., & Meybodi, M. R. (2018a). Link prediction in stochastic social networks: Learning automata approach. *Journal of Computational Science*, *24*, 313–328. https://doi.org/10.1016/j.jocs.2017.08.007

Moradabadi, B., & Meybodi, M. R. (2018b). Wavefront cellular learning automata. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *28*(2), 021101. https://doi.org/10.1063/1.5017852

Navid, A. H. F., & Aghababa, A. B. (2013). *Emerging Applications of Cellular Automata*, 85–111. https://doi.org/10.5772/52953

Newman, M. E. (2001). Clustering and preferential attachment in growing networks. *Physical Review E*, *64*(2), 025102. https://doi.org/10.1103/PhysRevE.64.025102

Patel, D. K., & More, S. A. (2013). Edge detection technique by fuzzy logic and Cellular Learning Automata using fuzzy image processing. *International Conference on Computer Communication and Informatics*.

Qiu, B., Ivanova, K., Yen, J., & Liu, P. (2010). Behavior evolution and event-driven growth dynamics in social networks. *IEEE Second International Conference on Social Computing*.

Rossetti, G., Guidotti, R., Pennacchioli, D., Pedreschi, D., & Giannotti, F. (2015). Interaction prediction in dynamic networks exploiting community discovery. *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*.

Saghiri, A. M., & Meybodi, M. R. (2017). A closed asynchronous dynamic model of cellular learning automata and its application to peer-to-peer networks. *Genetic Programming and Evolvable Machines*, *18*(3), 313–349. https://doi.org/10.1007/s10710-017-9299-7

Shang, K. -K., Li, T. -C., Small, M., Burton, D., & Wang, Y. (2019). Link prediction for tree-like networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *29*(6), 061103. https://doi.org/10.1063/1.5107440

Sherkat, E., Rahgozar, M., & Asadpour, M. (2015). Structural link prediction based on ant colony approach in social networks. *Physica A: Statistical Mechanics and Its Applications*, *419*, 80–94. https://doi.org/10.1016/j.physa.2014.10.011

Shetty, J., & Adibi, J., (2004). The Enron email dataset database schema and brief statistical report. *Information Sciences Institute Technical Report*, *4*(1), 120–128. *University of Southern California*

Tan, F., Xia, Y., Zhu, B., & Zhan, W. (2014a). Link prediction in complex networks: A mutual information perspective. *PLoS One*, *9*(9), e107056. https://doi.org/10.1371/journal.pone.0107056

Tan, F., Xia, Y., Zhu, B., & Zhan, W. (2014b). Link prediction in complex networks: A mutual information perspective. *PLoS One*, *9*(9), e107056. https://doi.org/10.1371/journal.pone.0107056

Thathachar, M. A. L., & Arvind, M. T. (1998). Parallel algorithms for modules of learning automata. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *28*(1), 24–33. https://doi.org/10.1109/3477.658575

Thathachar, M. A. L., & Sastry, P. S. (2002). Varieties of learning automata: An overview. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *32*(6), 1393–1407. https://doi.org/10.1109/TSMCB.2002.1049606

Vafashoar, R., Meybodi, M. R., & Azandaryani, A. H. M. (2012). CLA-DE: A hybrid model based on cellular learning automata for numerical optimization. *Applied Intelligence*, *36*(3), 735–748. https://doi.org/10.1007/s10489-011-0292-1

Valverde Rebaza, J., & de Andrade Lopes, A. (2013). Exploiting behaviors of communities of twitter users for link prediction. *Social Network Analysis and Mining*, *3*(4), 1063–1074. https://doi.org/10.1007/s13278-013-0142-8

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Reviews of Modern Physics*, *55*(3), 601. https://doi.org/10.1103/RevModPhys.55.601

Yang, S. -H., Long, B., Smola, A., Sadagopan, N., Zheng, Z., & Zha, H. (2011). Like like alike: Joint friendship and interest propagation in social networks. *Proceedings of the 20th international conference on World wide web*.

YounessZadeh, S., & Meybodi, M. R. (2018). A link prediction method based on learning automata in social networks. *Journal of Computer & Robotics*, *11*(1), 43–55.

Zhao, Y., Jiang, W., Li, S., Ma, Y., Su, G., & Lin, X. (2015). A cellular learning automata based algorithm for detecting community structure in complex networks. *Neurocomputing*, *151*, 1216–1226. https://doi.org/10.1016/j.neucom.2014.04.087

Zhou, T., Lü, L., & Zhang, Y. -C. (2009). Predicting missing links via local information. *The European Physical Journal B*, *71*(4), 623–630. https://doi.org/10.1140/epjb/e2009-00335-8

Appendix 1: Analysis of MWCLA's convergence

This part of the paper attends to the analysis of the convergence of the MWCLA. For analyzing the convergence, we assume an MWCLA with the following features. First, the structure of MWCLA, S, is a directed graph without any directed cycle (directed acyclic graph) with E as the collection of edges and V as the collection of vertices. Second, we assume a finite set of successor vertices at the predefined depth, $N(i)$, as the i^{th} cell's neighbourhood set.

The optimisation problem (9) proposed in Section 4 is considered as the goal of the MWCLA:

$$\text{maximize} f(P) = \sum_{\tau=1}^{n} E[\beta^{\tau}|P] = \sum_{\tau=1}^{n}\sum_{i=1}^{N}\sum_{j=1}^{r_i} E[\beta_i^{\tau}|\alpha_i^{\tau} = \alpha_{ij}, P] = \sum_{\tau=1}^{n}\sum_{i=1}^{N}\sum_{j=1}^{r_i}\sum_{y_1,\ldots,y_{n(i)}} \left(F_i^{\tau}(y_1,\ldots,y_{n(i)},\alpha_{ij}) \prod_{k\in N(i)} p_k^T\right) \text{ for any } \tau.$$

$$\text{subject to:}$$

$$p_{ij} \geq 0, \sum_{j=1}^{r_i} p_{ij} = 1, 1 \leq j \leq r_i 1 \leq i \leq N \tag{16}$$

where:

$P^T = (p_1^T, p_2^T, \ldots, p_N^T)$, $p_i^T = p_{i1}, p_{i2}, \ldots, p_{ir_i}$

The target of the MWCLA is to maximise the waves' reinforcement signals obtained from the random environment. We want to display that MWCLA maximises the reinforcement signals obtained from the environment.

Note that the analysis in this Section is independent of the MWCLA structure. Since in MWCLA, more than one LA in a cell is active at any time, and each has chosen an action, in this algorithm, only the distribution probability vectors of active LAs are updated, and the rest stay unchanged.

In essence, in this analysis, we need to understand how, over a long time, the LAs action probabilities evolve in the introduced learning algorithm (14). This is achieved by learning algorithm estimation using an *ordinary differential equation (ODE)*. Since the objective of the learning algorithm is to solve the optimisation problem Equation (9), a connection should be established between the asymptotic behaviour of the ODE and maximum $\text{off}(P) = \sum \tau = 1)^n = E[\beta^{\tau}P]$. We need to describe some definitions for approximating the learning algorithm.

Definition (7): The conditions for P, which is a solution to $f(P) = \sum_{\tau=1}^{n} E[\beta^{\tau}|P]$, is called Kuhn-Tucker conditions. In the following, we describe the First-Order Necessary Kuhn-Tucker conditions:

$$i,j\left\{\begin{array}{l} \frac{\partial f}{\partial p_{ij}} + \eta_i j + \gamma_i = 0 \\ \eta_i j p_i j = 0 \\ \eta_i j \geq 0 \\ p_i j \geq 0 \\ \sum_j p_{ij} = 1 \end{array}\right\} \tag{17}$$

In the MWCLA algorithm, a multiplicity of LAs in a module works in a parallel fashion in a random environment and is in some shared situation at each given moment. Each LA reacts to the situation by choosing an action and is then penalised or rewarded. Due to the fact that the LAs in a module of a cell have a shared internal state, they cooperate with each other in some cases. The actions chosen by each LA depend on the shared internal state, but the chosen actions are independent of each other. Here, considering the above definitions, the learning algorithm used with one wave is analysed and then developed the discussion for multiple waves that work in parallel. Each LA in the sequential case in a wave for updating its probability vector can use the L_{R-I} the learning algorithm is given below:

$$p_{ij}(k+1) = p_{ij}(k) + \lambda\beta(k)\big(1 - p_{ij}(k)\big) if \alpha_i(k) = \alpha_{ij}$$

$$p_{ij}(k+1) = (1 - \lambda\beta(k))p_{ij}(k) \ if \alpha_i(k) \neq \alpha_{ij} \tag{18}$$

Remark (1): Where (18) can be demonstrated in the following general form:

$$X(k+1) = X(k) + \lambda H(X(k), Y(k)); X(0) = x_0 \tag{19}$$

So that λ is the learning parameter and $0 \leq \lambda \leq 1$. $X(k)$ is $p(k)$. $Y(k)$ is a vector of the (α, β), which shows the chosen actions and reinforcement signal. H is the update rule which is made of the $H_{ij}, j = 1, 2, \ldots, r_i, i = 1, \ldots, N$ functions set.

Definition (8): For each $P = \left(p_1^T, \ldots, p_N^T\right)^T$ where p_i is a probability vector of r_i dimensions, we propose:

$$s_{ij}(P) = E\big[H_{ij}(P(k), Y(k))|P(k) = P\big] \tag{20}$$

Remark (2): For the learning algorithm (19), we have the following hypotheses:

$A_1 : \{(P(k), \beta(k)), k > 1\}$ is a Marco process.

$A_2 : prob[\beta(k) \in \beta|P(k), \beta(k-1)] = prob[\beta(k) \in \beta|P(k)]$ for Any appropriate Borel set

$A_3 : g(x) = E[H(P(k), \beta(k))|P(k) = x]$ is global Lipschitz and that is independent of k

$A_4 : H(P(k), \beta(k)) - g(P(k))^2 < M < \infty$ for some M and k

According to the above hypotheses, the approximating ODE for updating rule in algorithm (19) is defined as follows, and it has proven in (Thathachar & Arvind, 1998):

$$\frac{dx}{dt} = h(x), X(0) = x_0 \tag{21}$$

Theorem (1): For the learning algorithm (19) for one wave, the approximating ODE is:

$$\frac{dp_{ij}}{dt} = s_{ij}(p), j = 1, 2, \ldots, r, i = 1, 2, \ldots, N \tag{22}$$

Now, considering Remark (2), in (Moradabadi & Meybodi, 2018b), Theorem (1) has been proven for the algorithm (19).

As mentioned earlier, in the MWCLA, a number of waves work in parallel. Therefore, the algorithm for updating theL_{R-I} probability for MWCLA can be rewritten as (23). The L_{R-I} updating algorithm vector in MWCLA is achieved by adding up the update vectors in n waves, which uses the L_{R-I} algorithm in each wave individually.

Remark (3): So, the learning algorithm (19) for n waves in MWCLA can be rewritten as:

$$X(k+1) = X(k) + \tilde{\lambda}\sum_{\tau=1}^{n} H(X(k), \alpha^{\tau}(k), \beta^{\tau}(k)), X(0) = x_0 \tag{23}$$

Where $X(k)$ is replaced by *p(k)*.

According to the method for the one wave case and as we discussed earlier, the ODE for updating rule in MWCLA will be:

$$\frac{dx}{dt} = nh(x), X(0) = x_0 \tag{24}$$

Theorem (2): The approximating ODE for the learning algorithm in (23) for n waves is:

$$\frac{dp_{ij}}{dt} = ns_{ij}(p), j = 1, 2, \ldots, r, i = 1, 2, \ldots, N \tag{25}$$

ODE (22) and (25) include a set of stationary points with identical characteristics with no difference except that ODE (25) has been compressed n-fold on the time axis. If $\tilde{\lambda}$ enjoys the identical values for both ODEs, and n is thus that λ falls in the allowable range, for given time T, ODE (25) will be faster than ODE (22) based on factor n, so that using the (23), accuracy stands at the desired level.

As for approximate accuracy, there will be identical arguments for both sequential and parallel algorithms in the case of a unique stable equilibrium point. Please note that although the given algorithm works in a parallel fashion, it does not change the sequential algorithm's asymptotic properties. Please also note that as the value of n increases, the convergence speed does not necessarily increase because there are limits in the state space.

In this step, we want to study the ODE of MWCLA, so we calculate the s_{ij}.

Definition (9): Drift Δp_{ij} for the j^{th} component of the vector p_i is defined as:

$$\Delta p_{ij} = E\left[p_{ij}(k+1) - p_{ij}(k) | P(k) = P\right] \tag{26}$$

Based on the Definition (9) and Equation (20) Δp_{ij} for one wave can be defined:

$$\Delta p_{ij} = \lambda s_{ij}(P(k)) \tag{27}$$

And since $\tilde{\lambda} = \frac{\lambda}{n}$, then (27) for n waves in MWCLA has been written as:

$$\Delta p_{ij} = n\tilde{\lambda} s_{ij}(P(k)) \tag{28}$$

Lemma (1): The drift for (20) is

$$\Delta p_{ij} = \lambda p_{ij}\sum_{s} p_{is}\left(\frac{\partial f}{\partial p_{ij}} - \frac{\partial f}{\partial p_{is}}\right) = n\tilde{\lambda} p_{ij}\sum_{s} p_{is}\left(\frac{\partial f}{\partial p_{ij}} - \frac{\partial f}{\partial p_{is}}\right) \tag{29}$$

$f(.)$ is defined by (9), and its proof is given in Reference (Thathachar & Arvind, 1998).

Based on the (28) and (29), we have:

$$s_{ij}(P) = p_{ij}\sum_{s} p_{is}\left(\frac{\partial f}{\partial p_{ij}} - \frac{\partial f}{\partial p_{is}}\right) \tag{30}$$

So, the approximate ODE for the MWCLA algorithm will be demonstrated by (25), whereas s_{ij} has been demonstrated with (30). Now we have to show the solutions for ODE. As mentioned earlier, our goal is to confirm that the optimisation problem of maximising $f(P)$ would be solved by the ODE solutions. To do that, we first need to show that function $f(P(t))$ is an increasing monotonically function along ODE's path.

Lemma (2): Let's assume that u is a dummy variable with probable values such as space value on vector$P$. Now along ODE (20), we have:

$$\frac{df}{dt}(u) > 0 \tag{31}$$

Also, in the case that u is an equilibrium point for ODE, then $\frac{df}{dt}(u) = 0$. It has been proven in (Moradabadi & Meybodi, 2018b).

Lemma (3): If $u$satisfies the FONKT conditions in (17) for optimisation problem (9), then it will be an equilibrium point for ODE (25). However, if it is not satisfying FONKT and is also an equilibrium point for ODE, then it is unstable.

In Reference (Moradabadi & Meybodi, 2018b), this lemma has been proven for ODE (22), and since ODE (25) is an n-factor in comparison to ODE (22), this lemma can be easily proven through the given considerations.

Lemma (4): if u is a stable equilibrium point and satisfies FONKT conditions, then it is a maximum for optimisation problem (9), and vice versa. The proof is given in Reference (Thathachar & Arvind, 1998).