Computer & Robotics

Learning Automata-Based Approaches to Infer Urban Structure from Traffic Dynamics Considering Costs

Hamid Yasinian[a], Mansour Esmaeilpour[b,*]

[a] *Computer Engineering Department, Central Tehran Branch, Islamic Azad University, Tehran, Iran*
[b] *Computer Engineering Department, Hamedan Branch, Islamic Azad University, Hamedan, Iran*
Received 02 November 2022; Accepted 02 December 2022

Abstract
Successful future urban planning is highly dependent on optimal connectivity between important areas of cities. Discovering essential latent links will optimize the urban structure. Moving towards a better structure requires some information. There are a lot of sources of information for urban structure inferring, including the current structure, the time-varying traffic dynamics, and the construction costs, which are the basics of the optimization problem formulation. This paper presents a new formulation for the problem. The model problem to be solved tries to utilize all data sources needed for inferring. There are some methods for solving the formulated problem. The methods need some development to apply to the model. Methods utilizing learning automata (LA) are favorable in this field due to their interaction with the environment. This paper presents two LA-based approaches for the model: Distributed Learning Automata (DLA) and Cellular Learning Automata (CLA). The algorithms result in an optimal connectivity matrix considering urban structure, traffic dynamics, and costs, where the matrix must include the current urban structure and some new reasonable necessary links. Moreover, comparisons are possible because the model has a fitness value for evaluating the provided connectivity matrix. The CLA-based proposed method performed better than the others in most experiments.

Keywords: Urban Structure, Traffic Dynamics, Optimal Connectivity Structure, Distributed Learning Automata, Cellular Learning Automata.

1.Introduction

Social development and the urban economy depend highly on urban traffic network studies. In modern cities, we can see the effects of traffic and its consequences. The expansion of traffic networks has developed these cities, and urban transformation affects the urban traffic pattern in return. There are many essential processes in urban traffic planning, such as traffic forecasting, background prediction, project evaluation, traffic survey layout, and scheme design. The urban traffic network design is the most critical point in the layout scheme design due to its determining capability of future urban planning for operational efficiency and economic development [1]. Ever-changing city patterns and challenges in urban development have convinced researchers to pay more attention to studies about traffic network structure affecting urban traffic distribution in recent years [2].

A high degree of synchronicity is there in the urban traffic network. Therefore, to reach efficient traffic management strategies, we must consider many features for appropriate models of these systems. In particular, such systems can be examples of event-driven and synchronous ones. The complex interactions of different discrete events can be the basis for their dynamics [3]. The data gathered from dynamics can provide valuable information about the urban components' behavior. Helpful knowledge for developing more efficient cities and planning better decisions can is achievable by analyzing this data. So traffic can be considered a critical component of an urban ecosystem[4].

There are different patterns over time in urban traffic. Therefore, vehicle distribution could be more stable due to the diversity in traffic patterns, and global traffic pattern modeling in a complex road

* Corresponding Author. Email: ma.esmaeilpour@gmail.com

network full of intersections is challenging [5]. There is much information about the traffic flow of urban administrative areas in real-time traffic data. The ever-varying traffic dynamics is a valuable source for effectively deriving traffic connection structure representing a dynamic picture of areas closely associated with the city based on traffic flow information of different hours during weekdays [6].

The structure of an urban area imposes functional activities in cities. There might be considerable interactions between two city areas, but no direct or structural connection, so unconnected urban areas may interact. Finding these areas is a challenging task. An even more challenging issue is finding areas that interact strongly at particular times of the day. The optimal connection structure can explain a dense functional correlation of traffic flow in different city areas, indicating which areas are most similar in terms of their traffic flow characteristics at different times of the day. The functional traffic dynamics and fixed road network formation are the essential elements of the effective connection structure [7]. There is another determinative factor for optimal structure, and it is the cost of modifying the network. The cost has many different aspects, such as construction and demolition costs.

Formulating the problem is the first step of optimal urban structure, inferring from real-time traffic data. The second step is to design a method to discover the most correlated zones considering costs. Traffic dynamics, costs, and current structure are inputs of the method. Two LA-Based algorithms are the proposed methods of the research. Learning automaton, or LA, is a coputational intelligence technique that can solve many real-world and complex problems with high uncertainty or lack of knowledge about the environment [8, 9]. Reinforcement learning literature considers LA a stochastic model and classifies them as a temporal-difference (TD) learning method [10]. TD methods update estimates based partially on the other learned estimates without waiting for an outcome, like dynamic programming [11].

Distributed Learning Automata is the algorithm used in one of the proposed methods. Due to the LA's limited capability, the interconnection of multiple LAs in various forms, like a tree, a mesh, or an array, increases their ability. The cooperation of these automata solves a problem [12]. DLA has been used successfully for solving complex problems [13-17]. *Cellular Learning Automata* is the 0algorithm used to design the second proposed method. A Cellular Automaton (CA) model consists of many simple cells passing through a group of states based on local interaction. The group of a cell's neighborhood produces its local environment based on the definition of the cell's local rule. Each cell selects a state among a finite set of states. Each cell's state, along with its adjacency cells' state, is called that cell's configuration. The CAs' local rule indicates their evolution via time and implies CAs' configuration changes in a stage. CA is suitable for modeling systems with locally-connected, simple building blocks. The combination of CAs and LAs presents a new CLA model [18]. This combination creates a distributed computation model that utilizes CAs' computation power and LAs' learning capabilities. CLA is an enriched version of CA due to the learning ability of optimal action. It also outperforms LA because of LA's interactions [19]. CLA has found many applications in various areas [20-25].

The network structure in this work remains unchanged during the research period. The time-dependent traffic flow produces functional dynamics data. There is a cost factor in the proposed problem definition to develop a possibility of urban structure inferring close to reality using the data at hand. Therefore, the definition satisfies three criteria: similarity of the optimal and current structures, the optimal structure functionality in conducting the traffic flow, and cost-effectiveness. The paper's contribution is to formulate and solve a developing problem (inferring optimal city structure) where there needs to be more attentive to the costs of creating a new structure in the real world. As long as there is a need to track the performance of a new structure towards optimality, the methods here must be interactive to solve the problem. This section mentioned the optimal urban structure inferring and briefly described the research. Section 2 presents a literature review to determine the advances in this area. The formulation for the problem definition is in section 3. Section 4 proposes algorithms for inferring the effective urban structure. In section 5, we find the testbed and the experimental result. A conclusion takes place in the final section to discuss the achievements.

2. Literature Review

Despite recent improvements in intelligent transportation, analyzing and demonstrating traffic dynamics still needs to be improved. [26] considered urban intersection and multi-dimensional vehicle-vehicle interaction models, which are pretty complex and challenging on a large scale. They also optimized the urban traffic graph using efficient computational simulation and proposed basic innovative methods. Simulating Realistic Manhattan topologies with different types of intersections helps us understand different traffic patterns and calculate the average travel delay. As long as modeling massive urban data flows requires more rigorous approaches, traditional models suffer theoretical limitations in complex systems. [27] described a conceptual framework for modeling city traffic dynamics and offered a way to confine complexity based on the abstract power of Markov chains. The test suite of the proposed model was a real-world data set of taxis' GPS footprints in Beijing.

Novel technologies can gather real-time data from urban areas and estimate their distribution of traffic for further analysis. Nowadays, we can extract valuable scenarios from such data. [28] extracted congestion hotspots in urban areas using a model based on essential cases in complex networks. The model could identify sensitive points that can become hotspots increasing mobility demand. [29] Urban transportation networks were highly uncertain, unpredictable, complex, and stochastic. The author proposed a framework based on the Markovian approach to model dynamic network traffic and used public Google Maps data of downtown Baltimore to validate the model. A statistical method calculated the complexities of transportation.

Urban dynamics detection in multi-layer network growth is a research topic providing scholars with a new way of considering structure-changing trends and respected impacts. A quantitative research method examined network centrality, accessibility, and community partition in [30], focusing on the growth of the upper layer (rail network). The researchers found that when a rail network grows from a simple tree-like network to a more intricate form, the network diameter and the average shortest path length of multi-layer networks decrease dramatically in the case study of Kuala Lumpur. The network expansion ability has been changing, and more rail stations in the city center had a higher ability for future expansion. They discussed the different performances of these nodes added in the multi-layer network to show the impact on the repartition of network communities and the number of decreasing communities. The research could help scholars understand and apply these network dynamic computational techniques.

The mathematical computer modeling of cities has roots in the past. The core elements are the structural evolution dynamics and flow models (spatial interaction). [31] developed an urban structure stochastic model to deal with uncertainty arising from unexpected events. They presented two noteworthy results: the representation of the structural variables via a single potential function and evolution modeling by developing stochastic differential equations and showing the ability to estimate parameters of the spatial interaction model from the structure alone, independently of flow data, using the Bayesian inferential framework. Markov chain Monte Carlo methods were their solution to overcome significant computational challenges of the posterior distribution in the case study on the London, UK retail system.

[32] investigated the role of evolving urban spatial structure in commuting patterns of Beijing, China. They identified persisting, emerging, and non-center areas between 2000-2008 in Beijing's three subregions to describe the multi-dimensional and dynamic urban spatial evolution. They stated that commuting differs between persisting, emerging, and non-center areas and, more notably, across the subregions. Emerging center areas generated longer commutes than other similar areas. Commutes to emerging centers were shorter in the inner-ring suburbs compared to persisting centers. Emerging center areas incurred the longest commutes in the outer-ring suburbs, while persisting center areas were the shortest. They reflected changing economic and urban functions in distinct city subregions and warned about the increase in commutes and relevant adverse side effects in developing polycentric urban Chinese cities.

There are high maintenance and operation costs in the universal deployment of devices in urban flow monitoring systems. There is a demand for techniques to reduce the number of deployed devices without degenerating granularity and data accuracy. [33] presented an approach to infer the fine-grained, real-time crowd flows throughout a city based on coarse-grained observations. This task has two challenges: the spatial correlations between coarse- and fine-grained urban flows and the complexities of the external effects. A developed model called

UrbanFM handled these challenges. It consisted of two major parts: an inference network to generate fine-grained flow distributions from coarse-grained inputs using a novel distributional upsampling and feature extraction module and a general fusion subnet to increase the performance considering the influence of various external aspects. They reported the remarkable effectiveness and efficiency of the structure for small-scale upsampling. The single-pass upsampling used by UrbanFM was insufficient at higher upscaling rates. Hence, they presented UrbanPy, a cascading model for advanced inference of fine-grained urban flows decomposing the original tasks into numerous subtasks. This improved structure demonstrated more satisfactory performance in larger-scale inference tasks compared to UrbanFM.

Intelligent transportation systems guarantee public safety and are critical. Two aspects make such systems very complicated: complex Spatio-temporal correlations of traffic, including spatial correlations between locations among timestamps along with temporal correlations, and a variety of such spatiotemporal correlations depending on the surrounding geographical information (varying from location to location, e.g., points of interests and road networks). A deep-meta-learning-based model named ST-MetaNet was proposed in [34] to handle challenges mentioned collectively by predicting traffic in all locations simultaneously. ST-MetaNet employed a sequence-to-sequence architecture containing an encoder to learn historical information and a decoder to make step-by-step predictions. There were embedded recurrent neural networks in both the encoder and decoder of the traffic. The extensive experiments illustrated the ability of ST-MetaNet on two real-world datasets.

Surveying different characteristics and abilities of various network structures under traffic congestion were the goal of the research [35] in response to shortcomings in single-layer networks. Simulation and a comparative experiment guided us to obtain optimal multi-layer urban traffic network topology under different conditions. They found that scale-free interconnected multi-layer networks have a relatively powerful ability to support more traffic and have higher anti-congestion capacities. The research results helped deepen our understanding of the traffic network structures' characteristics.

Weather and human interactions are unpredictable observed elements in a vehicle traffic network. They create dynamic systems with high complexity and non-linearity. Consequently, modeling the evolution of traffic systems over time is complicated through mathematical or modern patterns. Robust traffic analysis and forecasting approaches are in great demand for transportation systems. [36] offered a model-free data-driven approach to analyze and forecast traffic dynamics using the Koopman mode decomposition. Their work included the reconstruction of observed data, distinguishing decaying or growing patterns, and obtaining hierarchies of previously identified and never before identified spatiotemporal patterns from data sets delivered by the California Department of Transportation and the Federal Highway Administration. They claimed that forecasting highway network conditions are possible using this methodology.

3. Problem Formulation

Definition of structural connectivity (S) depends on adjacency, travel time, distance, or other information about connections between zones. The $n \times n$ adjacency matrix of a weighted graph $G_S = (V, L_S)$ is the structural connectivity. V is the set of n nodes (zones) and L_S is the edges representing structural connectivity between zones. An $n \times m$ matrix B forms S , where $B_{ij} = 1$ if road j passes through zone i . Connection strength between zones is the implication of the structural connectivity equation [6]:

$$S = BB^T \tag{1}$$

S_{ij} is the number of main roads linking i and j .

Structural connectivity is not the only factor, and there is a functional connectivity factor between zones (F). An $n \times n$ symmetric adjacency matrix of a weighted $G_f = (V, L_f)$ graph is functional connectivity. L_f is the edges representing functional connectivity between zones. An $n \times T$ matrix D forms F where D_{it} represents the traffic volume or the number of vehicles passing the road i at time t .

An $n \times T$ matrix D F is obtained using from which D_{it} represents the traffic volume or the number of vehicles passing the road i at time t . Functional

connectivity reveals functional similarity between zones and it is as follows:

$$F = D \bullet D \tag{2}$$

is an operator to calculate the absolute value of one minus distance as the similarity. The proposed formulation needs a distance metric, and Bhattacharyya distance [37] measures the similarity or closeness between two discrete or continuous normal distributions:

$$F_{ij} = \begin{cases} \left| 1 - \left(\sqrt{1 - \sum_{t=1}^{T} \frac{\sqrt{D_{it} \times D_{jt}}}{\sqrt{\sum_{t=1}^{T} D_{it} \sum_{t=1}^{T} D_{jt}}}} \right) \right| & i \neq j \\ 1 & i = j \end{cases} \tag{3}$$

In equation**Error! Reference source not found.**, T is the number of periods and F is a symmetric matrix.

Inferring an optimal connectivity matrix (O) is the problem here. Although O can represent the existing structure (S), it should also reveal other time-dependent structural connectivity relationships that can represent high inferred connectivity between the zones demanding connection according to the traffic dynamics hidden in the present structure. O needs to represent the essential edges and some new links that are very high-demanded and rare.

The objective is to maximize a fitness function's value, consisting of two sub-functions. The first one (Y_1) is the affordability of creating the estimated optimal connectivity ($\hat{O}$) and the structural connectivity (S). Higher values are profitable (equation**Error! Reference source not found.**). The second sub-function (Y_2) is the quality added to the structure by changing links. Allocating more roads to more dynamically similar zones is considered the quality of the estimated structure (equation**Error! Reference source not found.**). Therefore the objective function (Y) is as follows:

$$Y = \alpha \times Y_1 + \beta \times Y_2 \tag{4}$$

Where:

$$Y_1 = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Aff_{ij} \tag{5}$$

$$Y_2 = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{\left(\hat{O}_{ij} \Big/ \max\left(\hat{O}\right) \right) \times F_{ij}}{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} F_{ij}} \tag{6}$$

In formula **Error! Reference source not found.** α and β are constant parameters regulating the influence of sub-functions on the objective function ($\alpha+\beta=1$).The affordability in equation **Error! Reference source not found.** is the inverse of the sum of the costs over time.

The formulation has two types of costs: demolition (Dem) and construction (Con) costs. The following formula takes advantage of the difference between the estimated optimal connectivity ($\hat{O}$) and the structural connectivity (S) to integrate normalized value of these costs:

$$Cost_{ij}(t)=\begin{cases} Dem_{ijk} \Big/ \max\left(\max(Dem),\max(Con)\right) & \hat{O}_{ij}(t+1)<\hat{O}_{ij}(t) \\ Con_{ijk} \Big/ \max\left(\max(Dem),\max(Con)\right) & \hat{O}_{ij}(t+1)>\hat{O}_{ij}(t) \\ 0 & \hat{O}_{ij}(t+1)=\hat{O}_{ij}(t) \end{cases} \quad (7)$$

If there is no connection between nodes $(i, j)\, k=0$, so when we want to connect them, $k=k+1$ and we must pay $Cost_{ijk}$. It means that the cost of linking two zones differs each time, e.g., the cost of constructing a road between two zones for the first time is different from the second time.

4. Problem Formulation

Our proposed methods are LA-based ones. A learning automaton, as a machine learning algorithm studied since the 1970s, selects its current action based on past experiences from the environment. It will fall into the range of reinforcement learning if the environment is stochastic and utilizes the Markov Decision Process (MDP). Before presenting the proposed LA-based methods, we need to explain some concepts.

4.1. Reinforcement Learning

Machine learning explores and engineers algorithms and methods on which systems and computers can learn. It is one of the comprehensive and widely used units of artificial intelligence. The machine learning goal is that computers, in the most general sense of them, can find a better performance by utilizing data gradually. Reinforcement learning, or RL, is an interactive strategy different from learning by monitoring. This type of learning uses actual measurements and signals for indirect learning to diagnose the inaccuracy or correctness of the learning procedure. In other words, the accumulated knowledge is reinforced or weakened by rewards or penalties [11].

4.1.1. Learning Automata

A learning automaton can be considered an abstract object with a limited action set. Learning automata works by choosing an action from the set of actions and applying it to the environment. A randomized environment evaluates the action, and automata use the environment's reaction to choose the next step. Automata learn to choose optimal action during this process. The automata learning algorithm determines the usage of the environment's reaction to the selected action that selects the following automata action. Two sorts of learning automata are automata with fixed structure and automata with variable structure. In a static structure stochastic automata, the probability of the actions is unchangeable, such as in types like Testline, Krinsky, TestlineG, and Krylov. While in variable structure stochastic automata, the actions' probability changes in each repeat.

4.1.2. Stochastic automata With Variable Structure

The modifications in the actions' probability are performed based on the learning algorithm in variable structure learning automata. In this type of automata, the representation of the automata's inner state (φ^i) is by the actions' probability, called the probability vector ($p^i=\left\{p_1^i, p_2^i, \ldots, p_r^i\right\}$). The learning algorithm is a repetitive relationship to modify the state probability vector. If the environment's response is proper, the action is rewarded (equation**Error! Reference source not found.**) and penalized otherwise (equation **Error! Reference source not found.**). The learning algorithm is a determining factor of the learning automata performance to update the actions' probability. There are different learning algorithms in the literature [38]. The general form of the learning algorithm is as follows (if the selected action is i in this step):

A: Appropriate Environment's Response

$$\begin{aligned} p_i(n+1)&=p_i(n)+a[1-p_i(n)] \\ p_j(n+1)&=(1-a)\,p_j(n) \qquad \forall j,\ j\neq i \end{aligned} \quad (8)$$

B: Inappropriate Environment's Response

$$p_i(n+1)=(1-b)\,p_i(n) \qquad (9)$$
$$p_j(n+1)=\frac{b}{r-1}+(1-b)\,p_j(n) \quad \forall j,\ j\neq i$$

Where r is the number of automata actions, a and b are the reward and penalty parameters respectively. Regarding to a and b values, three conditions can be assumed. If a and b are equal, learning automaton is called L_{RP}, if b equals zero, it is called L_{RI} and if $b<<a$ it is named $L_{R\varepsilon P}$.

4.1.3. Distributed Learning Automata

The distributed learning automata (DLA) is an automata network that concertedly collaborates to solve a particular problem. A directed graph models a DLA, and the set of nodes in the graph represents the set of automata. Each node's outgoing edges represent the corresponding automaton's group of actions. The automaton action selection causes a different automaton on the other side of the edge to be activated, corresponding to the selected action. At any time, there is just one active automaton in the network. We formally define DLA with n learning automata as a graph (A, E). $A=\{A_1, A_2,\ldots, A_n\}$ is the set of automata, and $E\subset A\times A$ is the set of edges in the graph. The edge (i, j) corresponds to the action α_j of the automaton A_i. The action probability vector for learning automaton A_j is p^j where p_m^j represents the probability of selecting action α_m by A_j.

4.1.4. Cellular Automata

Cellular Automata are a distributed computing model providing a platform for performing complex computations through local interactions. CA is a lattice of similar simple units or cells. Putting many cells together produces complex behavioral patterns because of the cells' interactions with their neighbors. The update of each cell's state is according to its current state and neighbors using the CA's updating rule. $\langle Z^d, \Phi, N, F\rangle$ represents a d-dimensional CA, where Z^d is a lattice of d-tuples from integer numbers. Z^d shows each cell in $(z_1; z_2;:::; z_D)$, and $F = f1; 2;:::; kg$ is a finite set of states. $N=\{X_1, X_2,\ldots, X_m\}$ or neighbor vector is a finite subset of the Z^d. The neighbor vector determines the relative positions of neighboring cells for any given cell in the Z^d. Neighbors of a cell u are the set $\{u+x_i \mid i=1,2,\ldots m\}$ and finally, $F:\Phi^m\rightarrow\Phi$ is the local rule of the CA, which produces the cell's new state considering the current neighbors' state [39].

There have been many different structural variations of CA proposed since its beginning. CA can be classified based on different features. According to the possible states for the cells, CAs can be binary and multi-valued. The applied local rule to each cell may be identical or different. We refer to these two possibilities as uniform and hybrid CA. Most local rules are deterministic, although there are variations of probabilistic [40] or fuzzy [41] local rules. Cells' states lay down the local rules; thus, all cells update their state simultaneously, and These CA are synchronized. On the contrary, local rules might be state and time-dependent, which results in asynchronous CA. Choosing an appropriate model depends on the given application.

4.1.5. Cellular Learning Automata

Every cellular learning automaton includes a set of learning automata. The learning automata in each cell determine the state of the corresponding cell based on its action probability vectors. There is an operation rule for CLA similar to cellular automata. The rule and the selected actions of the neighbor cells determine the reinforcement signal provided to the learning automata residing in that cell. In CLA, neighboring learning automata in any cell contain their local environment. This environment is not stationary as long as it changes and is affected by the learning automata actions [42]. A d-dimensional cellular learning automaton is a structure $\langle Z^d, \Phi, A, N, F\rangle$. A is a set of LA assigned to cells of the CLA. $F:\Phi^m\rightarrow\beta$ is a function to compute the reinforcement signal for each LA based on the state of the neighbouring LA, where β is the possible values for the reinforcement signal.

Cellular learning automata work as follows: the initial step is defining each cell's state according to the probability vectors of the LA in that cell. The

previous experiments arbitrarily determine the initial state of the CLA. In the second stage, each cell's local rule determines the cell's reinforcement signal. Finally, each LA updates its probability vector using the reinforcement signal and the cell's chosen action. This process continues until some condition is satisfied.

In most applications, all cells synch with a global clock and operate simultaneously. However, in some cases, activation of different cells' LA is asynchronous (asynchronous cellular learning automata (ACLA)) [43]. In standard CLA or close CLA, each cell only interacts with its local environment, including neighboring cells. Some applications also have a global environment, and cells interact with the local and external global environments. This kind is open cellular learning automata (OCLA) [44]. CLA containing multiple LA in each cell is another type of CLA. Several LA serves each cell in this kind of CLA, and the activation of LA can be synchronous or asynchronous [45].

4.2. Optimal Connectivity Inferring

In this section, we propose methods for inferring structural connectivity. A must-finding hidden structure is the optimal connectivity matrix that the proposed methods (DLA-based and CLA-based) try to discover. Each node in the graph $G_S=(V,L_S)$ represents a zone with a learning automaton. The graph is complete, so an automaton has $n-1$ actions (n is the number of zones), and the actions are selecting other city zones. To infer the optimal connectivity matrix, there are two proposed algorithms in this paper. $\hat{O}$ is the estimated optimal connectivity matrix, an $n\times n$ one in which all of its elements are zero when the algorithms begin.

4.2.1. Algorithm 1: Optimal Connectivity Inferring Using DLA

We modify the algorithm presented in [46] here to deal with inferring urban structure from traffic dynamics bringing costs up.

An automaton i related to the randomly chosen zone in the algorithm selects an action j first. After choosing the action by the automaton i, the update of $\hat{O}$ takes place, and $\hat{O}_{ij}=\hat{O}_{ij}+1$, $\hat{O}_{ji}=\hat{O}_{ji}+1$. When the automaton's action updates $\hat{O}$, that automaton is rewarded or penalized by changing the corresponding probability vector. Rewarding or penalizing is carried out based on the increase or decrease in the objective function value Y (formula**Error! Reference source not found.**) compared to the previous values of Y. If the Y value corresponding to the new $\hat{O}$ is higher than the highest value achieved, the algorithm rewards the automaton or penalizes otherwise. The algorithm repeats the process to constitute the optimal connectivity matrix until there is no progress in Y value for more than a predefined number of action selections (ne). The method uses a smoother vector (s) instead of the probability vector for action selection which is:

$$s_j^i = \frac{\left(\frac{p_{ij}}{r-1-\deg(v_i)}+\bar{V}_{ij}\right)^{\theta}}{\sum_{i=1}^{r}\left(\frac{p_{ij}}{r-1-\deg(v_i)}+\bar{V}_{ij}\right)^{\theta}} \tag{10}$$

In equation**Error! Reference source not found.**, $\theta\geq 1$ is a normalizing factor. $\deg(v_i)$ is the degree of node v_i, and $\bar{V}$ is the normalized functional connectivity covariance matrix:

$$\bar{V}_{ij} = \frac{V_{ij}}{\sum_{j=1}^{n} V_{ij}} \tag{11}$$

In equation **Error! Reference source not found.**, V is the functional connectivity covariance matrix defined by equation **Error! Reference source not found.** [6]:

$$V=DD^T \tag{12}$$

After reaching the stop criterion, the last $\hat{O}$ with the highest objective value is considered the best estimated optimal connectivity in this algorithm iteration. The method starts from another random point using an updated probability vector to form another $\hat{O}$. After repeating $n1$ times, the best $\hat{O}$ is the one with the highest Y value. The algorithm iterates $n2$ times to escape from trapping in local optimums likely to happen because of convergence to a non-optimal probability vector.

DLA Based Optimal Connectivity Inferring (Algorithm1)

Input: V, S, F, a, b, α, and β

Output: O

1: **Repeat**
2: Initialize P
3: **Repeat**
4: Initialize $\hat{O}$
5: Choose a random zone
6: **Repeat**
7: Select an action by the current automaton according to s (relation **Error! Reference source not found.**).
8: Update estimated optimal connectivity matrix $\hat{O}$
9: Calculate objective function Y (relation**Error! Reference source not found.**)
10: Update p (relations **Error! Reference source not found.** and **Error! Reference source not found.**)
11: **Until** reaching stop criterion
12: Compare $\hat{O}_{best}$ with the $\hat{O}$ and replace $\hat{O}_{best}$ with $\hat{O}$ if new Y value is higher.
13: **Until** $n1$ $n2$ iterations
14: Compare O with the $\hat{O}_{best}$ and replace O with $\hat{O}_{best}$ using objective function value Y.
15: **Until** $n1$ iterations

4.3. Algorithm 2: Optimal Connectivity Inferring using CLA

Algorithm 2 is a method to detect unconnected sections and then integrate them, maximizing the fitness function. CLA's learning nature results in optimal connectivity. The following passage discusses the proposed algorithm in 7 steps:

1. Creation of irregular cellular learning automata.
2. Assigning one cell to each zone and one LA to each cell.
3. Action selection by all cells in a parallel manner and adding the selected zones to the string inside each cell.
4. If the selection of an area is over the threshold (its degree here), checking the fitness function value before and after the selection is necessary. Increasing the value makes for updating the optimal structure.
5. Repeating step 3 until achieving the stop criterion.
6. Discovering the connected zones to each zone by the strings inside each cell.
7. Integrating the acquired strings by a greedy search algorithm maximizing the fitness value.
8. Calculating the final fitness value.
9. Checking improvement in the fitness function value (reward for all actions in case of success or otherwise penalization).
10. Returning to step 2 until reaching the stop criterion.In step 1 n cellular automata are generated each with an empty strings. These strings can be the neighbors of the zones discovered by the method. Each CLA is assigned to a zone and equipped with an LA. CLAs select actions using formula **Error! Reference source not found.** in step 3. The optimal connectivity matrix is updated if the condition described in step 4 is met. The internal loop is terminated should no progress occur in Y value after a predefined number of iterations (ne). The connected areas to each zone is determined by the string inside its corresponding CLA in step 6. If j is shown up s_n times in the string belonging to the CLA i, then $\hat{O}_{ij} = s_n, \hat{O}_{ji} = s_n$. In step 6, the fitness function value is first calculated for the current achieved structure, then two strings with the most connections between their vertices (the edges between nodes of two strings are maximal) are combined, and the fitness function is recalculated. If the fitness value increases, the combination process continues, otherwise it stops. The internal loop terminates and the estimated optimal structure is found.

CLA Based Optimal Connectivity Inferring (Algorithm 2)

Input: V, S, F, a, b, α, and β

Output: O

1: **Repeat**
2: Initialize P
3: **Repeat**
4: Initialize $\hat{O}$
5: Assign a cellular automaton to each zone and put an automaton in each CLA
6: **Repeat**
7: **For** all cells do in parallel
8: Select an action by the current automaton according to s (relation **Error! Reference source not found.**).
9: Check if selected areas are more than their degree and update $\hat{O}$ respectively.
10: **Until** reaching stop criterion
11: Calculate objective function Y (relation**Error! Reference source not found.**)
12: Update p (relations **Error! Reference source not found.** and **Error! Reference source not found.**)
13: Compare $\hat{O}_{best}$ with the $\hat{O}$ and replace $\hat{O}_{best}$ with $\hat{O}$ if new Y value is higher.
14: **Until** $n2$ iterations
15: Compare O with the $\hat{O}_{best}$ and replace O with $\hat{O}_{best}$ using objective function value Y.
16: **Until** $n1$ iterations

5. Experimental Results

There are two metrics to evaluate the methods: the similarity between the structural connectivity matrix and the optimal one (formula) and the analogy between the optimal structure matrix and the functional connectivity matrix (formula).

We carry out the experiments using two datasets (synthetic and real-world). The synthetic data experiments show the general effectiveness of the proposed approach and novel problem definition. The Camera data gathered using the municipality sensors deployed in the city of Hamadan's urban areas are to demonstrate the applicability of the mathematical formulation of the problem in real-world applications.

5.1. Experiments on Synthetic Data

Three synthetic sets of data with different numbers of zones are used here (n=10, n=100, n=1000). A random integer between zero and $2\log n$ is assigned to each edge that represents the number of roads connecting the corresponding zones. So S is a symmetric matrix and its elements are some numbers in the mentioned range.

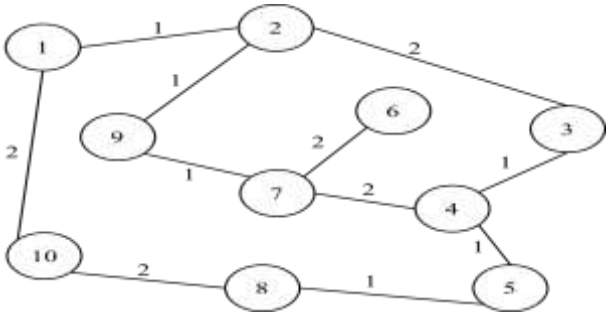

Fig. 1. An example of a synthetic urban area with $n = 10$.

To form matrix F, the traffic flow (matrix D) is needed. Each row in D represents the traffic pattern of an urban zone in different time periods. The number of vehicles crossing the roads of each urban zone varies over time. A random number production function should be applied to create such numbers. Various distributions can be used for random number generation and the selected function here is the normal distribution or Gaussian distribution which is widely used to model random processes such as traffic. There are two parameters in this probability function, one for location (μ) which is referred to as the means of distribution and another one for scale (σ) that is also called scattering. The Normal distribution probability density function formula is defined by equation **Error! Reference source not found.**.

$$f\left(x;\mu,\sigma^2\right)=\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2/2\sigma^2} \quad (13)$$

It is assumed that the C_i vehicles pass through zone i, and the required time for each one is a normal random integer derived from a normal distribution probability density function. Based on the assumptions, there are some decisions for each zone to generate functional connectivity matrix. The first one is the total number of vehicles C_i. The second one is the number of normal distribution probability density functions f and their μ and σ, and the final decision is about the time interval. In **Error! Reference source not found.** an example of the traffic generated during different time intervals of a day is presented. In this pattern 700 vehicles pass through the zone. The required time for 500 vehicles are derived from Gaussian function with $\mu = 12$ and $\sigma = 3$. Others are derived from another function with $\mu = 20$ and $\sigma = 1$.

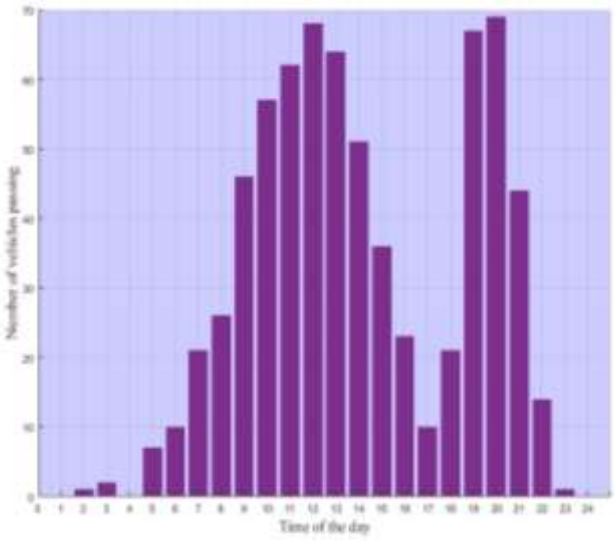

Fig. 2. An example of a city zone traffic pattern.

In all synthetic experiments, two Gaussian functions with different μ and σ can be utilized to create the pattern of the traffic. Each function creates a distinct

number of integers. Each data is produced by using a number of $[3\log_{10} n]$ different patterns, so the number of patterns for $n=10$, $n=100$, and $n=1000$ would be 3, 6, and 9. Each pattern has its own number of vehicles C_i which can be calculated using a random number between n^3+n^2 and n^3-n^2. A random proportion of C_i is dedicated to the first distribution and the rest are for the second one.

Table 1
An example of Parameters used to generate patterns for $n=10$.

i	C_i	Details
1	1083	C_{i1}=858, $\mu=12$, $\sigma=3$ C_{i2}=225, $\mu=20$, $\sigma=1$
2	1092	C_{i1}=716, $\mu=13$, $\sigma=3.5$ C_{i2}=376, $\mu=19$, $\sigma=2$
3	907	C_{i1}=770, $\mu=11.5$, $\sigma=4$ C_{i2}=137, $\mu=21$, $\sigma=1.5$

The traffic of each zone in the network generated by the traffic generation method with one of the parameter settings causes some zones to have the same pattern. The patterns generated by the introduced method with specific parameters (e.g.,) can be applied to form the traffic flow for a whole day. Performing the approach again creates weekly or monthly patterns. It is also possible to divide a day into arbitrary intervals. All experiments generate a set of weekly flow data with 15 minutes intervals using the method explained.

To start the experiments, some parameters need to be set. There are two adjustable parameters in the objective function (α and β), which represent the importance of each sub-function. If adding or removing a link from a network structure is so costly, the first objective function is of more importance and α has to have a greater value than β; otherwise if the current structure is not effective and inferring, the structure from dynamics is favorable (β is dominant to α). In all experiments here they are the same (α=β=0.5). Other parameters are reward and penalty parameters (a and b), which are set $a=0.1$ and β=0.01 in all experiments. These values are obtained after applying the proposed method upon different datasets many times.

After applying the proposed method on the synthetic datasets the optimal structure is found and represented by the optimal connectivity matrix. Value of the objective function is changed in each iteration of the internal loop of the algorithm because of automata action selection.

After passing iterations in the internal loop, there might be no improvement in the objective function value. The algorithm chooses the structure with the highest Y as the estimated optimal structure which an example is shown in **Error! Reference source not found.** The algorithm will terminate internal loop automatically if there will be no improvement in objective function value after passing ne iterations.

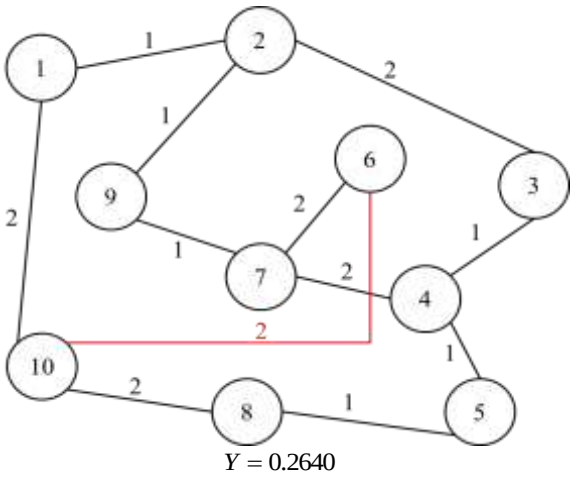

Fig. 3. Estimated optimal structure found in the internal loop of the algorithm executed on data with $n=10$.

The procedure of internal loop is repeated using the tuned probability vectors hoping to find a better structure with higher objective function value. During this loop, the probability matrix is updated again in accordance to the optimal structure. Number of iterations for external loop is $n1$. The whole process is repeated $n2$ times to examine the ability of algorithm. The stopping criterion parameters are specified in Table 2.

Table 2
Stopping criterion in synthetic data experiments.

Number of zones	ne	$n1$	$n2$
10	20	20	30
100	40	200	30
1000	60	2000	30

Three algorithms are employed in this section to solve the problem; these methods are retrieved from the following section:

Table 3
Algorithms to solve the problem.

#	reference	name
1	[47]	Urban Structure Accessibility Modeling and Visualization for Joint Spatiotemporal Constraints
2	[48]	Learning Urban Community Structures: A Collective Embedding Perspective with Periodic Spatial-temporal Mobility Graphs
3	Algorithm 1 [46]	Distributed Learning Automata Based Approach to Inferring Urban Structure via Traffic Flow
4	Algorithm 2	Inferring Urban Structure from Traffic Flow using Cellular Learning Automata

A set of points of interest and human mobility vectors taken from GPS trajectories are used in [48] to learn the vector representation of the community. This can describe static spatial configurations and the dynamic connectivity of human mobility in the community. In our problem definition, the community representation is consonant. We use the second term of the method defined in [48] to formulate and solve the problem based on the data described before. There is a similarity between the two POIs calculated using cosine similarity, and each edge gets a weight in that method. The learning framework outputs a representation that is considered an estimated optimal connectivity. Then we can calculate the fitness function value using the equation.

Computational models are developed in [47] to extract joint-constrained and dynamic structures in cities, which helps demonstrate accessibility. A model named USAGraph forms regions that satisfy some constraints. We define Primary Traffic Regions (PTRs) according to the constraints.

PTR areas are assumed to be connected in the estimated optimal connectivity structure. We use the areas to compute the fitness function value based on equation (4).

Table 4 reveals the minimum, maximum, mean, and variance of the fitness function value calculated after implementing the algorithms on three synthetics.

The min, max, and mean values of the objective function obtained by the proposed algorithm 2 are higher than the ones from other algorithms on synthetic data. The variance is lower in all cases, which is very favorable. The proposed algorithm learns traffic dynamics iteratively. The advantages of this algorithm result in finding the latent link based on the current structure.

Table 4
Results obtained by the algorithms on different datasets.

Objective function	Algorithm #	Dataset		
		10	100	1000
Min	1	0.2152	0.2286	0.2521
	2	0.2132	0.2415	0.2629
	3	0.2342	0.2744	0.2989
	4	0.2344	0.2785	0.3011
Max	1	0.2392	0.2621	0.3023
	2	0.2510	0.3028	0.3511
	3	0.2818	0.3272	0.3605
	4	0.2824	0.3281	0.3621
Mean	1	0.2265	0.2410	0.2611
	2	0.2314	0.2809	0.2759
	3	0.2546	0.3043	0.2978
	4	0.2557	0.3062	0.3211
Variance	1	0. 000581	0. 000667	0. 001023
	2	0. 000491	0. 000421	0. 000991
	3	0.000213	0.000287	0.000932
	4	0.000219	0.000282	0.000901

5.2. Experiments on Real World Data

In these experiments, the network structure consists of different zones of Hamadan that have 40 separated parts (**Error! Reference source not found.**) and the connections between them. The traffic dynamic data is the number of vehicles going through the paths that connect the zones in 15 minutes intervals obtained by cameras installed on streets. It is very costly to create or remove a new path from the city structure, so the first sub-function(γ_1) will be of

high importance and S must be as close to $\hat{O}$ as possible, therefore $\alpha = 0.9$ and $\beta = 0.1$.

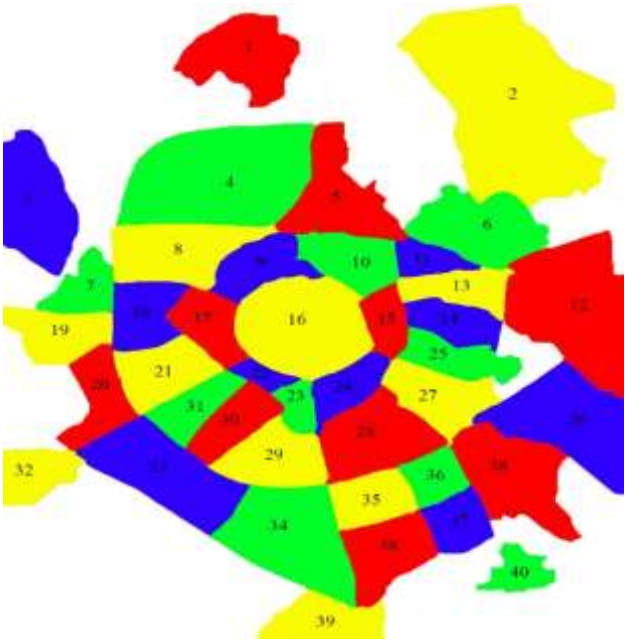

Fig. 4. Map of Hamedan city consisting 40 zones.

Reward and penalty parameters used in the proposed algorithm are $a = 0.1$ and β=0.01 in this test, which are obtained based on many examination. The stopping criterion parameters for this case can be found in Table 5.

Table 5
Stopping criterion in real data experiments.

Number of zones	ne	$n1$	$n2$
40	32	80	30

As it was explained the estimated optimal connectivity matrix which represents the optimal structure can be found during the internal loop causing a growing flow in the objective function value by automata action selection. The structure with the highest Y as the estimated optimal structure is chosen by the algorithm and the algorithm will terminate internal loop as it was described before.

The procedure of internal loop is repeated $n1$ times to find the best estimated optimal structure. After accomplishing the execution of the proposed algorithm on real world datasets Minimum, maximum, mean, and variance of fitness function value is calculated (Table 6).

Table 6
Results obtained by the algorithms on real datasets.

Objective function	Algorithm #	value
Min	1	0.8191
	2	0.8339
	3	0.8590
	4	0.8598
Max	1	0.8976
	2	0.9100
	3	0.9310
	4	0.9322
Mean	1	0.8511
	2	0.8962
	3	0.9213
	4	0.9218
Variance	1	0.000321
	2	0.000299
	3	0.000236
	4	0.000230

5.3. Time Complexity Survey

There are two fixed loops in algorithm 1 repeating $n1$ and $n2$ times. There is another loop with uncertain repetition cycles. This loop is highly dependant of the problem and (the network structure, traffic flow, and the regulating parameters in the fitness function). If we confine the algorithm to create just one link between two distinct nodes, it will repeat $ne + n$ times in worst case scenario. So the time complexity will be $n1\left(n2(ne+n)\right)$. Like algorithm 1, there are also two fixed loops in algorithm 2 one repeating $n1$ and $n2$ times. Considering the same limitation, the other loop repeates $ne+1$ to $ne+n$ times in worst case scenario, because of parallel nature of action selection. So the time complexity

will be $n1\left(n2\left(ne+\frac{n+1}{2}\right)\right)$.

6. Conclusions

Two methods have been proposed in this paper, trying to optimize an objective function that bears a paradigmatic structure. Two types of data, synthetic and real-world, have been used to show the effectiveness of the algorithms. A method generated synthetic data to create traffic patterns in 10, 100, and 1000 zones. Two kinds of data that included the static road network connectivity of zones and functional correlations of traffic flow between the city zones have been used and provided for the presented methods. These kinds of data are useful for finding hidden connections, which are very helpful in city planning. Using the traffic dynamics, some unconnected linkages in the current structure appeared. The zones connected with the new paths are beneficial because their traffic models are indifferent. We examined the proposed algorithms on different data compared to other algorithms. Algorithm 2 potentially has a better performance rate than the other algorithms on real-world and synthetic data using the objective function's min, max, mean, and variance. The experiments showed that the CLA-based algorithm outperforms others in all cases, and the DLA-based one is the second. In real datasets mean fitness function for the CLA-based one is 0.9218, and for the DLA-based one, it is 0.9213, and we can observe that the CLA-based algorithm is better than the DLA-based algorithm. This performance can be because of the learning nature of the algorithms to infer the optimal structure. Utilizing other MDP-based approaches to infer urban structures can be a future study. The presented algorithms can be beneficial in other forms of planning other than urban management in traffic optimization like management, engineering, construction, economics, and many more fields.

References

[1] C. Liu, "Advanced traffic planning," *People Traffic Publication, Beijing*, 2001.

[2] R. Ding et al., "Application of complex networks theory in urban traffic network researches," Networks and Spatial Economics, vol. 19, no. 4, pp. 1281-1317, 2019.

[3] M. Dotoli and M. P. Fanti, "An urban traffic network model via coloured timed Petri nets," Control Engineering Practice, vol. 14, no. 10, pp. 1213-1229, 2006.

[4] J. Poco, H. Doraiswamy, H. T. Vo, J. L. Comba, J. Freire, and C. T. Silva, "Exploring Traffic Dynamics in Urban Environments Using Vector- Valued Functions," in Computer Graphics Forum, 2015, vol. 34, no. 3: Wiley Online Library, pp. 161-170.

[5] O. K. Tonguz, W. Viriyasitavat, and F. Bai, "Modeling urban traffic: a cellular automata approach," IEEE Communications Magazine, vol. 47, no. 5, pp. 142-150, 2009.

[6] B. Baskar, "Effective Urban Structure Inference from Traffic Flow Dynamics," Software Engineering and Technology, vol. 10, no. 2, pp. 35-38, 2018.

[7] S. Sarkar et al., "Effective urban structure inference from traffic flow dynamics," IEEE Transactions on Big Data, vol. 3, no. 2, pp. 181-193, 2017.

[8] A.Rezvanian, A. M. Saghiri, S. M. Vahidipour, M. Esnaashari, and M. R. Meybodi, Recent advances in learning automata. Springer, 2018.

[9] A.Rezvanian, B. Moradabadi, M. Ghavipour, M. M. D. Khomami, and M. R. Meybodi, Learning automata approach for social networks. Springer, 2019.

[10] M. A. Thathachar and P. S. Sastry, Networks of learning automata: Techniques for online stochastic optimization. Springer Science & Business Media, 2003.

[11] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018.

[12] H. Beigy and M. R. Meybodi, "An iterative stochastic algorithm based on distributed learning automata for finding the stochastic shortest path in stochastic graphs," The Journal of Supercomputing, vol. 76, no. 7, pp. 5540-5562, 2020.

[13] H. Beigy and M. R. Meybodi, "Utilizing distributed learning automata to solve stochastic shortest path problems," International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 14, no. 05, pp. 591-615, 2006.

[14] M. R. Mollakhalili Meybodi and M. R. Meybodi, "Extended distributed learning automata," Applied Intelligence, vol. 41, no. 3, pp. 923-940, 2014.

[15] M. Hasanzadeh and M. R. Meybodi, "Grid resource discovery based on distributed learning automata," *Computing*, vol. 96, no. 9, pp. 909-922, 2014.

[16] A.Rezvanian and M. R. Meybodi, "Finding maximum clique in stochastic graphs using distributed learning automata," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 23, no. 01, pp. 1-31, 2015.

[17] B. Moradabadi and M. R. Meybodi, "Link prediction in fuzzy social networks using distributed learning automata," Applied Intelligence, vol. 47, no. 3, pp. 837-849, 2017.

[18] R. Vafashoar, H. Morshedlou, A. Rezvanian, and M. R. Meybodi, "Cellular Learning Automata: Theory and Applications," ed: Springer.

[19] M. Khaksar Manshad, M. R. Meybodi, and A. Salajegheh, "A new multi-wave continuous action-set cellular learning automata for link prediction problem in weighted multi-layer social networks," The Journal of Supercomputing, 2022/06/09 2022, doi: 10.1007/s11227-022-04615-z.

[20] M. Ghavipour and M. R. Meybodi, "Irregular cellular learning automata-based algorithm for sampling social networks," Engineering Applications of Artificial Intelligence, vol. 59, pp. 244-259, 2017.

[21] M. K. Manshad, A. K. Manshad, and M. R. Meybodi, "Memory/search RCLA-EC: A CLA-EC for moving parabola problem," in 2011 6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT), 2011: IEEE, pp. 732-737.

[22] M. Khaksar Manshad, M. R. Meybodi, and A. Salajegheh, "A new irregular cellular learning automata-based evolutionary computation for time series link prediction in social networks," Applied Intelligence, vol. 51, no. 1, pp. 71-84, 2021.

[23] R. Vafashoar, M. R. Meybodi, and A. H. Momeni Azandaryani, "CLA-DE: a hybrid model based on cellular learning automata for numerical optimization," Applied Intelligence, vol. 36, no. 3, pp. 735-748, 2012.

[24] D. K. Patel and S. A. More, "Edge detection technique by fuzzy logic and Cellular Learning Automata using fuzzy image processing," in 2013 International Conference on Computer Communication and Informatics, 2013: IEEE, pp. 1-6.

[25] B. Moradabadi and M. R. Meybodi, "Wavefront cellular learning automata," Chaos: An Interdisciplinary Journal of Nonlinear Science, vol. 28, no. 2, p. 021101, 2018.

[26] F. Angius, M. Reineri, C.-F. Chiasserini, M. Gerla, and G. Pau, "Towards a realistic optimization of urban traffic flows," in 2012 15th International IEEE Conference on Intelligent Transportation Systems, 2012: IEEE, pp. 1661-1668.

[27] V. Moosavi and L. Hovestadt, "Modeling urban traffic dynamics in coexistence with urban data streams," in Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing, 2013, pp. 1-7.

[28] A.Solé-Ribalta, S. Gómez, and A. Arenas, "A model to identify urban traffic congestion hotspots in complex networks," Royal Society open science, vol. 3, no. 10, p. 160098, 2016.

[29] X. Zhao, "Modeling transportation networks and urban traffic dynamics: A Markovian framework," Johns Hopkins University, 2017.

[30] R. Ding et al., "Detecting the urban traffic network structure dynamics through the growth and analysis of multi-layer networks," Physica A: Statistical Mechanics and its Applications, vol. 503, pp. 800-817, 2018.

[31] L. Ellam, M. Girolami, G. A. Pavliotis, and A. Wilson, "Stochastic modelling of urban structure," Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 474, no. 2213, p. 20170700, 2018.

[32] L. Hu, T. Sun, and L. Wang, "Evolving urban spatial structure and commuting patterns: A case study of Beijing, China," Transportation Research Part D: Transport and Environment, vol. 59, pp. 11-22, 2018.

[33] K. Ouyang et al., "Fine-grained urban flow inference," IEEE transactions on knowledge and data engineering, 2020.

[34] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1720-1730.

[35] R. Ding et al., "Optimal Topology of Multilayer Urban Traffic Networks," Complexity, vol. 2019, 2019.

[36] A.Avila and I. Mezić, "Data-driven analysis and forecasting of highway traffic dynamics," Nature communications, vol. 11, no. 1, pp. 1-16, 2020.

[37] A.Djouadi, O. Snorrason, and F. Garber, "The quality of training sample estimates of the bhattacharyya coefficient," IEEE Transactions on Pattern analysis and machine intelligence, vol. 12, no. 1, pp. 92-97, 1990.

[38] B. Masoumi and M. R. Meybodi, "Learning automata based multi- agent system algorithms for finding optimal policies in Markov games," Asian Journal of Control, vol. 14, no. 1, pp. 137-152, 2012.

[39] M. Ahangaran, N. Taghizadeh, and H. Beigy, "Associative cellular learning automata and its applications," Applied Soft Computing, vol. 53, pp. 1-18, 2017.

[40] P.-Y. Louis and F. R. Nardi, Probabilistic cellular automata. Springer, 2018.

[41] Gerakakis, P. Gavriilidis, N. I. Dourvas, I. G. Georgoudas, G. A. Trunfio, and G. C. Sirakoulis, "Accelerating fuzzy cellular automata for modeling crowd dynamics," Journal of Computational Science, vol. 32, pp. 125-140, 2019.

[42] J. A. Torkestani and M. R. Meybodi, "A cellular learning automata-based algorithm for solving the vertex coloring problem," Expert Systems with Applications, vol. 38, no. 8, pp. 9237-9247, 2011.

[43] H. Beigy and M. R. Meybodi, "Asynchronous cellular learning automata," Automatica, vol. 44, no. 5, pp. 1350-1357, 2008.

[44] H. Beigy and M. R. Meybodi, "Open synchronous cellular learning automata," Advances in Complex Systems, vol. 10, no. 04, pp. 527-556, 2007.

[45] H. Beigy and M. R. Meybodi, "Cellular learning automata with multiple learning automata in each cell and its applications," IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 40, no. 1, pp. 54-65, 2009.

[46] H. Yasinian and M. Esmaeilpour, "Distributed learning automata based approach to inferring urban structure via traffic flow," Applied Intelligence, pp. 1-13, 2021.

[47] F. Kamw et al., "Urban structure accessibility modeling and visualization for joint spatiotemporal constraints," IEEE Transactions on Intelligent Transportation Systems, 2019.

[48] P. Wang, Y. Fu, J. Zhang, X. Li, and D. Lin, "Learning urban community structures: A collective embedding perspective with periodic spatial-temporal mobility graphs," ACM Transactions on Intelligent Systems and Technology (TIST), vol. 9, no. 6, p. 63, 2018.