Wavefront cellular learning automata

Behnaz Moradabadi, and Mohammad Reza Meybodi

Citation: Chaos **28**, 021101 (2018); doi: 10.1063/1.5017852
View online: https://doi.org/10.1063/1.5017852
View Table of Contents: http://aip.scitation.org/toc/cha/28/2
Published by the American Institute of Physics

Wavefront cellular learning automata

Behnaz Moradabadi[a] and Mohammad Reza Meybodi[b]
Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran

(Received 30 November 2017; accepted 15 January 2018; published online 2 February 2018)

This paper proposes a new cellular learning automaton, called a wavefront cellular learning automaton (WCLA). The proposed WCLA has a set of learning automata mapped to a connected structure and uses this structure to propagate the state changes of the learning automata over the structure using waves. In the WCLA, after one learning automaton chooses its action, if this chosen action is different from the previous action, it can send a wave to its neighbors and activate them. Each neighbor receiving the wave is activated and must choose a new action. This structure for the WCLA is necessary in many dynamic areas such as social networks, computer networks, grid computing, and web mining. In this paper, we introduce the WCLA framework as an optimization tool with diffusion capability, study its behavior over time using ordinary differential equation solutions, and present its accuracy using expediency analysis. To show the superiority of the proposed WCLA, we compare the proposed method with some other types of cellular learning automata using two benchmark problems. *Published by AIP Publishing.*
https://doi.org/10.1063/1.5017852

Cellular Learning Automata (CLA) is a combination of Cellular Automata (CA) and Learning Automata (LA). LA is an adaptive decision making unit in unknown environments and CA is a structure of cells where each cell has a state and transitions over the set of possible states using information from its state and its neighbor states. The CLA is preferable to the CA because it tries to learn optimal actions, and it is also preferable to an LA because it can improve the learning capability using a set of learning automata that interact with each other. On the other hand, in many real-world learning problems, the learning process can proceed in stages in a particular order. For example, in a pattern classification with decision trees, when a node makes a decision, its decision is transmitted to its successor nodes as a wave. So, in this paper, we consider a CLA with a connected structure and one LA for each region and define the neighbors of each cell to be the cells of successor regions, such that each LA can send a wave to its neighbors if its chosen action is different from the previous action. This model of CLA is called a Wavefront CLA (WCLA). The WCLA enables us to propagate information through the CLA because it has both a connected neighbor structure and wave propagation properties. This paper introduces the WCLA and studies its convergence. Also to examine the proposed method, we have implemented some experiments. For future studies, we want to apply the proposed framework in online social network problems where a change in the network should be propagated.

I. INTRODUCTION

A cellular automaton (CA) is made of similar cells where each cell has a state and transitions over the set of possible states which are based on a local rule. The local rule of a cell is defined based on the local environment, usually the set of its neighbor cells.[1] For each cell, the state of all neighbor cells (including itself) is called the configuration of that cell, and the behavior of the CA through time is determined by the initial configuration of the CA and its local rule. A CA is generally appropriate for modeling and analyzing systems made of similar interrelated components.[2,3]

A learning automaton (LA) is also an adaptive decision-making unit in unknown random environments. Intuitively, the LA, during its learning mechanism, tries to choose the optimal action from its action set based on the environment's response to the previous action. To improve the local interactions of the LA in complex systems, the CA and LA are merged together and proposed in terms of a CLA.[4] The CLA is preferable to a CA because it tries to learn optimal actions, and it is also preferable to an LA because it can improve the learning capability using a set of learning automata that interact with each other. To date, the CLA has been applied in some real-world areas such as channel assignment,[5] call admission control on cellular networks,[6] and VLSI placement.[6] The behavior of the CLA through time was analyzed in Ref. 7 and it has been shown that the CLA can converge to a steady state using a set of rules called commutative rules.[6–9]

In many real-world learning problems, the learning process can proceed in stages in a particular order. For example, in a pattern classification with decision trees, when a node makes a decision, its decision is transmitted to its successor nodes as a wave. In problems related to sampling social networks, when we label a link or node as a sample instance, this decision can propagate into the network as a wave. Therefore, we consider a CLA with one LA for each region and define the neighbors of each cell to be the cells of successor regions, such that each LA can send a wave to its neighbors if its chosen action is different from the previous

[a]moradabadi@aut.ac.ir
[b]mmeybodi@aut.ac.ir

action. This model of CLA is called a wavefront CLA (WCLA). In a wavefront CLA, we partition the problem space into stages such that each stage tries to learn the optimal action and helps the neighbors to act accordingly in the environment.

A wavefront CLA (WCLA) is an extension of the asynchronous CLA model, with a connected neighbor structure and propagation property where each LA can send a wave to its neighbors and activate their learning automata to choose new actions if the chosen action is changed from the previous action. Furthermore, this procedure continues. Each cell that receives the wave is activated and its corresponding LA must choose a new action. The proposed WCLA is an asynchronous CLA because, at each iteration, only some learning automata are activated independently, rather than all of them in a parallel manner. The WCLA enables us to propagate information through the CLA because it has both a connected neighbor structure and wave propagation properties. This paper introduces the WCLA and studies its convergence.

The rest of the paper is structured as follows: Section II reviews the learning automaton and cellular learning automaton. Section III presents the WCLA. The experimental study of the CLA is presented in Sec. IV. Finally, Sec. V summarizes the main conclusions of the paper.

II. BACKGROUND

This section gives an introduction to the learning automaton and cellular learning automaton as the following:

A. Learning automata

A learning automaton is an adaptive decision-making unit that improves its performance by learning how to choose the optimal action from a finite set of allowed actions through repeated interactions with a random environment.[10] The action is chosen based on the probability distribution of the action set and at each instant the given action serves as the input to the random environment. The environment sends a reinforcement signal as a response to the action taken. The action probability distribution is updated using the environment reinforcement signal. The goal of a learning automaton is to discover the optimal action from the action set. Thus, the average reward received from the environment is maximized.

The environment in an LA is defined by the triple $E = \{\alpha, \beta, c\}$, where $a = \{a_1, a_2, \ldots, a_r\}$ represents a finite input set of actions, $\beta = \{\beta_1, \beta_2, \ldots, \beta_r\}$ is the set of reinforcement signal values, and $c = \{c_1, c_2, \ldots, c_r\}$ is a set of penalty probabilities such that each element $c_i = E[\beta|a = a_i]$ of c is the penalty probability for input a_i. If the reinforcement signal β takes values in the range [0, 1], then the environment is called an S-model environment. Learning automata are classified into two categories: fixed-structure learning automata and variable-structure learning automata. The following paragraph introduces the variable-structure LA.

The variable-structure LA is described by the quadruple $\{\alpha, \beta, p, T\}$, in which a and β are the same as they are in the LA definition, $p = \{p_1, p_2, \ldots, p_r\}$ gives the action probability distribution over the action set, and $p(k+1) = T[a(k), b(k), p(k)]$ defines the learning algorithm for updating the probability vector. Generally, in each iteration of the LA lifecycle, the LA chooses an action $a(k)$ using its action probability distribution, and sends this action to the environment. The environment evaluates the selected action and generates the reinforcement signal $(\beta(k))$. Finally, the LA updates its action-set probabilities based on the learning algorithm.[11]

To date, the LA has been applied in many learning problems in different areas.[12–17]

B. Cellular learning automata

A CLA is defined as a combination of CA and LA in which an LA is allocated to every cell in the CA. Like the CA, the CLA also works based on its local rule and because each cell contains an LA, the local rule of the cell with its configuration generates a reinforcement signal to the LA corresponding to that cell. The configuration of each cell is the same as it is in the CA. Furthermore, the environment of each LA in each cell is non-stationary because the learning automata change through the evolution of the CLA. The CLA operates as follows: in the initial phase, the initial state of every cell is specified using the initial action probability distribution of the corresponding LA. Then, using the local rule of the CLA, a reinforcement signal is generated for the LA residing in that cell, and the LA uses this reinforcement signal to update its action probability distribution using the learning algorithm. This procedure continues until a stopping criterion is reached. The CLA is preferable to the CA because it tries to learn the optimal actions and it is also preferable to single LA because it can improve the learning capability using a set of learning automata interacting with each other. Generally, a CLA model is characterized as follows:

Definition 1. A d-dimensional cellular learning automaton is a framework,[7] $\mathcal{A} = (Z^d, N, \Phi, A, \mathcal{F})$, where

- Z^d represents the lattice of d-tuples of integer numbers.
- $N = \{\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_m\}$ is a finite subset of Z^d called the neighborhood vector, where $\overline{x}_i \in Z^d$.
- Φ denotes the finite set of states and φ_i represents the state of the cell c_i.
- A represents the set of learning automata residing in the cells of the CLA.
- $F^i : \underline{\Phi_i} \rightarrow \underline{\beta}$ defines the local rule of the CLA for each cell c_i, where $\underline{\beta}$ is the set of possible values for the reinforcement signal, and it calculates the reinforcement signal for each LA using the chosen actions of its neighbors.

In the rest of this section, we briefly review some classifications of cellular learning automata:

- Asynchronous CLA vs. synchronous CLA: in a synchronous CLA, all cells use their local rules at the same time. This model assumes that there is an external clock which triggers synchronous events for the cells. In an asynchronous CLA, at a given time only some cells are activated and the state of the rest of the cells remains unchanged.[6] The learning automata may be activated in two ways: time-driven activation where each cell is assumed to have an internal clock which wakes up the LA associated with

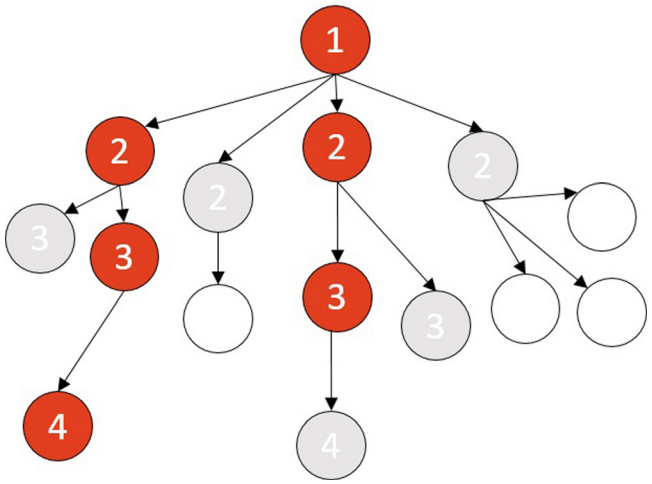


FIG. 1. WCLA example.

that cell or step-driven activation where a cell is selected in a fixed or random sequence.

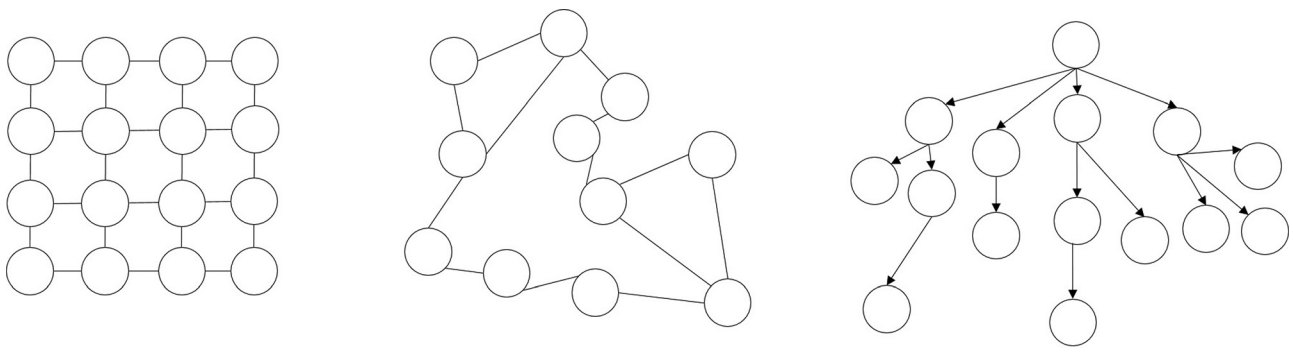
- Regular CLA vs. irregular CLA (ICLA): in a regular CLA, the structure of the CLA is represented as a lattice of d-tuples of integer numbers, while in an irregular CLA (ICLA), the structure regularity assumption is replaced with an undirected graph.[18–20]

III. WAVEFRONT CLA

A wavefront cellular learning automaton is a generalization of the asynchronous CLA which has a connected neighbor structure and the property of diffusion. The main features of a WCLA are that the neighbors of each cell are defined as its successor cells and also that each LA can send a wave to its neighbors and make them active if its chosen action is different from the previous action (diffusion property). Each cell that receives the wave is activated and its corresponding LA must choose a new action. Then, using the local rule of the WCLA and the selected actions of its neighbors, a reinforcement signal is generated for the LA residing in that cell and the LA uses this reinforcement signal to update its action probability distribution using the learning algorithm. This procedure continues until a stopping criterion is reached. Figure 1 shows the procedure of the WCLA in a simple view. In this figure, first the root learning automata that is labeled as 1 is activated. This learning automata chooses an action and because the new action is different from the previous one, it activates all of its children LAs in the next level (labeled as 2). Then, each activated child chooses a new action and if its action is the same as its previous action, then the chain is stopped at that node (colored as gray) and if the new action is different from the previous action, it activates its children (colored as red) and this procedure goes on. The proposed WCLA is asynchronous because at each iteration, one LA selects a new action and therefore the action selection does not occur in a parallel manner. It is also irregular and closed because it does not use a lattice for its structure and only uses the local environment to update the CLA behavior. Finally, it is static because the structure of the CLA remains fixed.

As mentioned previously, the WCLA utilizes two new concepts compared to the CLA:

1. Different neighbor structure: in the traditional CLA, the structure must be a lattice, while in the WCLA, the structure can be any connected structure; either a regular structure such as a lattice or an irregular structure such as a graph or tree. The only necessity is that the structure must be connected. The structure of a WCLA is connected when there is a path between every pair of cells. In a connected structure, there are no unreachable cells. The connectivity property of the structure of the WCLA ensures that a wave can reach any cell in the network, and therefore there is a learning capability throughout the network. Figure 1 shows some examples of valid structures for a WCLA. Figure 2(a) shows a lattice structure as an example of a regular structure. Figures 1(b) and 1(c) show the graph and tree structures, respectively, as examples of irregular structures. All the structures are connected.
2. Propagation property using waves: as previously mentioned, in a traditional CLA, each learning automaton chooses its action in each iteration. Then, the local rule of each learning automaton generates the reinforcement signal for



a) Lattice b) Graph c) Tree

FIG. 2. Examples of valid structures in a WCLA.

the corresponding learning automaton using its chosen action and the chosen actions of its neighbors. Finally, each learning automaton updates its action probability based on its reinforcement signal. As we see in the traditional CLA, there is no propagation and diffusion property. When a cell state is changed, the other cells are not triggered to choose new actions. To overcome this problem, the WCLA has a propagation property using waves. Each LA can send a wave to its neighbors and activate them if the chosen action is different from the previous action (diffusion property). Each cell that receives the wave is activated and its corresponding LA must choose a new action. Figure 3 shows different waves over different WCLA structures. Each wave starts from a cell in the network and moves to its neighbors. The diffusion path of the wave depends only on the neighboring structure of the WCLA. Because the structure is connected and the waves can move over the entire network, each cell can be activated and improve its state after receiving waves, thus also improving its learning capability. However, in order to control the movement of the waves over the network, we define an energy for each wave in the WCLA. The energy of each wave determines the wave's ability to propagate itself. The energy of the wave decreases as it moves through the network until it reaches zero, at which point the wave disappears and movement stops. In this way, each cell receiving the wave is activated and its LA is triggered to choose a new action.

Generally, a WCLA model is characterized as follows:

Definition 2. A wavefront cellular learning automaton is defined with the structure $\mathcal{A} = (S, \Phi, A, \mathcal{F}, W, C)$, where:

- S is the structure of the WCLA that is a connected network of learning automata.
- Φ is a finite set of states.
- A represents the set of learning automata residing in the cells of the CLA.
- $F^i : \underline{\Phi_i} \rightarrow \underline{\beta}$ defines the local rule of the WCLA for each cell c_i, where $\underline{\Phi_i}$ is the current state of LA_i and its successors. In addition, $\underline{\beta}$ is the set of possible values for the reinforcement signal, and the reinforcement signal is calculated for each LA using the chosen actions of successor learning automata.
- W defines the wave energy and determines when and where the wave stops moving through the network.
- C is a condition where an LA sends a wave over the network and triggers its children to choose a new action. In a WCLA, C is defined according to whether the chosen action of the LA is different from the previous action or not.

As previously mentioned, the WCLA differs in two main respects compared to the CLA. First, in the WCLA, the lattice Z^d and the neighborhood vector N are replaced by any connected network. Second, in the proposed model, each LA can send a wave to its neighbors and each cell that receives the wave is activated and triggers its LA to choose a new action. In the rest of this section, we present the learning algorithm we use for updating the WCLA action:

As mentioned above, in the WCLA we have a set of N learning automata indexed from 1 to N. The ith LA has the action set A_i with r_i actions as follows:

$$A_i = \{a_{i1}, \ldots, a_{ir_i}\}. \tag{1}$$

In addition, $p_i(k)$ and $\alpha_i(k)$ are its action probability vector and its chosen action in iteration k, respectively. The probability of choosing action a_{ij} at instance k is defined as

$$Prob\{\alpha_i(k) = a_{ij}\} = p_{ij}(k). \tag{2}$$

To update $p_i(k)$ for instance k, we use the linear reward-inaction (LR-I) algorithm according to the following equations:

$$\begin{aligned} p_{ij}(k+1) &= p_{ij}(k) + \lambda\beta(k)(1 - p_{ij}(k)) \quad \text{if } \alpha_i(k) = a_{ij}, \\ p_{ij}(k+1) &= (1 - \lambda\beta(k))p_{ij}(k) \quad \text{if } \alpha_i(k) \neq a_{ij}, \end{aligned} \tag{3}$$

where $\beta(k)$ is the reinforcement signal in instance k and the learning parameter λ satisfies the relation $0 < \lambda < 1$. Also in order to analyze the WCLA convergence and expediency using learning algorithm (3), we have prepared some proofs and theorems in the Appendix.

IV. EXPERIMENTS

In Secs. IV A and IV B, we will present two experiments to study the behavior of the WCLA. In the first experiment, we show the accuracy of the WCLA in the minimum spanning tree (MST) problem and analyze the initial parameters of the WCLA. In the second experiment, we choose the vertex coloring (VC) problem and compare the results with those of the traditional CLA and the irregular CLA.

A. Experiment 1

In this experiment, we use the proposed WCLA (with depth = 2) to find the minimum spanning tree (MST) of five

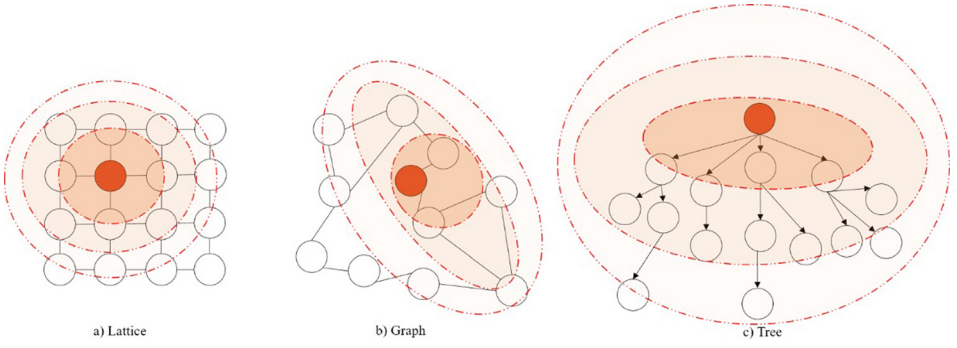


FIG. 3. Waves in different structures.

TABLE I. The accuracy of the MST in 4000 iterations of the WCLA.

Graph no.	Accuracy (%)
G(10)	100.00
G(20)	100.00
G(30)	100.00
G(40)	100.00
G(50)	100.00

sample graphs. We have used five complete graphs called G(N), where N is the number of vertices. We choose five configurations for N: {10, 20, 30, 40, and 50}. The edge weights are randomly generated with a uniform distribution within the range [10,100]. All maximum degrees are equal to three. For each graph, we extract a directed acyclic graph (DAG) as the structure of the WCLA from the connected graph and try to find the minimum spanning tree using the WCLA. Each learning automaton in the WCLA has two actions {0, 1} that determine whether the corresponding edge appears in the final MST or not. The reinforcement signal is also calculated based on the negative of the weighted sum of the edges in the results for the WCLA. Table I shows the accuracy of the MST obtained by the WCLA through 4000 iterations. It can be seen that the WCLA can find the best MST in G(10) through G(50). Furthermore, Fig. 4(a) shows the convergence rate of the WCLA for the first 4000 iterations for each graph used. It can be seen from the figure that for larger graphs, more time is needed for convergence and good accuracy. Also, we have tried to fit the convergence plot with an exponential function. The exponential function we chose is the following function:

$$Accuracy = a(N) \times x^{b(N)},$$

where x is the iteration number and $a(N)$, $b(N)$ are the parameters of the model based on the number of vertices, N. Then, we have tried to find the parameters $a(N)$ and $b(N)$ and obtained the following equations:

$$Accuracy = a(N) \times x^{b(N)},$$

$$a(N) = 73 - \frac{6n}{10},$$

$$b(N) = 0.05 + \frac{n}{1000}.$$

So, based on the above equations, we can estimate the number of iterations for converging the WCLA based on the number of vertices, N, as the following:

$$k = \left(\frac{1000}{730 - 6n}\right)^{\frac{1000}{50+n}},$$

where k is the estimation number of iterations. Also, Figs. 4(b)–4(f) show the fitted function and the real convergence data for graphs G(10) through G(50). From these figures, we can see that the estimated function is a good choice for calculating the needed iterations for a specific convergence rate and accuracy.

In addition, Table II gives the different initial action probability vectors for the WCLA and the result of the MST. For example, the initial action probability vector in configuration 3 is [0:0.5:1:0.5]. Table II also shows the accuracy of the WCLA for different initial action probability vectors. As can be seen from this table, in most cases, the WCLA is independent of the initial action probability vector and it converges to its equilibrium configuration.

Also because the WCLA is a distributed algorithm and it has a propagation property, it can be used in online

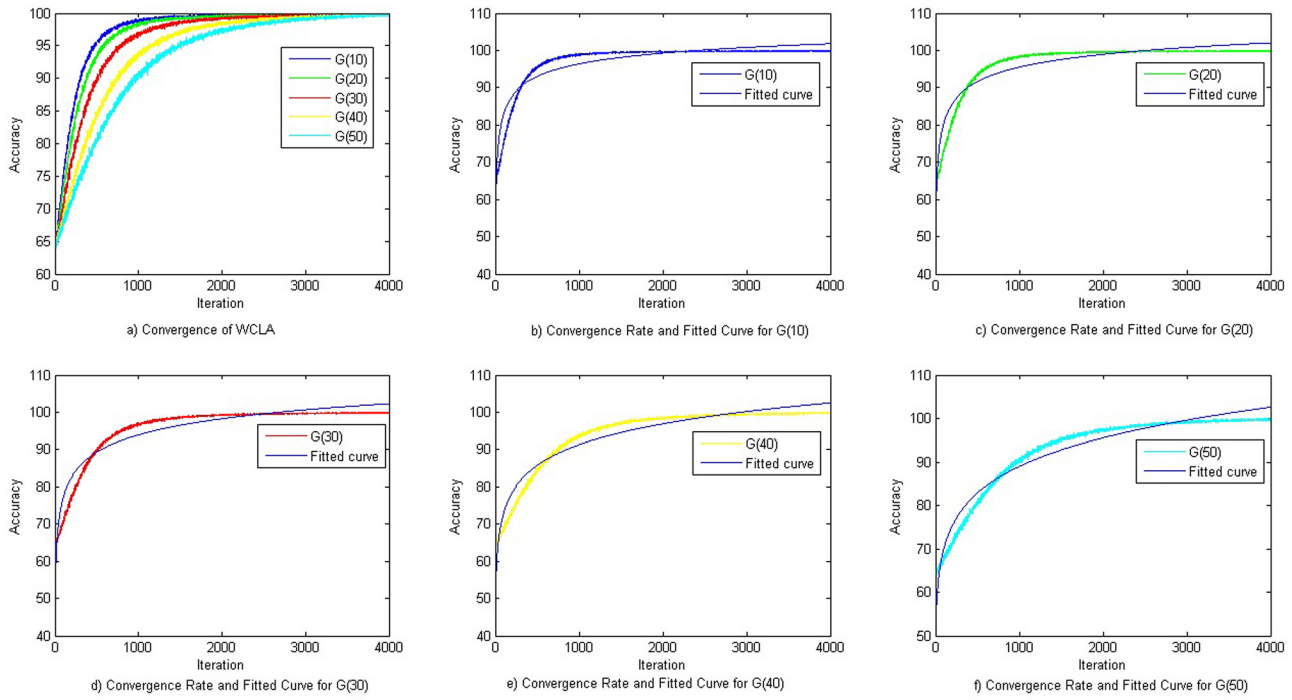


FIG. 4. Convergence rate of the WCLA.

TABLE II. The accuracy of the MST with regard to different initial configurations in the WCLA.

Graph no./action probability	[0:0.9 1:0.1] (%)	[0:0.7 1:0.3] (%)	[0:0.5 1:0.5] (%)	[0:0.3 1:0.7] (%)	[0:0.1 1:0.9] (%)
G(10)	100.00	100.00	100.00	100.00	100.00
G(20)	100.00	100.00	100.00	100.00	99.54
G(30)	100.00	100.00	100.00	100.00	97.80
G(40)	97.87	98.17	100.00	99.85	96.76
G(50)	96.81	97.48	99.93	98.54	95.62

TABLE III. The accuracy of the MST in online situations.

Graph no./action probability	Step1 (%)	Step2 (%)	Step3 (%)	Step4 (%)	Step5 (%)
G(10)	99.56	100.00	100.00	100.00	100.00
G(20)	98.32	100.00	99.51	99.31	98.98
G(30)	99.95	99.17	98.32	100.00	97.97
G(40)	100.00	99.01	100.00	100.00	100.00
G(50)	100.00	98.50	98.17	98.65	97.03

situations where we want to consider the network changes in the problem. So in the rest of this section, we conduct an experiment to evaluate the WCLA in online situations: to do this, first we calculate the MST solution for the initial graphs using the WCLA. Then in 5 steps, we change the weight of 5 edges in the graph and the corresponding LA of each changed edge is activated and the algorithm goes on until it finds another solution. Then, we evaluate the solution and go to the next step. Table III shows the accuracy of the obtained MST for each graph and each step. From the results, we can conclude that the WCLA is a good choice for the online MST problem.

B. Experiment 2

The vertex coloring (VC) problem is a combinatorial optimization problem in which a color is assigned to each vertex of the graph such that no two adjacent vertices have the same color. To solve the VC problem using a WCLA, we use the following procedure. First, we extract a DAG as the structure of the WCLA from the connected graph. Each learning automaton in the WCLA has n actions, where n is the maximum number of colors. The reinforcement signal is also calculated based on the following method. For each LA, if its selected color (action) is chosen by at least one of its neighbors, we penalize the LA, otherwise we calculate the color degree of the LA (the number of colors used by itself and its neighbors), d_i, and then if $d_i \leq t$, we reward the LA and update $t = d_i$, otherwise we penalize it. For this experiment, the learning rate of the WCLA is set to 0.1, and the algorithm is terminated when the probability of choosing one color in all automata is 0.95 or greater. In this experiment, we compare the proposed WCLA using the vertex color problem with the two following CLA algorithms:

- CLA-VC1[23] in which a CLA-based algorithm is proposed for finding a near-optimal solution of the vertex coloring problem. The proposed coloring algorithm is a fully distributed algorithm in which each vertex chooses its optimal color based solely on the colors selected by its adjacent vertices.
- CLA-VC2[20] in which an adaptive Petri net system based on an irregular CLA is proposed for the vertex coloring problem.

To perform the comparison, we evaluate each algorithm using a subset of hard-to-color benchmarks like DSJ[24] and WAP.[25] The performance of each algorithm is measured both in terms of the time and the number of colors required for coloring the benchmarks. It should be noted that the parameters of the algorithms are borrowed from their references.

The first data set, DSJ, is made up of a set of uniform (n,p) random graphs which are denoted DSJ $(n.p)$, where $n = \{125, 500, \text{and } 1000\}$ is the number of vertices, and $p = \{0.1, 0.5, \text{and } 0.9\}$ denotes the probability of connecting every pair of nodes in the graph. The second data set we consider includes WAP graphs which are denoted by Wap0ma, where $m = \{1, 2, \ldots, 5\}$. In this class, the graphs have a large number of vertices between 905 and 5231, and all instances have a clique of size 40. Table IV shows the results of the experiments conducted on DSJ and WAP benchmark graphs.

TABLE IV. Comparing the WCLA with two other algorithms for the VC problem.

		VC1		VC2		WCLA	
Graph no.	Best	CN	RT	CN	RT	CN	RT
DSJC125.1	5	5	0.0090	8	0.0050	5	0.0009
DSJC125.5	17	17	11.898	19	10.842	17	5.284
DSJC125.9	44	44	19.463	45	17.514	44	13.587
DSJC500.1	12	12	20.390	12	17.297	12	12.517
DSJC500.5	48	48	42.298	49	40.172	48	35.084
DSJC500.9	126	126	70.465	127	67.294	126	60.517
DSJC1000.1	20	21	38.670	21	35.184	20	29.568
DSJC1000.5	84	84	87.731	84	84.963	84	76.287
DSJC1000.9	224	224	119.10	224	115.42	224	100.574
Wap01a	42	42	15.785	43	13.517	42	07.518
Wap02a	41	41	27.902	41	24.854	41	16.847
Wap03a	44	43	50.563	45	47.521	44	42.287
Wap04a	43	43	46.576	45	42.517	43	36.756
Wap05a	41	40	26.150	41	22.587	41	17.249

In this table, the first column shows the number of colors required for coloring the graph (CN), and the second column shows the running time of each algorithm in seconds (RT).

Comparing the results of the algorithms, we observe that, in most cases, WCLA has the shortest running time, especially with increasing graph sizes. It also can be seen that, in almost all cases, WCLA chooses the smallest number of colors to color the graphs. Comparing the results of the WCLA with VC1, we find that the color sets chosen by the proposed algorithm are as small as those of the VC1, while the running time of the proposed algorithm is considerably shorter than the VC1. On the other hand, comparing the proposed algorithm with VC2, it can be seen that both the running time and the size of the color set are significantly smaller for the proposed algorithm than for VC2.

V. CONCLUSION

In this paper, a new extension to CLA, called the WCLA, is proposed. Generally, the WCLA differs from CLA with respect to two main features: (1) in the WCLA, the neighbors of an LA are defined as its successors in the network and (2) each LA can trigger its successors to choose a new action if its current action is different from the previous one. The latter property is needed for modeling real-world problems in areas such as social networks, computer networks, and web mining. We also analyzed the steady-state behavior of the WCLA through time and evaluated the WCLA via two experiments.

APPENDIX: WCLA CONVERGENCE ANALYSIS

In this appendix, we analyze the convergence of the WCLA. To do this, we consider a WCLA with the following properties:

- The structure S is considered to be a directed acyclic graph DAG, $G < E$, and $V>$, with V as the set of vertices (cells) and E as the set of directed edges (adjacency relations).
- The energy of the wave, W, is defined by a constant such that the wave each LA generates is sent to its successor learning automata to a particular depth.

In the rest of this appendix, we first present some definitions and then try to model and analyze the objective of the WCLA as an optimization problem.

Definition 1. In instance k, the configuration of the WCLA is defined by the following equation:

$$P(k) = [p_1^T, p_2^T, \ldots, p_N^T]^T, \tag{A1}$$

where p_1^T is the transpose of the action probability vector for the ith LA which should satisfy

$$\begin{aligned} &p_{ij} \geq 0 \quad 1 < j \leq r_i, \quad 1 \leq i \leq N, \\ &\sum_{j=1}^{r_i} p_{ij} = 1 \quad 1 \leq i \leq N. \end{aligned} \tag{A2}$$

Definition 2. The neighborhood set of the ith learning automaton, *N(i),* is defined as the set of its successor learning automata at a particular depth.

Now, we can consider the following optimization problem as the goal of the WCLA:

$$\begin{aligned} \textit{Maximize } f(P) &= E[\beta|P] = \sum_{i=1}^{N}\sum_{j=1}^{r_i} E[\beta_i|P, \alpha_i = a_{ij}] \\ &= \sum_{i=1}^{N}\sum_{j=1}^{r_i}\sum_{y_1,\ldots,y_{n(i)}} \left(F^i(y_1,\ldots,y_{n(i)},a_{ij}) \prod_{k\in N(i)} p_k^T\right). \end{aligned} \tag{A3}$$

Subject to

$$\begin{aligned} &p_{ij} \geq 0 \quad 1 < j \leq r_i, \quad 1 \leq i \leq N, \\ &\sum_{j=1}^{r_i} p_{ij} = 1 \quad 1 \leq i \leq N, \end{aligned}$$

where

$$P^T = [p_1^T, p_2^T, \ldots, p_N^T], \quad p_i{}^T = [p_{i1}, \ldots p_{ir_i}].$$

The goal of this optimization problem is to maximize the reinforcement signals received from the environment. In the following, we try to show that the proposed WCLA maximizes the expected reinforcement signals received from the environment. In our analysis, there are two special points:

1. The analysis reported in this paper regarding the WCLA is independent of the structure of the WCLA and of how the learning automata are interconnected.
2. In each iteration, only the learning automata that have chosen a new action update their action probabilities.

To analyze the behavior of the WCLA through time, we approximate the learning algorithm used in the WCLA by an ODE (Ordinary Differential Equation). Since the objective of the WCLA is to maximize the problem in Eq. (A3), we must generate a relation between the ODE solution and the maxima of $f(P) = E[\beta|P]$. To approximate the learning algorithm by an ODE, we need some definitions and remarks:

Definition 3. The First Order Necessary Kuhn-Tucker (FONKT) conditions that are defined as follows are the basic requirement conditions for the vector P to be one of the local maxima of $f(P)$:

$$\left.\begin{aligned} &\frac{\partial f}{\partial p_{ij}}(P) + \eta_{ij} + \gamma_i = 0 \quad (a) \\ &\eta_{ij} p_{ij} = 0 \quad (b) \\ &\eta_{ij} \geq 0 \quad (c) \\ &p_{ij} \geq 0 \quad (d) \\ &\textstyle\sum_j p_{ij} \quad (d) \end{aligned}\right\} \forall i, j. \tag{A4}$$

Remark 1. To simplify the analysis process, we redefine the learning algorithm in Eq. (3) with the following vector format:

$$P(k+1) = P(k) + \lambda G(P(k), \theta(k)), \tag{A5}$$

where θ is the set of chosen actions by the LA and its neighbors at instance k and G is the updating rule in Eq. (3) made of a set of functions $G_{ij}, j = 1, \ldots, r_i, i = 1, \ldots, N$.

Definition 4. For every $\mathrm{P}=(p_1^T \ldots p_N^T)^T$, where p_i is a r_i-dimensional probability vector, we define

$$s_{ij}(P) = E[G_{ij}\big(P(k),\theta(k)\big)|P(k)=P]. \tag{A6}$$

Remark 2. Assume for learning *algorithm L*, we have the following assumptions:

A1. $\{\big(P(k),\beta(k)\big),\ k>1\}$ is a Markov Process.
A2. $Prob[\beta(k)\in \mathrm{B}|P(k),\beta(k-1)] = Prob[\beta(k)\in \mathrm{B}|P(k)]$ for any appropriate Borel set, B.
A3. $g(x)=E[G(P(k),\beta(k))|P(k)=x]$ is independent of k and that is globally Lipschitz.
A4. $\|G\big(P(k),\beta(k)\big)-g(P(k))\|^2 < M < \infty$, for some M and $\forall k$.

In Ref. 21, it was shown that the approximation ODE for the updating rule is defined by the following equation:

$$\frac{dz}{dt} = g(z), \quad z(0)=P(0).$$

Theorem 1. The approximation ODE for the updating rule defined by (3) in the WCLA is given by

$$\frac{dP_{ij}}{dt} = s_{ij}(P), \quad j=1,\ldots,r_i, \quad i=1,\ldots,N. \tag{A7}$$

Proof: So based on Remark 2, we need to show the assumptions (A1)–(A4) are satisfied in our proposed method. Because in the WCLA like other CLA, the choosing action in each iteration is independent of the previous actions, so the assumption (A1) is already satisfied. Also because the $\beta(k)$ is also independent of previous β, so it can be seen that the assumption (A2) is also verified. Now for assumption (A3), because all components of P, G, and θ in our learning algorithm are bounded and all components of G are continuous, this assumption is also satisfied. Finally, the assumption (A4) is satisfied because the variances are bounded due to all components of G being bounded. So, we proved the theorem.

The next step in the analysis of the WCLA is to study the ODE. In order to do this, we try to calculate the functions s_{ij}.

Definition 5. The drift Δp_{ij} for the jth component of vector p_i is denoted as

$$\Delta p_{ij} = E\big[p_{ij}(k+1)-p_{ij}(k)|P(k)=P\big]. \tag{A8}$$

Combining this equation and Eq. (A6), we can see that $\Delta p_{ij}(k) = \lambda s_{ij}(P(k))$.

Lemma 1. The drift Δp_{ij} for algorithm (A6) is

$$\Delta p_{ij} = \lambda p_{ij}\sum_s p_{is}\left(\frac{\partial f}{\partial p_{ij}} - \frac{\partial f}{\partial p_{is}}\right), \tag{A9}$$

where f(.) is as defined by (A3). The proof is in Ref. 10.

Remark 3. It can see that Δp_{is} does not depend directly to the time instance k. Now, from (A9), we have

$$s_{ij}(P) = p_{ij}\sum_s p_{is}\left(\frac{\partial f}{\partial p_{ij}} - \frac{\partial f}{\partial p_{is}}\right). \tag{A10}$$

So, the approximating ODE for our algorithm is given by (A7), where s_{ij} are given by (A10). In the rest of this analysis we, try to specify the ODE solutions. As mentioned above, our goal is to show that the solutions of the ODE can solve the maximization problem $f(P)$. To do this, we first present that the function $f(P(t))$ is an increasing monotonically function along the paths of ODE.

Lemma 2. Let u be a fake variable with the possible values like the value space in vector P. Now, we have $\left(\frac{df}{dt}\right)(u) \geq 0$ along the paths of the ODE (A6). Also, if and if only u be an equilibrium point for the ODE, we have $\left(\frac{df}{dt}\right)(u) = 0$.

Proof: By using the differentiation chain rule, we have

$$\begin{aligned}\frac{df}{dt}(u) &= \sum_i\sum_j \frac{\partial f}{\partial p_{ij}}(u)s_{ij}(u)\\ &= \sum_{i,j,k} u_{ij}u_{ik}\left[\frac{\partial f}{\partial p_{ij}}(u) - \frac{\partial f}{\partial p_{ik}}(u)\right]\frac{\partial f}{\partial p_{ij}}(u).\end{aligned} \tag{A11}$$

Now, let $v_{ij} = \frac{\partial f}{\partial p_{ij}}(u)$, then

$$\begin{aligned}\frac{df}{dt}(u) &= \sum_{i,j,k<j} u_{ij}u_{ik}\big[(v_{ij}-v_{ik})v_{ij} + (v_{ik}-v_{ij})v_{ik}\big]\\ &= \sum_{i,j,k<j} u_{ij}u_{ik}\Big[(v_{ij}-v_{ik})^2\Big] \geq 0.\end{aligned}$$

Since all u_{ij}'s are nonnegative. Now, $\left(\frac{df}{dt}\right)(u)=0$ proves the following equation:

$$u_{ij}u_{ik}(v_{ij}-v_{ik}) = 0 \quad \forall i,j,k. \tag{A12}$$

So, we have

$$\sum_k u_{ij}u_{ik}(v_{ij}-v_{ik}) = 0. \tag{A13}$$

From (A10), the left-hand side of Eq. (A13) is the expression for $\left(du_{ij}/dt\right)$, and so it can be seen that u is one equilibrium point of the ODE. Also it can be easily seen that $\left(\frac{df}{dt}\right)(u)=0$ if u be one equilibrium point of the ODE. So, we have $\left(\frac{df}{dt}\right)(u)=0$ if and if only u be one equilibrium point of the ODE.

In the next step, we try to relate the FONKT conditions and the equilibrium points of the ODE for the considered maximization problem in the WCLA.

Lemma 3. In Ref. 22, it is shown that if u verifies the FONKT conditions in Eq. (A4) for the optimization problem (A3), then it is an equilibrium point of the ODE (A7) but if it does not verify the FONKT conditions and also it be an equilibrium point of the ODE, then it is unstable.

Remark 4. Until now we have only shown the relation between the equilibrium points of the ODE and satisfying FONKT conditions. Now, we should show that if u is a stable equilibrium point of an ODE and it satisfies the FONKT conditions, it is a maximum of the optimization problem (A3) and vice versa.

Lemma 4. If u is an asymptotically stable equilibrium point of an ODE and it satisfies the FONKT conditions, it is a maximum of the optimization problem (A3) and vice versa. The proof is shown in Ref. 22.

In the rest of this Appendix, we try to introduce the WCLA expediency:

Definition 6. We called an automaton, a pure-chance automaton if it selects its action using equal probability over its action set. In other words, if the cardinality of the action-set be r then an automata called pure-chance automaton if

$$p_i = \frac{1}{m}, \quad i = 1, 2, \ldots, m. \tag{A14}$$

Definition 7. Pure-chance WCLA is a WCLA, wherein each cell there is a pure-chance automaton rather that a learning automaton. Also, P_{pc} is the configuration of a pure-chance WCLA.

Definition 8. We called a WCLA an expedient with respect to ith LA if in long run the ith LA does better than pure-chance automaton based on the following inequaility:

$$lim_{k\to\infty} E[\beta | p_i^k] < \frac{1}{m} \mathrm{E}\left[\beta \,|\, p_{i_{pc}}^k\right]. \tag{A15}$$

Theorem 2. A WCLA is expedient using the learning algorithm in Eq. (3).

Proof. Because the $f(P(t))$ strictly increases on t along the ODE solutions, and the learning algorithm in Eq. (3) is an expedient with a small enough learning parameter λ, so based on lemma 2, the WCLA with learning algorithm (3) is also expedient.

[1]J. Kari, "Reversibility of 2D cellular automata is undecidable," Phys. D: Nonlinear Phenom. **45**(1–3), 379–385 (1990).
[2]N. H. Packard and S. Wolfram, "Two-dimensional cellular automata," J. Stat. Phys. **38**(5–6), 901–946 (1985).
[3]E. Fredkin, "An informational process based on reversible universal cellular automata," Phys. D: Nonlinear Phenom. **45**(1–3), 254–270 (1990).
[4]M. R. Meybodi, H. Beigy, and M. Taherkhani, "Cellular learning automata and its applications," Sharif J. Sci. Technol. **19**(25), 54–77 (2003).
[5]H. Beigy and M. R. Meybodi, "A self-organizing channel assignment algorithm: A cellular learning automata approach," in *International Conference on Intelligent Data Engineering and Automated Learning* (2003), pp. 119–126.
[6]H. Beigy and M. R. Meybodi "Asynchronous cellular learning automata," Automatica **44**(5), 1350–1357 (2008).
[7]H. Beigy and M. R. Meybodi, "A mathematical framework for cellular learning automata," Adv. Complex Syst. **7**(03n04), 295–319 (2004).
[8]H. Beigy and M. R. Meybodi, "Open synchronous cellular learning automata," Adv. Complex Syst. **10**(04), 527–556 (2007).
[9]H. Beigy and M. R. Meybodi, "Cellular learning automata with multiple learning automata in each cell and its applications," IEEE Trans. Syst., Man, Cybern. Part B: Cybern. **40**(1), 54–65 (2010).
[10]M. A. L. Thathachar and P. S. Sastry, *Networks of Learning Automata: Techniques for Online Stochastic Optimization* (Springer Science & Business Media, 2011).
[11]M. T. Setlin *et al*., *Automaton Theory and Modeling of Biological Systems* (Elsevier, 1973).
[12]A. G. Barto and P. Anandan "Pattern-recognizing stochastic learning automata," IEEE Trans. Syst., Man, Cybern. **3**, 360–375 (1985).
[13]K. S. Narendra and K. Parthasarathy, "Learning automata approach to hierarchical multiobjective analysis," IEEE Trans. Syst., Man, Cybern. **21**(1), 263–272 (1991).
[14]P. S. Sastry, V. V. Phansalkar, and M. A. L. Thathachar, "Decentralized learning of Nash equilibria in multi-person stochastic games with incomplete information," IEEE Trans. Syst., Man, Cybern. **24**(5), 769–777 (1994).
[15]O.-C. Granmo, B. J. Oommen, S. A. Myrer, and M. G. Olsen, "Learning automata-based solutions to the nonlinear fractional knapsack problem with applications to optimal resource allocation," IEEE Trans. Syst., Man, Cybern. Part B **37**, 166–175 (2007).
[16]B. J. Oommen and M. K. Hashem, "Modeling the "learning process" of the teacher in a tutorial-like system using learning automata," IEEE Trans. Cybern. **43**(6), 2020–2031 (2013).
[17]A. Yazidi, O.-C. Granmo, and B. J. Oommen, "Learning-automaton-based online discovery and tracking of spatiotemporal event patterns," IEEE Trans. Cybern. **43**(3), 1118–1130 (2013).
[18]M. Esnaashari and M. R. Meybodi, "Irregular cellular learning automata," IEEE Trans. Cybern. **45**(8), 1622–1632 (2015).
[19]M. Ghavipour and M. R. Meybodi, "Irregular cellular learning automata-based algorithm for sampling social networks," Eng. Appl. Artif. Intell. **59**, 244–259 (2017).
[20]S. M. Vahidipour, M. R. Meybodi, and M. Esnaashari, "Adaptive Petri net based on irregular cellular learning automata with an application to vertex coloring problem," Appl. Intell. **46**, 272–284 (2017).
[21]V. V. Phansalkar, "Learning automata algorithms for connectionist systems local and global convergence," Ph.D. thesis (Department of Electrical Engineering, Indian Institute of Science, 1991).
[22]M. Thathachar and V. V. Phansalkar, "Convergence of teams and hierarchies of learning automata in connectionist systems," IEEE Trans. Syst., Man Cybern. **25**(11), 1459–1469 (1995).
[23]J. A. Torkestani and M. R. Meybodi, "A cellular learning automata-based algorithm for solving the vertex coloring problem," Expert Syst. Appl. **38**(8), 9237–9247 (2011).
[24]D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon, "Optimization by simulated annealing: An experimental evaluation and part II, graph coloring and number partitioning," Oper. Res. **39**(3), 378–406 (1991).
[25]M. Caramia and P. Dell'Olmo, "Embedding a novel objective function in a two-phased local search for robust vertex coloring," Eur. J. Oper. Res. **189**(3), 1358–1380 (2008).