ISBN 978-1-84626-xxx-x
Proceedings of 2009 International Conference on Machine Learning and Computing
Perth, Australia, 10-12 July, 2009, pp. xxx-xxx

Solving the Bandwidth Multicoloring Problem: A Cellular Learning Automata Approach

Alireza Enami Eraghi[1], Javad Akbari Torkestani[2] and Mohammad Reza Meybodi[3]

[1]Computer Engineering Department, Islamic Azad University, Farahan Branch, Iran. alireza_enami@yahoo.com
[2]Department of Computer Engineering, Islamic Azad University, Science and Research Branch, Tehran, Iran. j-akbari@iau-arak.ac.ir
[3]Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. mmeybodi@aut.ac.ir

Abstract. In this paper, an approximation algorithm based on irregular cellular learning automata (ICLA) is proposed for solving the bandwidth coloring and bandwidth multicoloring problems. In the proposed algorithm, the multicoloring problem is first simplified as a vertex coloring problem, where each vertex is colored by only one color. To do so, the input graph must be transformed into a base graph. Then, a learning automaton is assigned to each vertex of the resultant graph. At each stage, each learning automaton randomly chooses one of its action according to its action probability vector. If the difference between the weight associated with the action selected by one cell and its neighboring cells is greater than or equal to the weight of the edge between the cells, then the selected action is rewarded; otherwise, the selected action is penalized. The proposed algorithm is repeated until all the cells are rewarded. The proposed algorithm is compared with the algorithms of Lim, Prestwich and Malaguti. Experimental results show that the proposed algorithm outperforms the others.

Keywords: Multicoloring problem, Bandwidth coloring problem, cellular learning automata.

1. Introduction

Given an undirected graph G=(V,E), where V is the vertex set and E is the edge set, the classical vertex coloring problem (VCP) requires to assign a color to each vertex (i.e. to label each vertex $i \in V$ with an integer c(i) corresponding to a color) in such a way that colors on adjacent vertices are different and the maximum color used is minimized. Vertex coloring is a well known NP-hard problem [1]. Legal coloring is to assign a color to each vertex; where colors on adjacent vertices are different and optimal coloring graph is the necessary minimum number for an legal graph coloring.

In the bandwidth coloring problem (BCP) distance constraints are imposed between adjacent vertices, replacing the difference constraints. A distance d(i,j) is defined for each edge $(i, j) \in E$, and the absolute value of the difference between the colors assigned to i and j must be at least equal to this distance: $|C(i) - C(j)| \geq d(i, j)$. In the multicoloring problem (MCP) a positive weight w(i) is defined for each vertex $i \in V$, representing the number of colors that must be assigned to vertex i, so that for each $(i, j) \in E$ the intersection of the color sets assigned to vertices i and j is empty. The bandwidth multicoloring problem (BMCP) is the combination of the two problems above. Each vertex i must be assigned w(i) colors, and each of these colors must respect the distance d(i,j) with all the colors assigned to any adjacent vertex j. In this case, loop d(i,i) represents the minimum distance between different colors assigned to the same vertex i [2][3].

BCP, MCP and BMCP are NP-hard because they generalize VCP, that is known to be NP-hard [4]. Clearly, a VCP instance is a BMCP instance where all the distances are equal to 1 and each vertex must receive only one color. BCP, MCP and BMCP allow complex situations to be modeled, like for example the assignment of frequencies to different cells in a mobile network [3].

In this paper , we propose an algorithm based on irregular cellular learning automata for solving the BCP and BMCP. The proposed algorithm is compared with algorithms of Lim [5][6], Prestwich [7] and Malaguti [8]. Simulation results show that the proposed algorithm provides better results.

The rest of this paper is organized as follows: in section 2, a brief review of irregular cellular learning automata is given. In section 3, we describe proposed algorithm based on irregular cellular learning automata and The performance of the proposed algorithm is evaluated through the simulation experiments in section 4. Section 5 is conclusion.

2. Irregular Cellular Learning Automata

An irregular cellular learning automata [9] is a CLA in which the restriction of rectangular grid structure in traditional CLA is removed. This generalization is expected because there are applications such as wireless sensor networks, immune network systems, graph related applications, etc. that cannot be adequately modeled with rectangular grids. An ICLA is defined as an undirected graph in which, each vertex represents a cell which is equipped with a learning automaton. The learning automaton residing in a particular cell determines its state (action) on the basis of its action probability vector. Like CLA, there is a rule that the ICLA operate under. The rule of the CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to the LA residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is nonstationary because the action probability vectors of the neighboring LAs vary during the evolution of the ICLA.

3. The Proposed Algorithm

In this algorithm, at first, input graph must be transformed into the base graph. Then, a learning automaton is assigned to each node of the resulting graph. For example, input graph shown in Fig. 1 with w(1)=1 ،w(2)=2 ،w(3)=2 ،d(1,1)=1 ،d(1,2)=2 ،d(1,3)=1 ،d(2,2)=2 ،d(2,3)=4 and d(3,3)=3, is transformed into the graph in figure 2. as shown in figure 2, after transformation, the weight of each vertex is 1.

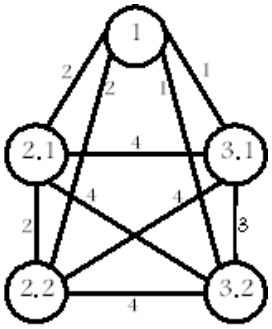

Fig. 2: Input graph after transformation

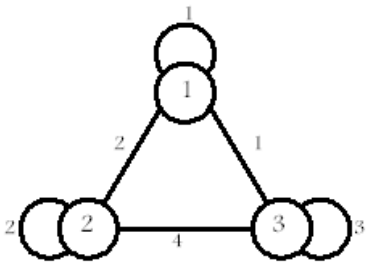

Fig. 1: Input graph before transformation

In this algorithm, we map the transformed graph (base graph) to into a irregular cellular learning automata. To do this, each node is associated with a cell, and a learning automaton is assigned to each cell. Two cells in the cellular learning automata are neighbour, if and only if they are adjacent to each other in the base graph. The number of actions for each automaton in the cellular learning automata is the number of colors required for coloring the graph, and so the color-set of each vertex denotes the action-set of its corresponding cell. Learning algorithm of each cell is assumed to be a L_{RP} algorithm, and reward and penalty parameters to be 0.01. for all learning automata, the initial probability of selecting an action is inversely proportional to the number of colors.

In [10], it is shown that χ (the number of used colors for a legal coloring) is less than or equal to Δ+1, where Δ is the degree of graph, then. It means that $\chi \le \Delta + 1$. In this algorithm, the cardinality of the action-set is $(\Delta+1)\times\left(\underset{\forall(i,j)\in E}{Max}\left(d(i,j)\right)\right)$ such as in pure graph coloring $\underset{\forall(i,j)\in E}{Max}\left(d(i,j)\right)=1$.

In this algorithm, at first, the input graph must be transformed into the base graph. If $|C(i)-C(j)| \ge d(i,j)$ for all j, where $(i,j)\in E$ then the action chosen by cell i is rewarded, otherwise this

action is penalized. $C(i)$ denotes the color selected by cell i. In our proposed algorithm, the number of actions can be taken by each learning automaton is $(\Delta+1)\times\left(\underset{\forall(i,j)\in E}{Max}\left(d(i,j)\right)\right)$.

If the action selected by the learning automaton of a given cell satisfies the condition above, then, it is rewarded, otherwise, cell action is penalized. The algorithm repeats until all cells are rewarded. Steps of the proposed algorithm have been illustrated bellow:

Step 1: Transforming the problem to a pure graph coloring problem.

In this step, a vertex in the input graph is mapped to $w(i)$ vertices in the base graph in such a way that they have not loop and their weight is 1, where $w(i)$ is the weight of cell i . Two neighboring vertices in the input graph are also neighbors in resultant graph, and the weight of the edges (i.e., distances) remain unchanged.

Step 2: Constructing an irregular CLA isomorphic to the input graph.

In this step, each vertex of the resultant graph is associated with a cell and a learning automaton is assigned to each cell. Two cells are neighbors, if and only if they are adjacent to each other in the base graph.

Step3: Color selection phase

In this step, each learning automaton selects one of its actions from its action-set according to its probability vector for coloring its cell.

Step4: Evaluating the legal colorings

Legal coloring is to assign a color to each vertex in such a way that the colors of adjacent vertices are different.

Step5: Updating the action probability vectors

If the difference between the action selected by one cell and its neighbors is greater than or equal to the weight of the edge between the cells, then the selected action is rewarded; otherwise, the selected action is penalized.

Step6: The stop condition

The algorithm repeats until all cells are rewarded, otherwise, algorithm resumes from step 3.

Pseudo code of this algorithm is given in Fig. 3.

```
Algorithm CLA
Input : Graph G(V,E)
Output : The number of colors needs to be chosen for coloring the graph
Begin
   Transfer problem to pure graph coloring
   Construct an irregular CLA isomorphic to the input graph
   Repeat
      For all cells do in parallel
         Each cell chooses one of its actions according to its action probability vector
         If ( |c_i − c_j| ≥ d_(i,j) , for each e_(i,j) ∈ E ) then
            Reward the action chosen by automaton A_i
         Else
            Penalize the action chosen by automaton A_i
         End if
   Until all cells are rewarded by the random Environment
   Return the number of used colors
End.
```

Fig. 3: Pseudo code of the proposed algorithm

4. Experimental Results

To study the performance of our proposed algorithm we have conducted some simulation experiments on a number of hard to color benchmark graphs reported in DIMACS Standards [11] like geometric graphs GEOMn and GEOMnb (where n represents the number of vertices of the graph). In these geometric graphs, the vertices are uniformly randomly generated in a 10,000 by 10,000 grid, and are connected by an edge if they are close enough together. GEOMn instances correspond to sparse graphs. GEOMnb instances

correspond to denser graphs. We executed our algorithm on a PIV 2.4 MHz computer, with 512 MB RAM, under Windows XP. The results of the proposed algorithm are compared with algorithm results of Lim [5][6], Prestwich [7] and Malaguti [8]. The evaluation criteria are number of used colors for coloring (C) and running time in second (T). Each result in table 1 and 2 is average of 50 runs. The last column in tables 1 and 2 (diff.) represents the difference between the number of colors used in our proposed algorithm and the best-known results reported in the literature.

Table 1: number of colors and running time for bandwidth graph coloring with several algorithms

Graph	*Proposed Algorithm*		*Lim [5]*		*Prestwich[7]*		*Malaguti[8]*		*Diff.*
	T	*C*	*T*	*C*	*T*	*C*	*T*	*C*	
GEOM20	82.13	9	1	21	0	21	0	21	-12
GEOM20b	83.75	13	0	14	0	13	0	13	0
GEOM30	85	16	0	29	0	28	0	28	-12
GEOM30b	85.72	18	0	26	0	26	0	26	-8
GEOM40	86.81	21	3	28	0	28	0	28	-7
GEOM40b	87.58	25	4	34	0	33	0	33	-8
GEOM50	87.9	26	5	28	0	28	0	28	-2
GEOM50b	88.92	38	7	38	0	35	0	35	3
GEOM60	89.44	27	1	34	0	33	0	33	-6
GEOM60b	90.46	40	5	46	0	43	29	41	-1
GEOM70	90.72	35	1	38	0	38	0	38	-3
GEOM70b	91.66	51	0	54	1	48	52	48	3
GEOM80	91.92	43	8	42	0	41	0	41	2
GEOM 80b	92.33	60	1	66	12	63	150	63	-3
GEOM90	93.43	45	1	46	3	46	0	46	-1
GEOM90b	94.4	63	20	77	2	72	1031	70	-7
GEOM100	94.71	48	33	51	0	50	2	50	-2
GEOM100b	95.54	73	9	83	15	73	597	73	0
GEOM 110	96.13	49	2	53	4	50	3	50	-1
GEOM 110b	96.95	78	17	88	2	79	676	78	0
GEOM 120	97.02	57	1	62	4	60	0	59	-2
GEOM 120b	97.93	81	4	98	9	86	857	84	-3

In BCP, the results on 22 instances reported in DIMACS show that our proposed algorithm decreases the number of used colors in 16 instances compared to the best results reported by the previous algorithms. Its does not change the best previous results in 3 instances and increases the number of required colors in 3 instances. The results also show that our proposed algorithm has a longer running time in comparison with Lim's algorithm and Prestwich's algorithm. The time complexity of our BCP algorithm is considerably shorter comparing Malaguti's algorithm, specifically for dense graphs.

Table 2: number of colors and running time for bandwidth multicoloring graph with several algorithms

Graph	*Proposed Algorithm*		*Lim [5]*		*Lim [6]*		*Prestwich[7]*		*Malaguti[8]*		*Diff.*
	T	*C*	*T*	*C*	*T*	*C*	*T*	*C*	*T*	*C*	
GEOM20	118.85	140	0	149	17	149	4	149	18	149	-9
GEOM20b	119.87	44	0	44	2	44	0	44	5	44	0
GEOM30	120.39	161	0	160	23	160	0	160	1	160	1
GEOM30b	121.41	76	0	77	7	77	0	77	0	77	-1
GEOM40	121.67	168	3	167	47	167	1	167	20	167	1
GEOM40b	122.61	74	8	76	10	74	4	74	1	74	0
GEOM50	122.87	227	41	224	77	224	1	224	1197	224	3
GEOM50b	124.01	84	53	87	15	87	1	86	197	83	1
GEOM60	124.38	258	46	258	96	258	77	258	139	258	0
GEOM60b	125.35	115	300	119	23	116	12	116	460	115	0
GEOM70	125.66	270	25	279	138	273	641	277	1413	272	-2
GEOM70b	126.49	118	136	124	30	121	55	119	897	117	1
GEOM80	127.08	380	4041	394	204	383	361	398	132	388	-3
GEOM80b	127.9	141	3230	145	39	141	37	141	1856	141	0
GEOM90	127.96	329	4095	335	248	332	44	339	4160	332	-3
GEOM90b	128.96	142	648	157	46	157	303	147	1750	144	-2
GEOM100	129.3	403	631	413	311	404	7	424	3283	410	-1
GEOM100b	129.87	157	4893	172	55	170	367	159	3699	156	1
GEOM 110	130.39	384	577	389	368	383	43	392	2344	383	1
GEOM 110b	131	209	12	210	68	206	5	208	480	206	3
GEOM 120	131.58	395	1825	409	408	402	9	417	2867	396	-1
GEOM 120b	132.43	190	869	201	97	199	3	196	3292	191	-1

The experiments on 22 instances reported in DIMACS also reveal that our BMCP algorithm decreases the size of the color set required for coloring the 9 instances compared to the best results reported by the previous algorithms. In 5 instances, our algorithm does not change the previous results, and in 7 instances performs worst. Comparing the results, we find that our proposed algorithm considerably outperforms the other existing algorithms in terms of the running time of algorithm.

5. Conclusion

In this paper, we proposed an algorithm for solving two types of graph coloring problems, namely BCP and BMCP based on irregular cellular learning automata with L_{RP} learning algorithm. The proposed algorithm was compared with the algorithms of Lim, Prestwich and Malaguti. The experiment results show the superiority of the proposed BCP algorithm over the existing methods both in terms of the number of colors and running time of algorithm. The results also show that the proposed BMCP algorithm colors the graph with a smaller number of colors in a shorter running time compared to the existing algorithms.

References

[1] R. Karp, "Reducibility among Combinatorial Problems", Complexity of computer computations, pp. 85-104, 1972.

[2] W. Hale, "Frequency Assignment: Theory and Applications", in Proceedings of the IEEE, Vol. 68, pp. 1497-1514, 1980.

[3] R. Opsut and F. Roberts, "On the Fleet Maintenance, Mobile Radio Frequency, Task Assignment, and Traffic Phasing Problems," in The Theory and Applications of Graphs, Wiley, New York, pp. 479-492, 1981.

[4] M.R. Garey and D.S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Freedman, New York, 1979.

[5] A. Lim, X. Zhang, Y. Zhu, "A Hybrid Methods for the Graph Coloring and its Related Problems", in: Proceedings of MIC2003: The Fifth Metaheuristic International Conference, Kyoto, Japan, 2003.

[6] A. Lim, Q. Lou, B. Rodrigues, Y. Zhu, "Heuristic Methods for Graph Coloring Problems", in: Proceedings of the 2005 ACM Symposium on Applied Computing, Santa Fe, NM, pp. 933–939, 2005.

[7] S. Prestwich, "Generalized Graph Colouring by a Hybrid of Local Search and Constraint Programming", Technical Report, Cork Constraint Computation Center, Ireland, 2005.

[8] E. Malaguti and P. Toth ,"An Evolutionary Approach for Bandwidth Multicoloring Problems", European Journal of Operational Research, Vol. 189, pp. 638–651, 2008.

[9] M. Asnaashari and M.R. Meybodi, "Irregular Cellular Learning Automata and Its Application to Clustering in Sensor Networks", Proceedings of 15th Conference on Electrical Engineering (15th ICEE), Volume on Communication, Telecommunication Research Center, Tehran, Iran, May 15-17, 2007.

[10] R. Diestel, "Graph Theory", 3rd Edition, Springer-Verlag, New York, 2005.

[11] M.A. Trick, Computational symposium: Graph coloring and its generalizations, 2002. <http://mat.gsia.cmu.edu/COLOR02/>.