Second International Congress on Technology, Communication and Knowledge (ICTCK2015)
November, 11-12, 2015 - Mashhad Branch, Islamic Azad University, Mashhad, Iran

Vertex Coloring Based on Artificial Bee Colony Algorithm

Vahid Chahkandi
Department of Computer Engineering
Islamic Azad University, Mashhad Branch
Mashhad, Iran
chahkandi.vahid@gmail.com

Omid Mirzaei
Computer Security Lab (COSEC)
Universidad Carlos III de Madrid
Madrid, Spain
omid.mirzaei@uc3m.es

***Abstract*— Given an undirected graph with a set of vertices and edges, vertex coloring, a well-known classical optimization problem in graph theory, consists of partitioning all vertices into independent sets and assigning unique colors to adjacent vertices with an effort to use the least number of colors. This paper presents a new algorithm based on artificial bee colony for solving the aforementioned problem. The proposed algorithm is evaluated on the DIMACS challenging benchmarks and computational results show that the presented method achieves highly competitive results. The outstanding outcomes of our algorithm are related to its computational time and also the number of colors it uses. Our simulations show that the computational time is considerably short and also the number of colors for vertex coloring is optimal in most of the cases.**

KEYWORDS: Graph Coloring, Vertex Coloring, Artificial Bee Colony, Optimization Problems

I. INTRODUCTION

Graph coloring, also known as vertex coloring, is one of the challenging optimization problems in graph theory. This problem is interesting not only because it is computationally intractable, but also because it has numerous practical applications, including task scheduling [1], register allocation [2], timetabling [3], and channel frequency assignment [4]. For these reasons, the graph coloring problem has been addressed by a number of approaches over the past few decades. However, none of them has turned out to be a clear winner.

A legal k-coloring of an undirected graph $G = (V, E)$, with V vertices and E edges, is the process of partitioning V into k independent sets in which each set is a subset of non-adjacent vertices of G. Considering this definition, graph coloring aims at finding the smallest k for a given graph G (its chromatic number $\chi(G)$) such that G has a legal k-coloring [5].

Greedy coloring shows simply that $\chi(G) \leq \Delta(G) + 1$, where $\Delta(G)$ denotes the maximum vertex degree of G. In other words, greedy coloring is looking for some ordering of vertices so that the least number of colors could be assigned to them (colors are assumed to be positive integers) having in mind not to assign similar colors to any adjacent vertices. A graph with maximum degree of Δ can be colored using at most $\Delta + 1$ colors; however, computing the chromatic number of a graph has proved to be an NP-hard problem. Thus, efficient algorithm that uses close to $\chi(G)$ colors is of our interest [6].

As mentioned earlier, minimizing the total number of colors is one of the primary objectives in the graph coloring problem. However, other criteria might also be considered. For instance, a second and modified objective might be to use the least possible colors at the least possible time which is an important objective in areas where time plays an important role. Problems with these kinds of objectives can be addressed with the help of parallelism and distributed computing although distributed environments could create some potential challenges for this optimization problem. The main challenge would be to obtain the least number of colors as the graph structure is unknown and only limited information regarding the neighbors of each vertex is available.

Several algorithms have been proposed for graph coloring problem most of which can be identified as local search methods. Well-known examples include the seminal TabuCOL algorithm [7], simulated annealing [8], GRASP [9], iterated local search [10], neighborhood search [5], reactive partial tabu search [6], variable space search [11] and clustering-guided tabu search [12]. Moreover, several algorithms based on swarm intelligence have been developed to deal with graph coloring problem, namely ant colony optimization [13][14][15][16][17][18], particle swarm optimization [19], and neural networks [20][21] to name a few.

On the other hand, swarm-based solutions such as Artificial Bee Colony (ABC) algorithm which was introduced by Karaboga in [22], have also been applied to this problem. Two major contributions based on ABC algorithm include [23], and [24]. The main difference between these approaches and the proposed ABC algorithm in this paper is related to the fitness function. In our method, the degrees of vertices (i.e. the number of edges intersect a specific vertex) are used as a unique numerical fitness value by bees to perform the local searches. However, in [23] and [24], fitness functions are calculated based on the assigned colors to vertices. Most importantly, the authors in [23] and [24] have tested their approaches on their own constructed graph samples and have not proved the efficiency of their methodologies by applying them to well-known benchmarks.

Here, in this paper, a novel artificial bee colony algorithm is presented to solve the problem of vertex coloring with the

978-1-4673-9762-9/15/$31.00 ©2015 IEEE

idea to save the computational time and also to use the least possible colors. It is expected that the suggested method can deal with the unknown structure of graphs because it consists of several local searches.

This paper is organized as follows. The proposed artificial bee colony algorithm is presented in section 2. Experimental results and the comparisons are all gathered in section 3. Finally, conclusions and some future extensions are presented in section 4.

II. The Proposed Artificial Bee Colony Algorithm

Generally speaking, the proposed Artificial Bee Colony (ABC) algorithm can be considered as a swarm intelligence based solution for this optimization problem. In what follows, the original version of ABC algorithm is described initially, and, next, the real concepts are correlated to the vertex coloring problem environment.

A. The original version of Artifical Bee Colony (ABC) algorithm

In the real ABC model, three groups of bees can be identified in each colony. These are employed bees, onlooker bees, and scouts. There are a number of food sources each of which represents a potential solution of an optimization problem. Moreover, sources have different nectar amounts which correspond to the quality (fitness) of various solutions. It is assumed that only one artificial employed bee can make use of each food source. Speaking in another way, there are some food sources around the hive and the number of employed bees in the colony is exactly the same as the number of these sources. Employed bees leave the colony for their food source and they start to dance around the hive as soon as they return. Later, at some point in time, employed bees try to create a modification on the source position in their memory with the purpose of finding new sources with higher amounts of nectars. Here, they leave the previous source by changing to scout bees in order to find new food sources with higher qualities. Scouts also replace abandoned sources with the new ones. On the other side, onlooker bees are monitoring the dancing behavior of employed bees upon which they will be able to find new food sources.

The main steps of the ABC algorithm can be summarized as below:

- A random population of positions is created for different food sources which are equal to the number of employed bees
- REPEAT
  - Each employed bee leaves the colony for a food source based on the position in her memory and evaluates its nectar amount. Then, she returns to the hive and starts to dance around it.
  - Each onlooker bee keeps an eye on the dance behavior of employed bees and picks out one of the sources according to her observations. Next, she flies to the chosen source and evaluates the nectar amount of that source.
  - Food sources which have been abandoned are identified, and they are replaced with the new food sources discovered by scouts.
  - The food source with the highest amount of nectar found so far is registered.
- UNTIL (requirements are met)

B. ABC algorithm adjusted for vertex coloring problem

The vertex coloring environment consists of numerous nodes and edges. The nodes of a graph are considered as food sources in the real ABC model and the degree of each vertex is considered as its nectar amount. Our proposed algorithm can be summarized as follows:

- Each employed bee chooses one vertex of the graph randomly, and then moves to that position (notice that only one employed bee is assigned to each node). Next, all the employed bees calculate the degrees of their own vertices (nectar amount) to share them with onlookers around the hive. Nectar values are used later by onlooker bees to move to the positions specified by employed ones as follows:

$$p_i = \frac{nectar_i}{\sum_{i=1}^{NB} nectar_i} \quad (1)$$

  Where NB is the number of bees (onlooker or employed), and $nectar_i$ is the degree of vertex i. The number of employed bees is considered half of the graph vertices in our algorithm.

- REPEAT
  - REPEAT
    - After the employed bees return to the hive, they share the information of nodes (their number) with onlooker bees.
    - The onlooker moves to the position (vertex) which is specified by employed bee using Eq. (1). When we want to decide for the position of each onlooker bee, a random variable is produced. If that random variable is shorter than or equal to the probability of vertex i, then the onlooker will move to that position. If the degree of that position is zero, an arbitrary color is assigned to it and we will not have any constraints in using that color for other vertices, because it is not adjacent with any vertices in the graph.
    - The neighbors of the current vertex are found and a random color is assigned to each of them from the palette.
    - The assigned colors are compared with adjacent vertices and if they are accepted, they will be added to Used_Color array which demonstrates the number of colors used so far.

- Else if they are not accepted, other colors are considered from previously used colors which are stored in Used_Color array.
- Else if onlookers are not yet able to find an appropriate color from the memory, they will refer to original palette and choose a new color from there. Then, the selected color is added to the memory (Used_Color array).
- After coloring all the neighbors of the assigned vertex (the nectar of the current vertex is subtracted when each of its neighbors are colored), the onlooker bee should choose a color for the current vertex itself. This color is either chosen from the Used_Color array or palette.
- The onlooker bee changes to a scout and goes to another unvisited vertex randomly after coloring the current vertex (the nectar of the current vertex is set to zero when an onlooker changes to a scout). For simplicity, one onlooker bee changes to a scout at each time instant.

  - UNTIL (for all onlooker bees)

- Onlooker bees will move to the positions specified by scouts and do the same coloring procedures.
- UNTIL (all vertices are colored and the used colors are less than Δ)

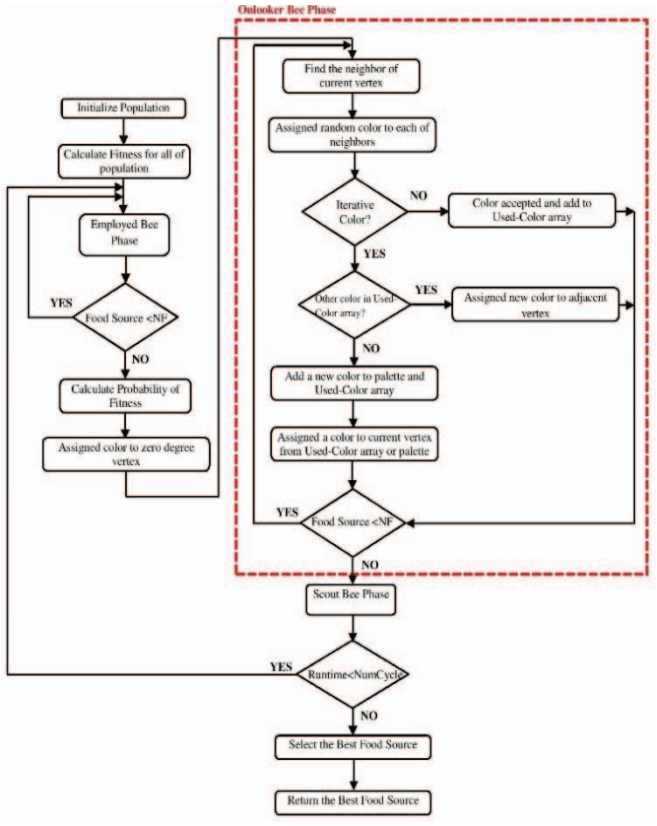

Figure 1. The flowchart of the proposed ABC algorithm

Note that the nodes of graph which play the role of food sources in artificial bee colony algorithm are not indeed a local solution as mentioned in the real model and they don't have any fitness values. They are just considered as food sources to make a simulation of the problem environment with real ABC algorithm.

To make all the theories of the proposed algorithm clearer, the whole process is illustrated as a flowchart in Fig. 1. Furthermore, the ABC algorithm has been implemented on a sample vertex coloring problem and the stages are all demonstrated in Fig 2.

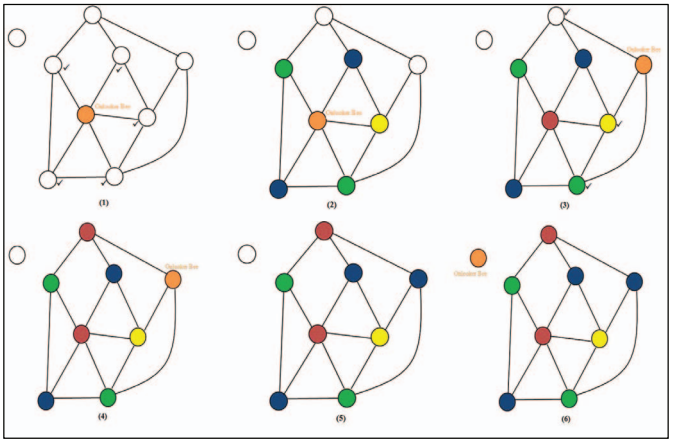

Figure 2. The stages of applying the proposed algorithm on a sample vertex coloring problem

III. EXPERIMENTAL RESULTS

In this section, several experiments are conducted to show the performance of the proposed ABC algorithm. The simulations are all applied on the well-known DIMACS coloring benchmarks and are divided into two separate sections. In the first section, the efficiency of the proposed ABC algorithm is presented using two different criterions which are the number of colors used by the algorithm in the process of vertex coloring and also its computational time. In the second section, the suggested method is compared with four other effective algorithms.

A. The simulation results of the proposed ABC algorithm

To show the effectiveness of the developed algorithm, it has been tested on 26 DIMACS datasets randomly. The simulations have been conducted in MATLAB environment in a computer with a Pentium 4 CPU 2.66 GHz, 4 GB Memory, 300 GB hard-disk capacity, and with Microsoft Windows Vista Business operating system. Furthermore, it should be mentioned that the provided results are indeed the average obtained by applying the ABC algorithm for 60 times on each sample. As it is clear from Table I, the presented algorithm has reached the optimal solutions (least number of colors) in all the cases. Moreover, the computational time of this method for these datasets has been demonstrated in Fig 3.

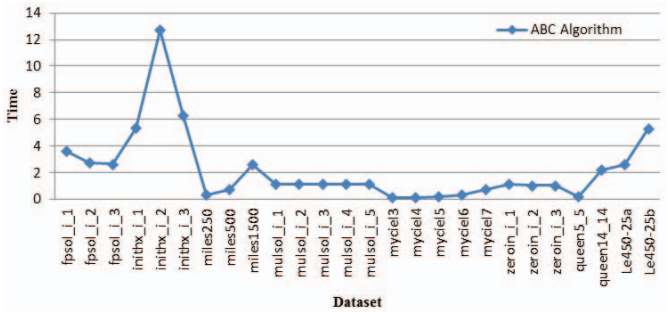

Figure 3. Computational time of the ABC algorithm for different datasets of DIMACS

TABLE I. OPTIMAL SOLUTIONS ON 26 RANDOM DIMACS DATASETS

No.	Datasets	N=|V|	M=|E|	Best K	Our Approach:	
					K	Time (sec)
1	fpsol.i.1	496	11654	65	65	3.6
2	fpsol.i.2	451	8691	30	30	2.71
3	fpsol.i.3	425	8688	30	30	2.63
4	inithx.i.1	864	18707	54	54	5.31
5	inithx.i.2	645	13979	31	31	12.74
6	inithx.i.3	621	13969	31	31	6.31
7	miles250	128	387	8	8	0.32
8	miles500	128	2340	20	20	0.69
9	miles1500	128	10396	73	73	2.61
10	mulsol.i.1	197	3925	49	49	1.1
11	mulsol.i.2	188	3885	31	31	1.09
12	mulsol.i.3	184	3916	31	31	1.12
13	mulsol.i.4	185	3946	31	31	1.1
14	mulsol.i.5	186	3973	31	31	1.11
15	myciel3	11	20	4	4	0.13
16	myciel4	23	71	5	5	0.14
17	myciel5	47	236	6	6	0.18
18	myciel6	95	755	7	7	0.3
19	myciel7	191	2360	8	8	0.7
20	zeroin.i.1	211	4100	49	49	1.15
21	zeroin.i.2	211	3541	30	30	1.02
22	zeroin.i.3	206	3540	30	30	1.02
23	queen5_5	25	160	5	5	0.2
24	queen14_14	196	8372	18	18	2.19
25	Le450-25a	450	8260	25	25	2.58
26	Le450-25b	450	8263	25	25	5.27

The most outstanding outcome we obtained in our simulations relates to the computational time of our method. The ABC algorithm proposed for vertex coloring is nearly the best among all other approaches suggested so far. This fact has been shown in Fig. 4 and is further discussed in remaining sections.

B. The comparative results between ABC algorithm and four other effective algorithms

As mentioned before, the performance of an algorithm proposed for this kind of problem is judged not only based on the number of colors it uses, but also based on its computational time. Therefore, four other effective algorithms have been selected in order to provide a fair comparison and to show the performance of the suggested solution. These are: (1) RBA algorithm [21], (2) EMA algorithm [8], (3) MACOL algorithm [25] and (4) ALS-COL algorithm [18]. It should be emphasized once again that we were determined to compare our method with the other ABC algorithm in [23]; however, we are unable to perform such a comparison currently since the other method has not been tested on a popular dataset.

1) ABC algorithm versus RBA algorithm

In recent years, neural networks have been applied to this problem, mostly based on Hopfield or Hopfield-like networks that employ binary neurons. The recursive binary adaptation (RBA) algorithm is the only other neural algorithm for which we have results on DIMACS benchmarks. All the other neural approaches that are proposed for minimum coloring so far either do not offer experimental results or their test graphs are not available. Consequently, the proposed algorithm has been applied on 7 random datasets and has been compared with RBA algorithm as it has been demonstrated in Table II.

TABLE II. EXPERIMENTAL RESULTS OF OUR ALGORITHM VS. RBA ALGORITHM ON 7 DIMACS DATASETS

No.	Datasets	N=|V|	M=|E|	Best K	Our Approach:		RBA	
							K	Time (sec)
1	R125.1	125	209	5	5	0.25	5	6
2	R250.1	250	867	8	8	0.44	9	103
3	R1000.1	1000	14378	20	22	4.67	26	3h.18m
4	Le450-15a	450	8168	15	20	3.55	22	1162
5	Le450-15b	450	8169	15	20	2.29	22	1168
6	DSJR500.1	500	7110	12	14	1.15	17	907
7	DSJC125.5	125	3891	17	25	1.09	26	42

The experimental results show that the ABC algorithm uses fewer colors than the RBA algorithm in all the cases and they are very close to optimal solutions. Furthermore, the computational time of the developed algorithm is significantly better than RBA in all the cases.

2) ABC algorithm versus EMA algorithm

In this section, the results of applying the ABC and EMA algorithms on 9 random datasets are presented. The outcomes are all gathered in Table III. Referring to this table, it is obvious that the number of colors used by ABC algorithm is less than the other method in most of the cases. However, it is equal with the number of colors used by EMA algorithm in some other cases. On the other side, the computational time of the proposed method is more than the EMA algorithm in all samples.

TABLE III. EXPERIMENTAL RESULTS OF OUR ALGORITHM VS. EMA ALGORITHM ON 9 DIMACS DATASETS

No.	Datasets	N=|V|	M=|E|	Best K	Our Approach:		EMA	
					K	Time (sec)	K	Time (sec)
1	DSJC250.1	250	6436	8	10	0.98	10.5	0.3
2	DSJR500.1	500	7110	12	12	1.15	13	0.6
3	fpsol2.i.1	496	11654	65	65	3.15	65	2.2
4	fpsol2.i.2	451	8691	30	30	3.2	30	1.4
5	inithx.i.1	864	18707	54	54	5.31	54	4.2
6	miles1500	128	10396	73	73	2.61	73	0.9
7	miles750	128	4226	31	31	1.14	31.8	0.3
8	mulsol.i.1	197	3925	49	49	1.1	49	0.5
9	myciel5	47	236	6	6	0.18	6	0.1

3) ABC algorithm versus MACOL algorithm

The third research work considered for our comparison is MACOL, which is a memetic based algorithm proposed for this kind of problem. This method is in fact a combination of a tabu search procedure with an evolutionary algorithm.

The experimental results of applying the ABC and MACOL algorithms on 5 random datasets have been presented in Table IV. Considering this table, it is clear that the two methods produce solutions very close to optimal solutions. However, the number of colors used by ABC algorithm is more than the other one in some cases. On the other hand, the computational time of the suggested method is considerably less than the other evolutionary algorithm.

TABLE IV. EXPERIMENTAL RESULTS OF OUR ALGORITHM VS. MACOL ALGORITHM ON 5 DIMACS DATASETS

No.	Datasets	N=|V|	M=|E|	Best K	Our Approach:		MACOL	
					K	Time (sec)	K	Time (sec)
1	DSJC125.1	125	736	5	5	0.3	5	60
2	DSJC125.5	125	3891	17	21	1.09	17	180
3	DSJC125.9	125	6961	44	50	1.78	44	240
4	DSJC250.1	250	6436	8	8	0.98	8	120
5	DSJR500.1	500	7110	12	12	1.15	12	240

4) ABC algorithm versus ALS-COL algorithm

The simulations end with the last comparison between the proposed algorithm and ALS-COL. The ALS-COL is a new and general ant methodology, where each ant is considered as a local search. The results of applying these two algorithms on 4 random datasets have been collected in Table V. As it is clear from this table, the ABC algorithm outperforms the other one in all the cases by using less number of colors while it is equal with optimal solutions in two cases and very close to optimal in others. Finally, the computational time of the proposed method is much better than the other one.

TABLE V. EXPERIMENTAL RESULTS OF OUR ALGORITHM VS. ALS-COL ALGORITHM ON 4 DIMACS DATASETS

No.	Datasets	N=|V|	M=|E|	Best K	Our Approach:		ALS-COL	
					K	Time (sec)	K	Time (sec)
1	DSJC500.1	500	24916	12	12	3.29	12	304
2	DSJR500.5	500	117724	123	127	15.44	128	181
3	flat300_28_0	300	21695	28	31	8.72	31	111
4	Le450-25c	450	17343	25	25	4.45	26	1847

IV. CONCLUSIONS AND FUTURE EXTENSIONS

In this paper, we have presented the ABC algorithm, a population based solution for vertex coloring which is a popular and challenging optimization problem. The proposed algorithm imitates the behavior of bees to deal with the aforementioned problem. In the suggested algorithm, a set of employed bees are initially assigned to some nodes of graph in a random manner. Next, these bees return their gathered information (degree of vertices) to the hive and share them with the onlookers. After this, the onlooker bees fly to the specified locations and start their processing operations for coloring the vertices. If some criterions are met, these employed bees become a scout one and will move to new locations. To show the efficiency of the proposed bee colony based algorithm, it has been applied on random samples of DIMACS dataset. The experimental results show that the ABC algorithm uses optimal number of colors in the majority of cases. Additionally, one noticeable outcome obtained from the simulations is that the computational time of the developed algorithm is significantly low when applying on this type of problem.

Ultimately, several future extensions can be considered to improve the performance of this algorithm as follows:

1. The assignment of employed bees to new vertices as scouts can be done more wisely. In our approach, this process is done randomly as the real model of bee colony algorithm. Maybe this change can lead to an improvement in the number of used colors.
2. The number of employed bees can be considered more than half of the vertices. This can also lead to less use of colors.

REFERENCES

[1] V. Lotfi and S. Sarin, "A graph coloring algorithm for large scale scheduling problems," *Comput. Oper. Res.*, vol. 13, no. 1, pp. 27–32, Jan. 1986.

[2] G. Chaitin, "Register allocation and spilling via graph coloring," *ACM SIGPLAN Not.*, vol. 39, no. 4, p. 66, Apr. 2004.

[3] D. de Werra, "An introduction to timetabling," *Eur. J. Oper. Res.*, vol. 19, no. 2, pp. 151–162, Feb. 1985.

[4] A. Gamst, "Some lower bounds for a class of frequency assignment problems," *IEEE Trans. Veh. Technol.*, vol. 35, no. 1, pp. 8–14, Feb. 1986.

[5] C. Avanthay, A. Hertz, and N. Zufferey, “A variable neighborhood search for graph coloring,” *Eur. J. Oper. Res.*, vol. 151, no. 2, pp. 379–388, 2003.

[6] I. Blöchliger and N. Zufferey, “A graph coloring heuristic using partial solutions and a reactive tabu scheme,” *Comput. Oper. Res.*, vol. 35, no. 3, pp. 960–975, 2008.

[7] S. Choudhary and G. N. Purohit, “Distributed algorithm for optimized vertex coloring,” *Proc. 2010 Int. Conf. Methods Model. Comput. Sci. ICM2CS-2010*, pp. 65–69, 2010.

[8] A. Di Blas, A. Jagota, and R. Hughey, “Energy function-based approaches to graph coloring,” *IEEE Trans. Neural Networks*, vol. 13, no. 1, pp. 81–91, 2002.

[9] M. Laguna and R. Martí, “A GRASP for coloring sparse graphs,” *Comput. Optim. Appl.*, vol. 19, no. 2, pp. 165–178, 2001.

[10] M. Chiarandini and T. Stützle, “An application of iterated local search to graph coloring problem,” *... Symp. Graph Color. its ...*, pp. 1–13, 2002.

[11] A. Hertz, M. Plumettaz, and N. Zufferey, “Variable space search for graph coloring,” *Discret. Appl. Math.*, vol. 156, no. 13, pp. 2551–2560, 2008.

[12] D. C. Porumbel, J. K. Hao, and P. Kuntz, “A search space ‘cartography’ for guiding graph coloring heuristics,” *Comput. Oper. Res.*, vol. 37, no. 4, pp. 769–778, 2010.

[13] D. Costa and A. Hertz, “Ants can colour graphs.”

[14] T. N. Bui, T. H. Nguyen, C. M. Patel, and K.-A. T. Phan, “An ant-based algorithm for coloring graphs,” *Discret. Appl. Math.*, vol. 156, no. 2, pp. 190–200, Jan. 2008.

[15] K. A. Dowsland and J. M. Thompson, “An improved ant colony optimisation heuristic for graph colouring,” *Discret. Appl. Math.*, vol. 156, no. 3, pp. 313–324, Feb. 2008.

[16] E. Salari and K. Eshghi, “An ACO Algorithm for Graph Coloring Problem,” in *2005 ICSC Congress on Computational Intelligence Methods and Applications*, 2005, pp. 1–5.

[17] SangHyuck Ahn, SeungGwan Lee, and TaeChoong Chung, “Modified ant colony system for coloring graphs,” in *Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint*, 2003, vol. 3, pp. 1849–1853.

[18] M. Plumettaz, D. Schindl, and N. Zufferey, “Ant Local Search and its efficient adaptation to graph colouring,” *J. Oper. Res. Soc.*, vol. 61, no. 5, pp. 819–826, Apr. 2009.

[19] G. Cui, L. Qin, S. Liu, Y. Wang, X. Zhang, and X. Cao, “Modified PSO algorithm for solving planar graph coloring problem,” *Prog. Nat. Sci.*, vol. 18, no. 3, pp. 353–357, Mar. 2008.

[20] P. M. Talaván and J. Yáñez, “The graph coloring problem: A neuronal network approach,” *Eur. J. Oper. Res.*, vol. 191, no. 1, pp. 100–111, Nov. 2008.

[21] A. Jagota, “An adaptive, multiple restarts neural network algorithm for graph coloring,” *Eur. J. Oper. Res.*, vol. 93, no. 2, pp. 257–270, 1996.

[22] D. Karaboga and B. Akay, “Artificial Bee Colony (ABC) Algorithm on Training Artificial Neural Networks,” *2007 IEEE 15th Signal Process. Commun. Appl.*, 2007.

[23] M. Dorrigiv and H. Yeganeh Markib, “Algorithms for the graph coloring problem based on swarm intelligence,” *AISP 2012 - 16th CSI Int. Symp. Artif. Intell. Signal Process.*, no. Aisp, pp. 473–478, 2012.

[24] R. S. Tomar, S. Singh, S. Verma, and G. S. Tomar, “A Novel ABC Optimization Algorithm for Graph Coloring Problem,” *2013 5th Int. Conf. Comput. Intell. Commun. Networks*, pp. 257–261, 2013.

[25] Z. Lü and J.-K. Hao, “A memetic algorithm for graph coloring,” *Eur. J. Oper. Res.*, vol. 203, no. 1, pp. 241–250, 2010.