O-BEE-COL: Optimal BEEs for COLoring Graphs

Piero Consoli[1] and Mario Pavone[2(✉)]

[1] School of Computer Science, University of Birmingham, Edgbaston Birmingham B15 2TT, UK
p.a.consoli@cs.bham.ac.uk
[2] Department of Mathematics and Computer Science, University of Catania, V.le A. Doria 6, 95125 Catania, Italy
mpavone@dmi.unict.it

Abstract. Graph Coloring, one of the most challenging combinatorial problems, finds applicability in many real-world tasks. In this work we have developed a new artificial bee colony algorithm (called O-BEE-COL) for solving this problem. The special features of the proposed algorithm are (i) a SmartSwap mutation operator, (ii) an optimized GPX operator, and (iii) a temperature mechanism. Various studies are presented to show the impact factor of the three operators, their efficiency, the robustness of O-BEE-COL, and finally the competitiveness of O-BEE-COL with respect to the state-of-the-art. Inspecting all experimental results we can claim that: (a) disabling one of these operators O-BEE-COL worsens the performances in term of the Success Rate (SR), and/or best coloring found; (b) O-BEE-COL obtains comparable, and competitive results with respect to state-of-the-art algorithms for the Graph Coloring Problem.

Keywords: Swarm intelligence · Artificial bee colony · Graph coloring problem · Combinatorial optimization

1 Introduction

Graph coloring is one of the most popular and challenging combinatorial optimization problems, playing a central role in graph theory. It can be formalized as follow: given an undirected graph $G = (V, E)$ a coloring of G is a mapping $c : V \rightarrow S$ ($\subseteq \aleph^+$) that assigns a positive integer to each vertex in V such that $c(u) \neq c(v)$ if u and v are adjacent vertices. The elements in S represent the available *colors*. The optimization version of *Graph Coloring Problem* (GCP) asks to find a mapping c with $S = \{1, 2, \ldots, k\}$ being of minimal size, i.e., finding the smallest integer k such that G has a $k - coloring$. This minimal cardinality of S is known as the *chromatic number* of G ($\chi(G)$). Thus formally, if $k > \chi$ then a graph G is called $k - colorable$, otherwise G is $k - chromatic$ if $k = \chi$. Computing the chromatic number of a graph is an NP–complete problem [17].

P. Legrand et al. (Eds.): EA 2013, LNCS 8752, pp. 243–255, 2014.
DOI: 10.1007/978-3-319-11683-9_19

Tackling and solving the GCP becomes crucial and important since it has a natural applicability in many real-world problems, such as scheduling [26], time tabling [12], manufacturing [19], frequency assignment [16], register allocation [8] and printed circuit testing [18]. The GCP can be tackled following two different approaches: *assignment* or *partitioning*. The first approach consist in the classical assignment of colors to vertices; whilst the latter one is based on partitioning the set of vertices V into k disjoint subsets $(V_1, V_2, \ldots, V_k)$ such that in any subset no two vertices are linked by an edge, i.e. if u and v are in V_i (for some $i \in \{1, \ldots, k\}$) then $(u, v) \notin E$. Every subset V_i represents a color class and forms an *Independent Set* of vertices. Although several pure population–based algorithms have been used to tackle the GCP, a hybrid approach where local search methods, specialized operators and evolutionary algorithms (EAs) are combined [25] might be more effective. This is, of course, due to the intractable nature of the GCP [5].

In this work we propose an *Artificial Bee Colony (ABC)* [24] algorithm for the GCP, based on three main features: (1) a new mutation operator, (2) an optimized version of the Greedy Partitioning Crossover (GPX) [15], and (3) a temperature mechanism. The ABC algorithm is a rather recent optimization technique inspired by the intelligent foraging behavior of a colony of bees, whose strength lies in the collective behavior of self-organized swarms that individually behave without any supervision. During the last decade, ABC has attracted quite a number of researchers, and it has been successfully applied mainly to continuous optimization problems [3,23], whilst, rather few works have appeared concerning discrete optimization problems (see, for example, [27,31]). In many cases the results obtained by ABC, including the ones of this work, demonstrate that this metaheuristic is able to compete with, and sometimes even outperforms, existing state-of-the-art algorithms for difficult optimization problems.

2 O-BEE-COL: An Artificial Bee Colony

The ABC algorithm takes inspiration from the intelligent foraging behavior of bees from a beehive. It is based on three main components: (1) *food source position*, corresponding to a feasible solution to the given problem; (2) *amount of nectar*, which indicates the quality of the solution; and (3) the bee types: *employed*; *onlooker*; and *scouts* bees. The first ones have the purpose to search for food sources, and, just found, storing their information. The onlooker bees select, and exploit the better food sources found taking advantage of the information learned from employed bees. Once one of the food sources is exhausted, the employed bees associated with it become scout bees, with the purpose to discover new food sources. Once discovered, they become again employed bees.

A new ABC heuristic has been developed in order to effectively coloring a generic graph. This algorithm is henceforth referred to as *"Optimal BEEs for COLoring"* (O-BEE-COL). The algorithm begins with the creation of the initial population, where each bee represents a permutation of vertices. Because the choice of the starting points in the search space become crucial we have designed,

and studied, three variants of O-BEE-COL in order to create the initial population. In the basic variant, it is randomly generated via a uniform distribution. The second variant, instead, uses a version partially randomize of RLF (Recursive Largest First) algorithm [10]. Of course, as expected, with this last variant O-BEE-COL shows better performances, because it begins the search from good solutions than the first variant. On the other hand, however using this second variant we have the disadvantage to get trapped into local optima easily, mainly in more complex instances. Thus, we have developed a third variant that is a mixed of the two previous ones. In this way, we introduce more diversity in the population in order to better exploration the search space, escaping from local optima, and exploiting good solutions at the same time. Analysing the comparisons among the three variants (not included in this paper due to limited space), the mixed one has produced the best performances obtaining better coloring in all instances tested. For example, if we take into account the *"le450_15c"* DIMACS instance [22], using the first variant the algorithm starts from a best solution found of 28 colors and improves the coloring until to reach a solution with 20 colors. Instead with the randomized RLF, although O-BEE-COL begins from 24 colors as best solution, it never improves this coloring found. If, however, O-BEE-COL incorporates the mixed variant, starting from a best solution of 24 colors (the one found by randomized RLF), at the end of the evolution it is able to coloring the graph with 15 colors, which is also the chromatic number for this instance.

The strength of O-BEE-COL is based on three main operators: mutation operator called *SmartSwap*; optimized version of GPX [15]; and Temperature mechanism, as in Simulated Annealing, which has the aim of self-regulating of some parameters of the algorithm. The mutation operator tries to reduce the number of colour classes deleting one of them, and reassigning its vertices inside other classes. Albeit is reasonable to think that this process might be easily performed in the smaller class, unfortunately often belong to it the most troublesome nodes, i.e. the ones harder to be handled. Thus, SmartSwap works primarily on these troublesome nodes with the aim to replace them with the ones more easy to be handled. In this way becomes easier the reassignment of the vertices, and therefore the delete of the class. To do that, SmartSwap allows a fixed number of constrains unsatisfied, which will be removed via the crossover operator: only *partLimit* constraints unsatisfied are allowed. With this operator we attempt to avoid that the solutions get trapped into local optima. Greedy Partitioning Crossover – GPX – is a well-known crossover originally proposed in [15], and based on strategy of considering more important the set of the vertices that belong to the same class rather than the colors assigned to each vertex. Via a round robin criterion two bees are selected for generating one offspring: the biggest colorclass of the two selected parents is copied into the new solution, and its vertices are removed from the color classes of the belonging parent. This process is performed until classes with only one vertex are encountered. In this case, the single node is inserted inside one of the existing classes. In O-BEE-COL we have designed an optimized version of GPX, which differs from the original

one basically in two aspects: (1) the number of solutions involved is determined by a parameter *partSol*; and (2) the cardinality of the colorclasses that must be copied into the new solution is determined by a parameter (*partLimit*). All colorclasses with cardinality greater or equal to *partLimit* will be copied inside the new solution. In this way, we want to force the transmission only of the best colorclasses to the offsprings. An experimental study conducted on the optimized GPX, also respect to the original one, confirmed us how these novelties introduced contribute significantly better on its performances (see plots in Fig. 3). The third novelty introduced in this work is the design of a Temperature mechanism that has the aim to dynamically self-handle some parameters during the evolution. The parameters bound to this self-regulating mechanism are: (1) number of parents involved in optimized GPX (*partSol*); (2) number of the improvement trails needed before to replace a solution (*evLimit*); (3) number of scout bees (*nScouts*); and (4) percentage of solutions that must be generated by randomized RLF during the scout bees phase (*percSol*). Whenever a better solution than the current one is found, the temperature mechanism sets the controlled parameters with their highest possible values, respectively $[100, 20, 5, 100\,\%]$. During the evolution, if no improvements occurred, then these values gradually decrease generation to generation until to reach their minimal values, which correspond to $[10, 5, 2, 10\,\%]$.

3 Results

In order to understand how the developed algorithm works, and how much is the contribution given by the novelties introduced we have performed many experiments using the classical DIMACS challenging benchmark[1]. O-BEE-COL has been tested on 22 instances (the most used), and it was compared with several algorithms, which represent the current state of the art for graph coloring problem. In this section we present all studies and experiments conducted, showing best tuning of the parameters; the impact factor contribution of the novelties designed; analysis on the running time; and comparisons conducted versus several algorithms. In most of the instances tested O-BEE-COL has found the best coloring, showing a robust convergence, and very competitive performances with respect the state of the art.

O-BEE-COL dynamics. One of the main goal when someone designs a generic EAs is to understand which is the best setting of the parameters because they strongly influence the performances of the algorithm. Thus many experiments have been performed with the aim to identify the best values of the parameters. As described in Sect. 2, O-BEE-COL depends on three parameters: population size ($popSize \in \{200,\ 500, 1000, 1500,\ 2000\}$) ; the lowest cardinality of the color classes allowed to be transmitted during the partitioning phase ($partLimit \in \{5,$ 10, 15, 18$\}$) ; and the percentage of Employed Bees ($percEmp \in \{10\,\%,\ 20\,\%,$ $50\,\%, 70\,\%,\ 90\,\%\}$) . To carefully analyse the proper tuning of the parameters, we

[1] http://mat.gsia.cmu.edu/COLOR/instances.html

conducted our study over several DIMACS instances, and for each combination of values we performed 10 independent runs. In Fig. 1 we show the convergence of O-BEE-COL on the instance *DSJC250.5* since it is challenging enough to make robust our study. Inspecting all 100 experiments over this instance, O-BEE-COL obtains the best performances in term of success rate (SR) with the combination $(200, 5, 10\,\%)$. Due to a limit space, we show for each parameter the convergence plots produced in combination with the other two best values. Analysing the left plot (varying $popSize$) is possible to see how with large population size, O-BEE-COL quickly gets down towards low values within few generations, after which it shows a steady-state. On the other hand, choosing small dimensions, albeit the algorithm needs more generations, it achieves still the best coloring. However, inspecting step-by-step the convergence for each value, $popSize = 200$, although is the slowest, it is the one that performs a better exploration of the search space with the result of producing a good trade-off for diversity into the population. In the middle plot, are shown the convergence curves produced varying the parameter $partLimit$. The lower bound to the color classes transmitted during the partitioning phase is the one that contributes most to the convergence speed of the algorithm, and it usually assumes values within the range $\left(2, \frac{|V|}{\chi}\right)$. In particular, assigning $partLimit = 5$, O-BEE-COL has a slower convergence but it reaches the best solution before than the others. In the right plot, and last of Fig. 1, is shown the contribution given by $percEmp$, which indirectly represents the exploitation phase of the best solutions found so far. For all curves, O-BEE-COL shows a good trend without presenting fast or slow convergences. Comparing the curves between them is possible to see how O-BEE-COL with low percentage of employed bees is able to better explore the search space, and, at the same time, exploit better the information gained so far. In fact, with the lowest percentage possible ($percEmp = 10\,\%$) the algorithm achieves the best solution before than the others. It is important to point out how the best values for the three parameters correspond to their minimal values tested. This indicates us that there exists a good balance of diversity into the population, which helps the algorithm to get out from local optima.

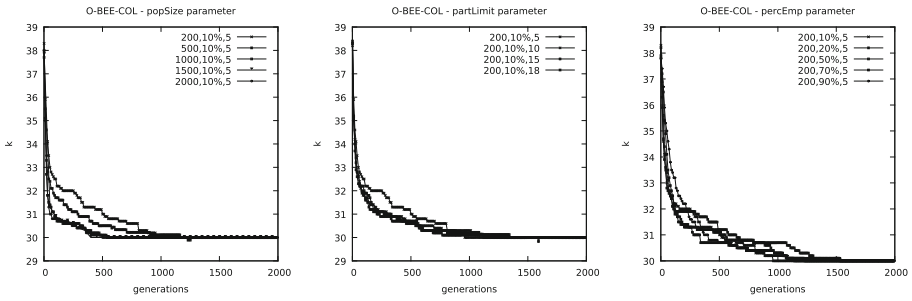

Fig. 1. Convergence behavior at varying the parameters: $popSize$, $partLimit$, and $percEmp$.

Table 1. Operating variants of O-BEE-COL, where $\hat{k}$ is the mean of the best colors found; k is the best coloring found in all runs; SR is the success rate, and AES is the average number of fitness function evaluations to the solution.

Variant	SmartSwap	Crossover	Temperature	$\hat{k}$	k	SR	AES
1	on	opt GPX	on	15	15	100 %	5,972,925
2	on	GPX	on	24	24	100 %	1,503,756
3	on	opt GPX	off	17.8	15	40 %	36,599,035
4	on	GPX	off	25	25	100 %	5
5	off	opt GPX	on	15.9	15	50 %	25,981,420
6	off	GPX	on	24	24	100 %	1,639,403
7	off	opt GPX	off	19.9	17	20 %	15,872,834
8	off	GPX	off	25	25	100 %	4

Several experiments have been conducted on the instance *le450_15c* in order to prove the effectiveness and utility of the features introduced in O-BEE-COL in terms of number of colors found; success rate; and average number of fitness function evaluations to the solution (AES). The aim of these experiments is to show that whatever the operators' combination chosen if we inhibit one of them, then its outcome will be negatively affected by this move. In Table 1 we show for any possible combination the average of the colors found ($\hat{k}$), best coloring found (k), SR and AES. In the next figures (Figs. 2, 3, and 4) we show a comparison of the several possible cases gradually disabling all the aforementioned features. The experiments have been averaged over 10 runs with different seeds. In the left plot of Fig. 2, we present the comparison of the convergence speed of O-BEE-COL with and without the SmartSwap operator (variants 1 and 5 of Table 1). It is possible to see how the first variant managed to reach the χ of the instance in every execution ($SR = 100\,\%$), whilst turning off the SmartSwap operator, O-BEE-COL is able to get the best coloring only in 50 % of the executions. Middle plot shows a version of the algorithm that does not use the temperature mechanism. If we disable also the SmartSwap operator (variant 7) the algorithm reaches an average of colors ($\hat{k}$) equal to 19.9, and the best result of 17 colors during all the executions; whilst using the mutation operator (variant 3) O-BEE-COL manages to reach the chromatic number in 40 % of the cases, with $\hat{k} = 17.8$. The right plot of the figure illustrates the contribution given by SmartSwap if instead we make use of the original GPX in O-BEE-COL (variants 2 and 6). Looking this plot is very clear, as both variants are not particularly efficient. The variant using the mutation operator ($2nd$ variant) manages to achieve an average of colors of 24, whilst the one that not using it ($6th$ variant) is not able to do better than 25. These three plots of Fig. 2 prove the usefulness of SmartSwap, and its benefits that affect positively on the overall performances, regardless on the operators combination enabled. The plots in Fig. 3 prove the real goodness of the optimized GPX proposed with respect to the original version [15] improving significantly

the performances of O-BEE-COL. The first plot on the left, presents a comparison of the speed convergences of O-BEE-COL using the proposed optimized crossover ($1st$ variant) versus the original one ($2nd$ variant). This comparison has been done on the fully enabled version of O-BEE-COL. The same comparison has been made also for the versions where the two other operators have been disabled ($7th$ and $8th$ variants), and it is shown in the second plot on the left of the figure. Looking both plots becomes very clear as the developed optimized version to equality of variant outperforms significantly the original one. The last two plots in Fig. 3 show respectively the analysis conducted when we turn off the temperature mechanism (penultimate plot), and SmartSwap mutation operator (last plot). The role played by the optimized GPX is clearly evident even in these plots. In particular, disabling the Temperature mechanism or SmartSwap operator, O-BEE-COL with the original version of GPX is not able to achieve a coloring with less than 25 colors; whilst with the designed GPX version O-BEE-COL performs better decreasing the colors number in average to $\hat{k} = 17.8$ (with only temperature enabled) and $\hat{k} = 15.9$ (with only mutation operator enabled). Finally in Fig. 4 we show the improvements produced, in using the temperature mechanism, which controls dynamically the values of some parameters. In the left plot of Fig. 4 is plotted the difference concerning of O-BEE-COL with, and without the temperature mechanism. In both variants the algorithm achieves successfully the chromatic number, $\chi = 15$ (see Table 1). However, whilst the fully enabled version is able to achieved always the chromatic number (variant 1), when this operator is turned off (variant 3) the algorithm manages to achieve the best coloring only in 40 % of the executions. In middle plot the two different versions of the algorithm make no use of the mutation operator. When the temperature mechanism is enabled ($5th$ variant) the algorithm finds the optimal coloring in one out of two cases ($\hat{k} = 15.9$), whilst the other combination ($7th$ variant) does not manage to do better than a 17-coloring ($\hat{k} = 19.9$). The right plot shows the behavior of the algorithm using the classical version of GPX ($2nd$ variant vs. $4th$). Despite the poor performances, O-BEE-COL obtains a slightly better result when using the temperature mechanism (variant 2). In the overall, inspecting all combinations in Table 1 is possible to claim that the Temperature mechanism developed is the one that gives a positive greater contribution with respect to SmartSwap mutation operator.

Time-To-Target plots [1] have been used for studying the running time of O-BEE-COL, comparing the empirical and theoretical distributions. They represent a classical tool for characterizing the running time of stochastic algorithms in order to solve a specific optimization problem. In particular, we have used a Perl program proposed in [2], which display the probability that an algorithm will find a solution as good as a target within a given running time. Through this program two kinds of plots are produced: $QQ-$plot with superimposed variability information, and superimposed empirical and theoretical distributions. This kind of analysis has been conducted on the instances *School1* and *DSJC250.1*, performing 200 independent runs for each instance. The produced plots are shown in Fig. 5 ($1st$ and $3rd$ plots for the first instance; $2nd$ and $4th$ plots for the last).

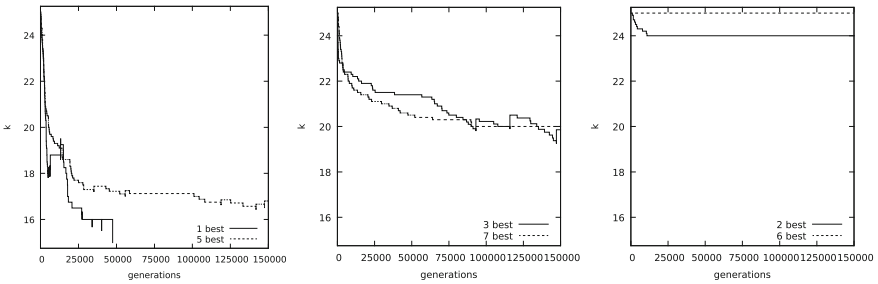

Fig. 2. Experimental analysis on the benefits provided by SmartSwap mutation operator.

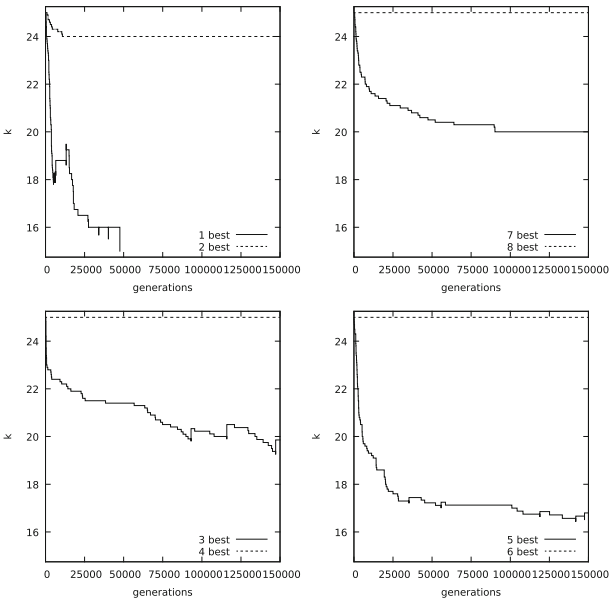

Fig. 3. Experimental analysis on the benefits provided by optimized GPX.

The plots show how for O-BEE-COL the empirical curve perfectly fits the theoretical one in both instances, except for very few worst cases (first two plots on the left). In the quantile-quantile plots, the O-BEE-COL results are in most of the cases equal to the theoretical ones, albeit a few less in *DSJC250.1* instance. This is explained because this last instance is more complex than the other one.

Experimental Comparisons. In order to evaluate the overall performances of O-BEE-COL, we have performed several experiments using the most known instances of the DIMACS benchmark [22]. The results in term of coloring found, SR obtained and AES needed are showed in Table 2. In this table we report for each instance its complexity characteristics; the chromatic number (χ); the

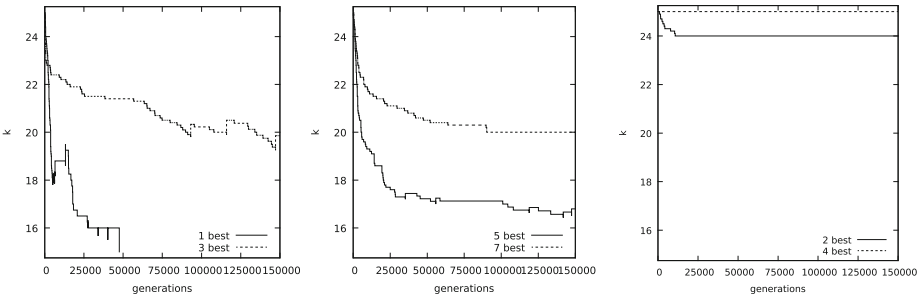

Fig. 4. Experimental analysis on the benefits provided by Temperature mechanism.

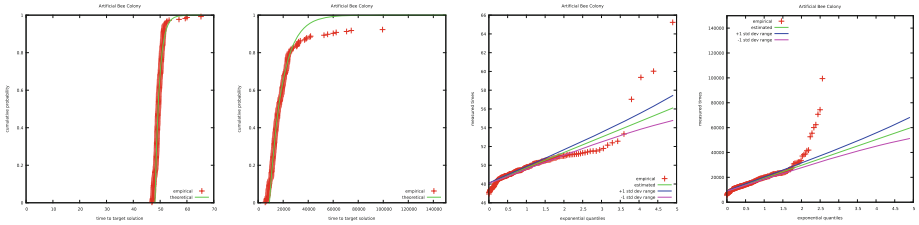

Fig. 5. Time to target plots for O-BEE-COL. The values have been obtained over 200 executions of the algorithm, respectively on the instance *School1* (1*st* and 3*rd*) and *DSJC250.1* (2*nd* and 4*th*).

best coloring known in literature (k^*); the best colors number found by O-BEE-COL (k), with SR and AES obtained. Each experiment has been performed on 10 independent runs. Inspecting such table, O-BEE-COL performs well on all instances *queen* and *school* finding the optimal coloring with a success rate of 100 %. On the class of the instances *DSJC*, instead, O-BEE-COL seems to have more difficulty in getting the best coloring known, except for *DSJC125.1*, where it manages to find the optimal solution in only 5 tests out of 10, and for *DSJC125.5* where only in one case out of 10 the algorithm finds a 17-coloring. On the instances *DSJC250.1* and *DSJC205.5*, instead, the algorithm finds as best solution a coloring with only one color in more; whilst for the instances *DSJC125.9* and *DSJC250.9* the difference with the best coloring known is of 2 and 3 colors respectively. The same performances are achieved also in *le450_15* family, where O-BEE-COL achieves the chromatic number in *le450_15c* and *le450_15d* instances, whilst for the other two its solution differs from the chromatic number only for one color in more. Finally, in *flat300_20* and *flat300_26* O-BEE-COL founds the chromatic number producing a success rate of 100 %, whilst in the last instance, *flat300_28*, it reaches a 31-coloring in 2 cases out of 10, where the chromatic number is however 28.

In Table 3 we present a comparison of O-BEE-COL with 6 different algorithms for the graph coloring problem, 4 of which nature-inspired: HPSO [30]; HCA [15]; GPB [20]; VNS [4]; VSS [21]; HANTCOL [13] (see the relative publications for major details). The best results are highlighted in boldface. Inspecting

Table 2. Experimental results on DIMACS benchmark instances [11,22].

Graph	$\mid V \mid$	$\mid E \mid$	χ	k^*	k	*SR*	*AES*
DSJC125.1	125	736	5	5	5	50 %	528, 715.6
DSJC125.5	125	3, 891	12	17	17	10 %	464, 633.0
DSJC125.9	125	6, 961	30	42	44	100 %	29, 817.4
DSJC250.1	250	3, 218	8	8	9	100 %	252, 538.7
DSJC250.5	250	15, 668	13	28	29	100 %	471, 823.0
DSJC250.9	250	27, 897	35	69	73	90 %	24, 403, 325.4
le450_15a	450	8, 168	15	15	16	100 %	17, 678, 139.9
le450_15b	450	8, 169	15	15	16	100 %	6, 188, 035.6
le450_15c	450	16, 680	15	15	15	100 %	5, 972, 925.6
le450_15d	450	16, 750	15	15	15	80 %	18, 630, 401.3
flat300_20	300	21, 375	20	20	20	100 %	4, 800
flat300_26	300	21, 633	26	26	26	100 %	$72.9K$
flat300_28	300	21, 695	28	28	31	20 %	$5.6M$
Queen5_5	25	320	5	5	5	100 %	1.9
Queen6_6	36	580	7	7	7	100 %	1, 741.66
Queen7_7	49	952	7	7	7	100 %	6, 636.84
Queen8_8	64	1, 456	9	9	9	100 %	22, 107.25
Queen8_12	96	2, 736	12	12	12	100 %	1, 212, 000.35
Queen9_9	81	1, 056	10	10	10	100 %	31, 243.28
School1.nsh	352	14, 612	14	14	14	100 %	1, 703.28
School1	385	19, 095	14	14	14	100 %	821.5

this table is possible to see how the performances of O-BEE-COL are competitive with the compared algorithms, achieving in all tested instances the best coloring except in *DSJC250.5*. Moreover, albeit on *flat300_28* the VSS algorithm has found the lower number of colors, O-BEE-COL achieves yet the same results as all others.

In Table 4, O-BEE-COL is compared with other 10 algorithms: IMMALG [11,28], MACOL [33], IGrAl [7], ACS [9], FCNS [29], IPM [14], ABAC [6], LAVCA, TPA and AMACOL [32]. The comparison has been performed with respect to the best coloring found. We have highlighted in boldface the colors found by O-BEE-COL, which are better or equal to the ones compared. Due a limit space, we refer the reader to each publication for more details on the algorithms. Also on these experiments is possible to see how O-BEE-COL is comparable with the state-of-the-art achieving the best coloring in 14 instances over 21. In the remaining instances nevertheless it isn't the worst.

Table 3. O-BEE-COL versus six different algorithms for graph coloring problem, with respect the best coloring found. The best results are highlighted in boldface.

Graph	O-BEE-COL	HPSO	HCA	GPB	VNS	VSS	HANTCOL
DSJC250.5	29	**28**	**28**	**28**	-	-	**28**
flat300_26	**26**	**26**	-	-	31	-	-
flat300_28	31	31	31	31	31	**29**	31
le450_15c	**15**	**15**	**15**	**15**	**15**	**15**	**15**
le450_15d	**15**	**15**	-	-	**15**	**15**	-

Table 4. O-BEE-COL versus state-of-the-art for graph coloring problem, with respect the best coloring found. The best or equal coloring obtained by O-BEE-COL is highlighted in boldface.

Graph	O-BEE-COL	IMMALG	MACOL	IGrAl	ACS	FCNS	IPM	ABAC	LAVCA	TPA	AMACOL
DSJC125.1	**5**	5	5	5	5	5	6	5	5	5	5
DSJC125.5	**17**	18	17	17	17	18	19	17	17	19	17
DSJC125.9	44	44	44	43	44	44	45	44	44	44	44
DSJC250.1	9	9	8	8	8	–	10	8	8	8	8
DSJC250.5	29	28	28	29	29	–	–	29	28	30	28
DSJC250.9	73	74	72	72	73	–	75	72	72	72	72
flat300_20_0	**20**	20	20	–	20	–	–	–	–	–	–
flat300_26_0	**26**	27	26	–	32	–	–	–	–	–	–
flat300_28_0	31	32	29	–	32	–	–	–	–	–	–
le450_15a	16	15	15	15	16	–	–	15	15	15	15
le450_15b	16	15	15	15	16	–	17	15	15	15	15
le450_15c	**15**	15	15	16	15	–	17	15	15	15	15
le450_15d	**15**	16	15	16	15	–	–	15	15	15	15
Queen5_5	**5**	5	–	5	–	–	–	5	–	–	–
Queen6_6	**7**	7	–	7	7	–	–	7	–	–	–
Queen7_7	**7**	7	–	7	7	–	–	7	–	–	–
Queen8_8	**9**	9	–	9	9	9	9	9	–	–	–
Queen8_12	**12**	12	–	12	12	–	–	12	–	–	–
Queen9_9	**10**	10	–	10	10	10	10	10	–	–	–
school1_nsh	**14**	15	14	14	14	–	–	14	–	–	–
School1	**14**	14	14	14	14	–	–	14	–	–	–

4 Conclusion

In this research paper we have developed a new Artificial Bee Colony heuristic, called O-BEE-COL, for the graph coloring problem. The novelties introduced in O-BEE-COL are basically: (1) SmartSwap mutation, which attempts to reduce the number of colorclasses, working primarily on the troublesome vertices; (2) optimized version of GPX, which works as multi-parents operator, forcing the transfer of the best colorclasses to the offsprings; and a (3) Temperature mechanism, which has the aim to dynamically handle some parameters.

Many experiments have been performed with the primary aim to evaluate the contribution, and benefits given by these new operators. Thus, all possible combinations of these three operators have been taken into account, and

have been tested; the obtained results prove us how inhibiting one of them the overall performances are negatively affected. In particular, we show, via figures, the significant improvements produced by the optimized version of GPX, and as the Temperature mechanism is the one that gives a greater positive contribution, respect to the SmartSwap operator. Via Time-To-Target plots are also analysed the running times of O-BEE-COL, comparing the empirical and theoretical curves. Finally, a comparison with the state-of-the-art has been conducted as well, in order to evaluate the robustness and efficiency of O-BEE-COL. Inspecting all results, and comparisons O-BEE-COL shows efficiency; robustness; and very competitive performances, achieving in the most of the instances the chromatic number, or the best coloring known.

References

1. Aiex, R.M., Resende, M.G.C., Ribeiro, C.C.: Probability distribution of solution time in GRASP: an experimental investigation. J. heuristics **8**, 343–373 (2002)
2. Aiex, R.M., Resende, M.G.C., Ribeiro, C.C.: TTTPLOTS: a perl program to create time-to-target plots. Optim. Lett. **1**, 355–366 (2007)
3. Akay, B., Karaboga, D.: A modified artificial bee colony algorithm for real-parameter optimization. Inf. Sci. **192**, 120–142 (2012)
4. Avanthay, C., Hertz, A., Zufferey, N.: A variable neighborhood search for graph coloring. Eur. J. Oper. Res. **151**(2), 379–388 (2003)
5. Bouziri, H., Mellouli, K., Talbi, E.-G.: The k-coloring fitness landscape. J. Comb. Optim. **21**(3), 306–329 (2011)
6. Bui, T.N., Nguyen, T.-V.H., Patel, C.M., Phan, K.-A.T.: An ant-based algorithm for coloring graphs. Discrete Appl. Math. **156**, 190–200 (2008)
7. Caramia, M., Dell'Olmo, P.: Coloring graphs by iterated local search traversing feasible and infeasible solutions. Discrete Appl. Math. **156**, 201–217 (2008)
8. Chow, F.C., Hennessy, J.L.: The priority-based coloring approach to register allocation. ACM Trans. Program. Lang. Syst. **12**, 501–536 (1990)
9. Consoli, P., Collerá, A., Pavone, M.: Swarm intelligence heuristics for graph coloring problem. In: IEEE Congress on Evolutionary Computation (CEC), vol. 1, pp. 1909–1916 (2013)
10. Costa, D., Hertz, A.: Ants can colour graphs. J. Oper. Res. Soc. **48**, 295–305 (1997)
11. Cutello, V., Nicosia, G., Pavone, M.: An immune algorithm with stochastic aging and kullback entropy for the chromatic number problem. J. Comb. Optim. **14**(1), 9–33 (2007)
12. de Werra, D.: An introduction to timetabling. Eur. J. Oper. Res. **19**, 151–162 (1985)
13. Dowsland, K.A., Thompson, J.M.: An improved ant colony optimisation heuristic for graph colouring. Discrete Appl. Math. **156**(3), 313–324 (2008)
14. Dukanovic, I., Rendl, F.: A semidefinite programming-based heuristic for graph coloring. Discrete Appl. Math. **156**, 180–189 (2008)
15. Galinier, P., Hao, J.: Hybrid evolutionary algorithms for graph coloring. J. Comb. Optim. **3**(4), 379–397 (1999)
16. Gamst, A.: Some lower bounds for a class of frequency assignment problems. IEEE Trans. Veh. Technol. **35**, 8–14 (1986)
17. Garey, M.R., Johnson, D.S.: Computers and Intractability: a Guide to the Theory of NP-completeness. Freeman, New York (1979)

18. Garey, M.R., Johnson, D.S., So, H.C.: An application of graph coloring to printed circuit testing. IEEE Trans. Circuits Syst. **CAS–23**, 591–599 (1976)
19. Glass, C.: Bag rationalization for a food manufacturer. J. Oper. Res. Soc. **53**, 544–551 (2002)
20. Glass, C.A., Prügel-Bennet, A.: Genetic algorithm for graph coloring: exploration of Galinier and hao's algorithm. J. Comb. Optim. **7**(3), 229–236 (2003)
21. Hertz, A., Plumettaz, M., Zufferey, N.: Variable space search for graph coloring. Discrete Appl. Math. **156**, 2551–2560 (2008)
22. Johnson, D.S., Trick, M.A.: Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge. American Mathematical Society, Providence (1996)
23. Karaboga, D., Basturk, B.: A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Global Optim. **39**(3), 459–471 (2007)
24. Karaboga, D., Basturk, B.: On the performance of Artificial Bee Colony (ABC) algorithm. Appl. Soft Comput. **8**, 687–697 (2008)
25. Krasnogor, N., Smith, J.E.: A tutorial for competent memetic algorithms: model, taxonomy and design issues. IEEE Trans. Evol. Comput. **9**(5), 474–488 (2005)
26. Leighton, F.T.: A graph coloring algorithm for large scheduling problems. J. Res. Natl. Bur. Stan. **84**, 489–505 (1979)
27. Oner, A., Ozcan, S., Dengi, D.: Optimization of university course scheduling problem with a hybrid artificial bee colony algorithm. In: IEEE Congress on Evolutionary Computation, pp. 339–346 (2011)
28. Pavone, M., Narzisi, G., Nicosia, G.: Clonal selection - an immunological algorithm for global optimization over continuous spaces. J. Global Optim. **53**(4), 769–808 (2012)
29. Prestwich, S.: Generalised graph colouring by a hybrid of local search and constraint programming. Discrete Appl. Math. **156**, 148–158 (2008)
30. Qin, J., Yin, Y., Ban, X.-J.: Hybrid discrete particle swarm algorithm for graph coloring problem. J. Comput. **6**(6), 1175–1182 (2011)
31. Rodriguez, F.J., García-Martínez, C., Blum, C., Lozano, M.: An artificial bee colony algorithm for the unrelated parallel machines scheduling problem. In: Coello, C.A.C., Cutello, V., Deb, K., Forrest, S., Nicosia, G., Pavone, M. (eds.) PPSN 2012, Part II. LNCS, vol. 7492, pp. 143–152. Springer, Heidelberg (2012)
32. Torkestani, J.A., Meybodi, M.R.: A new vertex coloring algorithm based on variable action-set learning automata. Comput. Inform. **29**(1), 447–466 (2010)
33. Zhipeng, L., Hao, J.-K.: A memetic algorithm for graph coloring. Eur. J. Oper. Res. **203**(1), 241–250 (2010)