RAIRO-Oper. Res. 59 (2025) 3683–3702
https://doi.org/10.1051/ro/2025142

RAIRO Operations Research
www.rairo-ro.org

AN IMPROVED BI-CRITERIA EVOLUTIONARY APPROACH FOR THE SUM COLORING OF GRAPHS

Olfa Harrabi[1,*], Jouhaina Chaouachi Siala[2] and Hend Bouziri[3]

Abstract. The sum coloring problem emerged as an interesting variant of the classical graph coloring problem since it is able to model several life applications. Interestingly, the studied problem could perform a straightforward implementation of many important complex problems such as timetabling, scheduling and resource allocation. In this work, we propose an original bi-criteria optimization method that tries to separately consider two objective functions when solving the sum coloring problem. The computational performance of our approach, based on a set of benchmark instances, shows a significant improvement of the chromatic sum bounds.

Received March 28, 2025. Accepted October 15, 2025.

1. Introduction

The Minimum Sum Coloring problem (MSCP) is a variant of the well-known Graph Coloring Problem (GCP) introduced by Kubicka *et al.* [18]. Formally, the MSCP could be defined using a simple undirected graph $G = (V, E)$ where V is the set of n vertices and E the set of m edges. Let us consider a set of colors $\{1, \ldots, k\}$. A *proper k-coloring* of the graph G is an application $c : V \to \{1, \ldots, k\}$ such as $c(x) \neq c(y)$, $\forall (x, y) \in E$. In the opposite case (*e.g.* $c(x) = c(y)$), the vertices x and y are termed *conflicting vertices*, the edge E is a *conflicting edge* and the obtained coloring is *non proper*. From another perspective, a k-coloring could also be considered as a partition of V into k disjoint independent sets or *color classes* $V = \{V_1, \ldots, V_k\}$. A minimum integer k for which a proper coloring of a graph exists is called the *chromatic number* and it is denoted by $\chi(G)$. Looking for the χ-coloring of the graph is the *graph coloring problem*. However, finding the k-coloring is known as *k-coloring problem*.

The minimum sum coloring problem is about finding a proper coloring of G such that the following sum of colors is minimized:

$$f(c) = \sum_{i=1}^{n} c(v_i) \tag{1}$$

Keywords. Minimum sum coloring problem, bi-criteria evolutionary method, perturbed tabu search.

[1] Higher Institute of Management and Computer Science ISIGK, Kairouan University, Kairouan, Tunisia.
[2] Institute of Advanced Business Studies of Carthage, Carthage University, IHEC Carthage Presidency-2017, Tunis, Tunisia.
[3] LARODEC laboratory, Higher School of Economic and Commercial Sciences of Tunis, Tunis University, Abou Zakaria Al Hafsi Street, Tunis, Tunisia.
*Corresponding author: olfaharrabikhlif@gmail.com

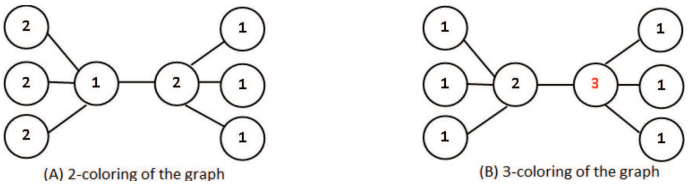

FIGURE 1. Relation between graph coloring problem and minimum sum coloring problem [26].

where $c(v_i)$ is the color of the vertex v_i. This minimal sum is called *chromatic sum* and denoted by $\sum(G)$. The minimal number of used colors in obtaining the chromatic sum is called *strength* $s(G)$ of the graph. For a better explanation, we provide the following Figure 1. The chromatic number of the representative graph is $\chi(G) = 2$ (left figure). With the given 2-coloring, we achieve a sub-optimal sum of 12. However, this graph requires 3 colors to achieve a lower sum equals to 11. This shows that minimizing the number of colors allocated to nodes does not necessarily mean minimizing the sum of colors.

1.1. Related works

Theoretically, the sum coloring problem and its variants are classified as NP-Hard except for some cases as it was detailed in [4,18]. Therefore, several approximations were proposed for the interval and bipartite classes of graphs [1, 17, 19]. Given the hardness of the MSCP, much interest has been adressed to propose heuristic and metaheuristic algorithms for computing, both upper or lower bounds of the underlying problem. These methods belong to three classes: greedy algorithms, local search heuristics and evolutionary algorithms. Li *et al.* [20] elaborated two greedy algorithms namely Minimum Degree SATuration (MDSAT) and Minimum Recursive Largest First (MRLF). The latters are basically related to Degree SATURation method (DSATUR) and Recursive Largest First (RLF) heuristics [7]. Experimental results showed that MDSAT and MRLF provide the first promising results for the MSCP and even overperform the classical DSATUR and RLF for some GCP instances. Wu *et al.* [29] elaborated also a greedy heuristic based on independent set extraction namely EXtraction of independent Set for COLoring (EXSCOL). This proposed method performs well on large graphs with more than 500 vertices. Moreover, we note the work of Moukrim *et al.* [23] who investigate lower bounds for the studied problem. For this purpose, authors used a greedy heuristic based on clique decomposition namely RMDS(n). The proposed approach improves some best known results and delivers optimality for some instances.

Concerning local search heuristics, a Tabucol coloring algorithm was adapted to MSCP [6]. The latter shows a good behavior on seven DIMACS instances compared to MDSAT and MRLF. Also, Helmar *et al.* [12] proposed a local search heuristic namely Multi-neighborhood Search with Local Search method (MDS(5)+LS). The latter is based on variable neighborhood search and iterated local search. MDS(5)+LS outperforms all recent methods and improves the best known results. In [3], Benlic *et al.* developed a Breakout Local Search algorithm (BLS). The proposed algorithm relies on the use of a local search method and an adaptive perturbation strategies. BLS improved upper bounds for 4 instances out of 27 tested graphs.

Besides, evolutionary algorithms were successfully proposed to tackle the minimum sum coloring problem. Firstly, a Parallel Genetic Algorithm (PGA) was proposed by Kokosiński and Kwarciany [16]. The PGA algorithm uses assignment and partition crossovers, first-fit mutation and proportional selection. PGA reports upper bounds on 16 small DIMACS graphs. Then, Moukrim *et al.* [24] introduced a Memetic Algorithm for the Minimum Sum Coloring problem (MA-MSC) using a two-parent crossover, hill-climber and destroy/repair procedure. The approach improves upper and lower bounds on tested instances. An other Memetic Algorithm (MASC) was elaborated by Jin *et al.* based on tabu search combined with and a multi-parent crossover [14]. MASC reported 15 new upper bounds for the MSCP. Recently, a Hybrid Evolutionary Search (HESA) was proposed by Yan *et al.* in [13]. HESA includes two crossover operators: diversification-guided crossover and a grouping-guided crossover. Also, the method employs an iterated double-phase tabu search procedure to ensure

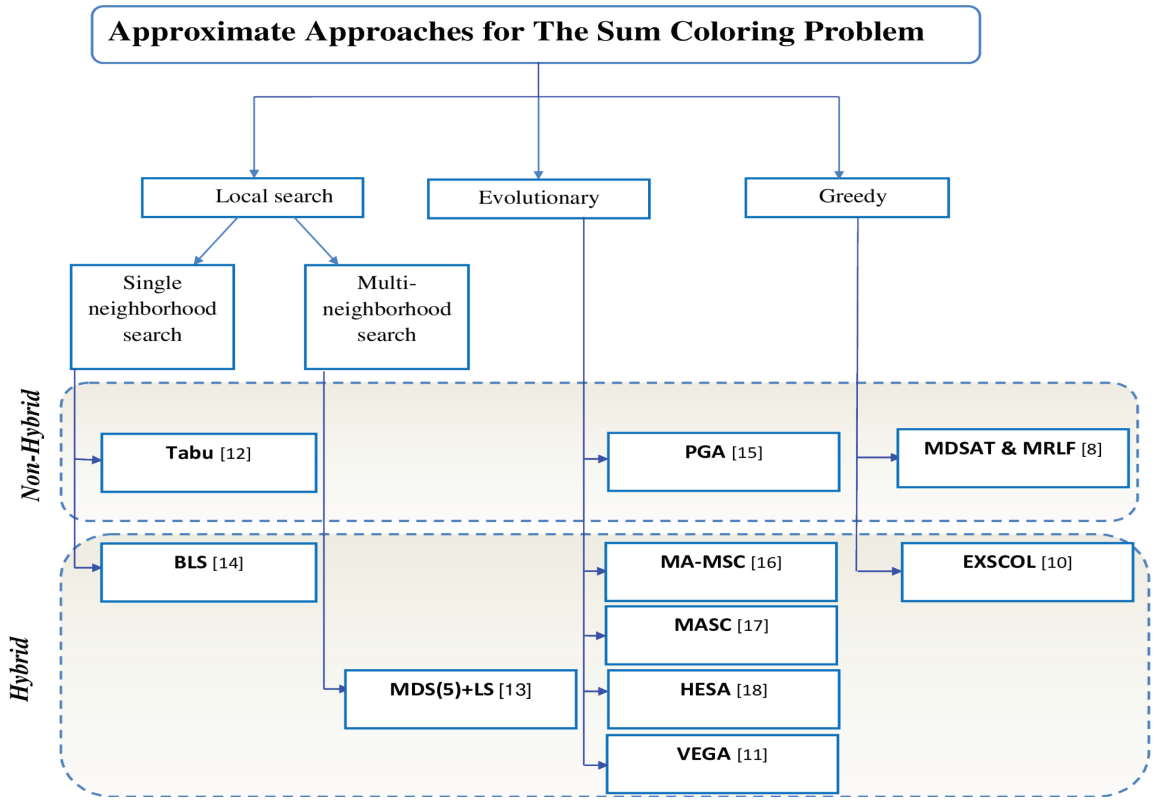

FIGURE 2. Cartography of the metaheuristic techniques to the sum coloring problem.

local optimization. Computational results show that HESA improves the best-known results for 51 instances out of 94. Moreover, authors in [11] have proposed a new bi-objective genetic algorithm VEGA and a heuristic approach to tune its parameter and therefore to guarantee its computational robustness.

For more details about the aforementioned works, one can refer to the more recent survey proposed in [15]. The literature review of the different approximate methods proposed for our problem is showed in Figure 2.

1.2. Scope and contributions

The focus of this paper is on improving existing upper bounds on the minimum sum coloring problem. To achieve this, we refer to the original graph coloring problem in which operating on non feasible solutions turned out to be extremely advantageous. That means, the search process is performed using a given k colors, until a feasible coloring is reached. This process is repeated by decreasing the number of colors k. This idea is used in almost all proposed metaheuristic and local search efficient techniques specially for hard instances [8,9]. Indeed, our work enables the search procedure for minimum coloring to use non feasible solutions. In fact, operating with non feasible solutions leads to promising results even with a very simple schema of resolution for the sum coloring problem as it was stated by the use of the tabu search algorithm in [6].

Therefore, we try to minimize the number of conflicts in the visited colorings while minimizing the sum of used colors. This leads us to consider a bi-criteria optimization method that minimizes the number of conflicts by looking for a proper k-coloring (number of conflicts is equal to zero) and simultaneously tries to minimize the sum of colors.

TABLE 1. Settings of Bi-EA parameters.

Parameter	Description	Value
p_1	Maximum number of Bi-EA generations	100
p_2	Maximum number of PTS iterations	10^4
p_3	Maximum number of non-improving moves for perturbation	5000
p_4	Maximum number of moves for TS	10^3
r_1	Crossover rate	0.9
r_2	Mutation rate	0.4

Our bi-criteria approach belongs to a category of the multi-objective optimization methods called criterion-based [21]. They seek to optimize each criterion separately and tend to generate *extreme solutions* that optimize each objective independently. This precisely corresponds to our expectations since we seek proper colorings with no conflicts and minimal sum of colors. Among these methods, we choose the VEGA (Vector Evaluated Genetic Algorithm) as a general schema to be adapted when solving the minimum sum coloring problem.

As consequence, we propose an evolutionary Bi-criteria (Bi-EA) method. This choice is motivated by the fact that evolutionary methods are known to be efficient for coloring problems. Also it can be hybridized easily with other local search techniques. Therefore the proposed Bi-EA:

- Explores a search space of k-colorings, that can be proper or not, and tries to minimize both the sum of colors and the number of conflicts in the graph.
- Uses a hybridization that couples evolutionary schema with tabu search as a potential search tool. This combination has proven to be effective in treating both the graph coloring problem and the sum coloring problem, as it was stated above.
- Applies adequate genetic operators, mainly the crossover which has to handle the *specificity* of the problem. Hence, we use a dedicated crossover operator called SPX (Sum Partition Crossover). Its efficiency is experimentally improved to deal with the sum coloring problem in [5].

The remainder of this paper is organized as follows. Section 2 presents the proposed solution approach. Section 3 presents the obtained simulation results and comparisons with some algorithms from the literature. Finally, Section 4 concludes the paper.

2. A Bi-criteria Evolutionary Approach for the minimum sum coloring problem: (Bi-EA)

Hybrid evolutionary algorithms (EA) are among the most powerful paradigms for solving NP-hard combinatorial optimization problems. In particular, they have been successfully applied to the tightly related graph coloring problem [8, 22, 25].

Hence, to optimize both objective functions, we use an EA schema based on the VEGA (Vector Evaluated Genetic Algorithm) [27]. Basically, the VEGA divides randomly the total population into a set of sub-populations equal to the number of objectives. Then, each sub-population is evaluated based on the given different objective functions. Subsequently, proportionate selection operator is used to create the mating pool.

We have adopted the same paradigm of the VEGA algorithm but with different features to ensure a high search performance. To this end, a new strategy of population division is introduced and a guided selection operator is investigated. The pseudo code of the Bi-EA method is detailed in Algorithm 1.

Bi-EA starts with an initial population randomly generated (line 5). Then, until a stopping criterion is satisfied, the proposed approach performs p_1 generations (see Tab. 1) to generate as good a coloring as possible, *i.e.* which is proper and having the minimum sum coloring. At each generation, Bi-EA divides the initial population into two equal-sized sub-populations (line 8). Contrary to the classic VEGA algorithm, our approach

Algorithm 1. Pseudo-code of the Bi-EA for MSCP.

1: **Input**: A graph G, population of size N, k the initial number of colors
2: **Output**: A proper coloring c^* with a minimal sum of colors and its sum of colors f^*
3: **Begin**
4: **While** Stopping condition is not met **do**
5: Random population initialization (P, N);
6: $f^* \longleftarrow \min_{c \in P}$ f(c)
7: **For** $i \in \{1, \cdots, p_1\}$ **do**
8: (Pop_1, Pop_2) $\longleftarrow$ **Divide** P into two sub-populations of sizes $\frac{N}{2}$;
9: $\text{Pop}_1' \longleftarrow$ **Greedy process** Pop_1;
10: $P_{\text{new}} \longleftarrow$ Combine (Pop_1', Pop_2);
11: (c_1, c_2) $\longleftarrow$ Selection (P_{new});
12: $o_1 \longleftarrow$ SPX (c_1, c_2);
13: $o_2 \longleftarrow$ PTS (o_1);
14: **If** $f(o_2) < f^*$ **then**
15: $f^* \longleftarrow f(o_2)$; $c^* \longleftarrow o_2$;
16: **end if**
17: Population Updating (P, o_2);
18: **If** $\nexists$ (proper colorings) **then** $k \longleftarrow k + 1$;
19: **End for**
20: Return f^*, c^*
21: **End**

performs the division after evaluating the population based on the objective function defined by equation (1). The reason behind this idea is identifying the sub-populations having the minimal sum of colors which are further enhanced using the greedy process (line 9). After obtaining a new combined population (line 10), two solutions are selected using a guided tournament selection operator (line 11). A crossover operator called SPX is then used to obtain an offspring solution (line 12). Next, a Perturbed Tabu Search procedure (PTS) is employed to improve the obtained offspring solution (line 13). Finally, according to a population updating rule, we decide whether the improved offspring will be inserted into the population or not (line 17). From the obtained population, we check existence of proper colorings. If no proper colorings were found, the number of used colors k is incremented. In the next, we present more details about these basic components.

2.1. Search space and evaluation function

The search space explored by Bi-EA is the set of both proper and non proper k-colorings. A k-coloring is proper when all adjacent vertices $\{x, y\} \in E$ are colored differently. Otherwise, the k-coloring is non proper. The objective value of a given coloring c is given by the function f defined by $f(c) = \sum_{i=1}^{n} c(v_i)$, where n is the number of vertices in the graph.

2.2. The division strategy

The proposed Bi-EA approach is a criteria based, which means that each objective is considered separately. In the Bi-EA procedure, our aim is firstly to minimize the sum of colors. Thus colorings having best sums are gathered in the first sub-population Pop_1 and the others are in the second sub-population Pop_2.

2.3. Greedy process

This component of the BI-EA method is a greedy heuristic which efficiently reconstructs individuals of the population in an iterative way. The considered process is described in Algorithm 2.

Individuals are restructed using their actual color classes. For each individual c_i of Pop, we sort color classes $V_j^i (j = 1, \cdots, k)$ in a decreasing order of their cardinalities $|V_j^i|$. Then, we assign, in an iterative way, the smallest color to the largest color class.

Algorithm 2. Pseudo-code of the Greedy process.

Input: Population Pop $= \{c_1, \cdots, c_N\}$
Output: A restructed population Pop$' = \{c'_1, \cdots, c'_N\}$
Begin
Pop$' \longleftarrow \emptyset$
For $i \in \{1, \cdots, N\}$ **do**
  Construct $\{V_1^i, \cdots, V_k^i\}$ from c_i $\{k$ *color classes of an individual* $c_i\}$
  Sort color classes V_i in a decreasing order of cardinality $|V_i|$;
  $c'_i \longleftarrow \{V_1^{i'}, \cdots, V_k^{i'}\}$ {*The restructed individual*}
  For $j \in \{1, \cdots, k\}$ **do**
    Assign the color j to $V_j^{i'}$ class of the coloring c'_i;
  End for
  Pop$' \longleftarrow$ Pop$' \sqcup \{c'_i\}$
End for
Return Pop$'$
End

2.4. Selection operator

The selection procedure uses the binary tournament selection. First, we randomly select p individuals from the population (p is an even integer). Then, we choose the best two individuals based on a score function. Hence, each coloring c_i in the population is evaluated by F_s where:

$$F_s(c_i) = f(c_i) + e^{|E_{c_i}|/|V|} \tag{2}$$

with E_{c_i} is the set of conflicting edges induced by the coloring c_i and $|V|$ is the number of the graph vertices. Accordingly, this introduced score function penalizes as most as possible non proper solution.

2.5. The crossover operator: SPX

Among basic operators that considerably influence the quality of genetic algorithms is the crossover. For our proposed Bi-EA, we use a Sum Partition Crossover (SPX). SPX considers a coloring as a partition of vertices into color classes.

Given two parent configurations: $P_1 = \{V_1^1, \cdots, V_k^1\}$ and $P_2 = \{V_1^2, \cdots, V_k^2\}$, the SPX operator successively constructs the partial color classes to build an offspring solution in a greedy way as it is explained in Algorithm 3.

Based on Algorithm 3, the SPX operator builds iteratively k classes $\{V_1, \cdots, V_k\}$ of the offspring in the following way. Firstly, the operator considers the parent P_1 or parent P_2 according to counter i. Then, it transmits to the offspring the class of colors having the maximum number of vertices. The transmitted vertices are subsequently removed for the parents. After proceeding k steps, the unaffected vertices are then assigned to a color class randomly chosen. Afterward, SPX reconstructs the obtained offspring o_1 by assigning, in an iterated way, the smallest available color k to the color class V_i with a maximum cardinality. In Figure 3, we show an illustrative example with 3 color classes and 10 vertices represented by $\{A, B, \cdots, J\}$. At step 1, the class $V_1^1 = \{A, B, C, D\}$ in parent P_1 is selected to be the first class V_1 of the offspring. Then, the assigned vertices $\{A, B, C, D\}$ are respectively removed from both P_1 and P_2. Similarly, we continue building classes V_2 and V_3 from respectively parents P_2 and P_1. After constructing all the offspring's color classes, we sort the obtained color classes in an increasing order of their cardinalities. Finally, in order to minimize the total sum of colors, we assign iteratively, the smallest color to the most numerous color class. Thus, the color class $V_1 = \{A, B, C, D\}$ receive the color 1, $V_2 = \{G, H, I, J\}$ the color 2 and $V_3 = \{E, F\}$ the color 3.

2.6. A Perturbed Tabu Search procedure: PTS

The purpose of a local optimization in the Bi-EA algorithm is to improve the quality of a solution returned by the SPX crossover operator. Since the proposed SPX operator may result in both proper and non proper

Algorithm 3. Pseudo-code of the SPX combination operator.

Input: Parents $P_1 = \{V_1^1, \cdots, V_k^1\}$ and $P_2 = \{V_1^2, \cdots, V_k^2\}$
Output: Offspring $o_1 = \{V_1, \cdots, V_k\}$
Begin
For $i \in \{1, \cdots, k\}$ **do**
  If i is odd, **then** $X := 1$, **else** $X := 2$;
  Choose j such that V_j^X has a maximum cardinality;
  $o_1 \longleftarrow \{V_i := V_j^X\}$;
  $P_1 \longleftarrow P_1 - \{V_i\}$;
  $P_2 \longleftarrow P_2 - \{V_i\}$;
End for
Assign randomly the vertices of $V - \{V_1 \sqcup \cdots \sqcup V_k\}$;
Sort V_i in a decreasing order of cardinality $|V_i|$;
$o_1 \longleftarrow \{V_1, \cdots, V_k\}$: the resulting restructed offspring;
For $i \in \{1, \cdots, k\}$ **do**
  Assign the color i to the V_i class of o_1;
End for
Return o_1
End

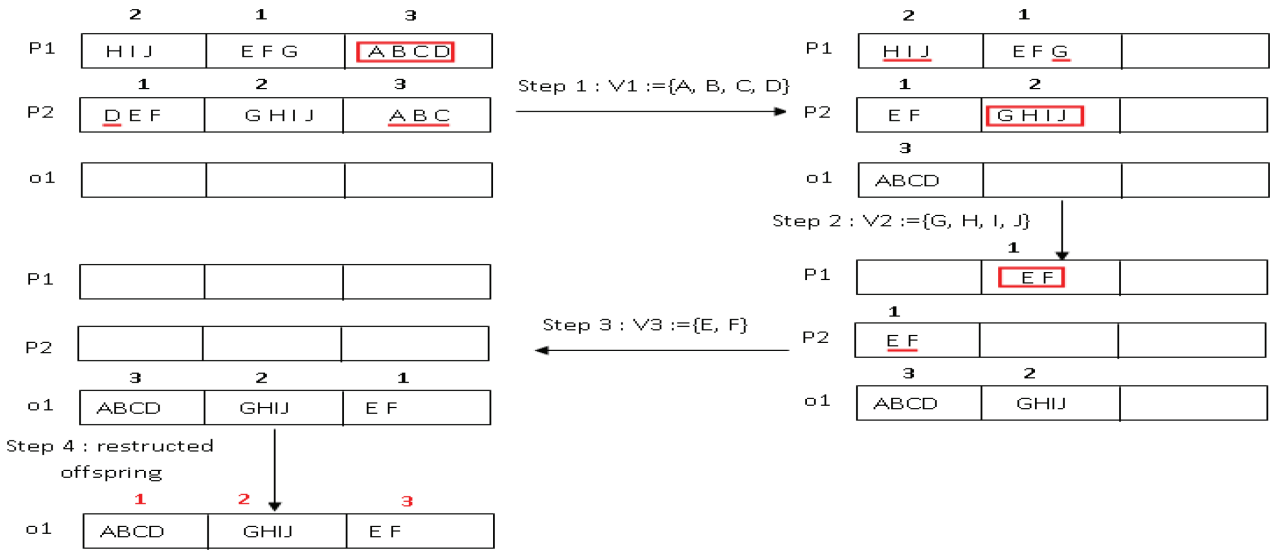

FIGURE 3. An illustrative example of the SPX crossover operator.

colorings, we elaborate a perturbed tabu search procedure to repair solutions while minimizing its sum of colors. The general scheme of PTS for MSCP is summarized in Algorithm 4.

Based on Algorithm 4, PTS starts checking the feasibility of a solution o_1 returned by the SPX operator. If the coloring is proper, then, the greedy process is applied to minimize its sum of colors. Otherwise, TS is called to repair color conflicts in order to obtain proper solutions which is further enhanced using the greedy process. This intensification phase stops when no improved local optimum is found after p_3 consecutive iterations (see Tab. 1). To escape from local optima, a perturbation mechanism is used. The stopping condition of PTS procedure is reaching p_2 maximum number of iterations (see Tab. 1). We detail in what follows the tabu search procedure and the perturbation mechanism.

Algorithm 4. Pseudo-code of the Perturbed Tabu Search for MSCP.

```
1: Input: Graph G, a k-coloring o1
2: Output: The best found coloring o2
3: Begin
4: o* ⟵ o1;
5: While (a stop condition is not met) do
6:   Repeat
7:     If (o1 is not proper) then
8:       o1 ⟵ TS(N, o1);
9:         If f(o1) < f(o*) then
10:          o* ⟵ o1;
11:        End if
12:    Else
13:      o1 ⟵ Greedy process (o1, k);
14:      If f(o1) < f(o*) then
15:        o* ⟵ o1;
16:      End if
17:    End if
18:  Until o* is not improved for p3 iterations
19:  o2 ⟵ Perturbation(o*);
20: Return o2
21: End
```

2.6.1. The tabu search procedure

The proposed procedure adapted the general framework of the tabu search algorithm [10]. Basically, TS explores the neighborhood of a given solution using the following characteristics:

- **Neighborhood:** The neighborhood structure employed by TS is presented by one-conflict operator CONF(v, V_i, V_j). The latter operates with conflicting vertices to repair non proper solutions. For this purpose, the operator chooses one non-tabu conflicting vertex v and changes its actual color class V_i by the smallest possible one V_j ($i \neq j$). Notice that, in order to get proper colorings, TS may use additional colors. The procedure stops after executing p_4 iterations (see Tab. 1).
- **Tabu list and aspiration criterion:** PTS uses a tabu list as a short term memory to avoid cycling [10]. More precisely, if a vertex $v \in V_i$ is displaced to a different color class V_j then v couldn't belong to V_i during TT iterations. The tabu tenure TT is specified using a random number from $\{2, \cdots, k-1\}$. Moreover, a forbidden move is accepted if the resulting neighboring solution is the best found one. This is what we call an aspiration in a tabu search algorithm.

2.6.2. Perturbation mechanism

To ensure diversification, our proposed PTS procedure applies a perturbation mechanism that is able to escape from local optima. The used mechanism consists in modifying the best solution o^* returned by the intensification search (line 19, Algorithm 4) in the following way. First, from a coloring o^*, we choose randomly a vertex $v \in V_i$ and a color class $V_j \neq V_i$. Then, we determine all adjacent vertices u' of v, $u' \in V_j$. Finally, we displace v to V_j and u' to V_i. This mechanism can result both proper and non proper colorings, which enables the search process to explore new feasible and unfeasible regions.

2.7. A stochastic population updating procedure

The population updating strategy plays a crucial role in our proposed hybrid algorithm since it can avoid premature convergence. In fact, our updating strategy controls the quality of the population. Basically, a solution quality is measured using the score function in equation (2). Whereas, proper colorings are evaluated using the sum coloring function f defined in equation (1). A proper coloring c_1 is better than c_2 if $f(c_1) < f(c_2)$. We

update the population as follows. We compare between o_2 and the best parent PA_{best} of o_2 in term of quality. In this context, we are faced by two scenarios:

- **Case 1**: The offspring o_2 is better than PA_{best}.
  In this case, we identify the worst individual P_{worst} in the population to be replaced by o_2. The rationale behind this step is to prevent a bad quality offspring from participating in the population updating mechanism. Also, we discard the worst individual to ensure a good quality of the population.
- **Case 2**: The offspring o_2 is worse than PA_{best}.
  Due to our selection strategy, the chosen parents PA_1 and PA_2 for the combination step are the best in term of quality. Obviously P_{best} is the closest individual to PA_1 and PA_2. To maintain a good quality of population, o_2 is compared to P_{best}. Subsequently, o_2 replaces P_{best} according to some probability p (randomly generated between $[0, 1]$).

2.8. Discussions

By observing the Bi-EA components and compared them to the most recent hybrid algorithms MASC, MA-MSC and HESA for minimum sum coloring problem, we can address the following remarks:

- Bi-EA starts with non proper initial solutions however, all recent hybrid algorithms employ different complex algorithms to ensure feasibility of colorings. Consequently, from the complexity perspective, our proposed Bi-EA approach marks its simplicity compared to MASC, MA-MSC and HESA. Moreover, a such starting strategy allows the diversification and therefore, prevents from premature convergence.
- Operating with non proper colorings could be beneficial for the search process. In fact, it provides a more "freedom" for better exploration of the search space of the problem.
- To ensure selection, Bi-EA performs a guided selection operator based on the well-known tournament selection operator. Moreover, our approach considers the two studied objective functions by using a score function for the selection. However, for MASC and HESA approach the selection was random. Concerning MA-MSC, it uses the same operator as Bi-EA, but, it only considers the function presented by equation (1). as it allows only feasible solutions.
- To recombine solutions, Bi-EA uses the crossover operator SPX which can generate both proper and non proper solutions. However, the recent HESA algorithm uses two different crossover operators.

3. Experimental study

3.1. Benchmark instances

Efficiency of BI-EA approach was tested using a set of benchmarks composed of 42 well-known graphs from COLOR 2002–2004 competitions[1] and DIMACS challenge[2]. These instances refer to various typologies and densities, which can be classified into the following types:

- Twelve classical random graphs (DSJC$n.d$, $n \in \{125; 250; 500; 1000\}$, $d \in \{1; 5; 9\}$).
- Six flat graphs (flat300-a- 0 with $a \in \{20; 26; 28\}$ and flat1000-b- 0 with with $b \in \{50; 60; 76\}$).
- Eight Leighton graphs (le450-xa, le450-xb, le450-xc, le450-xd, with $x \in \{5; 15; 25\}$).
- Eleven graphs from the Donald Knuth's Stanford GraphBase (miles$n$ with $n \in \{250; 500\}$), anna, david, huck, jean, homer, games120 and queen$a.a$, $a \in \{5; 6; 7; 8\}$.
- Five graphs based on the Mycielski transformation (myciel$a$, $a \in \{3; 4; 5; 6; 7\}$).

[1] http://mat.gsia.cmu.edu/COLOR02.
[2] http://dimacs.rutgers.edu/Challenges/.

3.2. Experimental protocol

Bi-EA algorithm was programmed in C++. The experiments were performed using a personal laptop with an Intel i5 CPU 2.53 GHz and 4 GB of RAM. The populations's size was set to 10 individuals (like many other memetic algorithms [14,24]). The stopping condition is a timeout limit set to be four CPU hours except for some hard graphs for which additional time limit is allowed. During experimentation, we run our algorithm 30 times on each instance with different seeds. In order to detect the appropriate value of each parameter, we varied the values within a range and compared the obtained solution in terms of both solution quality and computation time. Empirical experiments led to parameters setting shown in Table 1.

3.3. Used metric

Because of the stochastic feature of the proposed method, we have to evaluate its performance according to its improvements on the returned upper bounds. In addition, we have to measure the deviations of these measures to assess more precisely the effectiveness of our algorithm and also to compare it to related best known techniques. Indeed, we propose the RPD metric calculated as follows:

– Relative Percentage Deviation $\text{RPD} = (\frac{\text{UB} - f^b_{\text{LB}})}{f^b_{\text{LB}}}) * 100$

where f^b_{LB} is the current best lower bound [13].

3.4. Analysis of Bi-EA algorithm

Our proposed Bi-EA approach relies on the use of two important operators mainly the sum partition crossover (SPX) and the perturbed tabu search procedure (PTS). In this section, we exhibit influence of these operators. For this purpose, we only choose 16 DIMACS instances known to be hard for coloring.

3.4.1. Influence of the Sum Partition Crossover: SPX

A close analysis is performed to verify efficiency of the SPX operator. Indeed, we compare between Bi-EA and another version of our algorithm Bi-EA$_{\text{GPX}}$ employing GPX as a crossover operator. For more detail about the GPX operator, one may refer to [8].

The algorithms are run 30 times using the same parameters defined in Table 1. Results are displayed in Table 2. For each generated algorithm, we present the best upper bound f^*_{UB} and the relative percentage of RPD to upper bounds.

From Table 2, it is clear that Bi-EA outperforms remarkably Bi-EA$_{\text{GPX}}$. Indeed, the solutions qualities of Bi-EA are the best since it reports an RPD of 2.255% compared to 2.266% for Bi-EA$_{\text{GPX}}$. We should also note that for 14 instances Bi-EA is strictly better than Bi-EA$_{\text{GPX}}$. Only for flat300.20-0 and flat300.26-0, the two algorithms match the best known results.

3.4.2. Influence of Perturbed Tabu Search procedure: PTS

Local optimization is an important stage that mainly influences hybrid evolutionary algorithms. For our proposed approach, it is relevant to evaluate the impact of its perturbed tabu search procedure (PTS). For this purpose, we compare between Bi-EA and its counterpart Bi-EA$_{\text{nPTS}}$ schema without any local optimization operator. These two versions are run 30 times with the same parameters defined in Table 1. The results of our comparisons are summarized in Table 3 providing the same information as in Table 2.

Looking at Table 3, one observes that Bi-EA algorithm is more performant than Bi-EA$_{\text{nPTS}}$. This is clearly illustrated by the average RPD which is equal to 2.255% compared to 2.275% relative to Bi-EA$_{\text{nPTS}}$. Only for 2 cases out 16, the two versions were able to match the best known upper bounds.

TABLE 2. Comparative results of Bi-EA and Bi-EA$_{\mathrm{GPX}}$ algorithms on the set of 16 DIMACS instances.

Instances	f^b_{UB}	Bi-EA		Bi-EA$_{\mathrm{GPX}}$	
		f^*_{UB}	RPD	f^*_{UB}	RPD
DSJC125.5	1012	**1000**	0.821	1015	0.848
DSJC125.9	2503	**2506**	0.481	2512	0.485
DSJC250.5	3210	**3218**	1.500	3246	1.522
DSJC250.9	8277	**8277**	0.919	8289	0.922
DSJC500.1	2836	**2840**	1.272	2898	1.318
DSJC500.5	10 886	**10 905**	2.730	10 910	2.732
DSJC500.9	29 862	**29 910**	1.706	29 952	1.709
DSJC1000.1	8991	**9000**	2.258	9103	2.295
DSJC1000.5	37 575	**37 610**	4.606	37 698	4.619
DSJC1000.9	103 445	**103 470**	2.896	103 479	2.896
flat300.20-0	3150	3150	1.057	3150	1.057
flat300.26-0	3966	3966	1.562	3966	1.562
flat300.28-0	4238	**5557**	2.589	5582	2.605
flat1000.50-0	25 500	**26 900**	3.075	26 910	3.076
flat1000.60-0	30 100	**32 189**	3.847	32 200	3.849
flat1000.76-0	37 164	**38 203**	4.760	38 267	4.770
Average	–	–	**2.255**	–	**2.266**

TABLE 3. Comparative results of Bi-EA and Bi-EA$_{(\mathrm{nPTX})}$ algorithms on the set of 16 DIMACS instances.

Instances	f^b_{UB}	Bi-EA		Bi-EA$_{\mathrm{nPTS}}$	
		f^*_{UB}	RPD	f^*_{UB}	RPD
DSJC125.5	1012	**1000**	0.821	1020	0.857
DSJC125.9	2503	**2506**	0.481	2580	0.525
DSJC250.5	3210	**3218**	1.500	3279	1.547
DSJC250.9	8277	**8277**	0.919	8300	0.925
DSJC500.1	2836	**2840**	1.272	2910	1.328
DSJC500.5	10 886	**10 905**	2.730	10 913	2.733
DSJC500.9	29 862	**29 910**	1.706	29 965	1.711
DSJC1000.1	8991	**9000**	2.258	9120	2.301
DSJC1000.5	37 575	**37 610**	4.606	37 750	4.627
DSJC1000.9	103 445	**103 470**	2.896	103 528	2.898
flat300.20-0	3150	3150	1.057	3150	1.057
flat300.26-0	3966	3966	1.562	3966	1.562
flat300.28-0	4238	**5557**	2.589	5590	2.611
flat1000.50-0	25 500	**26 900**	3.075	27 040	3.096
flat1000.60-0	30 100	**32 189**	3.847	32 230	3.853
flat1000.76-0	37 164	**38 203**	4.760	38 312	4.776
Average	–	–	**2.255**	–	**2.275**

TABLE 4. Results of statistical tests for the Bi-EA algorithm analysis.

Instances	Upper Bounds	
	Bi-EA *vs.* Bi-EA$_{\text{GPX}}$ Significantly different?	Bi-EA *vs.* Bi-EA$_{(\text{nPTX})}$ Significantly different?
DSJC125.5	Yes	Yes
DSJC125.9	Yes	Yes
DSJC250.5	Yes	Yes
DSJC250.9	Yes	Yes
DSJC500.1	Yes	Yes
DSJC500.5	Yes	Yes
DSJC500.9	Yes	Yes
DSJC1000.1	Yes	Yes
DSJC1000.5	Yes	Yes
DSJC1000.9	Yes	Yes
flat300.20-0	No	No
flat300.26-0	No	No
flat300.28-0	Yes	Yes
flat1000.50-0	Yes	Yes
flat1000.60-0	Yes	Yes
flat1000.76-0	Yes	Yes

Statistical analysis of the Bi-EA study

In order to confirm our findings and draw accurate interpretations of the Bi-EA study performed above, we conduct a statistical analysis of the obtained results. For each tested instance, we attempt to analyze statistically the relevance of our results. First, it is required to verify the normality of the data in order to determine whether parametric or non parametric tests are appropriate. Parametric tests are performed if the data are normally distributed. In our context of study, three different normality tests were used: the Kolmogorov–Smirnov, the Shapiro–Wilk and the D'Agostino–Pearson normality tests. All of these tests show that the data were not normally distributed. Thus, Wilcoxon's tests are used to determine whether the observed difference between the compared algorithms is statistically significant when a 95% confidence level test is performed in terms of the best result obtained f^*_{UB}. Statistical results are displayed in Table 4.

The obtained statistical results confirm our initial finding. Indeed, The Wilcoxon's tests indicate that, in term of upper bounds, Bi-EA is statistically the best for 14 out of 16 cases except for the instances where the other versions can achieve the best known results f^*_{UB}. In summary, the use of SPX as a crossover operator has allowed Bi-EA to be more effective. Moreover, statistical results demonstrate the valuable role of the perturbed tabu search procedure compared to a simple genetic algorithm which did not refer to any local optimization for the mutation step.

3.5. Computational results

In this section, we evaluate the performance of Bi-EA regarding quality of obtained upper bounds for the sum coloring problem. In Table 5, columns 1–3 detail the main characteristics of each tested instance (instance name, number of vertices and number of edges). Column f^b_{UB} is the current best known upper bounds in literature. In the rest of the table, we present computational results of Bi-EA as follows: the best upper bound UB*, percentage of the relative RPD(%) and the average running time t_1 to reach UB* (in minutes).

Table 5 indicates that Bi-EA improves some best known results (indicated in bold). More precisely, Bi-EA improves 3 best known results and equals 19 others for some tested instances. In term of solution quality, Bi-EA exhibits unitary average RPD equal to 1.489% and 0.113% for DIMACS and COLOR instances respectively.

TABLE 5. Detailed computational results of Bi-EA algorithm on the 26 DIMACS instances (upper part) and 16 COLOR instances (bottom part).

Instances	$\|V\|$	$\|E\|$	f^b_{LB}	f^b_{UB}	Upper Bounds		
					UB*	RPD	t_1
DSJC125.1	125	136	247	326	**313**	0.267	13
DSJC125.5	125	3891	549	1012	**1000**	0.821	4
DSJC125.9	125	6961	1691	2503	2506	0.481	5
DSJC250.1	250	3218	570	970	975	0.713	16
DSJC250.5	250	15 668	1287	3210	3218	1.500	25
DSJC250.9	250	27 897	4311	8277	8277	0.919	7.5
DSJC500.1	500	12 458	1250	2836	2840	1.272	36
DSJC500.5	500	62 624	2923	10 886	10 905	2.730	53
DSJC500.9	500	112 437	11 053	29 862	29 910	1.706	41
DSJC1000.1	1000	49 629	2762	8991	9000	2.258	98
DSJC1000.5	1000	249 826	6708	37 575	37 610	4.606	21
DSJC1000.9	1000	449 449	26 557	103 445	103 470	2.896	59
Le450-15a	450	8168	2331	2632	**2628**	0.128	42
Le450-15b	450	8169	2348	2632	2640	0.124	38
Le450-15c	450	16 680	2610	3487	3839	0.480	39
Le450-15d	450	16 750	2628	3505	3514	0.340	55
Le450-25a	450	8260	3003	3153	3160	0.052	36
Le450-25b	450	8263	3305	3365	3366	0.018	59
Le450-25c	450	17 343	3657	4515	4698	0.284	43,5
Le450-25d	450	17 425	3698	4544	4559	0.233	86
flat300.20-0	300	21 375	1531	3150	3150	1.057	48
flat300.26-0	300	21 633	1548	3966	3966	1.562	1.5
flat300.28-0	300	21 695	1547	4238	5557	2.589	132.5
flat1000.50-0	1000	245 000	6601	25 500	26 900	3.075	9
flat1000.60-0	1000	245 830	6640	30 100	32 189	3.847	11
flat1000.76-0	1000	246 708	6632	37 167	38 203	4.760	20
Average	–	–	–	–	–	**1.489**	**38.423**
myciel3	11	20	16	21	21	0.312	0
myciel4	23	71	34	45	45	0.323	0 0
myciel5	47	236	70	93	93	0.328	0
myciel6	95	755	142	189	189	0.330	0
myciel7	191	2360	286	381	381	0.332	0
anna	138	493	273	276	276	0.010	0.3
david	87	406	234	237	237	0.012	0.2
huck	74	301	243	243	243	0	0
jean	80	254	216	217	217	0.004	0
queen5.5	25	160	75	75	75	0	0
queen6.6	36	290	126	138	138	0.095	0.2
queen7.7	49	476	196	196	196	0	0
queen8.8	64	728	288	291	291	0.010	0.1
miles250	128	387	318	325	325	0.022	1
miles500	128	1170	686	705	705	0.027	2
games120	120	638	442	443	443	0.002	0.5
Average	–	–	–	–	–	**0.113**	**0.268**

TABLE 6. Results of Spearman Correlation test between the instance's characteristics and the performance of Bi-EA algorithm.

	Upper Bounds	
	$\|V\|$ *vs.* Bi-EA	$\|E\|$ *vs.* Bi-EA
r_s	0.9280	0.9721
95% confidence interval	0.8669 to 0.9616	0.9475 to 0.9853
P (two-tailed)	< 0.0001	< 0.0001
Significant? (alpha = 0.05)	Yes	Yes

From tested instances, one notes large gaps for DIMACS compared to COLOR. This is mainly due to the hardness of DIMACS instances for most of sum coloring algorithms.

Statistical analysis of the Bi-EA performance according to the instance's features

As data are not normally distributed, a Spearman Correlation test r_s was conducted to know the statistical correlation between the instance's characteristics and the performance of Bi-EA algorithm in term of obtained upper bounds. Results are shown in Table 6.

Results of the Spearman Correlation test confirm that as the graph is more sophisticated and complex, the performance of Bi-EA degrades. This was fully shown through the reported RPD which change depending on the instances topology. A such statistical observation fully approve the hardness of the DIMACS instances compared to COLOR instances for upper bounds.

3.6. Comparative study

To further evaluate performance of Bi-EA, we compare obtained upper bounds with bounds reported by the best performing algorithms in the literature namely: HESA [13], EXSCOL [29], MA-MSC [24], and MASC [14]. The EXSCOL algorithm was programmed in C and compiled using GNUGCC on a PC with 2.8 GHz CPU and 2 GRAM. For the MA-MSC, the algorithm was in C and run on Intel Core 2 Duo T5450 1.66-1.67 with 2 GB Ram. Concerning HESA, the approach was coded in C++ and compiled with a 2.83 GHz processor and 8 GB RAM. The results are summarized in Table 7. The column headings are as follows: f^b_{UB}= the best known upper bound; f^*_{UB} = the best upper bound obtained by the reference algorithm; RPD(%) = relative percentage deviation of obtained upper bound; t = CPU time in minutes required to compute f^*_{UB}.

Looking at Table 7, we can draw the following observations:

- All shown methods tend to exhibit very similar performance. More precisely, for DIMACS instances, the average RPD are 1.424% for HESA, 1.440% for EXSCOL, 1.464% for MA-MSC, 1.438% for MASC and 1.489% for Bi-EA. Regarding COLOR instances, it is noteworthy that all the average RPD reported by the compared algorithms are exactly similar and equal to 0.113% except for the EXSCOL algorithm where the average RPD was equal to 0.121%. The observed large gaps reported by all the state-of-the-art algorithms approve again the complexity of the DIMACS instances comparing to COLOR instances during the coloring.
- Bi-EA outperforms all cited algorithms for 5 cases and successfully gets improved upper bounds (entries in bold). Also, our proposed approach was able to equal the best known upper bounds for 19 cases out of 42 (underlined entries).
- To evaluate the performance of Bi-EA on large graphs, one must refer to the well-known EXSCOL algorithm specifically designed to deal with these instances. Comparing Bi-EA to EXSCOL, one notes that our approach behaves favorably well and delivers an average RPD equal to 1.489% nearby 1.440% for DIMACS instances within a modest computing time. This slightly favors our proposed method even when dealing with large and hard instances.

TABLE 7. Results of comparisons analysis for the upper bounds on the 26 DIMACS instances (upper part) and 16 COLOR instances (bottom part). The dash "-" symbol indicates that the related statistics are not available.

Instances	f^b_{UB}	HESA			EXSCOL			MA-MSC			MASC			Bi-EA		
		f^*_{UB}	RPD	t	f^*_{UB}	RPD	t	f^*_{UB}	RPD	t	f^*_{UB}	RPD	t	f^*_{UB}	RPD	t
DSJC125.1	326	326	0.319	5.2	326	0.319	1	326	0.319	10	326	0.319	4.4	**313**	0.267	13
DSJC125.5	1012	1012	0.843	10.1	1017	0.852	1	1013	0.845	5	1012	0.843	3.5	**1000**	0.821	4
DSJC125.9	2503	2503	0.481	0.3	2512	0.487	1	2503	0.481	2	2503	0.481	1.9	2506	0.481	5
DSJC250.1	970	970	0.704	30.7	985	0.731	4	983	0.727	42	974	0.711	17.3	975	0.713	16
DSJC250.5	3210	3210	1.507	47.1	3246	1.535	6	3214	1.510	26	3230	1.523	23.1	3218	1.500	25
DSJC250.9	8277	8277	0.934	24.6	8286	0.936	7	8277	0.934	33	8280	0.935	5.6	8277	0.919	7.5
DSJC500.1	2836	**2836**	1.268	82.6	2850	1.28	9	2897	1.317	4	2940	1.352	50.4	2840	1.272	36
DSJC500.5	10 886	10 886	2.726	97	10 910	2.735	11	11 082	2.793	42	11 101	2.800	202.5	10 905	2.730	53
DSJC500.9	29 862	29 862	1.744	95.4	29 912	1.749	15	29 995	1.756	51	29 994	1.756	90.9	29 910	1.706	41
DSJC1000.1	8991	8991	2.255	101.6	9003	2.259	28	9188	2.326	31	8995	2.256	70	9000	2.258	98
DSJC1000.5	37 575	37 575	4.601	33.5	37 598	4.604	24	38 421	4.727	23	37 f594	4.604	200	37 610	4.606	21
DSJC1000.9	10 3445	10 3445	2.895	103.1	103 464	2.895	27	105 234	2.962	61	103 464	2.895	125	103 470	2.896	59
Le450-15a	2632	2634	0.130	91.5	2632	0.130	5	2681	0.151	19	2706	0.161	41	**2628**	0.128	42
Le450-15b	2632	2632	0.120	89.9	2642	0.125	7	2690	0.145	19	2724	0.160	40	2640	0.124	38
Le450-15c	3487	3487	0.344	86.7	3866	0.490	6	3943	0.520	6	3491	0.346	45	3839	0.480	39
Le450-15d	3505	3505	0.336	82.7	3921	0.495	5	3926	0.497	3	3506	0.337	59	3514	0.340	55
Le450-25a	3153	3153	0.051	88.5	3153	0.049	7	3178	0.058	5	3166	0.054	39	3160	0.052	36
Le450-25b	3365	3365	0.018	88.6	3366	0.018	6	3379	0.022	7	3366	0.018	40	3366	0.018	59
Le450-25c	4515	4553	0.251	84.8	4515	0.241	8	4648	0.277	16	4700	0.284	25	4698	0.291	43.5
Le450-25d	4544	4569	0.235	92.4	4544	0.229	7	4696	0.270	3	4722	0.277	27	4559	0.233	86
flat300.20-0	3150	3150	1.066	0	3150	1.066	3	3150	1.066	0	3150	1.066	0	3150	1.057	48
flat300.26-0	3966	3966	1.582	0.4	3966	1.582	3	3966	1.582	0	3966	1.582	0	3966	1.562	1.5
flat300.28-0	4238	4260	1.764	49.7	4282	1.778	3	4261	1.765	28	4238	1.750	22	5557	2.589	132.5
flat1000.50-0	25 500	25 500	2.863	0.3	25 500	2.863	9	25 500	2.863	28	25 500	2.863	0	26 900	3.075	9
flat1000.60-0	30 100	30 100	3.533	2.7	30 100	3.533	11	30 100	3.533	16	30 100	3.533	114	32 189	3.847	11
flat1000.76-0	37 164	37 164	4.603	36.8	37 167	4.604	19	38 213	4.761	8	37 167	4.604	1	38 203	4.760	20
Average	–	–	**1.424**	**54.853**	–	**1.440**	**8.961**	–	**1.464**	**18.769**	–	**1.438**	**47.984**	–	**1.489**	**38.423**
myciel3	21	21	0.312	0	21	0.312	1	21	0.312	0	21	0.312	0	21	0.312	0
myciel4	45	45	0.323	0	45	0.323	1	45	0.323	0	45	0.323	0	45	0.323	0
myciel5	93	93	0.328	0	93	0.328	1	93	0.328	0	93	0.328	0	93	0.328	0
myciel6	189	189	0.330	0	189	0.330	2	189	0.330	0	189	0.330	0	189	0.330	0
myciel7	381	381	0.332	0	381	0.332	2	381	0.332	0	381	0.332	1	381	0.332	0
anna	276	276	0.010	0.2	283	0.036	2	276	0.010	1	276	0.010	0	276	0.010	0.3
david	237	237	0.012	0.1	237	0.012	1	237	0.012	1	237	0.012	0	237	0.012	0.2
huck	243	243	0	0	243	0	1	243	0	0	243	0	0	243	0	0
jean	217	217	0.004	0	217	0.004	1	217	0.004	0	217	0.004	0	217	0.004	0
queen5.5	75	75	0	47.8	75	0	1	75	0	0	75	0	0	75	0	0
queen6.6	138	138	0.095	0	150	0.190	1	138	0.095	0	138	0.095	1	138	0.095	0.2
queen7.7	196	196	0	0	196	0	1	196	0	0	196	0	0	196	0	0
queen8.8	291	291	0.010	0	291	0.010	1	291	0.010	0	291	0.010	12	291	0.010	0.1
miles250	325	325	0.022	0.1	328	0.031	2	325	0.022	8	325	0.022	0	325	0.022	1
miles500	705	705	0.027	0	709	0.033	2	708	0.032	12	705	0.027	1	705	0.027	2
games120	443	443	0,002	0.5	443	0,002	2	443	0.002	0	443	0.002	0	443	0.002	0.5
Average	–	–	**0.113**	**3.043**	–	**0.121**	**1.375**	–	**0.113**	**1.375**	–	**0.113**	**0.937**	–	**0.113**	**0.268**
Best#/#G	–		38/42			21/42			22/42			24/42			22/42	

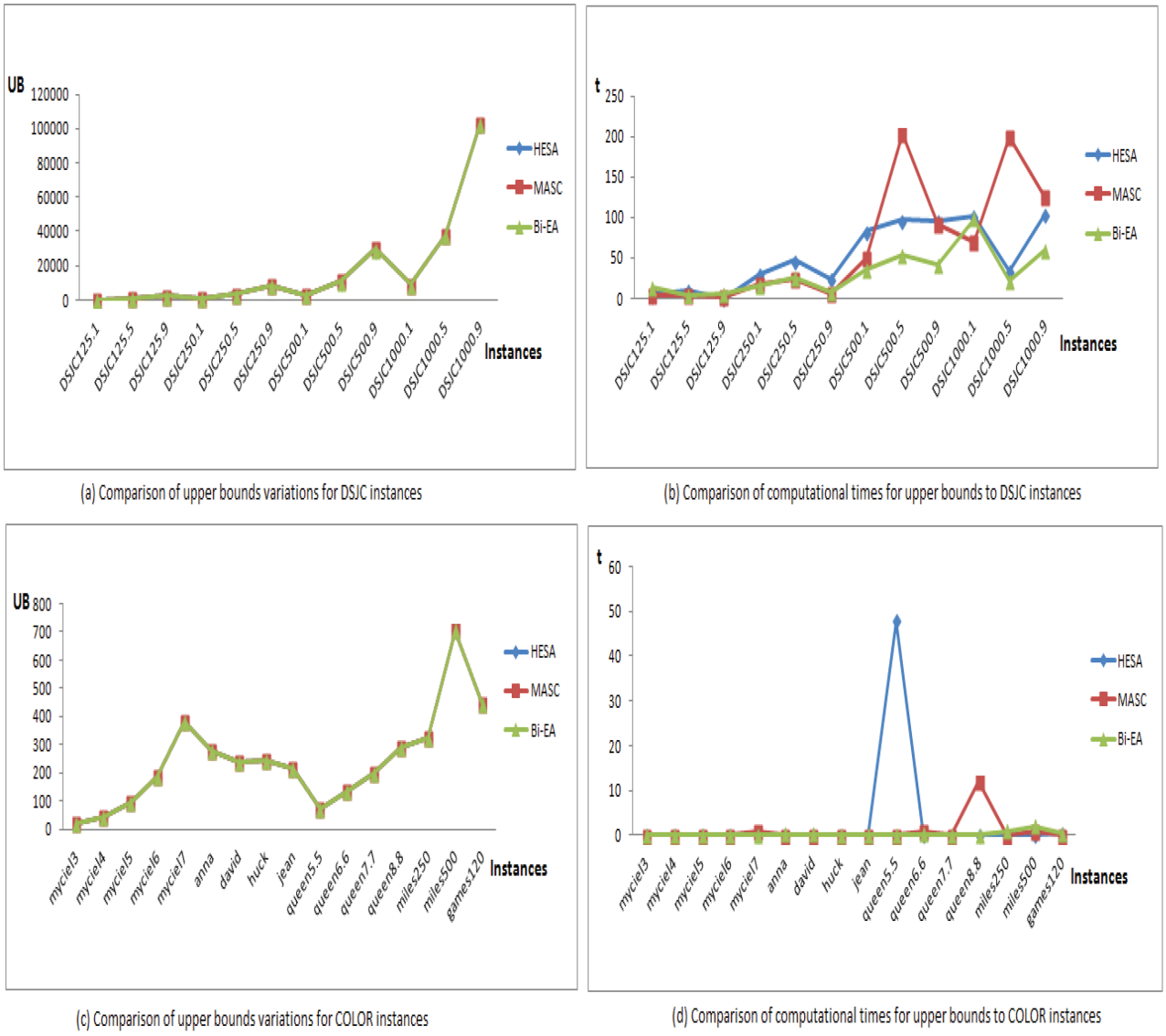

FIGURE 4. Comparisons of Bi-EA, HESA and MASC Upper bounds and related CPU time.

– When compared with similar techniques that are based on hybrid metaheuristics, Bi-EA performs well by improving its CPU time $t = 38.423$ min comparing to 47.984 min for MASC and 54.853 min for HESA. In other words, our algorithm can reach promising area in the search space more rapidly since it is allowed to move to non feasible solutions. This is fully shown for instances of the DIMACS Benchmark as well for the COLOR instances presented in Figure 4.

To better analyze the performance of our proposed algorithm, we compare between the state-of-the-art sum coloring algorithms one by one and illustrate the comparisons in Figure 5. Indeed, we provide three bars for each comparison. Each bar indicates respectively whether Bi-EA is better than, similar to or worse than the different reference algorithms.

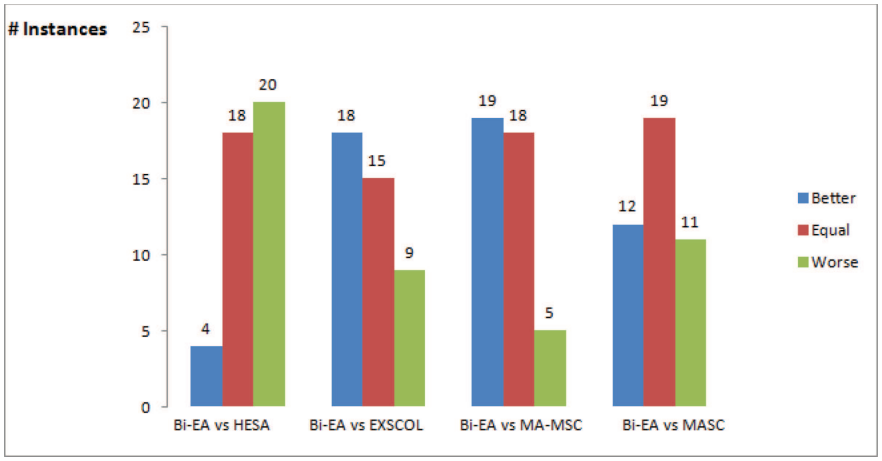

FIGURE 5. Comparisons of Bi-EA and four reference algorithms for the upper bounds.

From Figure 5, one notices that Bi-EA successfully gets better results than these reference algorithms for 4, 18, 19 and 12 instances respectively. Indeed, it is quite competitive with the most recent hybrid evolutionary algorithms HESA and MASC since it obtains better or equal upper bounds for 22 and 31 benchmark instances respectively.

Statistical analysis of the comparative study on upper and lower bounds

In what follows, an extensive statistical analysis is conducted to approve all the above observations. In this vein, we apply the same statistical tests as in Section 3.4 for the comparison which prove that data were not normally distributed. Simulation results presented in Table 8 confirm all the previous comments stated for the effectiveness of the BI-EA algorithm regarding the mentioned four reference algorithms in term of upper bounds.

The additional statistical methodology provide further empirical evidence that the Bi-EA method consistently produces tight upper bounds. Its results are statistically better than HESA, EXSCOL, MA-MSC and MASC for respectively 4, 18, 19 and 12 cases out of 42.

4. CONCLUSION

In this paper, we addressed the Minimum Sum Coloring Problem (MSCP), a combinatorial optimization problem with strong practical relevance in domains such as scheduling and resource allocation. To tackle this challenge, we proposed a hybrid bi-criteria evolutionary algorithm (Bi-EA) designed to compute high-quality upper bounds for the chromatic sum. The algorithm integrates several complementary mechanisms that enhance both its performance and robustness. A key feature of Bi-EA is its ability to explore non-feasible solutions, which significantly accelerates the search process compared to state-of-the-art methods. Furthermore, an adaptive selection strategy guided by a score function was employed to balance exploration and exploitation. The algorithm also introduces two novel operators: the Sum Partition Crossover (SPX) and the Perturbed Tabu Search (PTS). Extensive computational experiments confirmed the relevance of these operators, demonstrating that Bi-EA is not only fast but also consistently delivers robust, high-quality solutions.

At this point, it is worth mentioning that, while the Bi-EA algorithm successfully outperforms some best known bounds, our proposed technique faced a little bit difficulty especially with DIMACS challenge. This observation is strongly due to the computational hardness of the MSCP. This work can be extended to deal

TABLE 8. Results of statistical tests for comparison analysis of the upper and lower bounds.

Instances	Upper Bounds			
	Bi-EA *vs.* HESA Significantly different?	Bi-EA *vs.* EXSCOL Significantly different?	Bi-EA *vs.* MA-MSC Significantly different?	Bi-EA *vs.* MASC Significantly different?
DSJC125.1	Yes	Yes	Yes	Yes
DSJC125.5	Yes	Yes	Yes	Yes
DSJC125.9	Yes	Yes	Yes	Yes
DSJC250.1	Yes	Yes	Yes	Yes
DSJC250.5	Yes	Yes	Yes	Yes
DSJC250.9	No	Yes	No	Yes
DSJC500.1	Yes	Yes	Yes	Yes
DSJC500.5	Yes	Yes	Yes	Yes
DSJC500.9	Yes	Yes	Yes	Yes
DSJC1000.1	Yes	Yes	Yes	Yes
DSJC1000.5	Yes	Yes	Yes	Yes
DSJC1000.9	Yes	Yes	Yes	Yes
Le450-15a	Yes	Yes	Yes	Yes
Le450-15b	Yes	Yes	Yes	Yes
Le450-15c	Yes	Yes	Yes	Yes
Le450-15d	Yes	Yes	Yes	Yes
Le450-25a	Yes	Yes	Yes	Yes
Le450-25b	Yes	No	Yes	No
Le450-25c	Yes	Yes	Yes	Yes
Le450-25d	Yes	Yes	Yes	Yes
flat300.20-0	No	No	No	No
flat300.26-0	No	No	No	No
flat300.28-0	Yes	Yes	Yes	Yes
flat1000.50-0	Yes	Yes	Yes	Yes
flat1000.60-0	Yes	Yes	Yes	Yes
flat1000.76-0	Yes	Yes	Yes	Yes
myciel3	No	No	No	No
myciel4	No	No	No	No
myciel5	No	No	No	No
myciel6	No	No	No	No
myciel7	No	No	No	No
anna	No	Yes	No	No
david	No	No	No	No
huck	No	No	No	No
jean	No	No	No	No
queen5.5	No	No	No	No
queen6.6	No	Yes	No	No
queen7.7	No	No	No	No
queen8.8	No	No	No	No
miles250	No	Yes	No	No
miles500	No	Yes	Yes	No
games120	No	No	No	No

with other variants of the graph coloring problem by considering non feasible solutions and combining the tabu search technique into an evolutionary algorithm.

CONFLICTS OF INTEREST

No potential conflict of interest was reported by the authors.

DATA AVAILABILITY STATEMENT

No new data/codes were created or analyzed in this study.

References

[1] A. Bar-Noy and G. Kortsarz, Minimum color sum of bipartite graphs. *J. Algorithms* **28** (1998) 339–365.

[2] A. Bar-Noy, M. Bellare, M.M. Halldórsson, H. Shachnai and T. Tamir, On chromatic sums and distributed resource allocation. *Inf. Comput.* **140** (1998) 183–202.

[3] U. Benlic and J.K. Hao, A study of breakout local search for the minimum sum coloring problem, in Asia-Pacific Conference on Simulated Evolution and Learning. Springer (2012) 128–137.

[4] F. Bonomo, G. Durán, J. Marenco and M. Valencia-Pabon, Minimum sum set coloring of trees and line graphs of trees. *Discrete Appl. Math.* **159** (2011) 288–294.

[5] H. Bouziri and O. Harrabi, Behavior study of genetic operators for the minimum sum coloring problem, in 2013 5th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO). IEEE (2013) 1–6.

[6] H. Bouziri and M. Jouini, A tabu search approach for the sum coloring problem. *Electron. Notes Discrete Math.* **36** (2010) 915–922.

[7] D. Brélaz, New methods to color the vertices of a graph. *Commun. ACM* **22** (1979) 251–256.

[8] P. Galinier and J.K. Hao, Hybrid evolutionary algorithms for graph coloring. *J. Comb. Optim.* **3** (1999) 379–397.

[9] P. Galinier and A. Hertz, A survey of local search methods for graph coloring. *Comput. Oper. Res.* **33** (2006) 2547–2562.

[10] F. Glover, Tabu search and adaptive memory programming – advances, applications and challenges, in Interfaces in Computer Science and Operations Research. Springer (1997) 1–75.

[11] O. Harrabi and J.C. Siala, An effective parameter tuning for a bi-objective genetic algorithm to solve the sum coloring problem, in Soft Computing for Problem Solving. Springer (2019) 107–119.

[12] A. Helmar and M. Chiarandini, A local search heuristic for chromatic sum, in Proceedings of the 9th Metaheuristics International Conference. Vol. 1101 (2011) 161–170.

[13] Y. Jin and J.K. Hao, Hybrid evolutionary search for the minimum sum coloring problem of graphs. *Inf. Sci.* **352** (2016) 15–34.

[14] Y. Jin, J.K. Hao and J.P. Hamiez, A memetic algorithm for the minimum sum coloring problem. *Comput. Oper. Res.* **43** (2014) 318–327.

[15] Y. Jin, J.P. Hamiez and J.K. Hao, Algorithms for the minimum sum coloring problem: a review. *Artif. Intell. Rev.* **47** (2017) 367–394.

[16] Z. Kokosiński and K. Kwarciany, On sum coloring of graphs with parallel genetic algorithms, in International Conference on Adaptive and Natural Computing Algorithms. Springer (2007) 211–219.

[17] L.G. Kroon, A. Sen, H. Deng and A. Roy, The optimal cost chromatic partition problem for trees and interval graphs, in International Workshop on Graph-Theoretic Concepts in Computer Science. Springer (1996) 279–292.

[18] E. Kubicka and A.J. Schwenk, An introduction to chromatic sums, in Proceedings of the 17th conference on ACM Annual Computer Science Conference. ACM (1989) 39–45.

[19] E. Kubicka, G. Kubicki and D. Kountanis, Approximation algorithms for the chromatic sum, in Computing in the 90's. Springer (1991) 15–21.

[20] Y. Li, C. Lucet, A. Moukrim and K. Sghiouer, Greedy algorithms for the minimum sum coloring problem, in Logistique et transports (2009) LT–027.

[21] A. Liefooghe, L. Jourdan and E.G. Talbi, A unified model for evolutionary multi-objective optimization and its implementation in a general purpose software framework, in IEEE Symposium on Computational Intelligence in Miulti-Criteria Decision-Making, 2009. MCDM'09. IEEE (2009) 88–95.

[22] Z. Lü and J.K. Hao, A memetic algorithm for graph coloring. *Eur. J. Oper. Res.* **203** (2010) 241–250.

[23] A. Moukrim, K. Sghiouer, C. Lucet and Y. Li, Lower bounds for the minimal sum coloring problem. *Electron. Notes Discrete Math.* **36** (2010) 663–670.

[24] A. Moukrim, K. Sghiouer, C. Lucet and Y. Li, Upper and lower bounds for the minimum sum coloring problem (2014).

[25] D.C. Porumbel, J.K. Hao and P. Kuntz, An evolutionary approach with diversity guarantee and well-informed grouping recombination for graph coloring. *Comput. Oper. Res.* **37** (2010) 1822–1832.

[26] M.R. Salavatipour, On sum coloring of graphs. *Discrete Appl. Math.* **127** (2003) 477–488.

[27] J.D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms, in Proceedings of the First International Conference on Genetic Algorithms and Their Applications, 1985. Lawrence Erlbaum Associates. Inc., Publishers (1985).

[28] A. Sen, H. Deng and S. Guha, On a graph partition problem with application to VLSI layout. *Inf. Process. Lett.* **43** (1992) 87–94.

[29] Q. Wu and J.K. Hao, An effective heuristic algorithm for sum coloring of graphs. *Comput. Oper. Res.* **39** (2012) 1593–1600.