Minimum sum coloring for large graphs with extraction and backward expansion search

Qinghua Wu [a], Qing Zhou [a], Yan Jin [b,*] Jin-Kao Hao [c,d]

[a] *School of Management, Huazhong University of Science and Technology*

[b] *School of Computer Science, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China*

[c] *LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers, France*

[d] *Institut Universitaire de France, 1 Rue Descartes, 75231 Paris, France*

Applied Soft Computing, DOI: 10.1016/j.asoc.2017.09.043

Abstract

The Minimum Sum Coloring Problem (MSCP) is a relevant model tightly related to the classical vertex coloring problem (VCP). MSCP is known to be NP-hard, thus solving the problem for large graphs is particular challenging. Based on the general "reduce-and-solve" principle and inspired by the work for the VCP, we present an extraction and backward expansion search approach (EBES) to compute the upper and lower bounds for the MSCP on large graphs. The extraction phase reduces the given graph by extracting large collections of pairwise disjoint large independent sets (or color classes). The backward extension phase adds the extracted independent sets to recover the intermediate graphs while optimizing the sum coloring of each intermediate graph. We assess the proposed approach on a set of 35 large benchmark graphs with 450 to 4000 vertices from the DIMACS and COLOR graph coloring competitions. Computational results show that EBES is able to find improved upper bounds for 19 graphs and improved lower bounds for 12 graphs.

Keywords: Coloring problems; Hybrid search; Independent set; Problem reduction.

* Corresponding author.

Email addresses: qinghuawu1005@gmail.com (Qinghua Wu), qingzhou@hust.edu.cn (Qing Zhou), yanjin.china@hotmail.com (Yan Jin), jin-kao.hao@univ-angers.fr (Jin-Kao Hao).

1 Introduction

Let $G = (V, E)$ be a simple undirected graph with vertex set V and edge set E, and let k be an integer ($1 \leq k \leq n$) representing the number of colors. A *legal k-coloring* c of G is a partition of V into k mutually disjoint independent sets (color classes) $V_1, \ldots, V_k$ such that no two vertices of a color class are linked by an edge, i.e., for $\forall u, v \in V_i$ ($i = 1, \ldots, k$), $\{u, v\} \notin E$ holds. The Minimum Sum Coloring Problem (MSCP) involves finding a legal k-coloring c that minimizes the following sum of colors [11,14,23].

$$f(c) = \sum_{i=1}^{k} i|V_i| \tag{1}$$

where $|V_i|$ is the cardinality of V_i, $|V_1| \geq \ldots \geq |V_k|$. The minimum sum of colors for the MSCP is called the *chromatic sum* of G, and is denoted by $\sum(G)$.

The MSCP is derived from the conventional NP-hard Vertex Coloring Problem (VCP) that aims at finding a legal coloring with the smallest k [3,4], and is denoted by $\chi(G)$. Like the VCP, the MSCP is NP-hard [14]. In practice, the MSCP has a number of relevant applications in scheduling, resource allocation, VLSI design and so on [17].

From a perspective of exact methods for the MSCP, only three studies have been reported in the literature [15,19,25]. In [25], the MSCP was solved by an integer linear programming solver (CPLEX) applied to a simple linear model. In [15], the authors investigated different approaches including branch-and-bound, transformations to satisfiability problems and constraint programming. In [19], two approaches based on integer linear programming and constraint programming were experimented. These studies showed that existing exact approaches can only be applied to solve problem instances of limited sizes (roughly up to 150 vertices).

In order to deal with larger graphs, heuristic approaches are preferred to find suboptimal solutions (bounds). These include two greedy algorithms MDSAT & MRLF [16], a tabu search algorithm [2], a local search heuristic MDS(5)+LS [7], a breakout local search algorithm BLS [1] and a heuristic using independent set extraction EXSCOL [26]. More sophisticated hybrid algorithms were also developed recently including two memetic algorithms (MASC [8] and MA [21]) and a hybrid evolutionary search algorithm HESA [10]. MASC combines a tabu search local optimization procedure and a multi-parent crossover while MA uses a double-parent crossover combined with a hill-climber and "destroy and repair" procedure. HESA [10] uses an iterated double-phase tabu search

procedure combined with two dedicated crossover operators. These hybrid algorithms are among the state-of-the-art heuristics for the MSCP. Yet, all existing algorithms have trouble to effectively handle large graphs with more than several hundreds of vertices.

In this paper, we present an extraction and backward expansion search algorithm (denoted by EBES) to find minimum sum colorings for large graphs. This approach employs the general "reduce-and-solve" principle, which is known to be effective in coloring large graphs for the classical VCP [6,12,22,27]. Basically, the proposed EBES algorithm embeds an extraction phase that heuristically identifies and removes large collections of pairwise disjoint large independent sets. This is based on the consideration that assigning small colors to large independent sets minimizes the sum of colors defined by Eq. (1). However, this extraction approach may miss some relevant independent sets. To address this issue, we introduce a backward expansion phase that reconsiders the extracted independent sets while re-coloring the intermediate graphs to improve the sum of colors. Experiments on a set of 35 large benchmark graphs with 450 to 4000 vertices show that the proposed algorithm performs very well. Indeed, compared with the state-of-art results, EBES is able to find improved upper bounds for 19 instances and improved lower bounds for 12 instances out of the 35 tested benchmark graphs. We additionally analyze the behavior of the EBES algorithm on very large random graphs with 5000 to 10000 vertices.

The reminder of the paper is organized as follows. Section 2 presents the proposed EBES algorithm for computing upper bounds of the MSCP. Section 3 explain the way of using EBES to compute lower bounds of the MSCP. Section 4 presents computational results obtained by EBES and comparisons with state-of-the-art MSCP algorithms in the literature. Before concluding, Section 5 investigates one key ingredient of the proposed algorithm.

2 Extraction and backward expansion search for the MSCP

In this section, we present the proposed extraction and backward expansion approach for the MSCP. The approach is inspired by previous studies of applying similar ideas to the vertex coloring problem [6,27] and extends a basic independent set extraction approach for the MSCP [26].

2.1 General procedure

As we can observe in Eq. (1), the objective value (i.e., sum of colors) can be minimized by constructing large color classes and assigning small colors to them. This observation was first explored in [26] where an independent set algorithm is used to progressively extract sets of color classes until a null graph is reached. This extraction phase generates thus a series of intermediate graphs with decreasing number of vertices. The proposed EBES algorithm extends this approach in two directions. First, EBES combines independent sets extraction with an MSCP coloring algorithm, i.e., the extraction phase stops when the residual graph has no more than q (a parameter) vertices, which is then colored with an MSCP algorithm. Second and more importantly, EBES embeds a backward expansion phase to recover the intermediate graphs in the reverse order. This phase enables new opportunities for further minimization of the sum of colors.

The general scheme of our EBES algorithm for the MSCP is summarized in Algorithm 1, which is composed of three phases.

(1) **Independent set extraction:** Suppose that the initial graph is given by $G_0 = G$, EBES simplifies the graph G_0 by iteratively extracting a maximum collection φ_i of disjoint independent sets of the maximum size, thus progressively obtains smaller intermediate graphs G_i ($G_i = G_{i-1} - \varphi_i, i = 1, 2, \ldots, m$). This phase stops when G_m contains no more than a prefixed threshold of q vertices. For instance, as shown in Fig. 1, given the initial graph G_0 (and $q = 2$), a smaller intermediate graph G_1 is obtained by removing two maximum independent sets collected in $\varphi_1 = \{\{C, H, J\}, \{D, I, F\}\}$ (as well as the corresponding edges) from the graph G_0. Hereafter, G_1 is further reduced to the residual graph G_2 in a similar way.

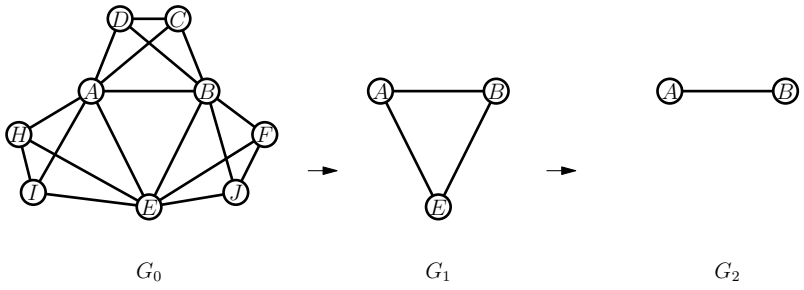


Fig. 1. An illustrative example of the extraction phase

(2) **Initial coloring:** For the smallest residual graph G_n, we can apply any MSCP algorithm to determine a legal coloring c' with the smallest sum of colors. In our case, we adopt the hybrid evolutionary search algorithm (HESA) presented in [10], which has obtained state-of-the-art results for

the MSCP. As mentioned previously, the number of independent sets extracted from the original graph is $\sum_{i=1}^{m} |\varphi_i|$. Suppose that the coloring c' for G_n is composed of k' color classes $V_1, V_2, \ldots, V_{k'}$ ($|V_1| \geq \ldots \geq |V_{k'}|$), a legal complete coloring c for the graph G is obtained by joining these $\sum_{i=1}^{m} |\varphi_i|$ extracted independent sets plus the k' color classes V_1, V_2, ..., $V_{k'}$. Notice that the independent sets (color classes) of the coloring c are re-sorted in non-increasing order of their cardinality if necessary before color assignment. Hereby, an initial coloring c for G is generated, which is submitted to the backward expansion and re-coloring phase for further improvement of sum of colors.

(3) **Backward expansion phase:** This phase extends the current graph G_i ($i = m, m-1, ...1$) by adding back the extracted independent sets to recover the intermediate graph G_{i-1} such that $G_{i-1} = G_i + \varphi_i$ in order to traverse all intermediate graphs in the reverse order of their creations. For each intermediate graph G_i, the MSCP algorithm is applied on G_i by starting from the current coloring of G_i extended with the newly added independent sets, with the purpose of improving the sum of colors. This phase stops when the initial graph G is recovered. Fig. 2 illustrates the expansion phase by carrying out the inverse of the extraction phase of Fig. 1. Starting from the residual graph G_2, an initial coloring c_0 is obtained by the MSCP algorithm and let its sum of colors be f_0. Then, the intermediate graph G_1 is obtained by adding back the independent sets $\varphi_2 = \{E\}$ to G_2. The corresponding coloring c_1 is again obtained by the MSCP algorithm with a sum of colors f_1 (that is hopefully better than f_0). The same operations are performed to recover the initial graph $G_0 = G$ by adding back the two extracted independent sets of $\varphi_1 = \{\{C, H, J\}, \{D, I, F\}\}$. From the coloring c_1 and the two color classes of φ_1, the MSCP algorithm is applied to obtain a new coloring c_2 (that is hopefully better than c_1). Consequently, the solutions are continuously optimized along with the expansion procedure, contrary to the basic extraction approach of [26] that stops at the end of the extraction phase.

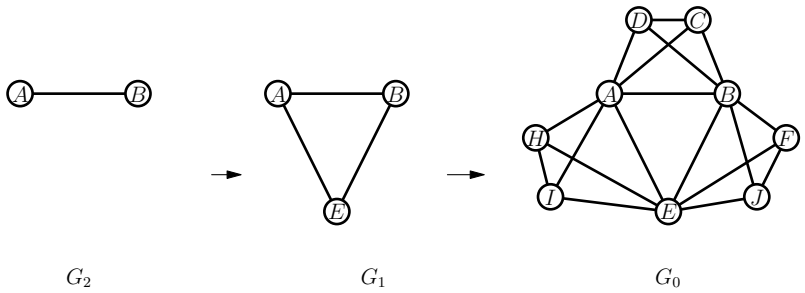


Fig. 2. An illustrative example of the backward expansion phase

Algorithm 1 The EBES algorithm for minimum sum coloring

Input: A graph $G = (V, E)$, a parameter q representing the number of remaining vertices in the smallest residual graph
Output: The best sum coloring c_* found and its sum of colors f_*
{*Extraction phase*}
$i \leftarrow 0$
$G_0 \leftarrow G$
while G_i has more than q vertices **do**
  $\varphi_{i+1} \leftarrow$ Extraction(G_i); /* A maximal collection φ_{i+1} of disjoint independent sets of the largest size is identified from graph G_i */;
  $G_{i+1} \leftarrow G_i - \varphi_{i+1}$; /* Create an intermediate subgraph by removing all vertices of φ_{i+1} as well as the corresponding edges from G_i */
  $i \leftarrow i + 1$;
end while
{*Initial sum coloring phase*}
$c_i \leftarrow$ Initial_Coloring($G_i, \bigcup_{j=1}^{i} \varphi_j$); /*An initial coloring c_i is generated by the color classes of G_i obtained by an MSCP coloring algorithm, plus the extracted independent sets */
$f_* \leftarrow f(c_i)$; $c_* \leftarrow c_i$
{*Backward expansion phase*}
repeat
  $i \leftarrow i - 1$;
  $G_i \leftarrow G_{i+1} + \varphi_{i+1}$; /* Expand the subgraph G_{i+1} by adding back the independent sets φ_{i+1} to obtain the intermediate graph G_i */
  $c_i \leftarrow$ Expansion(c_{i+1}, φ_{i+1}); /* Construct a coloring c_i of G_i from c_{i+1} */
  $c'_i \leftarrow$ Backward_Coloring(c_i); /* Apply the MSCP algorithm to improve the coloring c_i in terms of sum of colors */
  if $f(c'_i)$ is better than f_* **then**
    $f_* \leftarrow f(c'_i)$; $c_* \leftarrow c'_i$
  end if
until $G_i = G$
return f_*, c_*

2.2 Strategy for the extraction phase

Recall that a legal k-coloring of $G = (V, E)$ can be defined as a partition of V into k independent sets. In order to simplify the graph G, EBES applies the extraction phase to reduce G by extracting a maximal set of pairwise disjoint large independent sets. For this, EBES adopts the extraction strategy developed for the VCP [6,27] to deal with the MSCP. The overall procedure of the extraction phase is described as follows.

Step 1 Identify an independent set S as large as possible in G_i by employing

an independent set algorithm (or an equivalent clique algorithm). In our case, we use the adaptive tabu search algorithm (ATS) [28] to determine a large independent set S.

Step 2 Use a set CL to collect as many independent sets of size $|S|$ as possible by repeatedly employing ATS. This procedure stops when the number of independent sets reaches a prefixed number $|V| * \rho$ (ρ is the density of the graph) or no new independent set of $|S|$ can be generated.

Step 3 Find a maximal number of pairwise disjoint independent sets $\varphi_i = \{S_0, S_1, \ldots, S_{max}\}$ $(S_b, S_d \in \varphi_i, S_b \bigcap S_d = \emptyset, b \neq d)$ in CL, which is solved by ATS as the maximum set packing problem [4] (it is equivalent to the maximum clique problem).

Step 4 Remove the vertices of φ_i from G_i as well as the edges adjacent to any of these vertices and obtain the residual graph G_{i+1}, then $i = i + 1$.

Step 5 Repeat the above steps until the residual graph G_i contains no more than q vertices.

Fig. 3 illustrates how this procedure works on a graph with 11 vertices. First, we identify a maximum independent set S of size 3 with $S = \{C, M, N\}$. Then, we repeatedly employ ATS to collect as many independent sets of size 3 as possible in CL. Among those collected independent sets, we identify a maximal number of pairwise disjoint independent sets φ from CL, such as $\varphi = \{\{A, B, F\}, \{C, M, H\}, \{D, N, P\}\}$. Subsequently, we remove these vertices and the associated edges from the graph G and obtain a residual graph G' that contains the vertices J and L, as well as their connecting edge. We assign colors $1, 2, 3$ to the three extracted independent sets of φ and assign colors $4, 5$ to the vertices J and L of the residual graph G'. Hence, a coloring c with $sum(c) = 1 * 3 + 2 * 3 + 3 * 3 + 1 * 4 + 1 * 5 = 27$ is obtained.

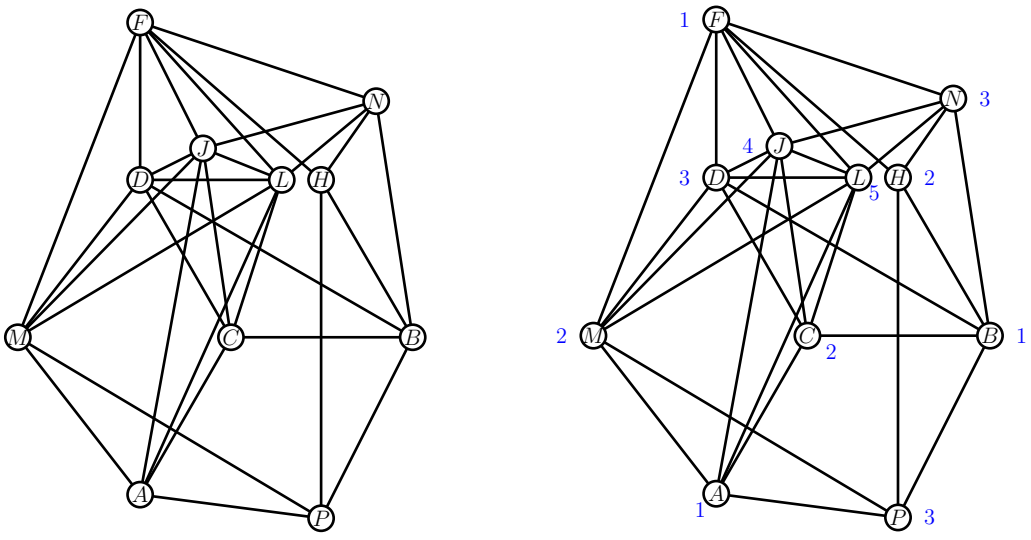


Fig. 3. Extracting independent sets of the extraction phase

2.3 Initial and intermediate MSCP coloring

In principle, our EBES algorithm can apply any MSCP algorithm to color the residual (generated at the end of the extraction phase) and intermediate graphs (generated by the backward expansion phase). Intuitively, it is advantageous to employ an effective MSCP algorithm to increase the overall performance of the approach. As mentioned above, we apply the state-of-the-art hybrid evolutionary search algorithm HESA introduced in [10]. HESA follows the general memetic framework that combines population-based evolutionary search and local optimization [5,18]. Starting from an initial population of legal colorings generated by the maximum independent set algorithm SBTS [9], HESA repeats a number of generations to improve the solutions. At each generation, HESA randomly selects two coloring to serve as parent solutions and uses two crossover operators to generate two offspring solutions. The offspring solutions are improved by an iterated double-phase tabu search procedure. Since HESA allows illegal colorings, a repairing procedure is employed to reestablish the solution feasibility when this situation occurs. More details about HESA can be found in [10].

2.4 Backward expansion phase

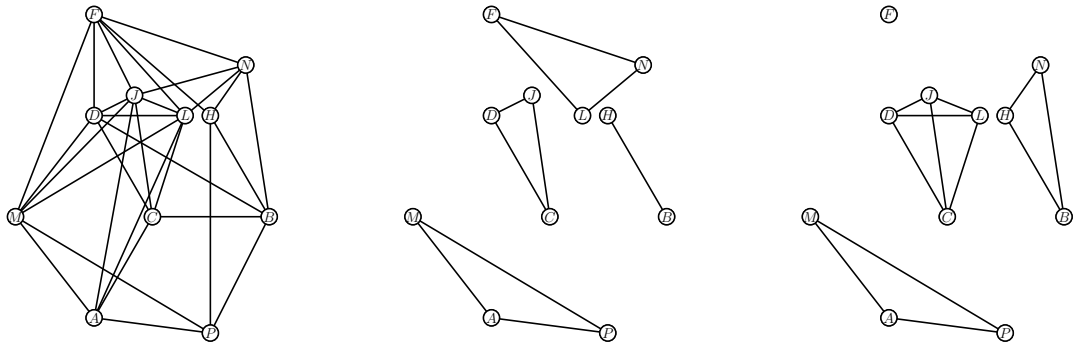
As explained in Section 2.1 (Step (3)), the backward expansion phase extends the current graph G_i $(i = m, m-1, ..., 1)$ to recover G_{i-1} by adding back the extracted independent sets of φ_i, i.e., $G_{i-1} = G_i + \varphi_i$ in order to traverse all intermediate graphs. For each extended intermediate graph G_{i-1}, we use the HESA algorithm to further improve the sum of colors of the solution by starting from the coloring of G_i extended with the newly added independent sets from φ_i. We also experimented with a simplified backward expansion strategy (denoted by a two-level strategy) that recovers G_0 from the residual graph G_i by adding back all the extracted independent sets at one time. This simplified strategy proved to perform well with respect to the successive backward expansion strategy and thus adopted for our experimental studies.

3 Lower bounds for the MSCP

Let $G' = (V, E')$ be a subgraph of the original graph $G = (V, E)$ with vertex set V and edge set E' $(E' \subset E)$. Obviously, the *chromatic sum* of G' is a lower bound for the *chromatic sum* of G. Hence, to calculate the lower bound of

the MSCP, we could try to find a subgraph G' whose *chromatic sum* can be efficiently and exactly computed.

One effective approach to obtain a subgraph G' is to decompose the set V of G into k pairwise disjoint cliques $S_1, S_2, \ldots, S_k$ such that $\forall i \neq j, S_i \bigcap S_j = \emptyset, \bigcup_i S_i = V$ [20,10]. Herein, the *chromatic sum* of the subgraph G' is computed by $\sum_{i=1}^{k} \frac{|S_i|(|S_i|+1)}{2}$ since each clique S_i with size $|S_i|$ needs exactly $|S_i|$ colors. As shown in Fig. 4, the graph G (Fig. 4(1)) has two different clique decompositions G'_1 and G'_2 (Fig. 4(2) and Fig. 4(3)), and the corresponding *chromatic sum* $\sum(G'_1) = 21$ and $\sum(G'_2) = 23$. One notices that the subgraph G'_2 of Fig. 4(3) has a larger *chromatic sum* than the subgraph G'_2 of Fig. 4(2), thereby Fig. 4(3) gives a tighter lower bound. Hence, the quality of the lower bounds depends on the clique decomposition.



$(1)\sum(G) = 27$ $(2)\sum(G'_1) = 21$ $(3)\sum(G'_2) = 23$

Fig. 4. Subgraphs obtained by clique decomposition

But finding a clique decomposition with a maximal *chromatic sum* is itself NP-hard [21]. In order to approximate this problem heuristically, in view of the fact that each color class of the complement graph $\bar{G}$ is a clique of G, one popular approach is to apply an MSCP algorithm to color the complement graph $\bar{G}$ and use the resulting color classes to define the clique decomposition of G [7,21,29].

In this work, we apply our EBES algorithm to color the complement graph $\bar{G}$ and uses the resulting clique decomposition for lower bound estimation. As we show in Section 4.2, this approach leads to tighter lower bounds for a number of benchmark graphs compared to the state-of-the-art results.

4 Experimental results

In order to evaluate the performance of the proposed EBES algorithm for the MSCP, we conduct experiments on a set of 35 large graphs with 450 to 4000

vertices. These graphs are from the well-known DIMACS and COLOR 2002-2004 graph coloring competitions [1], which are largely tested in the literature for both the vertex coloring and miminum sum coloring problems. The current known "theoretical" bounds of the chromatic sums of the tested instances are shown in Sections 4.1 and 4.2, which are calculated by the number of vertices and edges, as well as the chromatic number $\chi(G)$ of each graph [11,13,21,24].

Our experiments were performed on an Intel Xeon E5440 with 2.83 GHz and 8 GB RAM. To obtain our results, each instance was solved 30 times independently. The EBES algorithm stops when a fixed cutoff time (2 hours) is met. All the computational results were obtained with the same parameters, the threshold q is fixed to be 200. For the parameters required by the underlying HESA coloring algorithm, we adopted the settings from the original paper [10].

4.1 Computational results and comparisons for the upper bounds

We compare our EBES algorithm with four leading MSCP algorithms from the literature, which are reviewed in Section 1.

(1) The basic independent set extraction heuristic EXSCOL [26].
(2) The memetic algorithm MA presented in [21].
(3) The memetic algorithm MASC proposed in [8].
(4) The hybrid evolutionary search algorithm HESA [10] that is also used as our underlying coloring algorithm.

For the experimental platforms, EXSCOL was run on a 2.8 GHz computer with 2 GB RAM and iteratively extracts maximum independent sets until the null graph is reached. MA was run on an Intel Core 2 Duo T5450–1.66 GHz with 2 GB RAM and used a cutoff limit of 2 hours. MASC was run on a 2.7 GHz PC with 4 GB RAM and used 10^4 maximum iterations of its tabu search procedure and 50 maximum generations as its stopping condition. HESA was run on a computer with 2.83 GHz with 8 GB RAM and used a cutoff limit of 2 hours.

Table 1 reports the comparative results of our EBES algorithm together with EXSCOL, MASC, MA and HESA for the set of 35 graphs. Columns 2–3 show the current known theoretical upper bounds UB_t and the best-known upper bounds f^b_{UB} reported in the literature respectively. For each reference algorithm, columns f^*_{UB} and $Avg.$ indicate the best result in terms of sum of colors and the average result respectively, and column $t(min)$ gives the

[1] Available from `http://mat.gsia.cmu.edu/COLOR/instances.html`, `http://dimacs.rutgers.edu/Challenges/` and `http://mat.gsia.cmu.edu/COLOR04/`

Table 1. Comparisons of EBES with four state-of-art algorithms in terms of upper bounds of the MSCP on the set of 35 benchmark graphs

Graph			EXSCOL [26]			MASC [8]			MA [21]			HESA[10]			EBES		
Name	UB_t	f^b_{UB}	f^*_{UB}	*Avg.*	$t(min)$	f^*_{UB}	*Avg.*	$t(min)$	f^*_{UB}	*Avg.*	$t(min)$	f^*_{UB}	*Avg.*	$t(min)$	f^*_{UB}	*Avg.*	$t(min)$
fpsol2.i.1	12 150	3 403	–	–	–	3 403	3 403.0	8.7	3 403	3 403.0	4.0	3 403	3 403.0	8.5	3403	3403.0	13.3
fpsol2.i.2	6 990	1 668	–	–	–	1 668	1 668.0	5.7	1 668	1 668.0	3.0	1 668	1 668.0	0.8	1668	1668	18.6
inithx.i.1	19 571	3 676	–	–	–	3 676	3 676.0	7.6	3 676	3 679.6	5.0	3 676	3 676.0	1.3	3676	3676.0	25.5
inithx.i.2	10 320	2 050	–	–	–	2 050	2 050.0	4.4	2 050	2 053.7	10.0	2 050	2 050.0	1.3	2050	2050.0	12.4
inithx.i.3	9 936	1 986	–	–	–	1 986	1 986.0	1.8	1 986	1 986.0	2.0	1 986	1 986.0	0.0	1986	1986.0	9.5
wap05	23 077	13 656	*13 680*	13 718.4	21.0	*13 669*	13 677.8	3.3	–	–	–	13 656	13 677.8	1872.5	**13642**	13663.8	108.4
wap06	22 254	13 773	*13 778*	13 830.9	27.0	*13 776*	13 777.8	4.1	–	–	–	13 773	13 777.6	621.3	**13769**	13772.5	97.4
wap07	42 511	28 617	*28 629*	28 663.8	112.0	28 617	28 624.7	12.4	–	–	–	*29 154*	29 261.1	4.4	**28599**	28603.9	106.0
wap08	43 010	28 885	*28 896*	28 946.0	127.0	28 885	28 890.9	15.1	–	–	–	*29 460*	29 542.3	3.0	**28874**	28881.5	45.9
qg.order30	13 950	13 950	13 950	13 950.0	28.0	13 950	13 950.0	3.8	13 950	13 950.0	1.0	13 950	13 950.0	0.0	13 950	13 950.0	5.0
qg.order40	32 800	32 800	32 800	32 800.0	35.0	32 800	32 800.0	11.8	32 800	32 800.0	1.0	32 800	32 800.0	0.0	32 800	32 800.0	5.0
qg.order60	10 9800	10 9800	*110 925*	110 993.0	87.0	10 9800	10 9800.0	290.6	10 9800	10 9800.0	7.0	10 9800	10 9800.0	0.2	109800	109800.0	120.0
DSJC500.1	3 250	2 836	*2 850*	2 857.4	9.0	*2 841*	2 844.1	28.9	*2 897*	2 990.5	4.0	2 836	2 836.0	1997.9	**2835**	2837.1	65.4
DSJC500.5	12 250	10 886	*10 910*	10 918.2	11.0	*10 897*	10 905.8	73.3	*11 082*	11 398.3	42.0	10 886	10 891.5	4919.3	**10881**	10881.0	120.0
DSJC500.9	31 750	29 862	*29 912*	29 936.2	15.0	*29 896*	29 907.8	59.0	*29 995*	30 361.9	51.0	29 862	29 874.3	5513.3	**29857**	29871.4	108.9
															29856	29867.0	154.6
DSJC1000.1	10 500	8 991	*9 003*	9 017.9	28.0	*8 995*	9 000.5	70.7	*9 188*	9 667.1	31.0	8 991	8 996.5	5604.4	**8979**	8984.8	75.0
															8978	8981.2	137.4
DSJC1000.5	42 000	37 575	*37 598*	37 673.8	24.0	*37 594*	37 597.6	200.4	*38 421*	40 260.9	23.0	37 575	37 594.7	3090.3	**37567**	37570.9	61.5
DSJC1000.9	112 000	103 445	*103 464*	103 531.0	27.0	*103 464*	103 464.0	125.9	*105 234*	107 349.0	61.0	103 445	103 463.3	211.2	**103440**	103446.8	102.2
															103429	103443.1	138.3
DSJR500.1	3 250	2 156	–	–	–	–	–	–	*2 173*	2 253.1	4.0	2 156	2 170.7	99.8	**2155**	2172.4	106.9
DSJR500.1c	21 250	16 286	–	–	–	–	–	–	*16 311*	16 408.5	34.0	16 286	16 286.0	21.7	16 286	16 286.0	19.2
DSJR500.5	30 750	25 440	–	–	–	–	–	–	*25 630*	26 978.0	44.0	25 440	25 684.1	97.6	*25685*	25749.9	106.3
															25603	25747.1	128.4
flat1000_50_0	25 500	25 500	25 500	25 500.0	9.0	25 500	25 500.0	0.1	25 500	25 500.0	28.0	25 500	25 500.0	0.3	25 500	25 500.0	0.5
flat1000_60_0	30 500	30 100	30 100	30 100.0	11.0	30 100	30 100.0	114.6	30 100	30 100.0	16.0	30 100	30 100.0	2.7	30 100	30 100.0	0.5
flat1000_76_0	38 500	37 164	*37 167*	37 213.2	19.0	*37 167*	37 167.0	1.1	*38 213*	39 722.7	8.0	37 164	37 165.9	2237.0	**37151**	37164.6	119.5
le450_15a	3 600	2 632	2 632	2 641.9	5.0	*2 706*	2 742.6	41.3	*2 681*	2 733.1	19.0	*2 634*	2 648.4	91.5	**2626**	2626.1	50.8
le450_15b	3 600	2 632	*2 642*	2 643.4	7.0	*2 724*	2 756.2	40.3	*2 690*	2 730.6	19.0	2 632	2 656.5	89.9	*2635*	2636.7	61.4
le450_15c	3 600	3 487	*3 866*	3 868.9	6.0	*3 491*	3 491.0	45.3	*3 943*	4 048.4	6.0	3 487	3 792.4	86.7	*3852*	3852.0	7.0
le450_15d	3 600	3 505	*3 921*	3 928.5	5.0	*3 506*	3 511.8	59.8	*3 926*	4 032.4	3.0	3 505	3 883.1	82.7	*3887*	3898.9	32.8
le450_25a	5 850	3 153	3 153	3 159.4	7.0	*3 166*	3 176.8	39.2	*3 178*	3 204.3	5.0	*3 157*	3 166.7	88.5	**3152**	3153.6	61.4
le450_25b	5 850	3 365	*3 366*	3 371.9	6.0	*3 366*	3 375.1	40.3	*3 379*	3 416.2	7.0	3 365	3 375.2	88.6	**3356**	3358.8	76.2
le450_25c	5 850	4 515	4 515	4 525.4	8.0	*4 700*	4 773.3	75.3	*4 648*	4 700.7	16.0	*4 553*	4 583.8	84.8	**4501**	4503.6	42.5
le450_25d	5 850	4 544	4 544	4 550.0	7.0	*4 722*	4 805.7	63.4	*4 696*	4 740.3	3.0	*4 569*	4 607.6	92.4	**4532**	4538.0	65.4
latin_sqr_10	44 550	41 444	*42 223*	42 392.7	4.0	41 444	41 481.5	101.2	–	–	–	*41 492*	41 672.8	98.3	*41591*	41642.5	115.5
															41470	41547.5	272.2
C2000.5	149 000	132 483	*132 515*	132 682.0	656.0	–	–	–	–	–	–	132 483	132 513.9	161.8	**132478**	132478.0	131.9
C4000.5	544 000	473 234	473 234	473 211.0	2588.0	–	–	–	–	–	–	*513457*	514 639.0	75.3	**473205**	473212.2	86.6
Suc#/Total#			9/27			13/30			10/28			26/35			30/35		

run time in minutes that is only provided for indicative purposes. The last row *Suc#/Total#* shows the number of graphs for which an algorithm can achieve the best-known result over the total number of the tested graphs. The bold entries, italic entries and "–" marks in the table indicate respectively that an algorithm improves the best-known results, fails to reach the best-known results and does not report results on the tested graphs respectively. Note that when the cutoff limit of 2 hours is prolonged to 5 hours, the upper bounds on five instances (DSJC500.9, DSJC1000.1, DSJC1000.9, DSJR500.5 and latin_sqr_10) can be improved, which are also listed in Table 1 (shown in the next line of the result for each of these results). Besides, since the extraction phase is quite time-consuming for C2000.5 and C4000.5, we just set 2 hours for the cutoff time of EBES while excluding the extraction phase.

From Table 1, we can make the following observations. (Notice that for this study on upper bounds, smaller values correspond to better results.)

For the upper bounds of the considered large graphs, the first three reference algorithms (EXSCOL, MASC and MA) report results on a subset of graphs, the most recent HESA algorithm and the proposed EBES algorithm report results for all 35 graphs. EXSCOL, MASC, MA, HESA and EBES find the 9/27, 13/30, 10/28, 26/35 and 30/35 best-known solutions respectively. In particular, our EBES algorithm can improve the best-known results for 19 graphs (in bold, representing 54% of the tested set). Besides, we notice that the computational upper bounds are always (much) better than the current known theoretical upper bounds UB_t except for four instances (underlined) for which the computational and theoretical bound match perfectly.

Finally, since the EBES algorithm uses EXSCOL (extraction phase) and HESA (initial and intermediate graph coloring), it is interesting to compare the results of EBES against those of EXSCOL and HESA alone. When comparing EBES and EXSCOL, we notice that EBES finds a better or equal result for all the instances. This provides a very favorable indicator concerning the benefit of the backward expansion phase which is missing in EXSCOL. When comparing EBES and HESA, we observe that the results of EBES are equal or better for 30 out of 35 graphs. This shows that the mixed strategy of EBES combining independent sets extraction and backward expansion is highly effective.

4.2 Computational results and comparisons for the lower bounds

In this section, we assess our EBES algorithm for computing MSCP lower bounds with respect to three state-of-art algorithms EXCLIQUE [29], MA [21] and HESA [10], which reported the best-known lower bounds for the

tested graphs. Recall that MA was run on an Intel Core 2 Duo T5450–1.66 GHz with 2 GB RAM, HESA was run on a processor with 2.83 GHz and 8 GB RAM, both with a timeout limit of 2 hours. EXCLIQUE is a variant of EXSCOL for computing the lower bounds of the MSCP, which was run on a 2.8 GHz computer with 2 GB RAM and iteratively extracts maximum independent sets until the null graph is reached. Like in Section 4.1 for the upper bound, the cutoff time for the EBES algorithm is set to 2 hours (for C2000.5 and C4000.5, this excludes the running time of the extraction phase).

Table 2 reports the comparative results of EBES with respect to EXCLIQUE, MA and HESA for the lower bounds of the 35 test graphs. Like Table 1, columns 2–3 show for each graph the theoretical lower bounds LB_t and the best-known lower bounds f_{LB}^b reported in the literature. Columns f_{LB}^*, $Avg.$ and $t(min)$ of each compared algorithm report the best, average lower bound and the running time in minutes respectively. The last row $Suc\#/Total\#$ presents the number of cases where an algorithm can achieve the best-known result over the total number of the tested graphs. The bold entries, italic entries and "–" marks in the table show that each algorithm improves the best-known lower bounds, fails to reach the best-known lower bounds and does not report the lower bounds on the tested graphs respectively. Notice that larger bounds are better for this comparison of lower bounds.

We observe from Table 2 that the reference algorithms EXCLIQUE, MA and HESA can match the best lower bounds for 15/29 graphs, 11/28 graphs and 27/35 graphs respectively. On the other hand, our EBES algorithm can match or improve 32/35 best-known results, and fail to do so for only 3 instances. In particular, EBES improves the best lower bounds for 12 graphs (in bold).

Furthermore, one notices that the computational lower bounds, in particular those of our EBES algorithm, are (much) better than the theoretical lower bounds for all the tested instances, indicating clearly the usefulness of various heuristic approaches for lower bound approximation. Comparing the results of Tables 1 and 2 shows that the upper and lower bounds match for 8 instances (fpsol2.i.1, fpsol2.i.2, inithx.i.1, inithx.i.2, inithx.i.3, qg.order30, qg.order40 and qg.order60), proving the optimal chromatic sum for these graphs. For the remaining instances, we observe large gaps between the best upper and lower bounds in many cases, indicating that there remains considerable room for further improvement.

When comparing EBES against EXCLIQUE, one notices that EBES dominates EXCLIQUE for the tested graphs. Comparing EBES against HESA shows that EBES finds better or equal results for 32 out of the 35 tested graphs. Hence, this experiment demonstrates the efficiency of the EBES algorithm for computing high-quality lower bounds of the MSCP.

Table 2. Comparisons of EBES with three state-of-art algorithms in terms of lower bounds of the MSCP on the set of 35 benchmark graphs

Graph			EXCLIQUE [29]			MA [21]			HESA [10]			EBES		
Name	LB_t	f^b_{LB}	f^*_{LB}	Avg.	t(min)	f^*_{LB}	Avg.	t(min)	f^*_{LB}	Avg.	t(min)	f^*_{LB}	Avg.	t(min)
fpsol2.i.1	2 576	3 403	3 403	3 403.0	44.6	3 403	3 403.0	63.0	3 403	3 403.0	13.1	3403	3403.0	10.0
fpsol2.i.2	886	1 668	–	–	–	1 668	1 668.0	28.0	1 668	1 668.0	8.9	1668	1668.0	10.0
inithx.i.1	2 295	3 676	3 676	3 676.0	61.5	3 676	3 616.0	107.0	3 676	3 675.3	82.1	3676	3676.0	10.0
inithx.i.2	1 110	2 050	–	–	–	2 050	1 989.2	16.0	2 050	2 050.0	21.5	2050	2050.0	10.0
inithx.i.3	1 086	1 986	–	–	–	1 986	1 961.8	89.0	1 986	1 986.0	18.8	1986	1986.0	10.0
wap05	2 130	12 449	*12 428*	12 339.3	104.7	–	–	–	12 449	12 438.9	57.9	12449	12448.9	61.6
wap06	1 982	12 454	*12 393*	12 348.8	90.3	–	–	–	12 454	12 431.6	53.2	12454	12452.9	44.8
wap07	2 844	24 800	*24 339*	24 263.8	139.3	–	–	–	24 800	24 783.6	72.6	**24802**	24791.6	14.2
wap08	2 860	25 283	*24 791*	24 681.1	152.1	–	–	–	25 283	25 263.4	65.6	*25274*	25260.5	78
qg.order30	1 335	13 950	13 950	13 950.0	7.9	13 950	13 950.0	0.0	13 950	13 950.0	0.1	13950	13950.0	10.0
qg.order40	2 380	32 800	32 800	32 800.0	23.0	32 800	32 800.0	1.0	32 800	32 800.0	0.3	32800	32800.0	10.0
qg.order60	5 370	109 800	109 800	109 800.0	125.1	109 800	109 800.0	11.0	109 800	109 800.0	2.7	109 800	109800.0	10.0
DSJC500.1	566	1 250	1 250	1 246.6	21.2	*1 241*	1 214.9	22.0	1 250	1 243.4	62.0	**1253**	1251.4	35.4
DSJC500.5	1 628	2 923	*2 921*	2 902.6	1.0	*2 868*	2 797.7	50.0	2 923	2 896.0	65.6	**2925**	2915.9	67.1
DSJC500.9	8 375	11 053	*10 881*	10 734.5	4.6	*10 759*	10 443.8	16.0	11 053	10 950.1	68.6	*10999*	10937.8	73.4
DSJC1000.1	1 190	2 762	2 762	2 758.6	86.6	*2 707*	2 651.2	98.0	*2 719*	2 707.6	66.0	**2779**	2774.1	48.6
DSJC1000.5	4 403	6 708	6 708	6 665.9	2.6	*6 534*	6 182.5	34.0	*6 582*	6 541.3	44.6	**6725**	6721.1	52.4
DSJC1000.9	25 753	26 557	26 557	26 300.3	45.7	*26 157*	24 572.0	73.0	*26 296*	26 150.3	51.8	**26706**	26626.9	70.4
DSJR500.1	566	2 069	–	–	–	*2 061*	2 052.9	21.0	2 069	2 069.0	4.2	2069	2069.0	11.3
DSJR500.1c	3 986	15 398	–	–	–	*15 025*	14 443.9	11.0	15 398	15 212.4	65.0	*15268*	15216.0	61.4
DSJR500.5	7 881	22 974	–	–	–	*22 728*	22 075.0	30.0	22 974	22 656.7	32.0	**23609**	23065.2	46.4
flat1000_50_0	2 225	6 601	6 601	6 571.8	2.0	*6 433*	6 121.5	63.0	*6 476*	6 452.1	51.5	**6627**	6622.4	78.2
flat1000_60_0	2 770	6 640	6 640	6 600.5	6.9	*6 402*	6 047.7	42.0	*6 491*	6 466.5	46.2	**6648**	6636.4	31.9
flat1000_76_0	3 850	6 632	6 632	6 583.2	1.6	*6 330*	6 074.6	66.0	*6 509*	6 482.8	34.1	**6662**	6656.7	49.2
le450_15a	555	2 331	*2 329*	2 313.7	4.2	*2 329*	2 324.3	3.0	2 331	2 331.0	23.3	2331	2331.0	10.0
le450_15b	555	2 348	*2 343*	2 315.7	10.0	2 348	2 335.0	2.0	2 348	2 348.0	4.8	2348	2347.8	45.1
le450_15c	555	2 610	*2 591*	2 545.3	3.1	*2 593*	2 569.1	6.0	2 610	2 606.6	57.3	2610	2606.4	73.5
le450_15d	555	2 628	*2 610*	2 572.4	2.9	*2 622*	2 587.2	24.0	2 628	2 627.1	54.9	2628	2628.0	65.6
le450_25a	750	3 003	*2 997*	2 964.4	16.1	3 003	3 000.4	5.0	3 003	3 003.0	1.2	3003	3003.0	10.1
le450_25b	750	3 305	3 305	3 304.1	25.8	3 305	3 304.1	2.0	3 305	3 305.0	1.0	3305	3305.0	10.0
le450_25c	750	3 657	*3 619*	3 597.1	11.5	*3 638*	3 617.0	31.0	3 657	3 656.9	41.7	3657	3657.0	12.1
le450_25d	750	3 698	*3 684*	3 627.4	14.2	*3 697*	3 683.2	20.0	3 698	3 698.0	8.3	3698	3698.0	15.6
latin_sqr_10	5 653	40 950	40 950	40 950.0	0.3	–	–	–	40 950	40 950.0	0.0	40 950	40 950.0	10.0
C2000.5	12 878	15 091	15 091	15 077.6	66.6	–	–	–	*14 498*	14 442.9	24.2	**15106**	15099.3	48.0
C4000.5	40 585	33 033	33 033	33 018.8	240.2	–	–	–	*31 525*	31 413.3	66.5	**33063**	33058.5	17.6
Suc#/Total#			15/29			11/28			27/35			32/35		

5 Analysis of EBES

5.1 Analysis of backward expansion

In this section, we present an experiment to assess the usefulness of the backward expansion phase. For this, we compare EBES with a variant EBES$^-$ where the backward expansion phase of EBES is disabled. In other words, EBES$^-$ uses only the two first phases of the EBES algorithm, i.e., independent set extraction and initial coloring phases (see Section 2.1). For this experiment, EBES$^-$ was run exactly under the same condition with the same parameter setting as for EBES.

Table 3 summarizes the upper and lower bounds obtained by EBES$^-$ and EBES. Columns 2-3 recall the best-known upper and lower bounds. The following columns present for each graph the best upper bound, average upper bound, average time, best lower bound, average lower bound and average time achieved by EBES$^-$ and EBES respectively. First, if we compare these results with the best-known results, we observe that EBES$^-$ finds 7 equal and 13 improved best upper bounds, 11 equal and 10 improved best lower bounds, while EBES finds 11 equal and 19 improved best upper bounds, 20 equal and 12 improved best lower bounds. Hence, both EBES$^-$ and EBES have a quite competitive performance on the tested large graphs. Second, if we contrast the results of EBES$^-$ with those of EBES, we observe that EBES obtains 21 better upper bounds, 17 better lower bounds out of 35 instances while the reverse is true only for 7 upper bounds and 7 lower bounds. This experiment confirm that the backward expansion phase leads to valuable improvements of the computational results of the EBES algorithm.

5.2 Influence of the expansion strategy

As mentioned in Section 2.4, EBES employs a two-level strategy to add back all the extracted independent sets as new color classes of colorings at one time. In order to show the interest of this strategy, we perform an additional experiment to compare it with two other expansion strategies. The first compared strategy regains the extratced independent sets of the same size according to the extraction order (i.e., from the largest to the smallest, denoted by largest first strategy). The second compared strategy recovers the independent sets of the same size according to the reverse of extraction order (i.e., from the smallest to the largest, denoted by smallest first strategy).

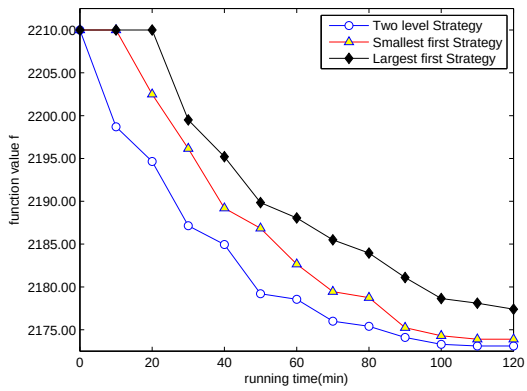
For this experiment, we created two EBES variants where only the

Table 3
Comparisons of upper and lower bounds obtained by EBES with and without the backward expansion phase on the set of 35 benchmark graphs

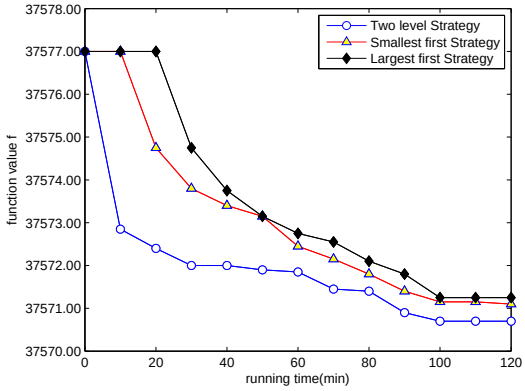
Graph			EBES$^-$						EBES					
Name	f^b_{UB}	f^b_{LB}	Upper Bound			Lower Bound			Upper Bound			Lower Bound		
			f^*_{UB}	Avg.	t	f^*_{LB}	Avg.	t	f^*_{UB}	Avg.	t	f^*_{LB}	Avg.	t
fpsol2.i.1	3403	3403	3987	3987.0	0.0	3403	3403.0	0.8	**3403**	3403.0	13.3	3403	3403.0	10.0
fpsol2.i.2	1668	1668	2136	2136.0	0.0	1668	1668.0	1.6	**1668**	1668.0	18.6	1668	1668.0	10.0
inithx.i.1	3676	3676	5968	5968.0	0	3676	3676.0	17.7	**3676**	3676.0	25.5	3676	3676.0	10.0
inithx.i.2	2050	2050	3193	3193.0	0.0	2050	2050.0	0.5	**2050**	2050.0	12.4	2050	2050.0	10.0
inithx.i.3	1986	1986	2990	2990.0	0.0	1986	1986.0	0.5	**1986**	1986.0	9.5	1986	1986	10
wap05	13656	12449	13656	13666.8	48.4	12411	12411.0	0.3	**13642**	13663.8	108.4	**12449**	12448.9	61.6
wap06	13773	12454	13769	13775.3	34.3	12452	12452.0	0	13769	13772.5	97.4	**12454**	12452.9	44.8
wap07	28617	24800	28605	28618.0	42.8	24791	24791.0	1.2	**28599**	28603.9	106.0	**24802**	24791.6	14.2
wap08	28885	25283	**28857**	28869.8	44.6	25255	25255	1.1	28874	28881.5	45.9	**25274**	25260.5	78.0
qg.order30	13950	13950	13950	13950.0	0.0	13950	13950.0	0.0	13950	13950.0	5.0	13950	13950.0	10.0
qg.order40	32800	32800	32800	32800.0	0.0	32800	32800.0	0.0	32800	32800.0	5.0	32800	32800.0	10.0
qg.order60	109800	109800	109800	109800.0	0.0	109800	109800.0	0.0	109800	109800.0	120.0	109800	109800.0	10.0
DSJC500.1	2836	1250	2836	2839.0	20.9	1251	1251.0	2.9	**2835**	2837.1	65.4	**1253**	1251.4	35.4
DSJC500.5	10886	2923	**10874**	10896.6	61.2	**2927**	2917.1	62.9	10881	10881	120.0	2925	2915.9	67.1
DSJC500.9	29862	11053	29912	29912.0	0.0	10944	10905.0	59.4	**29856**	29867.0	154.6	**10999**	10937.8	73.4
DSJC1000.1	8991	2762	8980	8987.6	44.0	2773	2773.0	4.9	**8978**	8981.2	137.4	**2779**	2774.1	48.6
DSJC1000.5	37575	6708	37571	37595.3	45.8	**6732**	6719.2	69.5	**37567**	37570.9	61.5	6725	6721.1	52.4
DSJC1000.9	103445	26557	103462	103462.0	0.2	26641	26607.4	62.6	**103429**	103443.1	138.3	**26709**	26626.9	70.4
DSJR500.1	2156	2069	2220	2220.0	0.0	2045	2045.0	0.4	**2155**	2172.4	106.9	**2069**	2069.0	11.3
DSJR500.1c	16286	15398	16502	16502.0	0.1	14968	14968.0	0.1	**16286**	16286.0	19.2	**15268**	15216.0	61.4
DSJR500.5	25440	22974	26295	26295.0	1.6	22968	22968.0	0.0	**25603**	25747.1	128.4	**23069**	23065.2	46.4
flat1000_50_0	25500	6601	25500	25500.0	0.0	**6635**	6623.9	67.1	25500	25500.0	0.5	6627	6622.4	78.2
flat1000_60_0	30100	6640	30100	30100.0	0.0	**6657**	6643.1	56.0	30100	30100.0	0.5	6648	6636.4	31.9
flat1000_76_0	37164	6632	**37139**	37154.8	49.3	**6670**	6659.2	70.4	37151	37164.6	119.5	6662	6656.7	49.2
le450_15a	2632	2331	2627	2627.0	2.8	2331	2331.0	0.3	**2626**	2626.1	50.8	2331	2331.0	10.0
le450_15b	2632	2348	2635	2636.3	24.9	2345	2345.0	0.3	2635	2636.7	61.4	**2348**	2347.8	45.1
le450_15c	3487	2610	**3850**	3857.0	9.4	2595	2595.0	0.2	3852	3852	7	**2610**	2606.4	73.5
le450_15d	3505	2628	3900	3901.4	8.9	2594	2594	0.4	**3887**	3898.9	32.8	**2628**	2628.0	65.6
le450_25a	3153	3003	**3150**	3153.2	28.1	2997	2997.0	0.4	3152	3153.6	61.4	**3003**	3003.0	10.1
le450_25b	3365	3305	3361	3361.2	10.6	3305	3305.0	0.3	**3356**	3358.8	76.2	3305	3305.0	10.0
le450_25c	4515	3657	4503	4503.9	12.1	3615	3615.0	0.2	**4501**	4503.6	42.5	**3657**	3657.0	12.1
le450_25d	4544	3698	**4534**	4536.8	19.7	3685	3685.0	0.2	4532	4538	65.4	**3698**	3698.0	15.6
latin_sqr_10	41444	40950	42193	42193.9	6.7	40950	40950.0	0.0	**41470**	41547.5	272.2	40950	40950.0	10.0
C2000.5	132483	15091	132510	132520.6	42	**15123**	15110.3	66.5	**132478**	132478.0	131.9	15106	15099.3	48.0
C4000.5	473234	33033	**473179**	473203.3	64.8	**33071**	33062.1	61.9	473205	473212.2	86.6	33063	33058.5	17.6

expansion strategy of EBES is changed. We ran each EBES variant 30 times with a cutoff time of 2 hours to solve a given graph. Fig. 5 presents the evolution profiles of the three compared strategies on four selected graphs

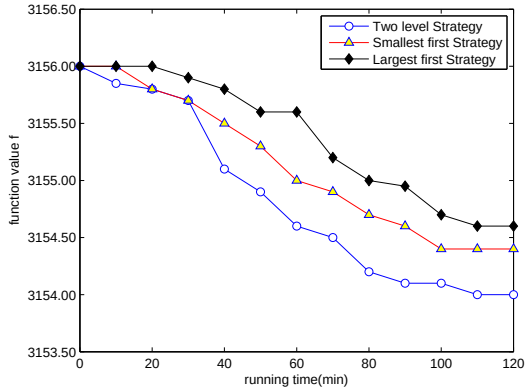
(DSJR500.1, DSJC1000.5, le450_25 and flat1000_76_0). The average best upper bounds are plotted against the running time. One observes that no strategy performs strictly better than the other expansion strategies. For DSJR500.1, DSJC1000.5 and le450_25, the two-level strategy has the best performance among the three expansion strategies. However, the smallest first strategy achieves the best result for flat1000_76_0 compared to the other two strategies. For the overall performance of these expansion strategies, this comparison shows that the two-level strategy performs globally best. Moreover, the two-level strategy has the advantage of simplicity and of being less-consuming. This justifies the choice of using this strategy within our proposed EBES algorithm for the backward expansion phase.



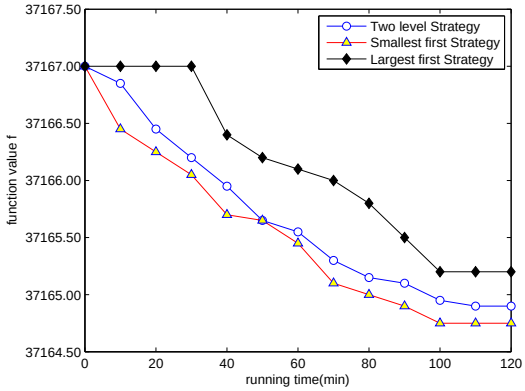
(1) DSJR500.1



(2)DSJC1000.5



(3) le450_25a



(4) flat1000_76_0

Fig. 5. Comparison between the two-level expansion strategy and two other expansion strategies

Table 4
Comparison of EBES with two state-of-the-art algorithms in terms of upper bounds of the MSCP on the set of 10 large random graphs

Graph	EXSCOL [29]			HESA [10]			EBES		
Name	f^*_{UB}	Avg.	t(min)	f^*_{UB}	Avg.	t(min)	f^*_{UB}	Avg.	t(min)
Random5000.05	89935	90155.5	662.4	90895	91171.2	287.8	**89896**	89989.2	755.3
Random5000.1	155911	156551.0	406.8	162959	163419.1	88.4	**155901**	156114.8	821.3
Random5000.2	285978	287151.6	216.4	304624	305154.4	110.4	**285777**	286340	466.1
Random5000.3	419015	426996.8	104.4	449564	450658.1	91.9	**418665**	*421543.4*	515.3
Random5000.5	754587	776007.5	89.6	778161	779763.8	158.5	**753835**	*763754.9*	592.5
Random6000.1	217540	217976.9	556.7	227008	227422.3	48.1	**217416**	217850.5	775.5
Random6000.2	399235	401060.2	344.9	425935	426730.3	58.7	**398051**	399907.2	686.8
Random8000.1	371070	371909.1	535.7	387122	388081.0	20	**371050**	371320.6	746.8
Random8000.5	1847799	1950911.2	127.0	1890040	1892547.9	359.5	**1846957**	*1847419.1*	556.9
Random10000.5	2819955	2946127.7	106.1	2833255	2834378.3	235.7	**2813352.0**	*2824099.8*	302.8

5.3 The effectiveness of the proposed approach on large random graphs

In this section, we study the behavior of the proposed EBES algorithm on large random graphs. Since the available benchmark graphs for the MSCP in the literature are limited to 4000 vertices, we generated 10 large random graphs with the number of vertices from 5000 to 10000 and the edge density from 0.05 to 0.5. We ran EXSCOL, HESA and the proposed EBES on these large graphs to calculate both upper and lower bounds of the MSCP. We used a cutoff limit of 15 hours for HESA and EBES (note that EXSCOL terminates automatically when the residual graph becomes empty, thus does not need a cutoff time). To obtain the comparative results, each instance was solved 30 times independently. The computational results in terms of upper and lower bounds are given in Tables 4 and 5.

From Table 4, we observe that the proposed EBES algorithm achieves the best upper bounds (in bold) for all the 10 large graphs compared to the reference algorithms EXSCOL and HESA (Notice that smaller bounds are better for this comparison of upper bounds). In particular, EBES improves on the results of HESA by more than 1% for each case. Besides, the average upper bounds of EXSCOL are improved by more than 1% for 4 out of 10 cases (in italic). Moreover, from Table 5, one notices that EBES performs the best among these three algorithms for each instance (in bold). Especially, EBES can improve the lower bounds of EXSCOL by more than 1% for each case, as well as the lower bounds of HESA on Random5000.5. This experiment shows that the proposed EBES algorithm is highly effective in solving the MSCP on large graphs.

Table 5
Comparisons of EBES with two state-of-the-art algorithms in terms of lower bounds of the MSCP on the set of 10 large random graphs

Graph	EXSCOL [29]			HESA [10]			EBES		
Name	f^*_{LB}	Avg.	t(min)	f^*_{LB}	Avg.	t(min)	f^*_{LB}	Avg.	t(min)
Random5000.05	12495	12437.6	159.9	13276	13264.96	172.8	**13295**	13280.66	243.5
Random5000.1	15321	15254.73	142.3	16123	16117.23	200.2	**16154**	16148.13	254.1
Random5000.2	19632	19537.2	125.3	21205	21191.4	79	**21243**	21223.5	286.6
Random5000.3	24041	23912.96	145.5	26362	26346.36	68.1	**26400**	26382.13	155.3
Random5000.5	39563	39493.8	12.9	39695	39676.83	39.6	**40360**	39802.4	121.1
Random6000.1	17985	17534.5	180.1	19669	19655.16	111	**19700**	19679.6	307.9
Random6000.2	23044	22548.43	98.8	25934	25916.86	125.6	**25997**	25959.5	202.3
Random8000.1	25567	25372	193.2	27143	27116	155.4	**27200**	27174.8	240.6
Random8000.5	65436	63390.26	172.8	66826	66780	81.8	**66950**	66866.33	196.8
Random10000.5	82464	82054.7	154.5	84840	84802.36	93.2	**84877**	84861.33	170.4

6 Conclusion

Minimum sum coloring is a computationally challenging problem both in theory and in practice. This is particular true when handling large graphs with more than several hundreds of vertices. Inspired by the "reduce-and-solve" principle and studies on the vertex coloring problem, we investigated for the first time the extraction and backward expansion approach for computing upper and lower bounds of the MSCP, especially for large graphs.

We assessed the performance of EBES on 35 large benchmark graphs from the well-known DIMACS and COLOR 2002-2004 competitions. EBES can match the best-known upper and lower bounds for most graphs except for 5 and 3 graphs respectively. In particular, EBES can find 19 improved upper bounds and 12 improved lower bounds. These improved bounds can serve as reference values for assessment of existing and new MSCP algorithms.

This study demonstrates that the "reduce-and-solve" principle is a powerful idea that can help design effective search algorithms for the MSCP. It is worthy of investigating similar ideas for solving other combinatorial search problems.

Acknowledgments

We are grateful to the reviewers for their insightful comments which helped us to improve the paper. This work was supported by the National Natural Science Foundation of China (Grants No. 61602196, 61472147, 61772219, 71401059, 71771099).

References

[1] U. Benlic, J.K. Hao, 2012. A study of breakout local search for the minimum sum coloring problem. In: L. Bui, Y. Ong, N. Hoai, H. Ishibuchi, P. Suganthan (eds.), Simulated Evolution and Learning, vol. 7673 of Lecture Notes in Computer Science, Springer, Berlin / Heidelberg, Germany, pp. 128–137.

[2] H. Bouziri, M. Jouini, 2010. A tabu search approach for the sum coloring problem. Electronic Notes in Discrete Mathematics 36: 915–922.

[3] P. Galinier, J.P. Hamiez, J.K. Hao, D.C. Porumbel, 2012. Recent advances in graph vertex coloring. In I. Zelinka, A. Abraham, V. Snasel (Eds.) Handbook of Optimization, Springer.

[4] M.R. Garey, D.S. Johnson, 1979. Computers and intractability: A guide to the theory of NP-completeness. W.H. Freeman and Company, San Francisco.

[5] J.K. Hao, 2012. Memetic algorithms in discrete optimization. In F. Neri, C. Cotta, P. Moscato (Eds.) Handbook of Memetic Algorithms. Studies in Computational Intelligence 379, Chapter 6, pages 73–94.

[6] J.K. Hao, Q. Wu, 2012. Improving the extraction and expansion method for large graph coloring. Discrete Applied Mathematics, 160(16), 2397-2407.

[7] A. Helmar, M. Chiarandini, 2011. A local search heuristic for chromatic sum. In: L. Di Gaspero, A. Schaerf, T. Stützle (eds.). Proceedings of the 9th Metaheuristics International Conference, pages 161–170.

[8] Y. Jin, J.K. Hao, J.P. Hamiez, 2014. A memetic algorithm for the minimum sum coloring problem. Computers & Operations Research 43: 318–327.

[9] Y. Jin, J.K. Hao, 2015. General swap-based multiple neighborhood tabu search for finding maximum independent set. Engineering Applications of Artificial Intelligence 37: 20–33.

[10] Y. Jin, J.K. Hao, 2016. Hybrid evolutionary search for the minimum sum coloring problem of graphs. Information Sciences 352: 15–34.

[11] Y. Jin, J.P. Hamiez, J.K. Hao, 2017. Algorithms for the minimum sum coloring problem: a review. Artificial Intelligence Review 2017, 47(3): 367–394.

[12] D.S. Johnson, C.R. Aragon, L.A. McGeoch, C. Schevon, 1991. Optimization by simulated annealing: an experimental evaluation; part II, graph coloring and number partitioning. Operations Research 39(3), 378–406.

[13] Z. Kokosiński, K. Kwarciany, 2007. On sum coloring of graphs with parallel genetic algorithms. In: B. Beliczynski, A. Dzielinski, M. Iwanowski, B. Ribeiro (eds.), Adaptive and Natural Computing Algorithms, vol. 4431 of Lecture Notes in Computer Science, Springer, Berlin / Heidelberg, Germany, pp. 211–219.

[14] E. Kubicka, A.J. Schwenk, 1989. An introduction to chromatic sums. In Proceedings of the 17th ACM Annual Computer Science Conference, pages 39–45, New York, NY, USA.

[15] C. Lecat, C.M. Li, C. Lucet, Y. Li, 2015. Exact methods for the minimum sum coloring problem. https://hal.archives-ouvertes.fr/hal-01326666/

[16] Y. Li, C. Lucet, A. Moukrim, K. Sghiouer, 2009. Greedy algorithms for the minimum sum coloring problem. In: Logistique et Transports Conference, Sousse, Yunisia, pp.LT–027.

[17] M. Malafiejski, 2004. Sum coloring of graphs. In: M. Kubale (ed.), Graph Colorings, vol. 352 of Contemporary mathematics, American Mathematical Society, New Providence (Rhode Island) USA, pp. 55–65.

[18] P. Moscato, C. Cotta, 2003. A gentle introduction to memetic algorithms. In F. Glover and G. Kochenberger (Eds.), Handbook of Metaheuristics, Kluwer, Norwell, Massachusetts, USA.

[19] M. Minot, S. N. Ndiaye, C. Solnon, 2016. Using CP and ILP with tree decomposition to solve the sum colouring problem. Doctoral program of CP 2016, September 2016, Toulouse, France. http://cp2016.a4cp.org/

[20] A. Moukrim, K. Sghiouer, C. Lucet, Y. Li, 2010. Lower bounds for the minimal sum coloring problem. Electronic Notes in Discrete Mathematics 36: 663–670.

[21] A. Moukrim, K. Sghiouer, C. Lucet, Y. Li, 2014. Upper and lower bounds for the minimum sum coloring problem, https://www.hds.utc.fr/ moukrim/dokuwiki/doku.php?id=en:mscp

[22] C. Morgenstern, 1996. Distributed coloration neighborhood search. Discrete Mathematics and Theoretical Computer Science 26, 335–358.

[23] K.J. Supowit, 1987. Finding a maximum planar subset of a set of nets in a channel. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 6(1): 93–94.

[24] C. Thomassen, P. Erds, Y. Alavi, P. Malde, A. Schwenk, 1989. Tight bounds on the chromatic sum of a connected graph. Journal of Graph Theory, 13: 353C-357.

[25] Y. Wang, J.K. Hao, F. Glover, Z. Lü, 2013. Solving the minimum sum coloring problem via binary quadratic programming. CoRR abs/1304.5876.

[26] Q. Wu, J.K. Hao, 2012. An effective heuristic algorithm for sum coloring of graphs. Computers & Operations Research 39(7): 1593–1600.

[27] Q. Wu, J.K. Hao, 2012. Coloring large graphs based on independent set extraction. Computers & Operations Research, 39(2), 283–290.

[28] Q. Wu, J.K. Hao, 2013. An adaptive multistart tabu search approach to solve the maximum clique problem. Journal of Combinatorial Optimization 26(1): 86–108.

[29] Q. Wu, J.K. Hao, 2013. Improved lower bounds for sum coloring via clique decomposition. CoRR abs/1303.6761.