Improving the Extraction and Expansion Method for Large Graph Coloring

Jin-Kao Hao and Qinghua Wu

LERIA, Université d'Angers, 2 Bd Lavoisier, 49045 Angers Cedex 01, France

Accepted to *Discrete Applied Mathematics*, 18 June 2012

Abstract

Graph coloring is one of the most studied combinatorial optimization problems. This paper presents an improved extraction and expansion method (IE^2COL) initially introduced in [47]. IE^2COL employs a forward independent set extraction strategy to reduce the initial graph G. From the reduced graph, IE^2COL triggers a backward coloring process which uses extracted independent sets as new color classes for intermediate subgraph coloring. The proposed method is assessed on 20 large benchmark graphs with 900 to 4000 vertices. Computational results show that it provides new upper bounds for 6 graphs and matches consistently the current best known results for 12 other graphs.

Keywords: graph coloring; graph k-coloring; independent set extraction; memetic coloring; progressive optimization.

1 Introduction

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. A subset I of V is an independent set if no two adjacent vertices belong to I. A legal k-coloring of G is a partition of V into k independent sets (color classes). The graph k-coloring problem is to find a legal k-coloring of G for a given k. The graph coloring problem is to determine the smallest integer k (the chromatic number $\chi(G)$) such that there exists a legal k-coloring of G. Notice that the graph coloring problem can be approximated by solving a series of k-coloring problems with increasing or decreasing k values [18].

Email addresses: hao@info.univ-angers.fr (Jin-Kao Hao), wu@info.univ-angers.fr (Qinghua Wu).

Graph k-coloring is a well-known NP-complete problem [21] and has a number of practical applications related to printed circuit testing [20], scheduling [31], register allocation [11], timetabling [5], frequency assignment [40] and bag rationalization [22]. In the general case, exact solution methods can be used only to solve problems of relatively small size. As a matter of fact, there are graphs with as few as 125 vertices that can not be solved optimally even by using the current best exact algorithms [29,34]. For larger graphs, heuristics and meta-heuristics are usually preferred to find approximate solutions. Comprehensive surveys of the most significant coloring methods can be found in [18,34].

There are a large number of heuristic approaches for graph coloring: greedy construction (DSATUR [3], RLF [31]), tabu search [2,12,24,27,37], iterated local search and variable neighborhood search [1,8], simulated and quantum annealing [7,29,41], variable space search [28], scatter search [25], multiagent fusion search [43], ant colony optimization [10,4,36,44], and evolutionary or population based hybrid search [13,17,19,23,32,33,38]. These coloring algorithms are based on diverse solution strategies and have led to continually improved results. Among these algorithms, population-based heuristics are certainly among the most competitive approaches. Nevertheless, large graphs with more than 900 vertices always represent a real challenge for any existing coloring algorithm.

A basic approach to deal with large graphs is to apply the general principle of "reduce-and-solve". Before the coloring process, this approach first removes, during a preprocessing step, some large independent sets from the original graph to obtain a reduced graph (called residual graph). Since each independent set can form a color class, to obtain a coloring of the initial graph, it suffices to find a legal coloring for the residual graph. This approach were explored with success in early studies like [15,27,29,35].

Very recently, an improvement has been proposed to enhance this basic independent set extraction approach [46]. The extraction phase was enhanced by removing at a time a *maximal collection* of disjoint independent sets of maximal size instead of only *one* independent set. The resulting EXTRACOL algorithm has obtained new improved colorings for several large and very large graphs (DSJC1000.9, C2000.5, C2000.9, C4000.5). However, extracting independent sets as a preprocessing technique suffers an inevitable limitation. Actually, if an independent set is wrongly extracted such that it is not part of the optimal coloring, the mistake can never be repaired. To remedy this difficulty, the work of [47] introduced an expansion phase which allows the coloring process to reconsider each extracted independent set on an one-by-one basis. The resulting E2COL algorithm has improved the best-known results for two very large graphs (C2000.5 and C4000.5).

This paper further extends these previous studies by proposing additional strategies, leading to the improved extraction and expansion algorithm (IE^{2}COL). We report experimental studies of IE^{2}COL on the set of 20 largest and most challenging benchmark graphs (with 900 to 4000 vertices) from the DIMACS and COLOR02/03/04 competitions. These results show that the proposed algorithm obtains new upper bounds for 6 graphs (flat1000_76_0, C2000.5, C4000.5, C2000.9, WAP04, WAP07) and matches consistently the current best-known results for 12 other graphs.

Section 2 presents the proposed algorithm. Section 3 is dedicated to extensive computational evaluations and comparisons. Section 4 investigates some key components of the proposed approach, followed by the concluding section.

2 Improved extraction and expansion coloring (IE^{2}COL)

2.1 General IE^{2}COL procedure

The proposed IE^{2}COL algorithm is based on and extends the basic extraction and expansion method of [47] and can be summarized by the following general procedure composed of three phases.

(1) The *extraction* phase simplifies the initial graph G by removing iteratively large independent sets (as well as the corresponding edges) from the original graph. To be effective, each iteration removes a collection of disjoint independent sets of the same size (the largest possible) according to the method developed in [46]. This phase stops when the residual graph contains no more than a fixed number of q vertices. The independent set extraction method is discussed in Section 2.2.
(2) The *initial coloring* phase applies a graph coloring algorithm (the memetic algorithm presented in [32]) to the residual graph G_z to determine a $(k-t)$-coloring where t is the number of extracted independent sets. If a legal $(k-t)$-coloring $C = \{c_1, ..., c_{k-t}\}$ for G_z is found, then C plus the t independent sets extracted during the phase 1 constitutes a legal k-coloring of the initial graph G, return this k-coloring and stop. Otherwise, continue to phase 3 to trigger the expansion and backward coloring phase. The memetic coloring algorithm applied to G_z and intermediate subgraphs (phase 3) is discussed in Section 2.3.
(3) The *expansion and backward coloring* phase extends the current subgraph G' by adding back some extracted independent sets S to obtain an extended subgraph G''. Then the coloring algorithm is run on G'' by starting from the current coloring of G' extended with the independent sets of S as new color classes. Once again, if a legal coloring is found for the subgraph G'', this coloring plus the remaining independent sets forms

a legal k-coloring of the initial graph G and the whole procedure stops. Otherwise, one repeats this expansion and backward coloring phase until no more independent set is left or a legal coloring is found for the current subgraph under consideration. Possible strategies to select independent sets for expansion are discussed in Sections 2.4 and 4.2.

The proposed IE^{2}COL algorithm, designed for the graph k-coloring problem, implements this general approach and is described in Alg. 1. In what follows, we show how the main components of IE^{2}COL are implemented.

2.2 Extraction of independent sets

The main goal of the extraction phase (Alg. 1, lines 4-8) of IE^{2}COL is to simplify the initial (large) graph G by removing from G large independent sets. For this purpose, IE^{2}COL applies the specific extraction strategy of [46] which proves to be effective in reducing a graph. This extraction strategy can be summarized by the following steps.

(1) Apply the Adaptive Tabu Search maximum clique algorithm (ATS) (see [45]) to identify a first maximal independent set I in G (recall that maximum clique and maximum independent set are two equivalent problems).
(2) Apply repeatedly ATS to obtain as many independent sets of size $|I|$ as possible. Then find among these independent sets a maximal set of *pairwise disjoint* independent sets $\mathcal{I} = \{I_1, \ldots, I_x\}$. This later problem is the well known maximum set packing problem [21], which itself is equivalent to the maximum clique (thus independent set) problem, ATS is thus used again to solve the problem.
(3) Remove from G all the vertices of $I_1, \ldots, I_x$ as well as all the edges adjacent to any of these vertices.

This extraction phase repeats the above steps until the residual graph contains no more than q vertices (see Alg. 1, line 5). In Section 4.1, we study the influence of q on the performance of our proposed algorithm.

2.3 Initial and intermediate graph coloring

The IE^{2}COL algorithm needs an algorithm to color the residual graph G_z and some intermediate subgraphs (Alg. 1, lines 12 and 23). For this purpose, we adopt MACOL, a recent and effective memetic algorithm [32] designed for the graph k-coloring problem.

For a given graph G and a fixed number k of colors, MACOL explores a search space Φ composed of all the k-colorings of the graph $G = (V, E)$, i.e., $\Phi = \{C : V \rightarrow \{1, \ldots, k\}\}$. MACOL tries to find a *legal* k-coloring by optimizing (minimizing) a simple function $f(C)$ which counts the number of color conflicts in a k-coloring C. Formally, let $C = \{c_1, c_2, ..., c_k\}$ be a (legal

---

Algorithm 1 The IE^{2}COL algorithm for large graph k-coloring

---

Input: An undirected graph $G = (V, E)$; an integer k
Output: A legal k-coloring of G or report failure
$\{$*EXTRACTION*$\}$
$\{$Each extraction iteration removes a maximal collection of disjoint independent sets of maximal size in G, see Sect. 2.2$\}$
while (G has more than q vertices) **do**
    Find in G a maximal collection $\mathcal{I}$ of *pairwise disjoint* independent sets of the largest size possible
    Simplify G by removing from G all the independent sets of $\mathcal{I}$ and the associated edges
end while
Let Ω contains all the extracted disjoint independent sets; let t the total number of the extracted independent sets ($t = |\Omega|$); let G_z be the residual graph from the extraction phase
$\{$*INITIAL COLORING*$\}$
$\{$A population of $(k-t)$-colorings is obtained by the MACOL coloring algorithm applied to the residual graph G_z, see Sect. 2.3$\}$
Generate a population $\mathcal{P}$ of $(k-t)$-colorings for graph G_z and run MACOL with the colorings of $\mathcal{P}$ to color G_z
if (A legal $(k-t)$-coloring $C \in \mathcal{P}$ for G_z is found by MACOL) **then**
    The coloring C, plus the t extracted independent sets, forms a legal k-coloring for the initial graph G. Return this k-coloring and stop
end if
$\{$*EXPANSION AND BACKWARD COLORING*$\}$
$\{$Backward coloring of intermediate subgraphs by reconsidering extracted independent sets of $\Omega\}$
Let $G' = (V', E')$ be the current subgraph of G under consideration, $\mathcal{P}$ be the set of (illegal) colorings of G' produced by MACOL
while ($\Omega \neq \emptyset$) **do**
    Select some independent sets S from Ω ($S \subset \Omega$) and recover the corresponding subgraph G'' induced by the vertices of $V' \cup S$ (see Sect. 2.4)
    $\Omega \leftarrow \Omega \setminus S$
    Extend each coloring $C \in \mathcal{P}$ by including the independent sets of S as new color classes
    Run MACOL with the extended colorings of $\mathcal{P}$ to color G'' (see Sect. 2.3)
    if (A legal coloring $C \in \mathcal{P}$ for G'' is found by MACOL) **then**
        The coloring C, plus the remaining extracted independent sets of Ω, forms a legal k-coloring for the initial graph. Return this k-coloring and stop
    end if
end while
Return (No legal k-coloring found)

---

or illegal) k-coloring, the evaluation function $f(C)$ is given by the following formula:

$$f(C) = \mid \{\{u, v\} \in E : \exists c_i \in C,\ u \in c_i, v \in c_i\} \mid \quad (1)$$

C is a legal k-coloring if and only if $f(C) = 0$, i.e., each color class c_i of C is an independent set (conflict free).

MACOL is composed of four basic components: a population of candidate solutions (each solution being a k-coloring in Φ) to sample the search space, a dedicated recombination operator (crossover) to create new candidate solutions (offspring) by blending two or more existing solutions, a tabu search based local optimization operator, and a population management strategy.

MACOL starts with an initial population of illegal k-colorings whose individual k-colorings are first improved by the tabu coloring algorithm which is a variant of the seminal TabuCOL [27]. MACOL improves the solutions of its population throughout a number of *generations*. At each generation, MACOL takes randomly $m \geq 2$ parents and uses the adaptive multi-parent crossover operator (AMPaX) to generate an offspring k-coloring. AMPaX builds one by one the color classes of the offspring solution by taking at each step the largest color class among the parents. During the crossover process, AMPaX takes care of using color classes from different parents in order to generate diversified offspring solutions. Once the new offspring coloring is created, it is immediately improved by the tabu coloring algorithm. The tabu coloring algorithm improves an illegal k-coloring by minimizing the above evaluation function f (formula 1). This is achieved by iteratively changing the color of a vertex that shares the same color with at least one adjacent vertex. To decide whether the improved offspring k-coloring can be added to the population, MACOL implements a distance-and-quality based replacement strategy for the pool updating.

As shown in [32], MACOL performs generally much better than local search algorithms. This is why we employ MACOL as our underlying coloring algorithm.

2.4 Expansion strategies for backward coloring

The expansion and backward coloring phase takes as its input the current subgraph G' of G and the colorings of G' in the population $\mathcal{P}$, extends G' to another subgraph G'' by adding some extracted independent sets S and color G'' with the colorings in $\mathcal{P}$ expanded by S (see Alg. 1, lines 16-27). The key issues concern the way to select the independents sets S and to rebuild the corresponding subgraph G''. We consider in this section possible strategies to determine the independent sets for expansion.

To determine the set S of independent sets, we can first consider how many independent sets that we pick for expansion. Basically, this decision can be made according to one of two rules: one independent set or several independent sets. This choice may have influences on the subsequent coloring process. Indeed, adding back one independent set at a time implies limited changes between subgraphs G' and G'' and limited extensions to the current colorings (only one new color class is added). This leads thus to a more gradual coloring optimization. On the other hand, using several independents sets to extend the current subgraph and colorings offers more freedom for coloring optimization.

We can also consider which independent set(s) are to be selected. This decision can be achieved following one of three (at least) rules: reverse of extraction order, extraction order and random order. Given the way independent sets are extracted during the extraction phase (see Sect. 2.2), applying the reverse of extraction order handles the independent sets from the smallest to the largest while applying extraction order does the opposite.

It is clear that any combination of the above two decisions defines a strategy that can be used to determine the independent set(s) for subgraph and coloring extensions. Based on experimental observations, we have decided for this work to use the following simplified strategy which proved to be effective for the set of graphs tested in the paper. After the initial coloring phase of the residual graph G_z, we backtrack directly to the initial graph G and add back all the extracted independent sets as new color classes of colorings of G. Experiments showed that this strategy performs quite well for the graphs used in the paper (See Section 4.2 for a computational analysis). In the general case, (e.g., if still larger and harder graphs are considered), it would be necessary to recover and color additional intermediate subgraphs during the expansion and backward coloring phase.

2.5 Discussions

In this section, we highlight the enhancements introduced in our proposed IE^{2}COL algorithm with respect to the basic extraction and expansion (E2COL) algorithm of [47].

First, as to the extraction phase, while E2COL generates one subgraph for each extracted independent set, IE^{2}COL does not store any intermediate subgraph. Instead, for each selected subset $S \subset \Omega$ of independent sets, the corresponding subgraph is reconstructed.

Second, the expansion and backward coloring phase of the proposed IE^{2}COL differs from that of E2COL. Actually, while E2COL traverses the extracted independent sets from the smallest to the largest and adds back exactly one independent set at a time, IE^{2}COL relies on a much more flexible strategy to decide how many and which independent sets are selected for each expansion iteration (E2COL is thus just a special case of this more general expansion policy). This difference is critical for two reasons. The proposed IE^{2}COL, by taking several independent sets for expansion, needs to consider (and color) fewer subgraphs than E2COL does, thus probably shortening the computing time. And more importantly, extending the current colorings with more new color classes at a time introduces naturally more freedom for the coloring algorithm to better optimize its solutions during subgraph coloring process.

Finally, IE^{2}COL uses MACOL to search for a legal coloring while E2COL replies on a perturbation based tabu search algorithm. By using MACOL, IE^{2}COL is able to color large subgraphs more effectively and achieve highly competitive results, as we will show in the next section.

3 Experimental Results

In this section, we assess the performance of the proposed IE^{2}COL algorithm. For this purpose, we present computational results on a collection of 20 largest benchmark graphs from the well-known DIMACS graph coloring Challenge [30] [1] and COLOR02/03/04 competitions [2]. We also report comparisons with respect to 10 top-performing coloring algorithms from the literature.

3.1 Experimental settings

Test instances. Since IE^{2}COL is designed to color large graphs, we only consider graph instances with at least 900 vertices. Moreover we retain only those graphs which are known to be difficult and challenging (see Table 1) and exclude those (easy) graphs (A graph is considered to be easy if the current best k^*-coloring can be reached by our tabu coloring algorithm). These large, but easy instances with at least 900 vertices include the following cases: 3 abb/ashxxxGPIA graphs, 1 Insertions graph, 3 FullIns graphs, 4 qg.order graphs.

The 20 large and hard graphs considered in this paper belong to the following six families.

- Three large random graphs (DSJC1000.1, DSJC1000.5, DSJC1000.9). The first and second number in the name of each graph represent respectively

[1] `http://www.info.univ-angers.fr/pub/porumbel/graphs/index.html`
[2] `http://mat.gsia.cmu.edu/COLOR04/`

the number of vertices and the edge density in the graph. The chromatic numbers of these graphs are unknown.
- Three large flat graphs (flat1000_50_0, flat1000_60_0, flat1000_76_0). They are structured graphs with known chromatic number (respectively 50, 60 and 76).
- Two large random geometric graphs (R1000.1c, R1000.5). These graphs are generated by picking random points (vertices) in a plane and by linking two points situated within a certain geometrical distance. The chromatic number is unknown for R1000.1c and is equal to 234 for R1000.5.
- Three very large random graphs (C2000.5, C2000.9, C4000.5). The chromatic numbers of these graphs are unknown. Due to the size and difficulty of these graphs, they are not always used in computational experiments in the literature.
- One latin square graph (latin_sqr_10) with unknown chromatic number.
- Eight WAP graphs (WAP01 to WAP08) from COLOR02/03/04 competitions. These graphs stem from real-life optical network design problems. Each vertex corresponds to a lightpath in the network; edges correspond to intersecting paths. These structured graphs have unknown chromatic number except WAP05 whose chromatic number is 50. These instances are used less often than the classical DIMACS graphs.

The graphs of families 1 to 5 were initially collected for the 2nd DIMACS challenge (on graph coloring and maximum clique) while the WAP graphs were made available for the COLOR02/03/04 competitions. One notices that contrary to most DIMACS graphs, the WAP graphs are much less studied in the literature [4,19,6,9].

Parameter. To run IE^{2}COL, we need to fix the threshold q, the number of vertices left in the smallest residual graph G_z. Based on preliminary experiments and as shown in Section 4.1, we have fixed q equal to 500 for all our experiments. In addition to q, MACOL (as well as its tabu coloring algorithm) requires also several parameters. In our case, we adopt those used in the original paper [32].

Stop condition. All experiments for this study were performed on a computer equipped with an Intel Xeon E5440 processor (2.83 GHz, 2GB RAM) running GNU/Linux. Following the DIMACS machine benchmark [3], our machine requires respectively 0.23, 1.42 and 5.42 CPU seconds for the graphs r300.5, r400.5 and r500.5. For all the tested graphs, the same parameter values are used. To report our computational results, 20 independent runs (5 runs for the three largest random graphs C2000.5, C2000.9 and C4000.5) of IE^{2}COL were performed on each graph with different random seeds. The IE^{2}COL algorithm stops if one of the following conditions is verified:

[3] dmclique, ftp://dimacs.rutgers.edu in directory /pub/dsj/clique

(1) A legal $(k-t)$-coloring is found in the initial coloring phase by MACOL which is limited to 300 generations.
(2) A legal coloring is found during the expansion and backward coloring phase.
(3) The processing time reaches its timeout limit. The timeout limit is set to be 5 CPU hours except for five large graphs C2000.5, C2000.9, C4000.5, WAP03 and WAP04. For WAP03 and WAP04 a limit of 1 day is allowed while for the three largest random graphs C2000.5, C2000.9 and C4000.5, the limit is set equal to 5 days. Notice that these timeout limits are comparable with those reported in the latest papers on large graph coloring like [32,33,38,43,46,47] to obtain state-of-the-art results.

3.2 Computational Results

Table 1 [4] summarizes the computational statistics of our IE^{2}COL algorithm on the set of 20 large benchmark instances. Columns 2–4 indicate the features of the tested instances: the number of vertices ($Node$), the number of edges ($Edge$) and the density of the graph ($Density$). Column 5 displays the current best known results k^* reported in the literature, i.e., the smallest k for which a legal k^*-coloring has ever been found by a coloring algorithm. In columns 6–9, the computational statistics of our IE^{2}COL algorithm are presented, including the smallest number of colors (k) for which IE^{2}COL obtains a legal k-coloring, the success rate (hit) and the average computation time in minutes over the runs where a solution with k colors is found. The last column shows the average number of iterations for the successful runs. If IE^{2}COL has a success rate inferior to 100%, we show additional results with larger k until a 100% success rate is reached.

From Table 1, we observe that the results obtained by IE^{2}COL (column 6, k) are highly competitive when compared to the current best known results reported in the literature (column 5, k^*). For the three huge random graphs C2000.5, C2000.9 and C4000.5, colorings with respectively $k = 146$, 409 and 260 were reported recently in [46]. It is noteworthy that IE^{2}COL is able to further improve these bounds and obtain colorings with $k = 145$, 408 and 259 respectively.

For the three flat graphs, IE^{2}COL can reach the current best known results consistently with a success rate of 20/20. More importantly, for flat1000_76_0, IE^{2}COL obtains for the first time a new 81-coloring, improving thus the current best-known result which requires 82 colors.

[4] The results of IE^{2}COL are available at `http://www.info.univ-angers.fr/pub/hao/ie2col.html`

Table 1
Computational results of IE^2COL on the set of 20 large and difficult benchmark instances. IE^2COL improves on the current best known results for 6 instances and matches the current best results for 12 instances. For 2 graphs, IE^2COL obtains a worse result.

Instance	*Node*	*Edge*	*Density*	k^*	IE^2COL			
					k	*hit*	*time(m)*	*Iterations*
DSJC1000.1	1000	49629	0.1	20	20	20/20	65	3.2×10^7
DSJC1000.5	1000	249826	0.5	83	83	20/20	116	1.2×10^8
DSJC1000.9	1000	449449	0.9	222^a	222	3/20	256	5.1×10^8
					223	20/20	216	4.3×10^8
flat1000_50_0	1000	245000	0.49	50	50	20/20	25	1.2×10^6
flat1000_60_0	1000	245830	0.49	60	60	20/20	25	1.3×10^6
flat1000_76_0	1000	246708	0.49	82	**81**	3/20	281	5.8×10^8
					82	20/20	26	5.3×10^7
R1000.1c	1000	485090	0.97	98	98	20/20	67	3.9×10^7
R1000.5	1000	238267	0.48	234	245	2/20	282	8.5×10^8
					246	8/20	251	6.8×10^8
					247	20/20	186	4.3×10^8
latin_sqr_10	900	307350	0.76	97^b	98	5/20	317	1.5×10^8
					99	20/20	171	7.9×10^7
C2000.5	2000	999836	0.5	146^c	**145**	1/5	1198	1.7×10^9
					146	5/5	223	1.4×10^8
C2000.9	2000	1799532	0.9	409^c	**408**	5/5	720	1.1×10^9
C4000.5	4000	4000268	0.5	260^c	**259**	2/5	6987	6.8×10^8
					260	5/5	5223	1.4×10^8
WAP01	2368	110871	0.04	42	42	20/20	159	1.8×10^8
WAP02	2464	111742	0.04	41	41	20/20	206	2.6×10^8
WAP03	4730	286722	0.03	44	44	20/20	1127	3.6×10^8
WAP04	5231	294902	0.02	43	**42**	3/20	1321	8.4×10^8
					43	20/20	1139	3.7×10^8
WAP05	905	43081	0.11	50	50	20/20	18	1.6×10^6
WAP06	947	43571	0.10	40	40	6/20	257	4.4×10^8
					41	20/20	139	2.2×10^8
WAP07	1809	103368	0.06	42	**41**	20/20	141	1.5×10^8
WAP08	1870	104176	0.06	42	42	20/20	135	1.4×10^8

Note a: This bound was reported very recently in [41,46].

Note b: This bound was reported very recently in [42].

Note c: These bounds were reported very recently in [46].

For the 3 random DSJC graphs which are known to be hard to color for many algorithms, IE^2COL can attain the current best known results for two of them (DSJC1000.1 DSJC1000.5) with a hit rate of 20/20. In particular, for DSJC1000.9, IE^2COL is able to find 222-colorings which were reported very recently for only two algorithms [41,46].

Table 2
Comparison of IE^{2}COL with three related algorithms on the set of 12 large DIMACS benchmark instances. In all the cases, IE^{2}COL obtains the same or improved results with respect to the compared algorithms.

Instance	k^*	IE^{2}COL			MACOL [32]			EXTRACOL [46]			E2COL [47]		
		k	*hit*	*Iter*	k	*hit*	*Iter*	k	*hit*	*Iter*	k	*hit*	*Iter*
DSJC1000.1	20	20	20/20	3.2×10^7	20	20/20	3.5×10^7	20	20/20	3.1×10^7	20	10/10	5.2×10^7
DSJC1000.5	83	20	20/20	1.2×10^8	20	20/20	2.2×10^8	20	20/20	2.0×10^8	20	4/10	7.2×10^8
DSJC1000.9	222	222	3/20	5.1×10^8	223	18/20	4.5×10^8	222	3/20	5.4×10^8	224	6/10	6.7×10^8
flat1000_50_0	50	50	20/20	1.2×10^6	50	20/20	3.2×10^5	50	20/20	3.2×10^5	50	10/10	1.2×10^6
flat1000_60_0	60	60	20/20	1.3×10^6	60	20/20	6.3×10^5	60	20/20	5.1×10^5	60	10/10	1.7×10^6
flat1000_76_0	82	**81**	3/20	5.8×10^8	82	20/20	7.2×10^7	82	20/20	6.7×10^7	82	10/10	3.5×10^8
R1000.1c	98	98	20/20	3.9×10^7	98	20/20	7.5×10^5	101	18/20	6.4×10^5	98	10/10	5.2×10^8
R1000.5	234	245	3/20	8.5×10^8	245	13/20	1.2×10^9	250	11/20	8.8×10^8	256	1/10	4.7×10^8
latin_sqr_10	97	98	5/20	1.5×10^8	99	5/20	6.7×10^7	99	11/20	1.2×10^8	98	10/10	2.7×10^8
C2000.5	146	**145**	1/5	1.7×10^9	148	1/5	8.8×10^8	146	5/5	1.7×10^8	147	5/5	1.1×10^9
C2000.9	409	**408**	5/5	1.1×10^9	413	2/5	7.5×10^8	409	2/5	4.5×10^8	413	2/5	1.3×10^9
C4000.5	260	**259**	2/5	6.8×10^8	272	3/5	1.2×10^9	260	4/5	1.8×10^8	262	5/5	1.8×10^9

Finally, it is interesting to observe that for the 8 large WAP graphs from COLOR02/03/04 competitions, IE^{2}COL is able to find improved upper bounds for 2 graphs (WAP4, WAP7) whose chromatic numbers are still unknown and match the current best known results for the 6 other graphs.

3.3 *Comparing IE^{2}COL with MACOL, EXTRACOL and E2COL*

In this section, we compare IE^{2}COL with three related approaches using the set of 12 DIMACS graphs: its underlying memetic coloring algorithm (MACOL [32]), the approach using independent set extraction as a preprocessing method (EXTRACOL [46]) and the initial basic extraction and expansion algorithm (E2COL [47]). The purpose of this comparison is to know to which extend IE^{2}COL can improve on the results of these related approaches and show the added value of the enhancements implemented in IE^{2}COL. Table 2 summarizes the computational results of these 4 algorithms.

When comparing IE^{2}COL against MACOL, we notice that they reach the same minimal k value for 6 graphs (DSJC1000.1, DSJC1000.5, flat1000_50_0, flat1000_60_0, R1000.1c and R1000.5). For the other 6 graphs, IE^{2}COL finds better solutions than MACOL. This shows the added value of embedding the memetic coloring algorithm into the proposed extraction and backward coloring approach.

When comparing IE^{2}COL and EXTRACOL, one observes that even though EXTRACOL performs very well on these graphs (except on the two R1000.x graphs), IE^{2}COL delivers better results in 7 out of 12 cases. In particular, thanks to the backward coloring strategy, IE^{2}COL is able to further improve on the current best known results of 3 very difficult graphs (C2000.5, C2000.9, C4000.5) which have been established by EXTRACOL. This highlights the critical role of the expansion-coloring strategy employed by IE^{2}COL.

Finally, when it comes to comparing IE^{2}COL and E2COL, the results are once again in favor of IE^{2}COL because IE^{2}COL improves on the results of E2COL in 6 out of 12 cases. This is possible thanks to the enhancements presented in Section 2, concerning particularly the improved strategies for the backward coloring phase. This also underscores the importance of the underlying coloring algorithm (recall that E2COL employs a perturbation-based tabu search coloring algorithm).

3.4 Comparison with other state of the art algorithms

In this section, we compare the results of our IE^{2}COL algorithm with 13 state-of-art coloring algorithms, which are based on diverse approaches: reactive tabu search with partial solutions (PCol) [2], iterated local search (ILS) [8], variable space search (VSS) [28], quantum annealing (QA) [41], hybrid evolutionary algorithms (HGA [15], HEA [17], MMT [33], Evo [38]), multiagent fusion search (MFS) [43], mimimal-state processing search (MSP) [16], distributed coloration neighborhood search (DCNS) [35], adaptive memory search (AmaCol) [19] and ant local search (ALS) [36]. For this experiment, we focus on the quality criterion, i.e., the lowest value of k for which a k-coloring can be found.

Table 3 presents the comparative results on the set of the DIMACS graphs (except C2000.9 for which no results are reported for the reference algorithms). Columns 2 and 3 recall the best known results (k^*) and the best results found by IE^{2}COL. Columns 4–13 give the best results reported by these reference algorithms. From Table 3, one observes that IE^{2}COL competes very favorably with these top-performing coloring algorithms. Indeed, if one compares IE^{2}COL with each of the reference algorithm, one finds that over these 11 hard graphs, IE^{2}COL can obtain one or more better solutions (smaller k) and at most one worse result (larger k).

Notice that a completely fair comparison is impossible since the reference algorithms are implemented by different authors and run under different conditions. This comparison is thus presented only for indicative purposes and should be interpreted with caution. Nevertheless, this experiment does show very positive indications about the competitiveness of IE^{2}COL when compared to these state-of-the-art algorithms.

Table 3
Comparisons between IE^{2}COL and 13 state-of-the-art coloring algorithms in the literature. '-' means unavailability of a result. For 10 of the 11 large DIMACS benchmark graphs, IE^{2}COL obtains the same or improved results with respect to the reference algorithms.

Graph	k^*	IE^{2}COL	state-of-the-art coloring algorithms												
			PCol	ILS	VSS	QA	Evo	MMT	MFS	MSP	HGA	DCNS	AmaCol	HEA	ALS
			[2]	[8]	[28]	[41]	[38]	[33]	[43]	[16]	[15]	[35]	[19]	[17]	[36]
DSJC1000.1	20	20	20	-	20	20	20	20	-	21	-	-	20	20	20
DSJC1000.5	83	83	89	89	86	83	83	83	84	88	84	89	84	83	84
DSJC1000.9	222	222	226	-	224	222	223	225	223	228	-	226	224	224	224
flat1000_50_0	50	50	50	-	50	-	50	50	50	50	84	50	50	-	50
flat1000_60_0	60	60	60	-	60	-	60	60	60	60	84	60	60	-	60
flat1000_76_0	82	**81**	87	-	85	82	82	82	83	87	84	89	84	83	83
R1000.1c	98	98	98	-	-	98	98	98	-	98	99	98	-	-	-
R1000.5	234	245	248	-	-	238	238	234	-	237	268	241	-	-	-
latin_sqr_10	97	98	-	99	-	98	98	101	104	99	106	98	104	-	-
C2000.5	146	**145**	-	-	-	-	148	-	150	162	153	151	-	-	-
C4000.5	260	**259**	-	-	-	-	271	-	-	301	280	-	-	-	-

4 Analysis of IE^{2}COL

4.1 Effect of the size of residual graph

We now turn our attention to a study on the influence of the size of residual graph on the performance of the IE^{2}COL algorithm. Recall that the extraction phase of IE^{2}COL stops when no more than q vertices are left in the residual graph from which the initial coloring and possibly backward coloring phases are launched. Different values of q may impact the outcome of IE^{2}COL. We carry out additional experiments on 4 instances (DSJC1000.5, DSJC1000.9, R1000.1c, flat1000_76_0) and run IE^{2}COL 10 times on each of these instances with $q \in \{300, 500, 600\}$ and show in Table 4 the computational results. In addition to k and *hit*, we also indicate the average number of iterations needed to find a k-coloring. For DSJC1000.9, we aim at finding a 223-coloring, for flat1000_76_0, we aim at finding a 82-coloring. For each run of the IE^{2}COL, the timeout limit is set to be 5 CPU hours.

¿From Table 4, we observe that all these q values allow the algorithm to find a legal k-coloring. Nevertheless, IE^{2}COL with $q = 500$ and $q = 600$ reaches more stable results (higher hits), but may require more iterations than with $q = 300$. Therefore, it seems that a relatively larger q makes the algorithm more robust but also slower. This implies that there may not be an absolute best value for this parameter and that a compromise between robustness and

Table 4
Influence of the size of residual graph (parameter q) on the performance of IE^2COL.

Graph	k^*	$q = 300$			$q = 500$			$q = 600$		
		k	*hit*	*Iterations*	k	*hit*	*Iterations*	k	*hit*	*Iterations*
DSJC1000.5	83	83	6/10	8.4×10^7	83	10/10	1.2×10^8	83	10/10	1.5×10^8
DSJC1000.9	222	223	10/10	4.1×10^8	223	10/10	4.3×10^8	223	9/10	4.6×10^8
flat1000_76_0	81	82	9/10	5.0×10^7	82	10/10	5.3×10^7	82	10/10	5.7×10^7
R1000.1c	98	98	10/10	4.1×10^7	98	10/10	3.9×10^7	98	10/10	4.2×10^7

speed could be possible.

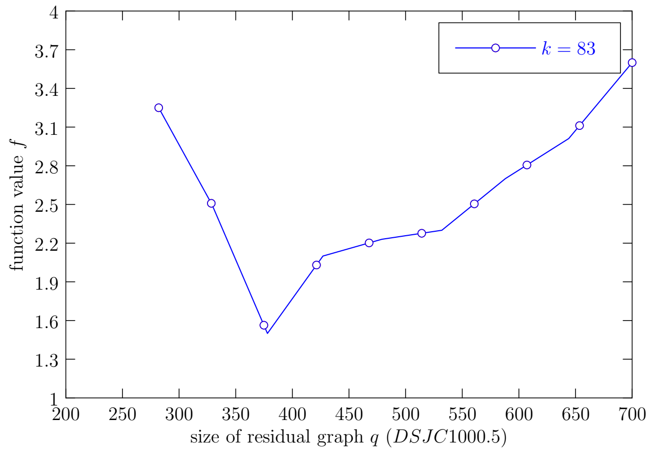

Fig. 1. Influence of the size of residual graph on the evaluation function f

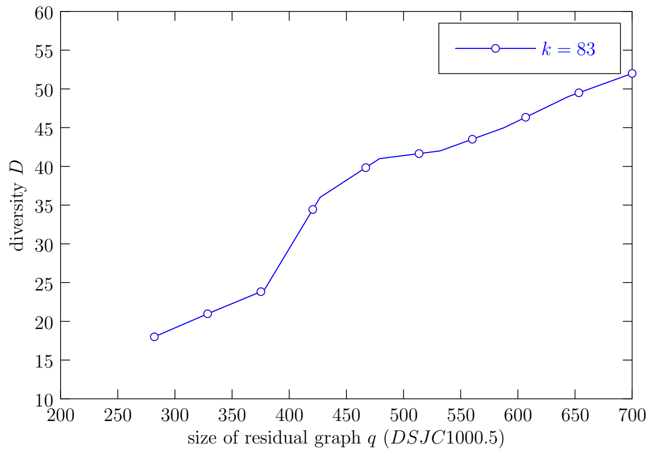

Fig. 2. Influence of the size of residual graph on the diversity D of the population

To complement this experiment and get more insight, we analyze the influence of q on two other interesting points: (1) the evaluation function f (Eq. 1, Sect. 2.3) and (2) the diversity of the population. For this purpose, we present below in detail the results on a single graph, but the observations remain valid for several other tested graphs.

The considered instance is DSJC1000.5 with $k = 83$. We show in Fig. 1 the influence of q on the evaluation function f using a running profile. The profile is defined by the function $q \longmapsto f_*(q)$ where q is the size of residual graph and $f_*(q)$ the best (smallest) f value at the end of the initial coloring phase (averaged over 10 independent runs). From Fig. 1, one can observe that a too large or too small q value can lead to worse (large) results for f. q values ranging from 350 to 500 seem to give the best results.

For memetic algorithms, it is well known that population diversity has an important influence on the performance [26]. A fast lost of the diversity in the population leads to a premature convergence. We show in Fig. 2 influence of q on the diversity D of the population. The population diversity is calculated according to the method described in [38,39]. The plotted profile in Fig. 2 is defined by the function $q \longmapsto D_*(q)$ where q is the size of the residual graph and $D_*(q)$ the population diversity at the end of the initial coloring phase (averaged over 10 independent runs). From Fig. 2, one observes that a larger value for q can better preserve the population diversity while a smaller value for q can lead to a fast lost of the diversity in the population, thus leading to premature convergence of the memetic algorithm.

Considering jointly Fig. 1 and 2, we conclude $q = 500$ is an appropriate value, which explains why this value was used for all the experiments reported in this paper. More generally, it is reasonable to believe that q may depend on the effectiveness of the underlying coloring algorithm and on the structure of the graphs to be colored.

4.2 Influence of the expansion strategy

As discussed in Section 2.4, if the initial coloring phase fails to find a legal coloring for the residual graph G_z, one can use different strategies to add back the extracted independent sets for the backward coloring phase. The computational results of Section 3 are obtained by our IE^{2}COL algorithm with a two-level strategy: backtrack from G_z directly to the initial graph G by adding back all the extracted independent sets as new color classes of colorings of G. In this section, we compare this strategy with two other multi-level expansion strategies which add back progressively the extracted independent sets in several steps.

With the first compared strategy, each expansion step reintegrates all the independent sets of the *same size* according to the extraction order (i.e., from the largest to the smallest, denoted by Largest first Strategy). For the second strategy, each expansion step brings back all 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).

As an illustration, Fig. 3 shows a detailed comparison of these three expansion strategies on the instance (DSJC1000.5, $k = 83$). We run IE^{2}COL with each of these expansion strategies until the coloring algorithm (MACOL) reaches 1000 generations. For the two-level strategy, we recover all the extracted independent sets at generations 300, while for the two other expansion strategies, the independent sets are progressively added at generations 300, 500 and 700 respectively in 3 steps.

We keep other ingredients unchanged in the IE^{2}COL algorithm and observe (like in Section 4.1) the evolution profile of each expansion strategy: the averaged best objective value (over 20 runs) *vs.* the number of generations. From Fig. 3, we observe that the two-level strategy performs better than the two multi-level expansion strategies. In particular, for the two-level strategy, as soon as all the independent sets are added back, the objective function value decreases (from generation 300 to 400) more importantly than with the two competing expansion strategies. This dominance continues until the end of the search. This could be explained by the fact that extending the current colorings with more new color classes at a time provides MACOL (which itself is a powerful coloring algorithm) with more freedom which allows it to better optimize its solutions during its coloring process.

Concerning the two multi-level recovery strategies, we observe that the Smallest first Strategy performs better than the Largest first Strategy. One possible reason could be the fact that the vertices of large independent sets have more chance to group together in an optimal coloring [17,32]. Thus, it seems wise to preserve these large independent sets and add them back only at late stages

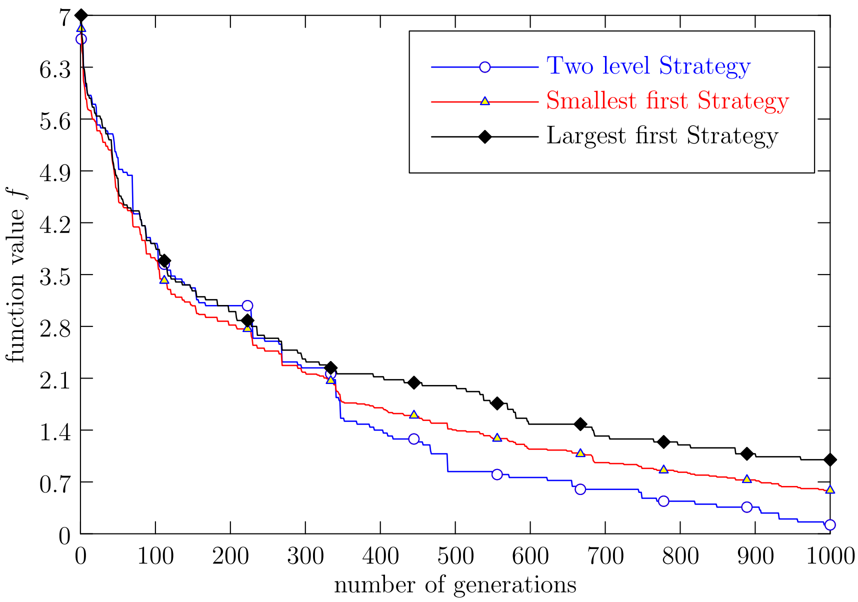

Fig. 3. Comparison between three different expansion strategies for backward coloring.

of the backward coloring process.

Finally, there seem no formal justifications to prefer one strategy over another. The above observations should be interpreted with caution. In particular, even though the two-level strategy showed a good performance on the set of instances used in this paper, the other expansion strategies discussed in Section 2.4 could be useful in other situations.

4.3 Analysis about the three different phases of IE^{2}COL

Our IE^{2}COL algorithm is composed of an independent set extraction phase (A), an initial coloring phase (B) and an backward-coloring phase (C). Given an instance (G, k), one may wonder at which step a legal coloring is reached. Clearly, the answer depends on the instance. For the set of 20 instances used in this paper, we observed that the extraction phase is quite helpful in general and is especially useful for random graphs. This observation is consistent with previous studies like [14,17,27,29,35] where an extraction phase is also implemented as a preprocessing step. On the other hand, we noted that the expansion and backward coloring phase is necessary for most of the instances, especially for the structured graphs (except some flat graphs). Finally, it is clear that the underlying coloring algorithm also impacts on the number of needed expansion steps. Complementary information can be found in two related studies [46,47].

To complement this discussion and for purely indicative purposes, we show in Table 5 the phases which are needed to obtain the results reported in Table 1 (Section 3.2) and the associated computing times for each phase.

5 Conclusion

In this paper, we have presented a general method for the graph vertex coloring problem able to handle large graphs. This method combines an independent set extraction phase with an expansion and backward coloring phase. The extraction phase relies on a dedicated strategy to identify and remove large independent sets from the initial graph. The expansion-coloring phase provides a way of reconsidering extracted independent sets as additional color classes for the purpose of progressive coloring optimization.

The proposed IE^{2}COL algorithm implementing this method has achieved noteworthy performance on the set of 20 largest benchmark graphs with 900 to 4000 vertices from DIMACS and COLOR02/03/04 competitions. IE^{2}COL im-

Table 5
An analysis about the independent set extraction phase (A), the initial coloring phase (B) and the backward-coloring phase (C) of the IE^{2}COL algorithm. The computing time T (in percentage) spent on each phase is indicated for indicative purposes.

Instance	*Node*	*Edge*	*Density*	k	IE^{2}COL			
					T_A	T_B	T_C	*Phases*
DSJC1000.1	1000	49629	0.1	20	85%	15%	0%	A+B
DSJC1000.5	1000	249826	0.5	83	15%	15%	70%	A+B+C
DSJC1000.9	1000	449449	0.9	222	8%	8%	84%	A+B+C
				223	9%	9%	82%	A+B+C
flat1000_50_0	1000	245000	0.49	50	80%	20%	0	A+B
flat1000_60_0	1000	245830	0.49	60	80%	20%	0	A+B
flat1000_76_0	1000	246708	0.49	81	10%	5%	85%	A+B+C
				82	80%	20%	0	A+B
R1000.1c	1000	485090	0.97	98	50%	30%	20%	A+B+C
R1000.5	1000	238267	0.48	245	7%	7%	86%	A+B+C
latin_sqr_10	900	307350	0.76	98	6%	5%	89%	A+B+C
				99	12%	9%	81%	A+B+C
C2000.5	2000	999836	0.5	145	9%	1%	90%	A+B+C
				146	49%	6%	45%	A+B+C
C2000.9	2000	1799532	0.9	408	64%	3%	33%	A+B+C
C4000.5	4000	4000268	0.5	259	68%	1%	31%	A+B+C
				260	90%	1%	9%	A+B+C
WAP01	2368	110871	0.04	42	20%	5%	75%	A+B+C
WAP02	2464	111742	0.04	41	20%	5%	75%	A+B+C
WAP03	4730	286722	0.03	44	41%	1%	58%	A+B+C
WAP04	5231	294902	0.02	42	37%	1%	62%	A+B+C
				43	43%	1%	56%	A+B+C
WAP05	905	43081	0.11	50	75%	25%	0	A+B
WAP06	947	43571	0.10	40	8%	4%	88%	A+B+C
				41	14%	7%	79%	A+B+C
WAP07	1809	103368	0.06	41	20%	10%	70%	A+B+C
WAP08	1870	104176	0.06	42	20%	10%	70%	A+B+C

proves on the current best colorings (new upper bounds) for 6 graphs and matches the current best results for 12 other graphs while its results is worse in two cases. The improved upper bounds, combined with the new development of lower bounds, constitute a step forward toward the goal of finding the chromatic number of these graphs.

Even though it is believed that it becomes more and more difficult to obtain better upper bounds for the tested benchmark graphs, this study shows that improvements are still possible with new solution strategies, in particular combined method.

Acknowledgment

We are grateful to the referees for their comments and questions which helped us improve the paper. The work is partially supported by the "Pays de la Loire" Region (France) within the RaDaPop (2009-2013) and LigeRO (2010-2013) projects.

References

[1] C. Avanthay, A. Hertz, N. Zufferey, A variable neighborhood search for graph coloring. European Journal of Operational Research 151(2) (2003) 379–388.

[2] I. Blöchliger, N. Zufferey, A graph coloring heuristic using partial solutions and a reactive tabu scheme. Computers and Operations Research 35(3) (2008) 960–975.

[3] D. Brélaz, New methods to color the vertices of a graph. Communications of the ACM 22(4) (1979) 251–256.

[4] T.N. Bui, T.H. Nguyen, C.M. Patel, K.A. T. Phan, An ant-based algorithm for coloring graphs. Discrete Applied Mathematics 156(2) (2008) 190–200.

[5] E.K. Burke, B. McCollum, A. Meisels, S. Petrovic, R. Qu, A graph-based hyper heuristic for timetabling problems. European Journal of Operational Research 176 (2007) 177–192.

[6] M. Caramia, P. Dell'Olmo. Embedding a novel objective function in a two-phased local search for robust vertex coloring. European Journal of Operational Research. 189:1358-1380, 2008.

[7] M. Chams, A. Hertz, D. de Werra, Some experiments with simulated annealing for coloring graphs. European Journal of Operational Research 32 (1987) 260–266.

[8] M. Chiarandini, T. Stützle, An application of iterated local search to graph coloring. In: D.S. Johnson, A. Mehrotra, M. Trick (Eds.) Proc. of the Computational Symposium on Graph Coloring and its Generalizations, Ithaca, New York, USA, 112–125, 2002.

[9] M. Chiarandini, I. Dumitrescu, T. Stützle. Stochastic local search algorithms for the graph colouring problem. In: TF. Gonzalez (Ed.) Handbook of Approximation Algorithms and Metaheuristics. Computer & Information Science Series, Chapman & Hall/CRC, 2007.

[10] D. Costa, A. Hertz, Ants can colour graphs. Journal of the Operational Research Society 48 (1997) 295-305.

[11] D. de Werra, C. Eisenbeis, S. Lelait, B. Marmol, On a graph-theoretical model for cyclic register allocation. Discrete Applied Mathematics 93(2-3) (1999) 191–203.

[12] R. Dorne, J.K. Hao, Tabu search for graph coloring, T-colorings and set T-colorings. Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, 77–92, 1998.

[13] R. Dorne, J.K. Hao, A new genetic local search algorithm for graph coloring. Lecture Notes in Computer Science 1498 (1998) 745–754.

[14] C. Fleurent, J.A. Ferland, Genetic and hybrid algorithms for graph coloring. Annals of Operations Research 63 (1996) 437–461.

[15] C. Fleurent, J.A. Ferland, Object-oriented implementation of heuristic search methods for graph coloring, maximum clique, and satisfiability. In: [30], (1996) 619–652.

[16] N. Funabiki, T. Higashino, A minimal-state processing search algorithm for graph coloring problems. IEICE Transaction Fundamentals E83-A (2000) 1420–1430.

[17] P. Galinier, J.K. Hao, Hybrid evolutionary algorithms for graph coloring. Journal of Combinatorial Optimization 3(4) (1999) 379–397.

[18] P. Galinier, A. Hertz, A survey of local search methods for graph coloring. Computers and Operations Research 33(9) (2006) 2547–2562.

[19] P. Galinier, A. Hertz, N. Zufferey, An adaptive memory algorithm for the K-colouring problem. Discrete Applied Mathematics 156(2) (2008) 267–279.

[20] M.R. Garey, D.S. Johnson, H.C. So, An application of graph coloring to printed circuit testing. IEEE Transactions on Circuits and Systems 23 (1976) 591–599.

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

[22] C. Glass, Bag rationalisation for a food manufacturer. Journal of the Operational Research Society, 53(2002) 544–551.

[23] C. Glass, A. Prügel-Bennett, Genetic algorithms for graph colouring: exploration of Galinier and Hao's algorithm. Journal of Combinatorial Optimization 7(3): 229–236, 2003.

[24] F. Glover, M. Parker, J. Ryan, Coloring by tabu branch and bound, In [30], (1996) 285-307.

[25] J.P. Hamiez, J.K. Hao, Scatter search for graph coloring. Lecture Notes in Computer Science 2310 (2002) 168–179, Springer-Verlag.

[26] J.K. Hao. Memetic algorithms in discrete optimization. In F. Neri, C. Cotta, P. Moscato (Eds.) Handbook of Memetic Algorithms. Studies in Computational Intelligence 379 (2011) Chapter 6, pages 73–94.

[27] A. Hertz, D. de Werra, Using tabu search techniques for graph coloring, Computing 39 (1987) 345–351.

[28] A. Hertz, M. Plumettaz, N. Zufferey, Variable space search for graph coloring. Discrete Applied Mathematics 156(13) (2008) 2551–2560.

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

[30] D.S. Johnson, M. Trick (Eds.), Cliques, coloring, and satisfiability: Second DIMACS implementation challenge, Volume 26 of Dimacs Series in Discrete Mathematics and Theoretical Computer Science (1996).

[31] F.T. Leighton, A graph coloring algorithm for large scheduling problems. Journal of Research of the National Bureau of Standards 84(6) (1979) 489–506.

[32] Z. Lü and J.K. Hao, A memetic algorithm for graph coloring. European Journal of Operational Research 200(1) (2010) 235–244.

[33] E. Malaguti, M. Monaci, P. Toth, A metaheuristic approach for the vertex coloring problem. INFORMS Journal on Computing 20(2) (2008) 302–316.

[34] E. Malaguti and P. Toth. A survey on vertex coloring problems. International Transactions in Operational Research, 17(1):1-34, 2010.

[35] C. Morgenstern, Distributed coloration neighborhood search. In [30], (1996) 335–357.

[36] M. Plumettaz, D. Schindl, N. Zufferey, Ant Local Search and its efficient adaptation to graph colouring. Journal of Operational Research Society 61(5) (2010) 819–826.

[37] D.C. Porumbel, J.K. Hao, P. Kuntz, A search space cartography for guiding graph coloring heuristics. Computers and Operations Research 37(4) (2010) 769–778.

[38] D.C. Porumbel, J.K. Hao, P. Kuntz, An evolutionary approach with diversity guarantee and well-informed grouping recombination for graph coloring. Computers and Operations Research 37(10) (2010) 1822–1832.

[39] D.C. Porumbel, J.K. Hao, P. Kuntz, An efficient algorithm for computing the distance between close partitions. Discrete Applied Mathematics 159 (2011) 53–59.

[40] D.H. Smith, S. Hurley, S.U. Thiel, Improving heuristics for the frequency assignment problem. European Journal of Operational Research 107(1) (1998) 76–86.

[41] O. Titiloye, A. Crispin, Quantum annealing of the graph coloring problem. Discrete Optimization 8(2) (2011) 376–384.

[42] O. Titiloye, A. Crispin, Graph coloring with a distributed hybrid quantum annealing algorithm. Lecture Notes in Computer Science, Volume 6682 (2011) 553–562.

[43] X.F. Xie, J. Liu, Graph coloring by multiagent fusion search. Journal of Combinatorial Optimization 18(2) (2009) 99–123.

[44] N. Zufferey, Optimization by ant algorithms: possible roles for an individual ant. Optimization Letters. DOI: 10.1007/s11590-011-0327-x, online since 2011.

[45] Q. Wu, J.K. Hao, Adaptive multistart tabu search for the maximum clique problem. Accepted to Journal of Combinatorial Optimization. DOI: 10.1007/s10878-011-9437-8, online since December 2011.

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

[47] Q. Wu, J.K. Hao, An extraction and expansion approach for graph coloring. Accepted to Asia-Pacific Journal of Operational Research (2012).