An Iterative Matheuristic Algorithm for the B-Coloring Problem

Roberto Montemanni
roberto.montemanni@unimore.it
Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia
Reggio Emilia, Italy

Derek H. Smith
derek.smith@southwales.ac.uk
School of Computing and Mathematics, University of South Wales
Pontypridd, United Kingdom

ABSTRACT

B-coloring is a problem in graph theory at the basis of several real applications and also used to improve solution methods for the classical coloring problem. Enhanced solutions for the classical coloring problem have in turn impacts on several other practical applications in scheduling, timetabling and telecommunications. Namely, given a graph $G = (V, E)$, the *b-coloring problem* consists of maximizing the number of colors used while assigning a color to every vertex in V such that no pair of adjacent vertices receive the same color and every color has a representative, called a b-vertex. A vertex can be a *b-vertex* if it is adjacent to vertices colored with all the colors apart from the one assigned to it.

In this paper we present a novel Iterative Matheuristic Algorithm based on considerations about the structure of promising solutions and a mathematical programming model. A vast section of computational experiments shows how the approach is able to find high quality solutions for commonly established datasets from the literature. In particular, the method we propose is able to improve the best known heuristic solution for 38 instances of the 137 considered. The optimality of the bounds previously known for another 5 instances has also been proved by running the approach we propose exhaustively.

CCS CONCEPTS

• **Mathematics of computing** → **Solvers**; **Graph theory**.

KEYWORDS

B-coloring; Graph Theory; Integer Linear Programming; Heuristic Algorithms

ACM Reference Format:
Roberto Montemanni and Derek H. Smith. 2022. An Iterative Matheuristic Algorithm for the B-Coloring Problem. In *2022 The 3rd International Conference on Industrial Engineering and Industrial Management (IEIM 2022), January 12–14, 2022, Barcelona, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3524338.3524379

1 INTRODUCTION

Given an undirected graph $G = (V, E)$, a b-coloring with K colors is a function that assigns to each vertex i of V a colour $c(i) \in C = \{1, 2, \ldots, K\}$ in such a way that $c(i) \neq c(j)$ for every $(i, j) \in E$ and for each $k \in C$ there exists a vertex $i \in V$ with $c(i) = k$ and such that $N(i) \cap \{j \in V | c(j) = h\} \neq \emptyset \; \forall h \in C \backslash \{k\}$, where $N(i) = \{j | (i, j) \in E\}$ is the neighbourhood of i. In other words, in a b-coloring for each color k used there is a vertex assigned to color k such that every other color used is assigned to at least one of its neighbors. The b-coloring problem consists of finding a proper b-coloring using the maximum possible numbers of colors. The *b-chromatic number* of a graph G, $X_b(G)$, is consequently defined as the maximum number of colors for which G admits a b-coloring. An example of an optimal b-coloring is provided in Figure 1.

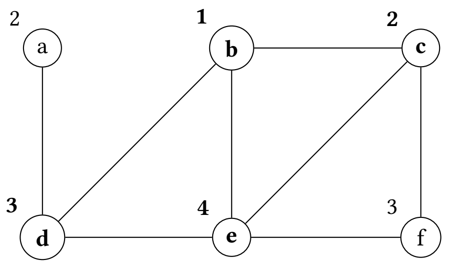

Figure 1: Example of a graph and an associated optimal b-coloring with 4 colors (top left of each vertex). Vertices with the name in bold are the b-vertices.

In [11] it is proved that the problem of finding $X_b(G)$, and consequently the b-coloring problem itself, are NP-hard problems. The difference between the optimal solution values of the classical coloring problem ([16]) and b-coloring for the same graph G can be arbitrarily large, as shown in [15]. The authors of [3] showed that the b-coloring problem can be largely influenced by the girth (length of a shortest cycle) of the graph. A property that is often exploited by constructive and enumerative methods for the coloring problem is the fact that one can have solutions with the number of colors ranging from the chromatic number to the cardinality of the vertex set. An important property, proved in [1] is that a b-coloring with k colors does not necessarily exist for every integer k ranging from the minimum number for which a b-coloring exists up to the b-chromatic number.

The authors of [6] proposed a hybrid evolutionary algorithm for the b-coloring problem. Some of the instances originally proposed for other graph problems in [12] are also considered here as a testbed for b-coloring. A first integer linear programming formulation for the b-chromatic index $X_b(G)$, the edge version of the problem, is introduced in [14]. A branch and cut algorithm based on the same formulation is provided in [13]. Results are however presented only for random graphs in both studies. Matheuristic approaches to the problem were finally discussed in [17], together with an effective new mixed integer linear programming model, on which the algorithms previously mentioned are actually based. Experiments on a large number of instances from [12] are reported.

B-coloring concepts are at the basis of several applications. It is worth mentioning here the works [7] and [8], where b-coloring is used within postal mail sorting systems. A new approach for address block localization that assists the software for address recognition, is presented. A novel clustering technique based on b-coloring is used by the French healthcare system to detect a new typology of hospital stays, as described in [5]. The authors of [2] presented an indirect practical motivation for attacking the b-coloring problem, namely, finding an upper bound for the b-algorithm which is a heuristic approach for the classical coloring problem, which in turn is a problem with several important practical applications such as scheduling [20], timetabling [4] and telecommunications [9, 18, 19]. Fast and effective methods for solving the b-coloring problem are therefore likely to have practical effects on such domains.

2 AN INTEGER PROGRAMMING MODEL

The model discussed in this section was originally proposed in [17]. The model uses a set of variables x such that $x_{ij} = 1$ if vertex j is coloured with the colour of the representative vertex i, 0 otherwise. Note that if vertex i is a representative, then $x_{ii} = 1$. Let $\bar{N}(i) = V \setminus \{\{i\} \cup N(i)\}$ be the anti-neighbourhood of i.

$$(BC) \quad \max \sum_{i \in V} x_{ii} \tag{1}$$

$$x_{ij} \leq x_{ii} \quad i \in V; j \in \bar{N}(i); \nexists k \in \bar{N}(i) : (j,k) \in E \tag{2}$$

$$\sum_{\substack{k \in N(j), \\ k \notin N(i)}} x_{ik} \geq x_{ii} + x_{jj} - 1 \quad i,j \in V; (i,j) \notin E \tag{3}$$

$$\sum_{j \notin N(i)} x_{ji} = 1 \quad i \in V \tag{4}$$

$$x_{ij} + x_{ik} \leq x_{ii} \quad i \in V; j,k \in \bar{N}(i); (j,k) \in E \tag{5}$$

$$x_{ij} \in \{0,1\} \quad i,j \in V \tag{6}$$

The objective function (1) maximizes the number of representative vertices, which are the b-vertices. Constraints (2) guarantee that a vertex can only give a color if it is a representative (notice that if this constraint is removed, a vertex that has a stable set as anti-neighborhood is allowed to represent all its anti-neighborhood without being a representative). Constraints (3) are the b-coloring restrictions which imply that if both vertices i and j are b-vertices, then there must be a neighbor of j which is represented by i. This is achieved due to the fact that if both i and j are representatives, the right-hand side is equal to one, implying that the summation in the left-hand side, which is composed by the neighbors of j that can be represented by i, should be at least one. Constraints (4) ensure that every vertex must have a color. Constraints (5) force the coloring to be proper. Constraints (6) ensure the integrality requirements on the variables.

3 AN ITERATIVE MATHEURISTIC ALGORITHM

One of the heuristic methods proposed in [17] is based the model *BC* presented in Section 2, and works by taking as input a feasible b-coloring solution, and setting the variables x_{ii} corresponding to b-vertices to 1 into the model. By solving the model modified in such a way, the feasible solution will be expanded to a local optimum containing the starting input set of b-vertices and potentially others.

The method we propose adopts the same strategy of iteratively setting the value of some b-vertices into an integer program, which is consequently solved. The main differences with respect to the method discussed in [17] are:

- we transform the *BC* model into a feasibility checker, only asking for a final solution with at least T b-vertices, with this parameter set therefore to a target on the number of b-vertices we want to achieve;
- at each iteration we set to 1 the variables x_{ii} corresponding to a subset of V, but without checking in advance if they lead to any feasible solution;
- we iterate by ideally analyzing all possible sets of T potential b-vertices.

Note that if all the sets of b-vertices can be enumerated, then the method is able to answer the question whether a solution with T colors exists at all for the b-coloring problems on the given graph. In practice, it is rarely possible to run such an extensive search (at least without some form of pruning) so we prefer to classify the method as a heuristic that stops after a given amount of running time has elapsed. The new algorithm we propose will therefore be referred to as the *Iterative Matheuristic Algorithm* (IMA) in the rest of the paper.

3.1 Selection of the potential b-vertices

The first step of the algorithm we propose is the selection of the vertices that can potentially take the role of b-vertices. Given a minimum target T for the number of colors, it is straightforward to see that only vertices $i \in V$ with at least $T-1$ neighbours in G can potentially be b-vertices. We can therefore safely consider the set of potential b-vertices as $P = \{i \in V || N(i)| \geq T-1\}$. This set will be at the basis of the algorithm we propose.

3.2 Ordering the potential b-vertices

The algorithm we propose works by iteratively selecting elements of the set P defined in Section 3.1 and imposing them as b-vertices in a model derived from the one discussed in Section 2. In order to speed up the convergence of the method, it is important to consider the most promising sets of b-vertices first. In practice, it is likely not possible to have a computationally exact selection of the T most promising vertices with low computational effort (selecting b-vertices is the most challenging task in the solution of b-coloring). Consequently, heuristic criteria to sort the elements of P from the most promising to the least one have to be considered. In this section we describe the one we adopted.

The elements of the candidate set P are ordered by non-increasing number of neighbours $N(i)$ (with ties broken at random). The rationale behind the choice is that promising candidates for a b-vertex role are vertices with many neighbours. In the remainder of the section we will indicate with $p(i)$ the position of vertex $i \in P$ in this heuristic order.

3.3 An integer programming model for the decision version of b-coloring

Formally, after having decided a target value T for the objective function (1) and having selected $S \subseteq P$ as the set of T candidate b-vertices, the method we present operates on a decision model $DBC(S)$ obtained from BC by neglecting the objective function (1) and by adding the following constraints:

$$\sum_{i \in V} x_{ii} \geq T \tag{7}$$

$$x_{ii} = 1 \quad i \in S \tag{8}$$

The constraint (7) implies that we accept only feasible solutions with at least T b-vertices. Constraints (8) set to 1 the variables corresponding to the b-vertices imposed by the current set S .

The rationale behind the new model is that the new constraint (7), paired with the elimination of the objective function, leads to a model $DBC(S)$ that is experimentally substantially easier to solve than BC, and consequently suitable to be used within an iterative framework. In comparison with most of the methods that previously appeared in the literature, the idea we propose has the burden of having to set a bound on the target T for the number of colors of b-coloring solutions. This requires either a pre-existing knowledge of each instance, or the preliminary calculation of lower and upper bounds to calibrate the parameter itself.

3.4 The overall algorithm

The elements presented in the previous sections are now combined into the overall algorithm we propose. The main idea is to work on sets S of candidate b-vertices, superimposing them in a sequence of models $DBC(S)$, and solving the decision problems so obtained. This idea is iteratively repeated for all possible combinations of T elements out of P, or until a maximum allowed computation time has elapsed. Note that if the first condition is met, the method is exact and allows us to claim whether a b-coloring solution with at least T colors exists or not. Otherwise, the method is heuristic, and given the combinatorial explosion of the possible subsets S, this is the normal outcome of the current algorithm. We will however see in Section 4 how IMA can sometimes also produce exact solutions for some particular instances.

The order in which the combinations of S are examined is imposed by the function o (see Section 3.2), in such a way that the most promising subsets are examined first, increasing the rate of (early) success. Formally, a set $\mathcal{W}$ of possible subsets of P with T elements not yet examined is considered, and the next subset S to be examined is selected as follows:

$$S = \arg\min_{W \in \mathcal{W}} \left\{ \sum_{i \in W} o(i) \right\} \tag{9}$$

Note that in our implementation the set $\mathcal{W}$ is not generated explicitly, but its elements are dynamically generated in the order given by (9) through a textbook procedure to produce all possible combinations of a given set.

Algorithm 1 Iterative Matheuristic Algorithm (IMA)

Input: The information that has to be provided to the algorithm:
- a graph $G = (V, E)$;
- a target number of colors T;
- A policy to rank the b-vertex candidates;
- a maximum allowed computation time TM.

Output: Three possible alternative outcomes are possible:
- a b-coloring of G with at least T colours;
- a proof that a coloring with more than $T - 1$ colors does not exist;
- an inconclusive outcome in case the method runs out of time.

```
1:  $P \leftarrow$ candidates for the role of b-vertices (see Section 3.1);
2:  $o \leftarrow$ ranking of the elements of $P$ (see Section 3.2);
3:  $\mathcal{W} \leftarrow$ all possible combinations of $T$ elements out of $P$;
4:  while $\mathcal{W} \neq \emptyset$ and the computation time elapsed is less than $TM$ do
5:      $S \leftarrow$ element of $\mathcal{W}$ selected according to (9);
6:      $\mathcal{W} \leftarrow \mathcal{W} \setminus \{S\}$;
7:      Solve the model $DBC(S)$ (see Section 3.3);
8:      if $DBC(S)$ is feasible then
9:          $x \leftarrow$ optimal solution of $DBC(S)$;
10:         $T' \leftarrow \sum_{i \in V} x_{ii}$;
11:         Conclusive computation: return the b-coloring solution with $T'$ colours identified by variables $x$;
12:         exit;
13:     end if
14: end while
15: if $\mathcal{W} = \emptyset$ then
16:     Conclusive computation: no b-coloring solution with at least $T$ colours exists for $G$;
17: else
18:     Inconclusive computation due to maximum computation time $TM$ elapsed;
19: end if
```

A pseudocode for the algorithm IMA is presented in Algorithm 1. The preprocessing phase selects the set S of vertices that are candidates for acting as b-vertices and sorts them by non-decreasing values of the number of neighbors. All the possible combinations of T vertices out of S are then calculated and stored in $\mathcal{W}$. The iterative part of the algorithm is entered, and the elements of $\mathcal{W}$ are evaluated in turn until $\mathcal{W}$ is empty or the maximum time has elapsed. At each iteration, a set S is considered and it is checked whether a feasible solution exists for $DBC(S)$. The process is repeated until a feasible solution is found, it is proved that no solution exists, or the maximum computational time allowed has elapsed.

4 EXPERIMENTAL RESULTS

A campaign aiming at improving the results presented in [17], taken here as reference results, is presented. The 137 instances considered for the experiments are based on the DIMACS benchmark set originally proposed for the minimum coloring and the maximum clique problems in [12]. We attempt to run the algorithm IMA with values of T strictly greater than the best known results presented in [17] for each instance, and incrementally enlarge them until no improvement is found. In such a context, the method we propose can be seen as an incremental step to be run after another algorithm is used to provides initial upper bounds for the value of T. Another way of running IMA could be to use it in a binary search fashion for each graph G, starting from proper lower and upper bounds for the value of $X_b(G)$. We have not investigated this solution in the present study.

The method IMA has been coded in ANSI C, and all the experiments reported have been run on a computer equipped with an Intel Core i5 processor running at 1.6 GHz and 4 GB of RAM. For each run an overall maximum computation time TM of 1800 seconds has been allowed.

The Mixed Integer Program $DBC(S)$ is iteratively solved by Gurobi 9.1 [10] running on one thread only and with a maximum allowed computation time of 180 seconds for each iteration. After such a time has passed without finding a solution, it is assumed (heuristically) that the current set S (see Section 3.4) does not provide a feasible b-coloring.

The improved results found over the 78 maximum clique instances adopted in [17] are reported in Table 1. Namely, the 17 instances (21.8% of the total) for which an improved solution was retrieved are reported in the table. Columns contain, for each instance, the following information:

- V: number of vertices of the graph;
- E: number of edges of the graph;
- $m(G)$: upper bound on the value of $X_b(G)$ (calculated as described in [17]);
- Melo et al. [17]: number of colors of the best heuristic solution reported in [17];
- IMA: number of colors of the improved solution retrieved by algorithm IMA (with the relative percentage improvement reported in brackets).

From Table 1 we can observe how instances with different size and characteristics have been improved by IMA with respect to the state of the art. It is currently not possible to clearly identify on which instances the new method is performing better than the previous one, since there is no correlation between the results and typical characteristics of the graphs (even taking edge density into account).

Table 2 reports the same information as the previous table, but for the minimum coloring problem instances already adopted in [17], for which an improved solution was retrieved. In total, 21 instances out of 59 (35.6%) were improved.

We carried out a final test, aiming at showing how IMA can be effectively used as an exact algorithm by changing the maximum time allowed to solve each Integer Program $DBC(S)$ encountered from 180 seconds to unlimited. In such a way the algorithm is able to certify that no b-coloring solution exists with more than T colors when the exhaustive search of all possible candidate b-vertices is completed without finding any feasible solution. With such a technique it was possible to certify for the first time the optimality of some solutions previously reported in the literature. The 5 instances closed for the first time are summarized in Table 3, where most of the columns have the same meaning as before, while column named *[17]* $X_b(G)$ contains the old ranges for the optimal number of colors, and column *IMA* $X_b(G)$ contains the values we found.

5 CONCLUSIONS

B-coloring is a problem in graph theory. In this paper we presented a solution approach based on some considerations about the structure of promising solutions and on an integer linear programming model.

Experimental results showed how the new method is able to find high quality solutions for commonly established datasets from the literature. In particular, the method we propose was able to improve the best known upper bound for 38 instances of the 137 considered in previous work. The first optimality proof for the solutions of another 5 instances was finally provided.

Future work will be in the direction of enhancing the techniques described in this paper, which has in our opinion vast room for refinements and improvements. In particular, one crucial factor for the method we propose to perform efficiently as a heuristic, is the order in which candidate b-vertices are examined. Such a decision would strongly benefit from the use of machine learning techniques able to predict promising vertices based on complex structures otherwise hidden. Similar techniques have recently been applied successfully to other optimization problems.

REFERENCES

[1] Dominique Barth, Johanne Cohen, and Taoufik Faik. 2007. On the b-continuity property of graphs. *Discrete Applied Mathematics* 155, 13 (2007), 1761–1768. https://doi.org/10.1016/j.dam.2007.04.011
[2] Victor A. Campos, Carlos V. Lima, Nicolas A. Martins, Leonardo Sampaio, Marcio C. Santos, and Ana Silva. 2015. The b-chromatic index of graphs. *Discrete Mathematics* 338, 11 (2015), 2072–2079. https://doi.org/10.1016/j.disc.2015.04.026
[3] Victor A. Campos, Carlos V. Lima, and Ana Silva. 2013. B-Coloring Graphs with Girth at Least 8. *Proceedings of the Seventh European Conference on Combinatorics, Graph Theory and Applications* (2013), 327–332. https://doi.org/10.1007/978-88-7642-475-5_52
[4] Amal Dandashi and Mayez Al-Mouhamed. 2010. Graph Coloring for class scheduling. In *ACS/IEEE International Conference on Computer Systems and Applications - AICCSA 2010*. 1–4. https://doi.org/10.1109/AICCSA.2010.5586963
[5] Haytham Elghazel, Veronique Deslandres, Mohand-Said Hacid, Alain Dussauchoy, and Hamamache Kheddouci. 2006. A New Clustering Approach for Symbolic Data and Its Validation: Application to the Healthcare Data. *Lecture Notes in Computer Science* 4203 (2006), 373–482. https://doi.org/10.1007/11875604_54
[6] Iztok Fister, Iztok Peterin, Marjan Mernik, and Matej Črepinšek. 2015. Hybrid evolutionary algorithm for the b-chromatic number. *Journal of Heuristics* 21 (2015), 501–521. https://doi.org/10.1007/s10732-015-9288-z
[7] Djamel Gaceb, Veronique Eglin, Frank Lebourgeois, and Hubert Emptoz. 2008. Improvement of postal mail sorting system. *International Journal of Document Analysis and Recognition* 11 (2008), 67–80. https://doi.org/10.1007/s10032-008-0070-8
[8] Djamel Gaceb, Veronique Eglin, Frank Lebourgeois, and Hubert Emptoz. 2009. Robust Approach of Address Block Localization in Business Mail by Graph Coloring. *International Arab Journal of Information Technology* 6, 3 (2009), 221–229.
[9] James S. Graham, Roberto Montemanni, Jim N.J. Moon, and Derek H. Smith. 2008. Frequency assignment, multiple interference and binary constraints. *Wireless Networks* 14, 4 (2008), 449–464. https://doi.org/10.1007/s11276-006-0730-x
[10] Gurobi Optimization, LLC. 2021. Gurobi Optimizer Reference Manual. https://www.gurobi.com
[11] Robert W. Irving and David F. Manlove. 1999. The b-chromatic number of a graph. *Discrete Applied Mathematics* 91, 1 (1999), 127–141. https://doi.org/10.1016/S0166-218X(98)00146-2
[12] David S. Johnson and Michael A. Trick. 1996. Cliques, coloring, and satisfiability: second DIMACS implementation challenge. *American Mathematical Society* 26 (1996).
[13] Ivo Koch and Javier Marenco. 2019. An integer programming approach to b-coloring. *Discrete Optimization* 32 (2019), 43–62. https://doi.org/10.1016/j.disopt.2018.12.001

Table 1: Improved solution costs for clique instances from [12].

Instance				Melo et al.	IMA
Name	$\|V\|$	$\|E\|$	$m(G)$	[17]	
brock200_1.clq	200	14834	146	73	76 (+4.1%)
brock200_3.clq	200	12048	120	57	60 (+5.3%)
brock200_4.clq	200	13089	129	63	66 (+4.8%)
hamming8-4.clq	256	20864	164	48	52 (+8.3%)
hamming10-4.clq	1024	434176	849	157	165 (+5.1%)
johnson16-2-4.clq	120	5460	92	37	38 (+2.7%)
keller4.clq	171	9435	106	48	52 (+8.3%)
p_hat300-1.clq	300	10933	91	39	47 (+20.5%)
p_hat300-2.clq	300	21928	149	71	82 (+15.5%)
p_hat300-3.clq	300	33390	209	113	117 (+3.5%)
p_hat500-1.clq	500	31569	152	57	72 (+26.3%)
p_hat500-2.clq	500	62946	252	114	127 (+11.4%)
p_hat700-1.clq	700	60999	208	74	83 (+12.2%)
p_hat700-2.clq	700	121728	353	153	157 (+2.6%)
p_hat700-3.clq	700	183010	487	201	210 (+4.5%)
p_hat1000-3.clq	1000	371746	694	271	288 (+6.3%)
sanr400_0.5.clq	400	39984	201	74	80 (+8.1%)

Table 2: Improved solution costs for coloring instances from [12].

Instance				Melo et al.	IMA
Name	$\|V\|$	$\|E\|$	$m(G)$	[17]	
dsjc1000.1.col	1000	49626	112	44	45 (+2.3%)
dsjc125.5.col	125	3891	63	35	39 (+11.4%)
dsjc250.1.col	250	3218	33	20	24 (+20.0%)
dsjc500.1.col	500	12458	59	36	38 (+5.6%)
flat300_20_0.col	300	21375	144	56	61 (+8.9%)
flat1000_60_0.col	1000	245830	493	152	153 (+0.7%)
flat1000_76_0.col	1000	246708	494	153	154 (+0.7%)
le450_15a.col	450	8168	57	35	40 (+14.3%)
le450_15b.col	450	8169	56	34	40 (+17.6%)
le450_15c.col	450	16680	93	41	54 (+31.7%)
le450_15d.col	450	16750	92	46	54 (+17.4%)
le450_25a.col	450	8260	63	41	54 (+31.7%)
le450_25b.col	450	8263	60	39	52 (+33.3%)
le450_25c.col	450	17343	101	48	59 (+22.9%)
le450_25d.col	450	17425	99	48	60 (+25.0%)
le450_5a.col	450	5714	34	24	28 (+16.7%)
le450_5b.col	450	5734	34	22	28 (+27.3%)
le450_5c.col	450	9803	52	32	35 (+9.4%)
le450_5d.col	450	9757	52	26	36 (+38.5%)
school1.col	385	19095	117	58	70 (+20.7%)
school1_nsh.col	352	14612	101	49	59 (+20.4%)

[14] Ivo Koch and Iztok Peterin. 2015. The b-chromatic index of direct product of graphs. *Discrete Applied Mathematics* 190-191 (2015), 109–117. https://doi.org/10.1016/j.dam.2015.04.003

[15] Jan Kratochvíl, Zsolt Tuza, and Margit Voigt. 2002. The b-chromatic number of a graph. *Lecture Notes in Computer Science* 2573 (2002), 310–320. https://doi.org/10.1007/3-540-36379-3_27

[16] Enrico Malaguti and Paolo Toth. 2010. A survey on vertex coloring problems. *International Transactions in Operational Research* 17, 1 (2010), 1–34. https://doi.org/10.1111/j.1475-3995.2009.00696.x

[17] Rafael A. Melo, Michell F. Queiroz, and Marcio C. Santos. 2021. A matheuristic approach for the b-coloring problem using integer programming and a multi-start multi-greedy randomized metaheuristic. *European Journal of Operational Research* 295, 1 (2021), 66–81. https://doi.org/10.1016/j.ejor.2021.02.049

[18] Roberto Montemanni, Derek H. Smith, and Stuart M. Allen. 2001. Lower bounds for fixed spectrum frequency assignment. *Annals of Operations Research* 107, 1 (2001), 237–250. https://doi.org/10.1023/A:1014911401612

Table 3: Instances from [12] closed for the first time.

Instance			Melo et al. [17]	IMA
Name	$\|V\|$	$\|E\|$	$X_b(G)$	$X_b(G)$
mulsol.i.3.col	184	3916	[52, 54]	52
mulsol.i.4.col	185	3946	[52, 54]	52
mulsol.i.5.col	186	3973	[53, 55]	53
R125.5.col	125	3838	[60, 61]	60
R250.1.col	250	867	[12, 13]	12

[19] Roberto Montemanni, Derek H. Smith, and Stuart M. Allen. 2004. An improved algorithm to determine lower bounds for the fixed spectrum frequency assignment problem. *European Journal of Operational Research* 156, 3 (2004), 736–751. https://doi.org/10.1016/S0377-2217(03)00127-9

[20] Roberto Montemanni, Derek H. Smith, Andrea E. Rizzoli, and Luca Maria Gambardella. 2009. Sequential ordering problems for crane scheduling in port terminals. *International Journal of Simulation and Process Modelling* 5, 4 (2009), 248–261. https://doi.org/10.1504/IJSPM.2009.032597