A column generation based algorithm for the robust graph coloring problem

Discrete Applied Mathematics 217 (2017) 340–352

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

A column generation based algorithm for the robust graph coloring problem

Birol Yüceoğlu [a,1], Güvenç Şahin [b,*], Stan P.M. van Hoesel [c]

[a] R&D Center, Information Technologies Department, Migros T.A.Ş, Ataşehir, 34758, İstanbul, Turkey

[b] Industrial Engineering, Faculty of Engineering and Natural Sciences, Sabanci University, Orhanli, Tuzla, 34956, Istanbul, Turkey

[c] Department of Quantitative Economics, Maastricht University, P.O. Box 616, 6200 MD, Maastricht, The Netherlands

ARTICLE INFO

Article history:
Received 17 June 2015
Received in revised form 27 June 2016
Accepted 4 September 2016
Available online 24 September 2016

Keywords:
Robust graph coloring
Representatives formulation
Set-covering formulation
Column generation
Reduced cost fixing

ABSTRACT

Given an undirected simple graph G, an integer k, and a cost c_{ij} for each pair of non-adjacent vertices in G, a robust coloring of G is the assignment of k colors to vertices such that adjacent vertices get different colors and the total penalty of the pairs of vertices having the same color is minimum. The problem has applications in fields such as timetabling and scheduling. We present a new formulation for the problem, which extends an existing formulation for the graph coloring problem. We also discuss a column generation based solution method. We report computational study on the performance of alternative formulations and the column generation method.

1. Introduction

For a given undirected graph $G = (V, E)$, with V the set of vertices and E the set of edges, a (*vertex*) *coloring* of a graph is an assignment of colors to vertices such that no two vertices of an edge get the same color. A *k-coloring* of a graph uses k colors. The minimum number of colors necessary to color a graph is called the *chromatic number* of G and is denoted by $\chi(G)$.

The graph coloring problem has its origins from the need to color maps with a minimum number of colors. The problem dates back to 1852, when Francis Guthrie conjectured that a map could be colored using four colors, Kubale [13]. The problem of coloring a map (which can be transformed into a planar graph) is polynomially solvable, however the problem is *NP*-hard in general graphs, Karp [11]. The problem has applications in register allocation problem for compiler optimization [8], exam scheduling [15], frequency assignment in telecommunications [1], and course timetabling [5].

In scheduling, activities (such as duties to be assigned to operators) are represented by the vertices of a graph. Two activities that cannot be assigned to the same operator because of a time slot or equipment clash, are connected by an edge. Operators are assigned sets of tasks they can carry out without a clash. In timetabling, activities (such as courses or conference sessions to be assigned to time slots) are represented by the vertices of a graph. If two courses are taken by the same students or two conference sessions are similar in content the corresponding vertices are connected by an edge and those activities cannot be assigned to the same time slot.

Finding an optimal coloring in the context of scheduling and timetabling requires the problem data to be known with certainty apriori. In scheduling, activities are subject to delays. Therefore it may be undesirable to assign two activities to

* Corresponding author.
E-mail address: guvencs@sabanciuniv.edu (G. Şahin).

[1] The research was carried while the author was a post-doctoral researcher at Sabancı University.

http://dx.doi.org/10.1016/j.dam.2016.09.006
0166-218X/

the same operator if probability of a delay is high or a possible delay is costly. Similarly, in a course timetabling problem, it may not always be possible to create a timetable without any clashes. Furthermore, it is difficult to predict the choices of students beforehand. Therefore, it is desirable to create a timetable where number of clashes are minimized. In conference timetabling, two sessions that can be assigned to the same time slot can still attract the same set of people. Even though eliminating every possible clash may be difficult, it is still possible to minimize clashes.

The robust graph coloring problem is a generalization of the original graph coloring problem. As in the original problem, adjacent vertices are not allowed to take the same color while having the same color may only be undesirable and penalized for other vertices in the robust version. The major difference is that the objective function is not to minimize the number of colors but to minimize the sum of penalties due to pairs of vertices with the same color. The problem is introduced by Yáñez and Ramírez [21]. They show that the original graph coloring problem can be too restrictive if one also considers secondary objectives. They present an assignment type of integer linear programming (ILP) formulation and use a genetic algorithm to solve the problem. Lim and Wang [14] use the robust graph coloring problem to model the robust aircraft assignment problem; they employ heuristics to solve the problem. Guo et al. [9] and Kong et al. [12] also present heuristics. Wang and Xu [20] model the problem as an unconstrained quadratic programming problem and develop new heuristics.

To the best of our knowledge, the only exact approach for the robust graph coloring problem is the column generation approach in [2]. Archetti et al. [2] use the ILP formulation of Yáñez and Ramírez [21]. After establishing that the formulation is only suitable for small instances, they present a branch-and-price algorithm. They use both heuristics and exact methods to generate new columns.

We present a new ILP formulation for the robust graph coloring problem based on the asymmetric representatives formulation of Campêlo et al. [6], which is originally used for the graph coloring problem. The formulation, introduced by Campêlo et al. [7], uses the idea that vertices with the same color can represent each other. Furthermore, Campêlo et al. [6] use an ordering of the vertices to create the asymmetrical representative formulation. This enhanced formulation reduces the number of variables and different representation of color classes compared to the original representatives formulation. Furthermore, the asymmetric representatives formulation eliminates symmetries that result from the interchangeability of colors compared to the original formulation of Méndez-Díaz and Zabala [18]. We compare asymmetric representatives formulation with the original formulation in [21]; we show that it performs better than the original as it yields a much improved lower bound for the problem but is still heavily restricted by the size of the instances solved to optimality.

Even though the asymmetric representatives formulation performs better than the original formulation, it does not have a noticeable impact on the size of the instances solved to optimality. For this reason, we use the set-covering formulation in [2] and develop a column generation-based algorithm to solve it. Unlike Archetti et al. [2], we do not use branch-and-price; we employ a method in [19]. Even though this method enumerates all columns in the worst case, it performs well empirically.

In Section 2, we present our notation and the new ILP formulation based on the asymmetric representatives formulation. In Section 3, we develop a set-covering based formulation and our column generation based solution method. Computational experiments are presented in these sections. We discuss the results and conclude in Section 4.

2. Asymmetric representatives formulation

Given a simple, undirected, and connected graph $G = (V, E)$, where $n = |V|$ is the number of vertices and $m = |E|$ is the number of edges, two vertices i and j are *adjacent* if $\{i, j\} \in E$. $N(i) = \{j \in V \mid \{i, j\} \in E\}$ is called the *neighborhood* of i. An *ordering* of G is a mapping $\sigma : V \to \{1, \ldots, n\}$, where $\sigma(i)$ denotes the position of i in the ordering; we use an ordering of the vertices to eliminate the symmetries in the problem. We identify each vertex with its position in the ordering, i.e., the vertices are numbered $1, 2, \ldots, n$. For a given ordering of G, we call $N^-(i) = \{1, 2, \ldots, i-1\} \cap N(i)$ the *in-neighborhood* of i and $N^+(i) = \{i+1, i+2, \ldots, n\} \cap N(i)$ the *out-neighborhood* of i. $\bar{G} = (V, \bar{E})$ denotes the complement of G, where $\bar{E}$ consists of $\{i, j\} \notin E$; $\bar{N}(i) = \{j \in V \mid \{i, j\} \in \bar{E}\} \setminus \{i\}$ is called the *antineighborhood* of i. The in- and out-antineighborhoods of i in $\bar{G}$ are defined similarly as $\bar{N}^+(i)$ and $\bar{N}^-(i)$. The closed (anti)neighborhoods, where i is included, are denoted by $N[i]$, $N^-[i]$, $N^+[i]$, $\bar{N}[i]$, $\bar{N}^-[i]$, $\bar{N}^+[i]$. $\bar{N}^+[i]$ corresponds to the vertices that can be represented by i (including itself). $\bar{N}^-[i]$ corresponds to the vertices that can represent i (including itself).

We call $H = (V_H, E_H)$ an *induced subgraph* of G if $V_H \subseteq V$ and $\{i, j\} \in E_H$ if and only if $i \in V_H, j \in V_H$ and $\{i, j\} \in E$. H is called a *clique* if all vertices in H are pairwise adjacent. Each vertex of a clique has to have a different color. An *independent set* is a set of vertices, in which no two vertices are adjacent. In other words, H is an independent set if $E_H = \emptyset$. In any coloring, vertices having the same color form an independent set.

2.1. Mathematical model

We modify the asymmetric representatives formulation introduced by Campêlo et al. [6] which selects a subset of the vertices to represent the colors. Representative vertices can represent other vertices in their out-antineighborhood. The vertices represented by a representative vertex must form an independent set. Vertices that do not represent a color are identified by the color of their representatives, i.e., a vertex in the in-antineighborhood of i. c_{ij} denotes a non-negative cost associated with two vertices i and j such that $\{i, j\} \notin E$, it can be considered as the penalty of coloring two vertices with the same color.

We define two types of decision variables:

- Representation variables are used in order to determine colors of the vertices. For each vertex i, we define representation variables only for the vertices $j \in \bar{N}^+[i]$; for $i \leq j$ and $\{i, j\} \notin E$,

$$x_{ij} = \begin{cases} 1, & \text{if vertex } i \text{ represents vertex } j, \\ 0, & \text{otherwise.} \end{cases}$$

  If $x_{ii} = 1$, the vertex i is called a representative vertex; otherwise, i is represented by a representative vertex.
- Color class variables are used in order to determine whether two vertices have the same color. For each vertex i, we define color class variables only for the vertices $j \in \bar{N}^+(i)$; for $i < j$ and $\{i, j\} \notin E$,

$$y_{ij} = \begin{cases} 1, & \text{if vertex } i \text{ and } j \text{ have the same color,} \\ 0, & \text{otherwise.} \end{cases}$$

The asymmetric representatives formulation of the robust graph coloring problem is

$$\min \sum_{i \in V} \sum_{j \in \bar{N}^+(i)} c_{ij} y_{ij}, \tag{1}$$

$$\text{subject to} \sum_{i \in V} x_{ii} = k, \tag{2}$$

$$\sum_{i \in \bar{N}^-[j]} x_{ij} = 1, \qquad \forall j \in V, \tag{3}$$

$$\sum_{j \in C} x_{ij} \leq x_{ii}, \qquad \forall i \in V, \forall C \subseteq \bar{N}^+(i), C \text{ maximal clique}, \tag{4}$$

$$x_{ij} + x_{ik} \leq 1 + y_{jk}, \qquad \forall i \in V, \forall j, k \in \bar{N}^+[i], j < k, \{j, k\} \notin E \tag{5}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i \in V, \forall j \in \bar{N}^+[i], \tag{6}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall i \in V, \forall j \in \bar{N}^+(i). \tag{7}$$

The objective function (1) minimizes total weight of coloring the vertices. Constraint (2) limits the number of colors used (or the number of representative vertices) to k. Constraints (3) ensure that each vertex j is represented only once, either by itself or by a representative vertex that can represent j. Constraints (4) guarantee that a representative vertex i represents at most one vertex of any maximal clique in its (open) out-antineighborhood. In our formulation, we check pairs and triplets of vertices to generate the inequalities since it is not possible to generate all maximal cliques in a graph in polynomial time, Bron and Kerbosch [4]. Note that a maximal clique can be a singleton vertex or two vertices that are connected to each other. Constraints (5) assert that for two distinct vertices j and k with the same representative, the variable y_{jk} is selected. Constraints (6)–(7) are the domain constraints for the variables.

2.2. Computational results

We conduct computational experiments in order to compare the asymmetric representatives formulation with the original formulation in [21]. We use two sets of instances that are also used by Archetti et al. [2]. The first set of instances (see Table 1) are taken from the DIMACS graph coloring library by setting $c_{ij} = ij$, for distinct vertices i and j that are not adjacent. The number of colors is set to $\lceil \frac{3}{2}\bar{k} \rceil$ and $\lceil 2\bar{k} \rceil$, where $\bar{k}$ is the best known upper bound on the chromatic number. We consider instances with at most 150 vertices. The problem instance miles1500 with $k = \lceil 2\bar{k} \rceil = 156$ cannot be solved as the number of colors exceeds the number of vertices; corresponding cells are shaded in gray Table 1. The second set of instances are generated by Archetti et al. [2] for the robust graph coloring problem (see Tables 2 and 3). The name of the instances contains information on the number of vertices n, the number of colors k, and an index for the instance (with five instances for each values of n and k). The number of colors is set to $\lceil \frac{n}{3} \rceil$ and $\lceil \frac{n}{3} \rceil + 1$. The cost coefficients are generated according to a uniform distribution in the interval (0, 1) and the graphs have a density of 0.5. rcp_10_4_2 and rcp_10_4_4 are infeasible since they contain a clique of size 5.

The experiments are carried out on an Intel Core i7, 3.2 GHz Windows desktop using CPLEX 12.6 to solve the problem with default settings [10]. The processing time is limited to one hour. In order to speed up the solver, we find a maximum clique as part of preprocessing, as suggested by Campêlo et al. [6]. This allows us to fix some variables, as each vertex of a clique must have a different color. Even though the formulations do not necessarily require this process, it allows us to solve more instances to optimality.

The results are presented in Tables 1–3. In the tables, $\underline{z}^*$ denotes the lower bound and $\bar{z}^*$ denotes the upper bound. We also provide an optimality gap ($\frac{\bar{z}^* - \underline{z}^*}{\bar{z}^*} 100$) in percentages. We give the computation time (in seconds) for instances solved to optimality in the tables. For the other instances, the one hour time limit is reached. On DIMACS instances, as shown in Table 1, the asymmetric representatives formulation almost always gives a better lower bound and is able to

solve larger instances: miles1500 with $k = 100$ and DSJC125_9 with $k = 88$. In some sparse graphs, such as 1-FullIns_3 and 2-Insertion_3 with $k = 6$, the original formulation provides a better lower bound. The asymmetric representatives formulation performs better as the edge density increases and the number of variables decreases. A similar observation is made by Matsui et al. [16]. We do not observe significant differences in terms of the quality of the upper bounds, even though the asymmetric representatives formulation performs slightly better.

For the second set of instances, as shown in Tables 2 and 3, the asymmetric representatives formulation performs much better than the original formulation. The largest gap with the asymmetric representatives formulation was less than 55%; 46 instances out of 148 are solved to optimality. Only the smallest 28 instances are solved to optimality using the original formulation. For other instances, the gap is always greater than 80%. The instances in this set are easier to solve due to high values of k. However, the asymmetric representatives formulation is still limited on the size of instances solved to optimality. We observe that the formulation provides a good upper bound by comparing the results with the bounds in [2]; but, we are not able to prove optimality.

3. Set-covering type formulation and column generation

Both column generation and branch-and-price are frequently used to solve the graph coloring problem [17], and the robust graph coloring problem [2]. As an alternative to both the traditional formulation in [21] and asymmetric representatives formulation explored in Section 2, we use the set-covering type formulation in [2] and a column generation based solution approach using this formulation.

Our algorithm is composed of two phases: a column generation phase where the LP relaxation of the master problem (MP) is solved to optimality and an integer solution phase where the original integer problem is solved. For the column generation method, we use independent sets, which correspond to color classes. An independent set is a set of vertices in which no two vertices are adjacent. Obviously, an independent set can be colored using only one color. We use $\mathbb{S}$ to denote the set of all independent sets. With all independent sets, it is possible to formulate a set-covering type of formulation for the robust graph coloring problem. As the number of independent sets in a graph is typically exponential in the number of vertices, it is computationally inefficient (and possibly not doable) to generate the problem including all independent sets that is the MP. Therefore, we employ column generation, which does not necessarily require generating all independent sets apriori.

The column generation phase is initiated with a restricted master problem (RMP) containing only a subset of independent sets, $\bar{\mathbb{S}} \subseteq \mathbb{S}$. We solve the LP relaxation of RMP, denoted as RMP_{LP}. Given the optimal values of dual variables, new independent sets represented by columns with negative reduced cost are added to RMP until there is no such column. When there exists no such column left out, an optimal solution to RMP_{LP} is also an optimal solution to the LP relaxation of MP and provides a lower bound for the problem. If an optimal solution to RMP_{LP} is integer, it is also an optimal solution to the original integer problem. If this is not the case, we continue with the second phase of our algorithm. Traditionally, a branch-and-bound method is employed along with the column generation procedure for each node of the branch-and-bound tree. This method is called branch-and-price [3]. In our approach, we use a result of Nemhauser and Wolsey [19] which proves optimality of the integer MP by comparing the reduced cost of a column to be added to RMP against the difference between the lower bound and the upper bound of the problem. As a result, we do not resort to a branch-and-price algorithm.

3.1. Mathematical model

An independent set $S \in \mathbb{S}$ corresponds to a color class if it is used in the coloring of the graph. The cost of choosing the independent set S is denoted by $c_S = \sum_{i \in S} \sum_{j \in S: i<j} c_{ij}$. The decision variables are

$$\sigma_S = \begin{cases} 1, & \text{if independent set } S \text{ is used in coloring,} \\ 0, & \text{otherwise.} \end{cases}$$

The ILP formulation for the robust graph coloring problem as in [2] is

$$\min \sum_{S \in \mathbb{S}} c_S \sigma_S, \tag{8}$$

$$\text{subject to} \sum_{S \in \mathbb{S}: i \in S} \sigma_S \geq 1, \qquad \forall i \in V, \tag{9}$$

$$\sum_{S \in \mathbb{S}} \sigma_S = k, \tag{10}$$

$$\sigma_S \in \{0, 1\}, \qquad \forall S \in \mathbb{S}. \tag{11}$$

The objective function (8) minimizes the total weight of coloring the vertices. Constraints (9) ensure that each vertex i is colored by one color. Constraint (10) limits the number of colors that we use to k. Constraints (11) are the domain constraints for the variables. Solving the robust graph coloring problem using formulation (8)–(11), corresponding to MP, requires knowing all independent sets in G beforehand. Therefore, we start only with an initial set of columns (RMP) and use column generation to find new columns with negative reduced cost.

Table 1
Results with the original formulation and asymmetric representatives formulation for the DIMACS instances.

Instance			k	Asymmetric representatives				Original				k	Asymmetric representatives				Original			
Graph	$\|V\|$	$\|E\|$		$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)		$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)
myciel3	11	20	6	110.00	110		0.13	110.00	110		0.05	8	28.00	28		0.02	28.00	28		0.05
myciel4	23	71	8	1645.33	2600	36.72		980.25	2600	62.30		10	1136.80	1648	31.02		103.00	1648	93.75	
queen5_5	25	160	8	3850.00	3850		65.52	3850.00	3850		53.51	10	2521.00	2521		318.06	2521.00	2521		1187.74
1-FullIns_3	30	100	6	5508.96	13296	58.57		10277.90	13292	22.68		8	2827.53	8789	67.83		4069.00	8787	53.69	
queen6_6	36	290	11	6022.26	12078	50.14		4118.16	12084	65.92		14	3450.03	7750	55.48		385.00	7752	95.03	
2-Insertions_3	37	72	6	4802.41	32401	85.18		10008.70	32404	69.11		8	2466.48	22108	88.84		1178.56	22109	94.67	
myciel5	47	236	9	5293.58	52854	89.98		137.00	52919	99.74		12	4152.60	35295	88.23		0.00	35331	100.00	
queen7_7	49	476	11	22617.90	48073	52.95		20737.60	48104	56.89		14	11591.40	33447	65.34		4190.86	33575	87.52	
2-FullIns_3	52	201	8	6636.77	94613	92.99		5096.70	94583	94.61		10	5296.18	70955	92.54		333.00	71004	99.53	
3-Insertions_3	56	110	6	10504.20	182310	94.24		19005.70	182402	89.58		8	5500.42	129280	95.75		489.00	129294	99.62	
queen8_8	64	728	14	31866.00	109976	71.02		29577.90	110178	73.15		18	7712.14	75628	89.80		1476.00	75793	98.05	
1-Insertions_4	67	232	8	9027.98	274026	96.71		100.00	273736	99.96		10	8013.80	208321	96.15		0.00	209217	100.00	
huck	74	602	17	10404.30	157841	93.41		1025.00	158892	99.35		22	8319.04	106354	92.18		0.00	106897	100.00	
4-Insertions_3	79	156	6	23304.70	749642	96.89		29315.40	748457	96.08		8	10957.50	542507	97.98		0.00	541131	100.00	
3-FullIns_3	80	346	9	18412.90	496298	96.29		5541.75	497182	98.89		12	11840.10	351389	96.63		0.00	353835	100.00	
jean	80	508	15	11711.70	263573	95.56		1954.30	264294	99.26		20	9694.55	179278	94.59		3.00	177077	100.00	
queen9_9	81	1056	15	70841.90	279519	74.66		63405.90	282678	77.57		20	11686.20	188386	93.80		1682.00	189199	99.11	
david	87	812	17	12918.60	322863	96.00		232.00	322483	99.93		22	10915.10	224535	95.14		0.00	223536	100.00	
mug88_1	88	146	6	62478.70		100.00		139080.00		100.00		8	14407.30	845810	98.30		427444.00	844650	49.39	
mug88_25	88	146	6	59996.30		100.00		137487.00		100.00		8	15880.40	845124	98.12		1157.75	846248	99.86	
1-FullIns_4	93	593	8	26982.70		100.00		1195.00		100.00		10	16743.50	819462	97.96		0.00	824155	100.00	
myciel6	95	755	11	22025.80	810339	97.28		0.00	801705	100.00		14	19790.50	606964	96.74		0.00	611073	100.00	
queen8_12	96	2736	18	156679.00	455840	65.63		154935.00	459659	66.29		24	22063.40	305624	92.78		7606.00	308575	97.54	
mug100_1	100	166	6	84379.90		100.00		183508.00		100.00		8	16084.30		100.00		0.00		100.00	
mug100_25	100	166	6	85461.40		100.00		205995.00		100.00		8	16321.60		100.00		0.00		100.00	
queen10_10	100	1470	17	122471.00	588679	79.20		110622.00	591109	81.29		22	20366.00	415267	95.10		740.80	416169	99.82	
4-FullIns_3	114	541	11	25635.20		100.00		1417.00		100.00		14	20648.00		100.00		0.00		100.00	
games120	120	1276	14	37699.40		100.00		19064.00		100.00		18	27373.00		100.00		510.50		100.00	
queen11_11	121	1980	17	94524.40		100.00		75020.50		100.00		22	37343.00	995871	96.25		652818.00	973030	32.91	
DSJC125_1	125	736	8	46267.10		100.00		53390.30		100.00		10	26852.20		100.00		741621.00		100.00	
DSJC125_5	125	3891	26	54959.10	871159	93.69		0.00	978204	100.00		34	41893.30	631611	93.37		0.00	681919	100.00	
DSJC125_9	125	6961	66	135405.00	142722	5.13		0.00	155902	100.00		88	36351.00	36351	100.00	16.02	0.00	41867	100.00	
miles1000	128	6432	63	39240.30	190474	79.40		1661.00	201278	99.17		84	12256.60	59137	79.27		0.00	61729	100.00	
miles1500	128	10396	110	4284.00	4284		88.02	28.00	4285	99.35		156								
miles250	125	774	12	52641.00		100.00		46692.60		100.00		16	28020.70		100.00		520.00		100.00	
miles500	128	2340	30	36543.70	787691	95.36		2953.28	799781	99.63		40	24681.70	517685	95.23		0.00	516171	100.00	
miles750	128	4226	47	45463.00	379453	88.02		3317.50	390249	99.15		62	22266.80	214455	89.62		0.00	214181	100.00	
anna	138	986	17	28919.90		100.00		504.00		100.00		22	25279.10		100.00		0.00		100.00	
queen12_12	144	5192	18	606136.00		100.00		590505.00		100.00		24	56270.00		100.00		14235.00		100.00	
2-Insertions_4	149	541	6	39402.90		100.00		14006.00		100.00		8	32126.70		100.00		0.00		100.00	

Table 2
Results with the original formulation and asymmetric representatives formulation for the instances of Archetti et al. [2] (part 1).

Instance	Asymmetric representatives				Original				Instance	Asymmetric representatives				Original			
	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)		$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)
rcp_10_4_1	1.747	1.747		0.02	1.747	1.747		0.01	rcp_10_5_1	1.929	1.929		0.00	1.929	1.929		0.00
rcp_10_4_2									rcp_10_5_2	0.997	0.997		0.00	0.997	0.997		0.02
rcp_10_4_3	3.612	3.612		0.02	3.612	3.612		0.00	rcp_10_5_3	1.796	1.796		0.00	1.796	1.796		0.00
rcp_10_4_4									rcp_10_5_4	1.599	1.599		0.00	1.599	1.599		0.01
rcp_10_4_5	1.733	1.733		0.02	1.733	1.733		0.06	rcp_10_5_5	2.659	2.659		0.02	2.659	2.659		0.02
rcp_20_7_1	5.931	5.931		0.20	5.931	5.931		0.19	rcp_20_8_1	4.838	4.838		0.22	4.838	4.838		0.33
rcp_20_7_2	6.290	6.290		0.27	6.290	6.290		0.31	rcp_20_8_2	3.184	3.184		0.20	3.184	3.184		0.44
rcp_20_7_3	5.020	5.020		0.22	5.020	5.020		0.23	rcp_20_8_3	2.169	2.169		0.06	2.169	2.169		0.23
rcp_20_7_4	5.914	5.914		0.20	5.914	5.914		0.20	rcp_20_8_4	3.669	3.669		0.19	3.669	3.669		0.22
rcp_20_7_5	5.645	5.645		0.22	5.645	5.645		0.14	rcp_20_8_5	3.815	3.815		0.20	3.815	3.815		0.22
rcp_30_10_1	6.091	6.091		9.52	6.091	6.091		18.94	rcp_30_11_1	5.378	5.378		1.56	5.378	5.378		20.17
rcp_30_10_2	7.407	7.407		7.49	7.407	7.407		48.88	rcp_30_11_2	5.108	5.108		0.78	5.108	5.108		3.57
rcp_30_10_3	8.483	8.483		11.72	8.483	8.483		13.51	rcp_30_11_3	4.897	4.897		1.36	4.897	4.897		67.42
rcp_30_10_4	6.570	6.570		3.45	6.570	6.570		15.07	rcp_30_11_4	5.068	5.068		1.45	5.068	5.068		10.19
rcp_30_10_5	6.737	6.737		12.49	6.737	6.737		275.45	rcp_30_11_5	4.897	4.897		0.72	4.897	4.897		12.20
rcp_40_14_1	5.970	5.970		26.71	0.642	6.519	90.16		rcp_40_15_1	5.367	5.367		13.90	0.000	18.778	100.00	
rcp_40_14_2	6.242	6.242		108.34	0.346	6.712	94.85		rcp_40_15_2	4.819	4.819		12.65	0.000	20.633	100.00	
rcp_40_14_3	7.657	7.657		215.05	1.008	7.721	86.95		rcp_40_15_3	5.224	5.224		26.44	0.000	18.178	100.00	
rcp_40_14_4	6.146	6.146		26.47	0.985	6.153	83.99		rcp_40_15_4	5.544	5.544		37.88	0.000	23.350	100.00	
rcp_40_14_5	6.185	6.185		29.52	0.767	6.510	88.22		rcp_40_15_5	4.939	4.939		17.57	0.000	16.893	100.00	
rcp_50_17_1	7.687	7.687		908.60	0.000	16.290	100.00		rcp_50_18_1	6.777	6.777		464.26	0.000	27.467	100.00	
rcp_50_17_2	7.364	7.364		1887.88	0.000	20.343	100.00		rcp_50_18_2	6.678	6.678		235.91	0.000	21.837	100.00	
rcp_50_17_3	6.852	6.852		66.71	0.000	17.463	100.00		rcp_50_18_3	5.529	5.529		106.11	0.000	22.432	100.00	
rcp_50_17_4	7.015	8.189	14.33		0.000	15.764	100.00		rcp_50_18_4	6.812	6.812		201.49	0.000	21.511	100.00	
rcp_50_17_5	7.414	8.179	9.35		0.000	16.187	100.00		rcp_50_18_5	6.466	6.466		195.24	0.000	22.278	100.00	
rcp_60_20_1	9.054	9.123	0.75		0.000	20.682	100.00		rcp_60_21_1	6.643	7.112	6.59		0.000	48.205	100.00	
rcp_60_20_2	7.519	8.983	16.30		0.000	19.573	100.00		rcp_60_21_2	7.887	8.626	8.57		0.000	45.562	100.00	
rcp_60_20_3	7.737	8.604	10.07		0.000	20.289	100.00		rcp_60_21_3	6.951	8.109	14.28		0.000	46.956	100.00	
rcp_60_20_4	7.518	7.816	3.81		0.000	15.677	100.00		rcp_60_21_4	7.725	7.725		2281.53	0.000	61.797	100.00	
rcp_60_20_5	7.202	8.406	14.33		0.000	21.197	100.00		rcp_60_21_5	6.487	8.022	19.13		0.000	55.719	100.00	
rcp_70_24_1	8.208	9.320	11.93		0.000	41.536	100.00		rcp_70_25_1	6.212	6.749	7.96		0.000	62.086	100.00	
rcp_70_24_2	6.844	7.753	11.73		0.000	48.353	100.00		rcp_70_25_2	6.973	6.973		3463.52	0.000	65.438	100.00	
rcp_70_24_3	7.361	8.001	7.99		0.000	53.098	100.00		rcp_70_25_3	6.462	6.890	6.21		0.000	62.640	100.00	
rcp_70_24_4	7.394	8.823	16.20		0.000	47.172	100.00		rcp_70_25_4	7.178	7.963	9.85		0.000	59.702	100.00	
rcp_70_24_5	7.274	8.385	13.25		0.000	40.466	100.00		rcp_70_25_5	7.136	8.214	13.12		0.000	55.843	100.00	
rcp_80_27_1	8.230	10.659	22.79		0.000	61.191	100.00		rcp_80_28_1	6.961	7.816	10.94		0.000	69.989	100.00	
rcp_80_27_2	8.436	10.565	20.15		0.000	51.997	100.00		rcp_80_28_2	7.351	9.424	22.00		0.000	76.407	100.00	
rcp_80_27_3	7.703	9.300	17.17		0.000	60.758	100.00		rcp_80_28_3	7.390	8.848	16.48		0.000	67.699	100.00	
rcp_80_27_4	7.685	9.484	18.97		0.000	58.254	100.00		rcp_80_28_4	6.827	7.621	10.41		0.000	64.630	100.00	
rcp_80_27_5	7.707	11.142	30.83		0.000	56.301	100.00		rcp_80_28_5	6.484	7.731	16.14		0.000	73.206	100.00	

Table 3
Results with the original formulation and asymmetric representatives formulation for the instances of Archetti et al. [2] (part 2).

Instance	Asymmetric representatives				Original				Instance	Asymmetric representatives				Original			
	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)		$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)
rcp_90_30_1	8.455	11.366	25.61		0.000	62.470	100.00		rcp_90_31_1	7.486	9.526	21.41		0.000	80.818	100.00	
rcp_90_30_2	7.720	10.673	27.67		0.000	63.474	100.00		rcp_90_31_2	7.492	9.338	19.77		0.000	65.049	100.00	
rcp_90_30_3	8.501	11.097	23.39		0.000	68.596	100.00		rcp_90_31_3	7.175	9.347	23.24		0.000	63.968	100.00	
rcp_90_30_4	7.783	11.072	29.70		0.000	71.900	100.00		rcp_90_31_4	7.214	9.230	21.84		0.000	83.000	100.00	
rcp_90_30_5	7.339	9.842	25.43		0.000	59.161	100.00		rcp_90_31_5	6.331	7.667	17.43		0.000	59.838	100.00	
rcp_100_34_1	7.546	10.429	27.64		0.000	60.969	100.00		rcp_100_35_1	7.068	8.545	17.28		0.393	5.391	92.71	
rcp_100_34_2	7.360	9.572	23.11		0.000	78.657	100.00		rcp_100_35_2	7.473	10.228	26.94		0.666	4.958	86.57	
rcp_100_34_3	7.622	9.592	20.53		0.000	73.122	100.00		rcp_100_35_3	7.090	9.125	22.31		0.609	5.383	88.69	
rcp_100_34_4	7.884	10.398	24.17		0.000	77.736	100.00		rcp_100_35_4	6.419	8.717	26.36		0.865	5.996	85.58	
rcp_100_34_5	7.596	10.778	29.52		0.000	65.471	100.00		rcp_100_35_5	6.966	8.105	14.05		0.741	5.086	85.44	
rcp_110_37_1	8.076	11.712	31.05		0.364	9.461	96.15		rcp_110_38_1	7.091	10.274	30.98		0.117	7.879	98.52	
rcp_110_37_2	7.054	10.614	33.54		0.045	8.555	99.47		rcp_110_38_2	7.652	10.674	28.31		0.202	6.820	97.04	
rcp_110_37_3	7.967	11.693	31.87		0.170	7.733	97.80		rcp_110_38_3	7.162	9.792	26.86		0.236	6.269	96.24	
rcp_110_37_4	7.648	12.293	37.78		0.097	8.453	98.85		rcp_110_38_4	6.522	9.066	28.06		0.058	7.163	99.19	
rcp_110_37_5	8.218	11.478	28.40		0.297	8.875	96.65		rcp_110_38_5	7.594	11.045	31.24		0.108	7.023	98.46	
rcp_120_40_1	7.781	12.844	39.42		0.138	11.956	98.85		rcp_120_41_1	7.422	12.584	41.02		0.023	9.379	99.75	
rcp_120_40_2	8.073	13.416	39.83		0.019	10.824	99.82		rcp_120_41_2	8.110	11.695	30.65		0.030	10.186	99.71	
rcp_120_40_3	7.944	14.259	44.29		0.131	9.984	98.69		rcp_120_41_3	7.162	10.887	34.22		0.025	8.893	99.72	
rcp_120_40_4	8.014	13.563	40.91		0.184	9.166	97.99		rcp_120_41_4	7.627	10.974	30.50		0.083	8.690	99.04	
rcp_120_40_5	8.496	13.753	38.23		0.028	10.400	99.73		rcp_120_41_5	7.581	10.689	29.08		0.055	8.988	99.39	
rcp_130_44_1	8.433	14.731	42.76		0.004	11.404	99.96		rcp_130_45_1	7.173	10.577	32.19		0.000	10.968	100.00	
rcp_130_44_2	8.004	13.818	42.08		0.000	11.370	100.00		rcp_130_45_2	7.384	11.101	33.48		0.012	8.649	99.86	
rcp_130_44_3	7.987	13.021	38.66		0.021	12.439	99.83		rcp_130_45_3	7.403	11.249	34.19		0.018	8.809	99.80	
rcp_130_44_4	7.642	12.862	40.59		0.009	11.894	99.92		rcp_130_45_4	7.009	10.169	31.08		0.018	10.523	99.83	
rcp_130_44_5	7.330	11.965	38.74		0.000	10.354	100.00		rcp_130_45_5	7.180	11.021	34.85		0.000	11.425	100.00	
rcp_140_47_1	8.454	14.106	40.07		0.000	16.173	100.00		rcp_140_48_1	7.162	10.624	32.58		0.000	13.891	100.00	
rcp_140_47_2	8.181	14.724	44.44		0.000	17.504	100.00		rcp_140_48_2	7.887	13.710	42.47		0.000	15.931	100.00	
rcp_140_47_3	8.677	14.593	40.54		0.000	16.274	100.00		rcp_140_48_3	7.884	12.322	36.02		0.000	15.811	100.00	
rcp_140_47_4	7.865	13.461	41.57		0.000	13.552	100.00		rcp_140_48_4	7.856	12.653	37.91		0.000	14.586	100.00	
rcp_140_47_5	7.461	13.800	45.94		0.000	16.479	100.00		rcp_140_48_5	7.185	12.117	40.70		0.000	12.537	100.00	
rcp_150_50_1	8.100	16.354	50.47		0.000	19.095	100.00		rcp_150_51_1	8.496	15.558	45.39		0.000	16.889	100.00	
rcp_150_50_2	7.431	12.765	41.79		0.000	20.209	100.00		rcp_150_51_2	8.031	16.602	51.63		0.000	16.169	100.00	
rcp_150_50_3	7.398	16.418	54.94		0.000	21.169	100.00		rcp_150_51_3	8.118	14.054	42.24		0.000	16.308	100.00	
rcp_150_50_4	8.588	18.130	52.63		0.000	17.842	100.00		rcp_150_51_4	7.989	14.799	46.02		0.000	15.362	100.00	
rcp_150_50_5	8.270	15.106	45.26		0.000	17.589	100.00		rcp_150_51_5	7.834	13.616	42.47		0.000	14.997	100.00	

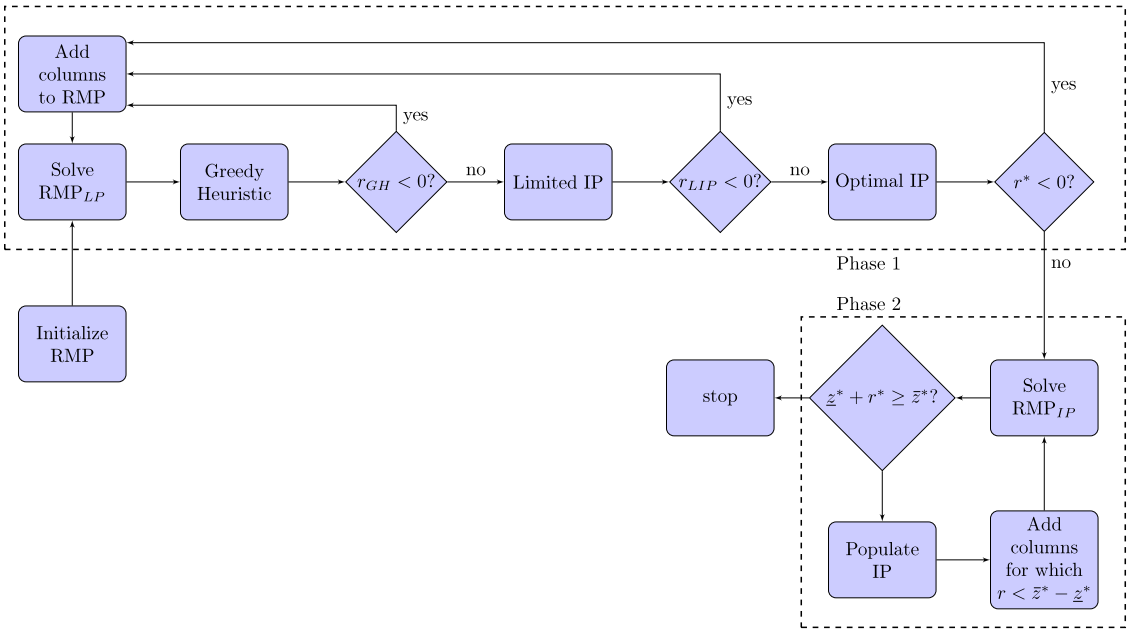

Fig. 1. Flowchart for our solution approach.

3.2. Algorithm

Our algorithm is composed of two phases as mentioned earlier. The column generation phase requires iteratively generating columns with negative reduced cost, which corresponds to solving a pricing subproblem. The pricing subproblem is solved at every iteration; it either generates new columns to expand RMP or signals that an optimal solution to the LP relaxation of MP is obtained. In second phase, information from the pricing subproblem and the optimal solution to MP is used to find an integer feasible solution.

In order to formulate the pricing subproblem, the dual of RMP corresponding to a restricted version of formulation (8)–(11) with $\bar{\mathbb{S}} \subseteq \mathbb{S}$, instead of $\mathbb{S}$, is used, where u_i denote the dual variables for constraints (9) and w denotes the dual variable for constraint (10). Given the optimal solution of a RMP at some iteration, a new column has to be added to the problem corresponding to an independent set S if $c_S - \sum_{i \in S} u_i - w < 0$. We solve an independent set problem with weights on vertices and pairs of vertices in order to find such independent sets. We define the decision variables for the vertices of the independent set as

$$v_i = \begin{cases} 1, & \text{if vertex } i \text{ is in the independent set,} \\ 0, & \text{otherwise.} \end{cases}$$

With variables y_{ij} as in Section 2, we formulate the pricing subproblem as

$$\min \sum_{i \in V} \sum_{j \in V : i < j} c_{ij} y_{ij} - \sum_{i \in V} u_i v_i - w, \tag{12}$$

$$\text{subject to } v_i + v_j \leq 1, \qquad \forall i, j \in V, \{i, j\} \in E, \tag{13}$$

$$v_i + v_j \leq 1 + y_{ij}, \qquad \forall i, j \in V, \{i, j\} \notin E \tag{14}$$

$$v_i \in \{0, 1\}, \qquad \forall i \in V, \tag{15}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall i \in V, j \in \bar{N}^+(i). \tag{16}$$

The objective function (12) looks for a column with the most negative reduced cost. Constraints (13) ensure that a solution form an independent set by allowing at most one endpoint of an edge to be selected. Constraints (14) guarantee that for two vertices i and j that are in the independent set, the corresponding y variable is selected. Constraints (15)–(16) are the domain constraints for the variables. If the objective function value of a feasible solution to (13)–(16) is negative, the corresponding independent set of vertices can be added to RMP as a new column. If the optimal objective function value is greater than or equal to 0, then we have an optimal solution for the LP relaxation of MP.

We illustrate the algorithm in Fig. 1. To obtain an initial RMP with a pseudo-feasible solution, we create a dummy color class with a very large cost including all vertices. We also create color classes of single vertices for every individual vertex such that we satisfy constraints on coloring the vertices and the number of colors used. This corresponds to initializing RMP in Fig. 1. After initializing RMP, we solve RMP$_{LP}$ iteratively and obtain the values of the dual variables. At each iteration, we choose either one of the following methods to generate new columns:

- Greedy Heuristic: We make a list of the vertices in V in nonincreasing order of the corresponding dual variables u_i. We select the vertex with the largest u_i value and make one pass through the list by adding the vertices in a new independent set; we add a vertex only if it decreases the reduced cost of the independent set. If the reduced cost of the independent set obtained after one pass, r_{GH}, is negative, we add the independent set as a new column to RMP. We follow the same procedure by selecting the remaining vertices in the list as the starting vertex and add negative reduced cost columns obtained after each pass.
- Limited IP: We solve the pricing subproblem (12)–(16) by restricting the number of nodes in the branch-and-bound tree in order to find independent sets with negative reduced cost without having to prove optimality. If the objective function value of the best solution found, r_{LIP}, is negative, we add the corresponding independent set to RMP along with other independent sets with negative reduced cost represented by the incumbent solutions obtained during the branch-and-bound procedure.
- Optimal IP: We solve the pricing subproblem (12)–(16) to optimality. If the objective function value of the solution, r^*, is negative, we add the corresponding independent set to RMP. Again, we also add the independent sets represented by incumbent solutions with negative objective function values. If the objective function value r^* is not negative, we have an optimal solution to RMP_{LP} and a lower bound. As a result, the algorithm transitions to the second phase.

In all three methods, we potentially add more than one column in order to reduce the number of iterations until we solve RMP_{LP} to optimality. However, we do not make an additional effort to find more columns while we are solving the pricing subproblem.

In the second phase, instead of using branch-and-price as in [2], we continue generating new columns (with zero or positive reduced cost) in order to improve both the lower bound and the upper bound. Our methodology is a direct result of reduced cost fixing idea in [19]; it uses the reduced costs of the decision variables c_S.

Corollary 1. *If a binary decision variable c_S with a reduced cost r_S is nonbasic, i.e. $c_S = 0$, in the optimal solution of RMP_{LP} with an objective function value $\underline{z}^*$, a solution to RMP_{LP} with $c_S = 1$ cannot have an objective function value less than $\underline{z}^* + r_S$. If an upper bound of $\bar{z}^*$ is known, such that $\underline{z}^* + r_S \geq \bar{z}^*$ then c_S can be fixed to 0.*

Solving RMP_{LP} to optimality, we obtain a lower bound $\underline{z}^*$ at the end of the first phase. We initiate the second phase by adding the integrality constraints (Solve RMP_{IP} in Fig. 1) in order to obtain a feasible solution, i.e. an upper bound $\bar{z}^*$, to the problem. If $\bar{z}^* = \underline{z}^*$, we have an optimal solution. Otherwise, we solve the pricing subproblem by using *populate* method in CPLEX [10] in order to generate multiple columns at each iteration. If r^* is the objective function value of an optimal solution found by the *populate* method, we obtain a new lower bound for our problem with objective function value $\underline{z}^* + r^*$. If $\underline{z}^* + r^* \geq \bar{z}^*$, we stop; the current integral solution is also an optimal solution. Otherwise, we add any columns S, with reduced cost r_S, obtained for the pricing subproblem for which $\underline{z}^* + r_S < \bar{z}^*$. The algorithm terminates when $\underline{z}^* + r^* \geq \bar{z}^*$. In the worst case, we may generate all columns for the original master problem.

3.3. Computational results

We conduct the computational study on an Intel Core i7, 3.2 GHz Windows desktop using CPLEX 12.6 to solve RMP and the pricing subproblem with its default settings. The algorithm is implemented in C++. The computation time is limited to two hours for a fair comparison with Archetti et al. [2] and the best upper and lower bounds are reported in Tables 4–6, where columns under Root Node header show the values associated with RMP_{LP}. We present the lower bound $\underline{z}^*(\%)$ as percentage of the best lower bound value, $\underline{z}^*$, and computation time required to solve RMP_{LP} to optimality. We also provide best lower and upper bound values under Final Solution header, as well as the overall computation time and the integrality gap $(\frac{\bar{z}^* - \underline{z}^*}{\bar{z}^*} 100)$. The number of colors exceeds the number of vertices for miles1500 instance when $k = \lceil 2\bar{k} \rceil = 156$, and this instance cannot be solved. In addition to that, two instances created by Archetti et al. [2] are not feasible (rcp_10_4_2 and rcp_10_4_4) since the size of a largest clique in these instances exceeds the number of colors that are allowed. Corresponding cells are shaded with gray. "–" indicates that the information for the corresponding cell is not available, i.e., the algorithm failed to provide the corresponding information in the time limit.

Overall, we can solve larger instances when compared to Archetti et al. [2] such as queen8_8, myciel_5 with $k = 9$ and some of the larger instances generated by Archetti et al. [2] such as rcp_150_50_1. Among these instances, we solve all but four to optimality. For the instances that we cannot solve to optimality, the gap is always less than 0.5% and we obtain better lower bounds compared to Archetti et al. [2]. For DIMACS instances we are able to solve 26 instances to optimality compared to 19 instances for Archetti et al. [2]. We are able to solve RMP_{LP} to optimality in 35 instances compared to 31 instances for Archetti et al. [2]. We are able to obtain a lower bound for instances such as queen10_10 with $k = 17$, or queen8_12 with $k = 18$ for which Archetti et al. [2] do not report lower bounds.

Our observations can be summarized as follows:

- Even though the algorithm may enumerate all columns of the original problem in the worst case, empirically, it performs well. We improve on the results of Archetti et al. [2].
- We resort to optimal solution of the pricing subproblem only at the terminating iteration of the column generation; this result is an evidence for the success of the heuristics.

Table 4
Results of the column generation approach for DIMACS instances.

Instance			k	Root node		Final solution				k	Root node		Final solution			
Graph	$\|V\|$	$\|E\|$		$\underline{z}^*$ (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)		$\underline{z}^*$ (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)
myciel3	11	20	6	100.00	0.31	110.0	110		0.33	8	100.00	0.23	28.0	28		0.24
myciel4	23	71	8	100.00	3.39	2600.0	2600		3.40	10	100.00	3.27	1648.0	1648		3.28
queen5_5	25	160	8	99.99	2.48	3850.0	3850		3.63	10	100.00	2.65	2521.0	2521		2.66
1-FullIns_3	30	100	6	100.00	19.39	13292.0	13292		19.46	8	100.00	8.80	8787.0	8787		25.53
queen6_6	36	290	11	100.00	7.94	12061.0	12061		11.07	14	100.00	7.85	7741.0	7741		7.87
2-Insertions_3	37	72	6	100.00	182.43	32397.0	32397		185.39	8	100.00	90.37	22101.0	22101		423.76
myciel5	47	236	9	100.00	366.76	52829.0	52829		367.59	12	100.00	384.21	35156.0	35156		1762.05
queen7_7	49	476	11	100.00	38.75	48002.0	48002		63.64	14	100.00	22.63	33383.0	33383		56.37
2-FullIns_3	52	201	8	100.00	1841.36	94566.0	94749	0.19		10	100.00	644.00	70830.0	70830		1822.23
3-Insertions_3	56	110	6	–	–	–	–	–	–	8	–	–	–	–	–	–
queen8_8	64	728	14	100.00	259.25	109796.0	109796		260.62	18	100.00	181.22	75460.0	75460		676.51
1-Insertions_4	67	232	8	–	–	–	–	–	–	10	–	–	–	–	–	–
huck	74	602	17	–	–	–	–	–	–	22	100.00	1396.16	106103.0	106104	0.00	
jean	77	508	15	–	–	–	–	–	–	20	–	–	–	–	–	–
4-Insertions_3	79	156	6	–	–	–	–	–	–	8	–	–	–	–	–	–
3-FullIns_3	80	346	9	–	–	–	–	–	–	12	–	–	–	–	–	–
queen9_9	81	1056	15	100.00	918.55	277416.0	280797	1.20		20	100.00	1133.27	185506.0	185506		4619.09
david	87	812	17	–	–	–	–	–	–	22	–	–	–	–	–	–
mug88_1	88	146	6	–	–	–	–	–	–	8	–	–	–	–	–	–
mug88_25	88	146	6	–	–	–	–	–	–	8	–	–	–	–	–	–
1-FullIns_4	93	593	8	–	–	–	–	–	–	10	–	–	–	–	–	–
myciel6	95	755	11	–	–	–	–	–	–	14	–	–	–	–	–	–
queen8_12	96	2736	18	100.00	3283.24	452410.0	466498	3.02		24	100.00	2047.40	301864.0	301867	0.00	
mug100_1	100	166	6	–	–	–	–	–	–	8	–	–	–	–	–	–
mug100_25	100	166	6	–	–	–	–	–	–	8	–	–	–	–	–	–
queen10_10	100	1470	17	100.00	6097.41	580899.0	665711	12.74		22	100.00	3139.90	410430.0	428337	4.18	
4-FullIns_3	114	541	11	–	–	–	–	–	–	14	–	–	–	–	–	–
games120	120	1276	14	–	–	–	–	–	–	18	–	–	–	–	–	–
queen11_11	121	1980	17	–	–	–	–	–	–	22	–	–	–	–	–	–
DSJC125_1	125	736	8	–	–	–	–	–	–	10	–	–	–	–	–	–
DSJC125_5	125	3891	26	100.00	6842.04	863172.0	1003639	14.00		34	100.00	5133.70	582561.0	584531	0.34	
DSJC125_9	125	6961	66	99.98	60.05	142721.0	142721		116.53	88	99.99	37.79	36351.0	36351		62.29
miles250	125	774	12	–	–	–	–	–	–	16	–	–	–	–	–	–
miles500	128	2340	30	–	–	–	–	–	–	40	–	–	–	–	–	–
miles750	128	4226	47	–	–	–	–	–	–	62	100.00	6470.65	196069.0	196069		6470.72
miles1000	128	6432	63	100.00	2835.60	187394.0	187394		2835.65	84	100.00	370.72	58768.0	58768		370.76
miles1500	128	10396	110	100.00	25.58	4284.0	4284		25.60	156						
anna	138	986	17	–	–	–	–	–	–	22	–	–	–	–	–	–
queen12_12	144	5192	18	–	–	–	–	–	–	18	–	–	–	–	–	–
2-Insertions_4	149	541	6	–	–	–	–	–	–	8	–	–	–	–	–	–

Table 5
Results of the column generation approach for instances from [2] (part 1).

Instance	Root node		Final solution				Instance	Root node		Final solution			
	$\underline{z}^*$ (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)		$\underline{z}^*$ (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)
rcp_10_4_1	100.00	0.17	1.747	1.747		0.18	rcp_10_5_1	100.00	0.14	1.929	1.929		0.16
rcp_10_4_2							rcp_10_5_2	100.00	0.03	0.997	0.997		0.04
rcp_10_4_3	100.00	0.18	3.612	3.612		0.19	rcp_10_5_3	96.94	0.04	1.796	1.796		0.24
rcp_10_4_4							rcp_10_5_4	100.00	0.12	1.599	1.599		0.13
rcp_10_4_5	100.00	0.30	1.733	1.733		0.31	rcp_10_5_5	100.00	0.19	2.659	2.659		0.21
rcp_20_7_1	98.01	0.92	5.931	5.931		1.78	rcp_20_8_1	99.18	0.60	4.838	4.838		1.84
rcp_20_7_2	99.88	0.65	6.290	6.290		2.84	rcp_20_8_2	100.00	0.79	3.184	3.184		0.80
rcp_20_7_3	100.00	0.86	5.020	5.020		0.87	rcp_20_8_3	100.00	0.54	2.169	2.169		0.55
rcp_20_7_4	99.17	0.87	5.914	5.914		1.71	rcp_20_8_4	98.79	0.58	3.669	3.669		1.77
rcp_20_7_5	100.00	1.27	5.645	5.645		1.28	rcp_20_8_5	97.54	0.95	3.815	3.815		1.68
rcp_30_10_1	100.00	1.55	6.091	6.091		1.56	rcp_30_11_1	100.00	1.76	5.378	5.378		1.78
rcp_30_10_2	100.00	2.05	7.407	7.407		2.07	rcp_30_11_2	97.67	1.17	5.108	5.108		3.66
rcp_30_10_3	97.94	1.90	8.483	8.483		8.02	rcp_30_11_3	100.00	1.15	4.897	4.897		1.16
rcp_30_10_4	100.00	1.92	6.570	6.570		1.93	rcp_30_11_4	98.48	1.08	5.068	5.068		3.92
rcp_30_10_5	98.47	1.90	6.737	6.737		4.97	rcp_30_11_5	100.00	0.89	4.897	4.897		0.90
rcp_40_14_1	99.92	2.37	5.970	5.970		3.78	rcp_40_15_1	100.00	1.34	5.367	5.367		1.35
rcp_40_14_2	99.96	2.49	6.242	6.242		4.24	rcp_40_15_2	99.51	2.44	4.819	4.819		8.96
rcp_40_14_3	100.00	4.66	7.657	7.657		4.67	rcp_40_15_3	99.29	2.00	5.224	5.224		7.10
rcp_40_14_4	99.41	3.00	6.146	6.146		7.90	rcp_40_15_4	99.04	4.42	5.544	5.544		9.45
rcp_40_14_5	99.58	4.70	6.185	6.185		8.30	rcp_40_15_5	99.25	2.07	4.939	4.939		5.41
rcp_50_17_1	99.94	11.12	7.687	7.687		28.31	rcp_50_18_1	99.06	5.75	6.777	6.777		12.72
rcp_50_17_2	98.65	6.92	7.364	7.364		20.65	rcp_50_18_2	99.26	6.71	6.678	6.678		23.37
rcp_50_17_3	100.00	5.88	6.852	6.852		5.90	rcp_50_18_3	98.62	2.98	5.529	5.529		13.05
rcp_50_17_4	98.67	9.53	7.923	7.923		34.02	rcp_50_18_4	100.00	5.91	6.812	6.812		5.93
rcp_50_17_5	97.09	6.46	8.073	8.073		35.80	rcp_50_18_5	99.68	6.58	6.466	6.466		11.01
rcp_60_20_1	99.38	14.32	9.123	9.123		27.96	rcp_60_21_1	100.00	9.43	7.112	7.112		9.44
rcp_60_20_2	98.09	15.01	8.910	8.910		48.47	rcp_60_21_2	99.33	13.13	8.550	8.550		31.70
rcp_60_20_3	99.18	12.05	8.485	8.485		26.18	rcp_60_21_3	97.64	12.96	8.109	8.109		47.66
rcp_60_20_4	98.89	11.55	7.810	7.810		28.67	rcp_60_21_4	100.00	14.09	7.725	7.725		14.11
rcp_60_20_5	99.21	15.00	8.406	8.406		33.72	rcp_60_21_5	98.54	14.61	7.770	7.770		40.87
rcp_70_24_1	99.39	17.82	9.106	9.106		44.43	rcp_70_25_1	99.44	25.64	6.743	6.743		70.09
rcp_70_24_2	98.03	22.41	7.753	7.753		86.82	rcp_70_25_2	100.00	15.32	6.973	6.973		15.34
rcp_70_24_3	100.00	18.77	8.001	8.001		18.78	rcp_70_25_3	98.57	22.49	6.874	6.874		77.40
rcp_70_24_4	98.40	13.59	8.823	8.823		47.85	rcp_70_25_4	98.71	13.64	7.963	7.963		58.55
rcp_70_24_5	98.78	15.31	7.976	7.976		50.56	rcp_70_25_5	98.64	23.01	7.853	7.853		55.54
rcp_80_27_1	98.16	44.22	10.078	10.078		180.61	rcp_80_28_1	99.32	27.18	7.769	7.769		72.58
rcp_80_27_2	98.37	35.65	10.260	10.260		196.66	rcp_80_28_2	99.27	22.57	9.272	9.272		86.10
rcp_80_27_3	98.50	28.59	9.126	9.126		114.86	rcp_80_28_3	98.70	28.37	8.564	8.564		104.27
rcp_80_27_4	98.61	38.38	9.233	9.233		135.74	rcp_80_28_4	99.71	16.17	7.621	7.621		40.27
rcp_80_27_5	97.84	39.75	10.143	10.143		229.34	rcp_80_28_5	99.45	30.54	7.404	7.404		68.97

- For the instances that we cannot solve, solving the pricing subproblem to optimality (at the end of the column generation) and solving RMP with integrality constraints are the main bottlenecks.
- Solving the pricing subproblem in CPLEX by restricting the number of nodes in the branch-and-bound tree made the most significant contribution in reducing the computation times.
- At the end of the column generation phase, the lower bound (obtained by solving RMP_{LP}) and the upper bound (obtained by an integer feasible solution to RMP_{IP}) for MP are very close for most of the instances.

Overall, our column generation-based algorithm turns out to be effective for the robust graph coloring problem with added efficiency of the heuristic methods for solving the pricing subproblem. When compared to both the traditional formulation and asymmetric representatives formulation, the set-covering type formulation is more effective when it is solved with our algorithm. With respect to quality of the solutions, we increase the number of problems solved to optimality in the literature.

4. Conclusion

The robust graph coloring problem is an extension to the original graph coloring problem where it may be undesirable and penalized for some pairs of non-adjacent vertices to be assigned the same color. The extension enables us to model problems with uncertain data or problems where clashes are allowed but need to be minimized.

We first extend the asymmetric representatives formulation in [6] to model the robust graph coloring problem. We compare the new formulation with the original formulation in [21] in terms of their computational efficiency. The asymmetric representatives formulation provides a better lower bound than the original formulation and is able to solve

Table 6
Results of the column generation approach for instances from [2] (part 2).

Instance	Root node		Final solution				Instance	Root node		Final solution			
	$\underline{z}^*$ (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)		$\underline{z}^*$ (%)	Time (s)	$\underline{z}^*$	$\bar{z}^*$	Gap (%)	Time (s)
rcp_90_30_1	97.94	59.32	10.795	10.795		386.26	rcp_90_31_1	98.39	38.97	9.263	9.263		256.72
rcp_90_30_2	98.54	70.37	10.316	10.316		270.43	rcp_90_31_2	98.92	50.62	8.924	8.924		163.69
rcp_90_30_3	98.24	65.07	10.530	10.530		304.33	rcp_90_31_3	98.91	45.73	8.966	8.966		189.16
rcp_90_30_4	98.99	72.09	10.113	10.113		231.29	rcp_90_31_4	98.50	53.97	8.949	8.949		187.75
rcp_90_30_5	98.63	57.33	9.481	9.481		213.27	rcp_90_31_5	98.39	60.20	7.667	7.667		217.41
rcp_100_34_1	99.07	109.88	9.758	9.758		346.81	rcp_100_35_1	98.69	77.43	8.479	8.479		265.68
rcp_100_34_2	99.73	79.83	8.990	8.990		193.05	rcp_100_35_2	98.77	70.70	9.897	9.897		302.27
rcp_100_34_3	98.54	62.39	9.500	9.500		307.16	rcp_100_35_3	98.45	115.82	8.805	8.805		437.61
rcp_100_34_4	97.63	108.83	10.032	10.032		734.90	rcp_100_35_4	97.89	89.35	8.219	8.219		513.94
rcp_100_34_5	98.70	103.81	9.858	9.858		371.69	rcp_100_35_5	98.11	61.38	8.105	8.105		340.70
rcp_110_37_1	97.12	111.70	10.291	10.291		1484.46	rcp_110_38_1	98.53	169.30	9.260	9.260		676.69
rcp_110_37_2	98.76	129.14	9.820	9.820		491.22	rcp_110_38_2	99.96	138.56	9.533	9.533		326.64
rcp_110_37_3	99.11	148.42	10.619	10.619		393.63	rcp_110_38_3	98.27	155.20	9.097	9.097		674.77
rcp_110_37_4	98.94	140.43	10.247	10.247		495.69	rcp_110_38_4	97.59	129.06	8.344	8.344		869.61
rcp_110_37_5	99.12	146.91	10.365	10.365		421.44	rcp_110_38_5	97.82	119.72	9.649	9.649		981.29
rcp_120_40_1	98.82	192.79	10.343	10.343		719.54	rcp_120_41_1	99.13	153.53	10.184	10.184		599.95
rcp_120_40_2	97.57	217.37	11.194	11.194		2331.33	rcp_120_41_2	98.20	165.19	10.721	10.721		1224.84
rcp_120_40_3	98.88	250.90	10.819	10.819		942.11	rcp_120_41_3	98.39	213.13	9.487	9.487		916.22
rcp_120_40_4	99.39	183.35	10.886	10.886		472.91	rcp_120_41_4	99.76	200.87	9.387	9.387		425.09
rcp_120_40_5	98.85	254.84	11.933	11.933		885.40	rcp_120_41_5	98.57	149.69	9.717	9.717		962.88
rcp_130_44_1	97.86	320.29	11.729	11.729		2515.07	rcp_130_45_1	98.81	268.95	9.791	9.791		975.61
rcp_130_44_2	98.59	376.18	10.915	10.915		1423.38	rcp_130_45_2	99.13	224.44	9.686	9.686		713.50
rcp_130_44_3	97.51	259.00	11.256	11.256		4335.10	rcp_130_45_3	98.21	298.73	10.227	10.227		1795.03
rcp_130_44_4	99.02	296.35	10.887	10.887		882.98	rcp_130_45_4	99.41	353.16	9.370	9.370		782.24
rcp_130_44_5	98.93	279.40	9.896	9.896		943.89	rcp_130_45_5	98.08	228.48	9.478	9.478		1783.16
rcp_140_47_1	98.55	444.05	11.743	11.743		2274.28	rcp_140_48_1	98.01	373.96	9.643	9.643		2365.63
rcp_140_47_2	97.57	409.23	11.547	11.549	0.02		rcp_140_48_2	97.80	411.62	10.785	10.785		3117.54
rcp_140_47_3	97.73	414.75	12.223	12.252	0.23		rcp_140_48_3	98.12	346.93	10.615	10.615		2456.53
rcp_140_47_4	98.06	578.69	11.243	11.243		3692.29	rcp_140_48_4	98.11	251.06	10.459	10.459		2719.90
rcp_140_47_5	98.74	380.58	10.548	10.548		1640.81	rcp_140_48_5	98.60	418.21	9.356	9.356		1480.66
rcp_150_50_1	97.98	674.36	11.556	11.556		5917.76	rcp_150_51_1	98.78	617.66	11.296	11.296		2326.12
rcp_150_50_2	98.17	641.55	10.070	10.119	0.48		rcp_150_51_2	98.70	602.99	11.643	11.643		3410.83
rcp_150_50_3	98.28	631.31	10.543	10.543		3865.35	rcp_150_51_3	98.42	664.12	10.701	10.701		3657.17
rcp_150_50_4	98.56	755.28	12.700	12.700		4360.66	rcp_150_51_4	97.88	623.59	11.273	11.273		6868.46
rcp_150_50_5	98.30	922.58	11.350	11.392	0.37		rcp_150_51_5	97.88	565.61	10.771	10.771		5633.11

larger instances to optimality. However, even the asymmetric representatives formulation is restricted with respect to the size of instances solved to optimality.

As both formulations are quite ineffective, we resort the set-covering type formulation in [2] and develop a column generation-based approach. Instead of using branch-and-price, we continue generating columns in order to prove optimality. The approach performs better than the branch-and-price algorithm of Archetti et al. [2], and we are able to solve larger instances.

Our contributions are twofold:

- We develop a new formulation based on an existing approach used for the traditional graph coloring problem and test the computational efficiency of this formulation against the traditional formulation.
- We design a new column-generation based algorithm and our computational experiments show that the new approach is superior to existing formulation and solution approaches.

We believe that there is enough evidence to claim that set-covering type formulation approach is effective and efficient to solve the robust graph coloring problem. Future work in this domain may aim to focus on further improving the column-generation phase of the algorithm.

References

[1] K. Aardal, S.P.M. van Hoesel, A.M.C.A. Koster, C. Mannino, A. Sassano, Models and solution techniques for frequency assignment problems, Ann. Oper. Res. 153 (1) (2007) 79–129.
[2] C. Archetti, N. Bianchessi, A. Hertz, A branch-and-price algorithm for the robust graph coloring problem, Discrete Appl. Math. 165 (0) (2014) 49–59. 10th Cologne/Twente Workshop on Graphs and Combinatorial Optimization (CTW 2011).
[3] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, P.H. Vance, Branch-and-price: Column generation for solving huge integer programs, Oper. Res. 46 (3) (1998) 316–329.
[4] C. Bron, J. Kerbosch, Algorithm 457: finding all cliques of an undirected graph, Commun. ACM (ISSN: 0001-0782) 16 (9) (1973) 575–577. http://dx.doi.org/10.1145/362342.362367.
[5] E.K. Burke, J. Marecek, A.J. Parkes, H. Rudová, A supernodal formulation of vertex colouring with applications in course timetabling, Ann. Oper. Res. 179 (2010) 105–130.

[6] M. Campêlo, V.A. Campos, R.C. Corrêa, On the asymmetric representatives formulation for the vertex coloring problem, Discrete Appl. Math. 156 (7) (2008) 1097–1111.
[7] M. Campêlo, R.C. Corrêa, Y. Frota, Cliques, holes and the vertex coloring polytope, Inform. Process. Lett. 89 (4) (2004) 159–164.
[8] G.J. Chaitin, M.A. Auslander, A.K. Chandra, J. Cocke, M.E. Hopkins, P.W. Markstein, Register allocation via coloring, Comput. Lang. 6 (1) (1981) 47–57.
[9] S. Guo, Y. Kong, A. Lim, F. Wang, A new neighborhood based on improvement graph for robust graph coloring problem, in: G.I. Webb, X. Yu (Eds.), AI 2004: Advances in Artificial Intelligence, in: Lecture Notes in Computer Science, vol. 3339, Springer, Berlin, Heidelberg, 2005, pp. 636–645.
[10] IBM. IBM ILOG CPLEX Optimizer. www.ibm.com/software/commerce/optimization/cplex-optimizer/ Last retrieved April 2015.
[11] R. Karp, Reducibility among combinatorial problems, in: R. Miller, J. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, 1972, pp. 85–103.
[12] Y. Kong, F. Wang, A. Lim, S. Guo, A new hybrid genetic algorithm for the robust graph coloring problem, in: T.D. Gedeon, L.C.C. Fung (Eds.), AI 2003: Advances in Artificial Intelligence, in: Lecture Notes in Computer Science, vol. 2903, Springer, Berlin, Heidelberg, 2003, pp. 125–136.
[13] M. Kubale, Graph Colorings, in: Contemporary Mathematics (American Mathematical Society), vol. 352, American Mathematical Society, 2004.
[14] A. Lim, F. Wang, Robust graph coloring for uncertain supply chain management, in: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 2005. HICSS'05, Jan 2005, p. 81b.
[15] V. Lotfi, S. Sarin, A graph coloring algorithm for large scale scheduling problems, Comput. Oper. Res. 13 (1) (1986) 27–32.
[16] T. Matsui, N. Sukegawa, A. Miyauchi, Fractional programming formulation for the vertex coloring problem, Inform. Process. Lett. 114 (12) (2014) 706–709.
[17] A. Mehrotra, M.A. Trick, A column generation approach for graph coloring, INFORMS J. Comput. 8 (1995) 344–354.
[18] I. Méndez-Díaz, P. Zabala, A branch-and-cut algorithm for graph coloring, Discrete Appl. Math. 154 (5) (2006) 826–847.
[19] G.L. Nemhauser, L.A. Wolsey, Integer and Combinatorial Optimization, Wiley-Interscience, New York, NY, USA, 1988.
[20] F. Wang, Z. Xu, Metaheuristics for robust graph coloring, J. Heuristics 19 (4) (2013) 529–548.
[21] J. Yáñez, J. Ramírez, The robust coloring problem, European J. Oper. Res. 148 (3) (2003) 546–558.