Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Discrete Mathematics 41 (2013) 447–454

Electronic Notes in DISCRETE MATHEMATICS

www.elsevier.com/locate/endm

# $(k, c)$ – *coloring* via Column Generation

E. Malaguti [1]

*DEI, University of Bologna*
*Bologna, Italy*

I. Méndez-Díaz [2,3]

*Computer Science Department, University of Buenos Aires*
*Buenos Aires, Argentina*

J. J. Miranda-Bront and P. Zabala [2,3]

*Computer Science Department, University of Buenos Aires*
*Consejo Nacional de Investigaciones Científicas y Técnicas*
*Buenos Aires, Argentina*

**Abstract**

In this paper we study the $(k, c)$-*coloring* problem, a generalization of the well known *Vertex Coloring Problem* (VCP). We propose a new formulation and compare it computationally with another formulation from the literature. We also develop a diving heuristic that provides with good quality results at a reasonable computational effort.

*Keywords:* $(k, c)$-*coloring*, column generation, diving heuristic

## 1 Introduction

In the Vertex Coloring Problem (VCP), one is required to assign a color to each vertex of an indirected graph in such a way that adjacent vertices re-

[1] Email: enrico.malaguti@unibo.it
[2] Email: (imendez,jmiranda,pzabala)@dc.uba.ar
[3] This work was partially supported by UBACYT 20020100100666, PICT 304 and 817

1571-0653/$ – see front matter 
http://dx.doi.org/10.1016/j.endm.2013.05.124

ceive different colors, and the objective is to minimize the number of the used colors. Several problems where a resource (color) has to be shared among conflicting users (vertices connected by an edge in the graph) can be modeled as VCPs. Some problems can be represented as generalizations of the VCP. In the multicoloring problem, for example, users require more than one copy of the resource (see, e.g., [4]). In some applications the resource cannot be duplicated more than a fixed amount of times, and it is acceptable to have the resource partially shared among conflicting users (this happens in frequency assignment, see, e.g., Aardal et al., [1]). In this paper we consider a generalization of the VCP where each vertex has to receive more than one color, and each pair of conflicting vertices can share a limited number of colors.

Formally, the $(k, c)$-*coloring* problem is defined as follows. Let $G = (V, E)$ be an undirected graph, with $V = \{1, \ldots, n\}$ the set or vertices and $E = \{uv : u, v \in V, u \neq v\}$ the set of edges, and $R = \{1, \ldots, |R|\}$ the set of available colors. Each vertex $v \in V$ is required to be assigned exactly $k$ different colors and each pair of adjacent vertices $u, v$ cannot share more than $c$ colors. The objective is to minimize the total number of colors used. The problem is $NP - hard$ and reduces to the VCP when $k = 1$ and $c = 0$ (see Garey and Johnson [2] for complexity results on VCP, and Malaguti [6] and Malaguti and Toth [8] for other $NP - hard$ generalizations of the VCP).

In this paper, we propose a new formulation for $(k, c)$-*coloring* problem. Based on the results recently achieved by Malaguti et al. [7], Gualandi and Malucelli [3] and Held et al. [5] for the VCP using set covering formulations, we propose a model for the $(k, c)$-*coloring* following these ideas. We experimentally evaluate and compare it with another model from the related literature, and develop a diving heuristic, which produces good quality solutions.

The rest of the paper is organized as follows. In Section 2 we describe the model from Méndez-Díaz and Zabala [9] and propose a new formulation for the $(k, c)$-*coloring*. In Section 3 we describe the diving heuristic and in Section 4 we present some preliminary computational results, making a comparison of both formulations and evaluating the diving heuristic. Finally, in Section 5 we present some conclusions and propose future lines of research regarding the $(k, c)$-*coloring*.

## 2 Models

We begin by showing the model proposed by Méndez-Díaz and Zabala [9], slightly modified for the particular case of the $(k, c)$-*coloring* problem. They consider three different sets of variables. The first one regards binary variables

$x_{vj}$, for $v \in V$ and $j \in R$, taking value 1 if and only if color $j$ is assigned to vertex $v$. Secondly, for each arc $u, v \in E$ and a color $j \in R$, binary variable $y_{uvj} = 1$ iff color $j$ is assigned to both $u$ and $v$. Finally, binary variables $w_j$, $j \in R$, take value 1 iff color $j$ is used by some vertex. The $(k, c)$-*coloring* problem is formulated as:

$$\min \sum_{j \in R} w_j \tag{1}$$

$$\text{s.t.:} \sum_{j \in R} x_{vj} = k \qquad v \in V \tag{2}$$

$$\sum_{j \in R} y_{uvj} \leq c \qquad uv \in E \tag{3}$$

$$x_{uj} + x_{vj} - y_{uvj} \leq 1 \qquad uv \in E,\ j \in R \tag{4}$$

$$x_{vj} \leq w_j \qquad v \in V,\ j \in R \tag{5}$$

$$x_{vj} \in \{0, 1\} \qquad v \in V,\ j \in R \tag{6}$$

$$y_{uvj} \in \{0, 1\} \qquad uv \in E,\ j \in R \tag{7}$$

$$w_j \in \{0, 1\} \qquad j \in R \tag{8}$$

The objective function (1) minimizes the number of colors used. Constraints (2) establish that exactly $k$ colors are assigned to vertex $v$ and constraints (3) restrict the number of colors that can be shared by two adjacent vertices. Constraints (4) impose $y_{uvj} = 1$ if color $j$ is assigned to both $u$ and $v$, and constraints (5) set $w_j = 1$ if color $j$ is used by some vertex. Finally, constraints (6) - (8) establish that all variables must be binary.

Similarly as with the VCP, this formulation allows symmetric solutions. To eliminate some symmetries, the authors propose to include in the formulation inequalities $w_j \leq w_{j+1}$, for $1 \leq j \leq |R| - 1$.

In this paper, we propose a new formulation which does not suffer from symmetry issues. Let $S = 2^V$ be the power set of $V$, and define integer variables $x_s$, $s \in S$, representing the number of colors assigned to all vertices in $s$. Regarding the standard formulation for the VCP, it is important to note that in our model any subset of vertices can be feasibly colored by one color when $c > 0$. The $(k, c)$-*coloring* problem can be formulated as follows:

$$\min \sum_{s \in S} x_s \tag{9}$$

$$\text{s.t.} \sum_{s \in S : v \in s} x_s \geq k, \qquad v \in V \tag{10}$$

$$\sum_{s \in S: uv \in s} x_s \leq c, \qquad uv \in E \tag{11}$$

$$x_s \in \mathbb{Z}^+, \qquad s \in S \tag{12}$$

The objective function (9) minimizes the number of colors used. Constraints (10) establish that a vertex must receive at least $k$ colors. It is important to note that these constraints modify slightly the definition $(k, c)$-*coloring*, where each vertex is required to have assigned exactly $k$ colors, in order to be able to consider maximal sets $s \in S$ and therefore speed up the column generation algorithm. In this context, we let a set $s \in S$ to be maximal if no vertex $v$ can be included into the set without adding new edges. Furthermore, any solution obtained by this formulation can be easily transformed into one having exactly $k$ colors. Finally, constraints (11) restrict the number of colors assigned to pairs of adjacent vertices in $G$ and constraints (12) require variables $x_s$, $s \in S$, to be nonnegative integers.

Model (9) - (12) has an exponential number of variables and therefore it cannot be formulated explicitly, even for medium size instances (i.e., having a few hundreds of nodes). Branch-and-Price algorithms have been proven to be quite effective in solving this kind of models, using column generation to solve the LP relaxation (usually called *master problem*). The main idea behind this technique is to start with a restricted set of columns, obtaining as a result a restricted master problem, and iteratively add columns with negative reduced cost until the master problem is solved to optimality. For this purpose, let $(\pi, \rho)$ be the optimal values of the dual variables associated with constraints (10) and (11), respectively. To account for variables $x_s$ with negative reduced costs, we formulate the following slave problem where binary variables $z_v$, $v \in V$, take value one if and only if vertex $v$ is in the set and, for each $uv \in E$, $y_{uv}$ is forced to 1 when both $u$ and $v$ belong to the set.

$$\max \sum_{v \in V} \pi_v z_v - \sum_{uv \in E} \rho_{uv} y_{uv} \tag{13}$$

$$\text{s.t.} \quad z_u + z_v - y_{uv} \leq 1, \qquad uv \in E \tag{14}$$

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

$$y_{uv} \in \{0, 1\}, \qquad uv \in E \tag{16}$$

If the objective value of the optimal solution of the slave problem is less than or equal to one, then the master problem has been solved to optimality. Otherwise, the optimal solution represents a feasible set with negative reduced cost, we add the column to the restricted master problem and iterate again.

## 3 Diving heuristic

Aiming to obtain good solutions for the $(k, c)$-*coloring*, in this section we present a diving heuristic developed using model (9) - (12). The algorithm iteratively tries to fix variables and updates the lower and upper bounds using the information provided by the LP relaxation, expecting to obtain at the end a feasible solution for the problem.

In order to solve the LP relaxation of model (9) - (12), we initialize the restricted master problem with the following columns:

- Sets $s = \{i\}$, for $i = 1, \ldots, n$, and
- sets $s$ represented by the solution obtained by the greedy heuristic proposed in Méndez-Díaz and Zabala [9]. This heuristic starts with an initial color list $L = \{1, \ldots, k\}$ and iteratively selects a vertex to be colored with the first $k$ compatible colors in $L$. In case this is not possible, $L$ is expanded by including new colors.

Another relevant aspect of the column generation algorithm is to be able to determine whether the master problem has been solved to optimality or if there exists a column with negative reduced cost. For this purpose, we solve the slave problem (13) - (16) using a general purpose MIP algorithm. The sketch of the diving heuristic is described in Algorithm 1.

**Algorithm 1** (i) *Initialize the master problem setting as columns sets* $s = \{i\}$, *for* $i = 1, \ldots, n$ *and the solution provided by the heuristic. Set* $bestUB$ *with the number of colors used.*

(ii) *Solve the master problem of model (9) - (12). Let* $x^*$ *be the optimal solution and* $z^*$ *the objective value. If* $\lceil z^* \rceil \geq bestUB$, *then exit the algorithm and return* $bestUB$.

(iii) *If* $x^*$ *is integer, exit the algorithm and return* $z^*$. *Otherwise, set in the master problem lower bounds* $x_s = x^*_s$ *for all variables* $x_s$ *having integer values in* $x^*$.

(iv) *Let* $\bar{s}$ *be the index of the fractional variable which is closest to its rounded-up value. Set* $x_{\bar{s}} \geq \lceil x^*_{\bar{s}} \rceil$ *and solve the modified restricted master problem. If the problem is feasible, update the master problem with* $x_{\bar{s}} \geq \lceil x^*_{\bar{s}} \rceil$ *and go to step (ii). Otherwise, Set* $x_{\bar{s}} \leq \lfloor x^*_{\bar{s}} \rfloor$ *in the master problem and go to step (ii)*

## 4 Computational results

We conducted experiments in order to evalute the quality of the lower bounds obtained by the linear relaxation of model (9) - (12) as well as the solutions obtained by the diving heuristic.

The experiments were run on a workstation with an Intel(R) Core(TM) i7 CPU (3.40GHz) and 16 Gb of RAM. The algorithms are coded in C++ using CPLEX 12.1 Callable Library as LP and MIP solver. We consider a set of instances with random generated graphs having 20 vertices and varying the density in terms of the number of edges, considering values of $10, 20, \ldots, 90$ percent of the edges. For each of these values we consider 10 instances, and we group them in Low (10% - 30%), Medium (40% - 60%) and High (70% - 90%) density, considering increasing values for this parameter.

We evaluated 3 different methods: (i) the Branch and Cut algorithm from Méndez-Díaz and Zabala [9] (BC), (ii) the diving heuristic described in Section 3 (DH-GH), and (iii) the diving heuristic but without including the columns obtained from the greedy heuristic (DH-noGH). We impose a time limit of 600 seconds for the three algorithms.

In Table 1 we present the average optimality gaps (100*(UB - LB)/LB) and computational times obtained by each algorithm. The average computational times are calculated only over instances for which the corresponding algorithm finishes before the time limit. A cell filled with $***$ means that the algorithm was not able to solve any of the instances, and a number between parenthesis stands for the number of instances effectively solved within the time limit. We first observe that the diving heuristic obtains better results in all cases when the columns provided by the greedy heuristic are considered to initialize the master problem. As regards the comparison between BC and DH-GH, the results are somehow mixed. BC produces the best results when $k$ and $c$ are close, but when this difference tends to grow DH-GH outperforms BC (see, e.g., $(4, 1)$, $(5, 1)$ and $(5, 2)$) finding the optimal solution in many cases. Furthermore, it also improves the initial solution provided by the greedy heuristic in more cases than BC.

In terms of computational times, although not reported due to space limitations, the behavior of DH-GH is reasonable and could be improved by devolping an effective heuristic algorithm for finding columns with negative reduced costs instead of solving the slave problem to optimality. Furthermore, in several cases we observed that the quality of the lower bounds of the LP relaxation of model (9) - (12) tends to be better than the one produced by BC without a particular strengthening obtained by adding valid inequalities (see [9]) and, on some instances, it also obtains similar values even considering it.

| $(k,c)$ | Density | BC | | DH-GH | | DH-noGH | |
|---|---|---|---|---|---|---|---|
| | | Time | %gap | Time | %gap | Time | %gap |
| $(3,1)$ | Low | 0.28 | 0.00 | 3.03 | 1.11 | 3.04 | 0.56 |
| | Med. | (29) 7.70 | 0.56 | 25.63 | 0.95 | 28.80 | 0.95 |
| | High | (11) 138.53 | 12.14 | (26) 177.75 | 22.92 | (26) 181.10 | 104.58 |
| $(3,2)$ | Low | 0.04 | 0.00 | 0.87 | 0.00 | 1.11 | 0.00 |
| | Med. | 0.40 | 0.00 | 5.93 | 2.67 | 6.96 | 4.67 |
| | High | 40.30 | 0.00 | 108.83 | 22.00 | 112.38 | 27.33 |
| $(4,1)$ | Low | (29) 0.68 | 0.33 | 2.50 | 0.00 | 3.02 | 0.33 |
| | Med. | (10) 79.27 | 11.33 | 39.34 | 1.80 | 40.87 | 2.71 |
| | High | * * * | 39.00 | (28) 273.85 | 4.43 | (28) 271.49 | 37.72 |
| $(4,2)$ | Low | 0.60 | 0.00 | 1.55 | 0.00 | 1.38 | 0.48 |
| | Med. | 58.67 | 0.00 | 26.59 | 2.38 | 28.37 | 2.38 |
| | High | (14) 165.26 | 7.86 | (23) 194.69 | 16.67 | (25) 225.22 | 151.67 |
| $(4,3)$ | Low | 0.12 | 0.00 | 0.62 | 0.00 | 0.88 | 0.67 |
| | Med. | 23.15 | 0.00 | 5.38 | 0.00 | 5.89 | 0.00 |
| | High | 12.76 | 0.00 | 54.74 | 11.11 | 55.92 | 15.00 |
| $(5,1)$ | Low | (28) 0.89 | 0.52 | 1.02 | 0.00 | 1.33 | 0.24 |
| | Med. | (6) 15.26 | 12.62 | 39.81 | 0.43 | 41.61 | 0.82 |
| | High | * * * | 44.64 | (27) 223.63 | 4.67 | (28) 243.99 | 30.59 |
| $(5,2)$ | Low | (29) 1.94 | 0.37 | 2.31 | 0.33 | 2.32 | 0.67 |
| | Med. | (13) 74.01 | 6.67 | 44.04 | 5.58 | 42.45 | 5.88 |
| | High | * * * | 33.00 | (25) 239.38 | 10.05 | (25) 234.75 | 129.39 |

Table 1
Percentage optimality gap and computational times (averages).

## 5 Conclusions

In this paper, we present a new formulation for the $(k,c)$-*coloring* problem and a diving heuristic which obtains good quality results in a reasonable amount of time. Furthermore, the diving heuristic outperforms one of the recent algorithms proposed in the literature on instances where the difference between $k$ and $c$ tends to grow.

As future research, and aiming to obtain a Branch-and-Price algorithm, it is necessary to focus on two major aspects of the problem. Firstly, it is important to develop a heuristic procedure for solving the slave problem (13) - (16) efficiently in order to accelerate the overall time required by the column generation algorithm.

Secondly, it would be intersting to obtain a deeper insight on the structure of the $(k,c)$-*coloring* problem in order to derive a *robust* branching rule. For the case where $c = k-1$, one possibility is to consider a generalization of the idea used for the VCP (see, e.g., [7], [3], [5]). Let $i,j \in V$ be two non-adjacent vertices, and consider splitting the original problem into two new subproblems

in the following fashion:

- $i$ and $j$ have share exactly $k$ colors, so we can collapse them into one new vertex, or
- $i$ and $j$ share at most $k-1$ colors, so we can add an edge between $i$ and $j$.

However, when $c \neq k-1$ this rule cannot be applied directly, since we obtain as a result a subproblem where the number of maximum colors that two adjacent vertices can share is not fixed. In this regard, further theoretical and practical developments are required.

## References

[1] Aardal, K., S. van Hoesel, A. Koster, C. Mannino and A. Sassano, *Models and solution techniques for the frequency assignment problem*, 4OR **1** (2003), pp. 261–317.

[2] Garey, M. and D. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Freedman, New York, 1979.

[3] Gualandi, S. and F. Malucelli, *Exact solution of graph coloring problems via constraint programming and column generation*, INFORMS Journal on Computing **24** (2012), pp. 81–100.

[4] Halldórsson, M. and G. Kortsarz, *Multicoloring: Problems and techniques*, in: *Mathematical Foundations of Computer Science 2004*, Lecture Notes in Computer Science **3153**, 2004, pp. 25–41.

[5] Held, S., W. Cook and E. Sewell, *Maximum-weight stable sets and safe lower bounds for graph coloring*, Mathematical Programming Computation **4** (2012), pp. 363–381.

[6] Malaguti, E., *The vertex coloring problem and its generalizations*, 4OR **7** (2009), pp. 101–104.

[7] Malaguti, E., M. Monaci and P. Toth, *An exact approach for the vertex coloring problem*, Discrete Optimization **8** (2011), pp. 174–190.

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

[9] Méndez-Díaz, I. and P. Zabala, *Solving a multicoloring problem with overlaps using integer programming*, Discrete Applied Mathematics **158** (2010), pp. 349–354.