Implementing cutting planes for the chromatic violation problem

Diego Delle Donne
ESSEC Business School of Paris
Cergy-Pontoise, France
delledonne@essec.edu

Mariana Escalante
FCEIA - Universidad Nacional de Rosario and CONICET
Rosario, Santa Fe, Argentina
mariana@fceia.unr.edu.ar

María Elisa Ugarte
FCEIA - Universidad Nacional de Rosario
Rosario, Santa Fe, Argentina
ugarte@fceia.unr.edu.ar

ABSTRACT

We consider the minimum chromatic violation problem (MCVP) studied from a polyhedral point of view. A previous paper presents the classical study of the convex hull of feasible solutions in it, introducing several exponentially sized families of valid inequalities and conditions under which they induce facets. Here we address the separation procedures for these families to find proper cuts and add them to the formulation in a cutting-plane fashion for solving the MCVP. As the inequalities analyzed in this work are associated with cliques in the given graph, we explore different clique-finding approaches and compare them within a computational experimentation over a set of DIMACS and random instances.

KEYWORDS

chromatic violation, graph coloring, polyhedral study, separation routines.

1 INTRODUCTION

A k-coloring of a graph is a partition of the vertex set in k stable sets (i.e., pairwise non-adjacent set of vertices). The k-coloring problem asks whether a given graph has a k-coloring or not, and it is known to be NP-Complete if $k \geq 3$. The classical vertex coloring problem (VCP) asks for the smallest k needed to color a given graph and it has many known applications such as frequency assignment problems, course timetabling, scheduling problems, among others. In practice, it is not difficult to find situations where the value for k is actually fixed and the goal is to minimize a *conflict-like* notion among some vertices in the same color class. See [2, 3] for a reference on polyhedral results for VCP.

In order to asses these type of problems, [1] proposes a generalization of the k-coloring problem, namely the *minimum chromatic violation problem* (MCVP), which considers a graph $G = (V, E)$, a set of colors C and a subset of *weak* edges $F \subseteq E$, and asks for a $|C|$-coloring of $G' = (V, E \setminus F)$ minimizing the number of edges from F with both endpoints in the same color class. Also, an edge is *strong* if it belongs to $E \setminus F$.

In [1] a polyhedral study of a formulation for MCVP is initiated. In particular, the authors perform a straightforward adaptation of the classical formulation for the vertex coloring problems studied in [5, 6] which uses binary variables x_{ic} for each vertex $i \in V$ and each color $c \in C$, to indicate whether vertex i is assigned to color c or not. Binary variables z_{ij} for each $ij \in F$ are also introduced, specifying whether vertices i and j receive the same color. Thus, the formulation aims to minimize $\sum_{ij \in F} z_{ij}$ subject to the following constraints:

$$\sum_{c \in C} x_{ic} = 1 \qquad \text{for } i \in V, \tag{1}$$

$$x_{ic} + x_{jc} \leq 1 + z_{ij} \qquad \text{for } ij \in F, c \in C, \tag{2}$$

$$x_{ic} + x_{jc} \leq 1 \qquad \text{for } ij \in E \setminus F, c \in C, \tag{3}$$

$$x_{ic} \in \{0, 1\} \qquad \text{for } i \in V, c \in C, \tag{4}$$

$$z_{ij} \in \{0, 1\} \qquad \text{for } ij \in F. \tag{5}$$

The objective function minimizes the number of (weak) edges whose endpoints receive the same color in a coloring of the graph. Constraints (1) ensure that each vertex is colored with exactly one color, constraints (2) ensure that when the endpoints of a weak edge have the same color, the corresponding binary variable associated to this edge assumes the value 1. Constraints (3) ensure that for each strong edge $ij \in E \setminus F$ and color $c \in C$, at most one endpoint of the edge receives color c.

The *chromatic violation polytope* associated with G, F and C, $P_{CV}(G, F, C) \subseteq \mathbb{R}^{|V||C|+|F|}$ is the convex hull of feasible solutions (x, z) satisfying (1)-(5). We simply write $P_{CV}(G)$ or $P_{CV}(G, F)$ instead of $P_{CV}(G, F, C)$ when the sets F and/or C are clear from the context.

Throughout this paper, we consider the graph $G = (V, E)$, $F \subseteq E$ the set of weak edges in G and C, the set of available colors. If $U \subset V$, the *graph induced* by U, $G[U]$, is the subgraph of G with vertex set U and such that two nodes in U are connected by an edge if they are in the original graph G. For a vertex $i \in V$, $\Gamma(i)$ is the set of nodes adjacent to i in G. We also define $\Gamma_S(i)$ as the set of *strong neighbours* of i, i.e., $\Gamma_S(i) = \{j \in \Gamma(i) : ij \in E \setminus F\}$, and equivalently, $\Gamma_W(i) = \{j \in \Gamma(i) : ij \in F\}$ is the set of *weak neighbours* of i. For $U \subseteq V$, $F(U)$ is the set of weak edges of the graph $G[U]$.

2 KNOWN FAMILIES OF VALID INEQUALITIES FOR THE CHROMATIC VIOLATION PROBLEM

In [1], several exponentially sized families of valid inequalities are introduced and proved to define facets, under some additional hypothesis. We next focus on those which are associated with cliques in the given graph. The results concerning the validity of these inequalities correspond to Theorem 2.8 and Propositions 2.11 and 2.14. from [1], and we present them below for completeness.

Weak clique inequalities. Let $K \subseteq V$ be a set of nodes inducing a complete subgraph in G. For $c \in C$, the *weak clique inequality* (WKI)

$$\sum_{i \in K} x_{ic} \leq 1 + \sum_{ij \in F(K)} z_{ij}$$

10.48786/alioeuro.2022.04

is valid for $P_{CV}(G)$. Moreover, if K is a maximal clique in $G - (F \setminus F(K))$ and $|C| > \chi(G - (F \setminus F(K)))$, then it defines a facet of $P_{CV}(G)$.

Multicolor clique inequalities. Assume that $K \subseteq V$ is a clique in G and $T \subseteq C$. If $|C| > \chi(G - (F \setminus F(K))) + 1$ and $1 \leq |T| \leq |K| \leq |C| + |T|$ then the *multicolor clique inequality* (MKI),

$$\sum_{t \in T} \sum_{i \in K} x_{it} \leq |T| + \sum_{ij \in F(K)} z_{ij} \tag{6}$$

induces a non-empty face of $P_{CV}(G)$. Moreover, under these conditions (6) induces a facet of $P_{CV}(G)$ if and only if $1 \leq |T| < |K| < |C| + |T|$ and there is no $r \in V \setminus K$ such that $K \subseteq \Gamma_S(r)$.

Multicolor combinatorial clique inequalities. Given $K \subseteq V$ inducing a complete subgraph with all weak edges, a set of colors $T \subseteq C$ and values $q_t \in \mathbb{N}$ for each $t \in T$, the *multicolor combinatorial clique inequality* (MCKI)

$$\sum_{t \in T} \sum_{i \in K} q_t x_{it} \leq \sum_{t \in T} \frac{q_t(q_t + 1)}{2} + \sum_{ij \in F(K)} z_{ij}$$

is valid for $P_{CV}(G)$. Furthermore, if $|C| > \chi(G - (F \setminus F(K))) + 1$, then the MCKI defines a facet of $P_{CV}(G)$ if and only if there is no $r \in V \setminus K$ with $K \subseteq \Gamma_S(r)$ and $q_\Sigma < |K| < |C| + q_\Sigma$, where $q_\Sigma = \sum_{t \in T} q_t$, and either $|T| > 1$ or $q_\Sigma + 1 < |K|$.

In the next sections, we present separation routines for these families of valid inequalities and our computational results applying them in a cutting plane framework.

3 SEPARATION ROUTINES FOR MCVP

Given a point $(x^*, z^*) \in \mathbb{R}^{|V||C|+|F|}$, the *separation* problem for a family of valid inequalities consists of finding an inequality of this family violated by (x^*, z^*), or determining that such inequality does not exist.

In order to develop separation routines for the above mentioned families of valid inequalities, we focus on heuristics for finding a clique of maximum weight in the given graph. We then incorporate these procedures in a branch-and-cut algorithm to solve MCVP. As far as the branching rules are concerned, we resort to the standard branching rules on the CPLEX framework used in the experimentation (see Section 4).

The separation phase is the central part of a cutting plane algorithm and efficient separation algorithms are crucial for the success of this approach.

Clique-related inequalities are used as cutting planes in many problems [6, 7], and associated separation problems are usually NP-Hard.

A first approach that may be considered is to construct in a preprocessing phase a list of maximal cliques and keep them in a pool. At each separation round, this list may be scanned in an attempt to find violated cuts. This approach has the advantage of looking for cliques once, but the clique construction does not exploit the information of the current fractional solution being considered.

As a more interesting approach, we consider the search for cliques by using information provided by the current point to be cut off. We implemented two kinds of separation routines: a greedy heuristic and a backtracking-based search. In particular, for the greedy heuristic, we consider different orderings to traverse the list of vertices. Additionally, we implemented a local search procedure to improve the solution of the greedy heuristic. For this procedure, we propose 2 different neighborhood definitions and a combination of them.

Our backtracking-based clique-search procedure looks for all possible cliques in the graph. At each level in the backtracking tree, a new vertex is considered for addition into the clique under construction. As this search may require an exponential number of steps, in our implementation we limit the operations made by the search with a predefined value, which depends on the size of the instance being solved. We give more details on this subject in Section 4.

On the other hand, the heuristic separation procedure we use consists of a standard greedy algorithm for clique searching. For that purpose, we define *weights* $w(i)$ for each vertex i of G. The algorithm starts with the vertex i with maximum weight, defining a singleton clique $K = \{i\}$ and, in each iteration, it visits the vertices of G in decreasing order of their weights, greedily adding each vertex j to K if $K \cup \{j\}$ is a clique.

In addition, different local search techniques are then applied to improve the results of the greedy algorithm, described in the next section.

3.1 Weak clique inequalities

Let (x^*, z^*) be the fractional solution obtained after solving the linear relaxation of MCVP. In order to solve the separation problem for WKI, we search for a clique $K \subseteq V$ and a color $c \in C$ such that

$$\sum_{i \in K} x^*_{ic} - \sum_{ij \in F(K)} z^*_{ij} > 1$$

In our implementations, for the greedy algorithms, we consider three different weights to estimate the contribution of each vertex to a weak clique inequality, for a fixed $c \in C$:

- $w^c(i) = x^*_{ic}$
- $w^c_Z(i) = x^*_{ic} - \sum_{ij \in F} z^*_{ij}$
- $w^c_N(i) = x^*_{ic} - \sum_{j \notin \Gamma(i)} x^*_{jc}$

The first two weight functions take into account values associated only to the vertex i. The third one considers the weight of the vertex i in relation to the weights of its non-neighbours, as they will not be considered for a clique containing i. If a preliminary clique is found, the weights can be refined by neglecting non-neighbours of the nodes in the clique that are not eligible for insertion. In addition, local search techniques are then applied to improve the results of the greedy algorithm. In our implementations, we perform $(1, 1)$-swaps, $(1, 2)$-swaps, and a combination of both, considering when these exchanges improve the clique greedily obtained. That is, we make a swap of nodes in the given clique if the weight of the new clique is bigger than the weight of the original one. The *weight of a clique* K we consider is $w(K) = \sum_{i \in K} x^*_{ic} - \sum_{ij \in F(K)} z^*_{ij}$.

In the $(1, 1)$-swap, a vertex $i \in K$ is replaced by a non-neighbour j, if $K \setminus \{i\} \cup \{j\}$ is a clique and

$$x^*_{jc} - x^*_{ic} + \sum_{s \in K} \bar{z}_{is} - \sum_{s \in K} \bar{z}_{js} > 0,$$

where $\bar{z}_{ij} = z^*_{ij}$, if $ij \in F$ and $\bar{z}_{ij} = 0$ otherwise. In a similar way, the $(1, 2)$-swap, consists on replacing a vertex $i \in K$ by two non-neighbouring vertices j and k, if $K \setminus \{i\} \cup \{j, k\}$ is a clique and

$$x^*_{jc} + x^*_{kc} - x^*_{ic} + \sum_{s \in K} \bar{z}_{is} - \sum_{s \in K} \bar{z}_{js} - \sum_{s \in K} \bar{z}_{ks} - \bar{z}_{jk} > 0.$$

We consider a third strategy combining these two swap techniques. Given a clique K found greedily with some of the above mentioned weights, try to perform a $(1,1)$-swap on its nodes. If this cannot be done for any node in K, it looks for $(1,2)$-swaps.

Let us remark that we look for cliques using greedy techniques with the three different weights, trying to find a clique for which the weak clique inequality is violated. Once we find it, we apply a local search procedure to try to improve it.

3.2 Multicolor clique inequalities

The separation of the MKI inequalities consists of finding a clique K and a set of colors T such that the MKI inequality associated with K and T is violated. However, to tackle these two issues (finding the clique and the set of colors) at the same time seems to be extremely difficult. Based on this observation, we decided to focus on the separation problem assuming that the clique is fixed before-hand. We explain in Section 3.4 how this is used in our implementation.

Given a clique K, for each color c, we compute $S_c = \sum_{i \in K} x^*_{ic}$ and we construct an ordered list $S_{c_1} \geq S_{c_2} \geq \ldots \geq S_{c_k}$. We finally consider $T_1 = \{c_1\}, T_2 = \{c_1, c_2\}, \ldots, T_{|K|-1} = \{c_1, c_2, \ldots c_{|K|-1}\}$, and check whether

$$\sum_{c \in T_r} \sum_{i \in K} x_{it} > |T_r| + \sum_{ij \in F(K)} z_{ij}. \tag{7}$$

If (7) holds for some $r \in 1, \ldots, |K|-1$ we choose T_r as the set of colors to associate with the MKI cut. If such an r does not exist we conclude that, for the given clique K there is no MKI violated by (x^*, z^*). This fact can be proved with really simple arguments. Therefore, the proposed routine can be considered an exact separation routine, provided that K is fixed.

3.3 Multicolor combinatorial clique inequalities

For this family of inequalities we follow almost the same strategy as before. In this case, let us assume that we have a clique K with all its edges in F, and rewrite the MCKI as follows:

$$\sum_{t \in T} \left(q_t \sum_{i \in K} x_{it} - \frac{q_t(q_t+1)}{2} \right) \leq \sum_{ij \in F(K)} z_{ij}.$$

Consider a fractional solution (x^*, z^*), the clique K, and let

$$X_t = \sum_{i \in K} x^*_{it} \quad , \quad \bar{z} = \sum_{ij \in F(K)} z^*_{ij}.$$

We shall look for $T \subseteq C$ and values of $q_t \in \mathbb{N}$, with $t \in T$, such that

$$\sum_{t \in T} \left(q_t X_t - \frac{q_t(q_t+1)}{2} \right) > \bar{z}.$$

Defining $m_t(q) = -\frac{1}{2}q^2 - \frac{1}{2}q + X_t q$ the goal is to maximize m_t, for each $t \in C$. Since the roots of m_t are 0 and $2X_t - 1$, the maximum is reached at $q = X_t - \frac{1}{2}$. Then,

$$q_t = \max \left\{ 0, \arg \max \left\{ m_t \left(\left\lfloor X_t - \frac{1}{2} \right\rfloor \right), m_t \left(\left\lceil X_t - \frac{1}{2} \right\rceil \right) \right\} \right\}.$$

Let us consider an ordered list in $\{m_t(q_t) : t \in C \wedge m_t(q_t) > 0\}$ in decreasing order, i.e. $m_{t_1}(q_{t_1}) > m_{t_2}(q_{t_2}) > \ldots > m_{t_r}(q_{t_r})$. If $T' = \{t \in C : m_t(q_t) > 0\}$ and $\sum_{t \in T'} m_t(q_t) \leq \bar{z}$, then there is no MCKI associated with the clique K that cuts (x^*, z^*). Otherwise, we take $T = \{t_1, t_2, \ldots, t_l\}$, where l is the first value such that $\sum_{i=1}^{l} m_{t_i}(q_{t_i}) > \bar{z}$.

As in the case of the MKI, the proposed routine for the MCKI can be considered an exact separation routine, provided that K is fixed.

3.4 Implementation details

As we remarked above, for a fixed clique K, the way we choose the set of colors for MKI and also the combinatorial parameters q_t for MCKI, determine exactly if there is an inequality in the corresponding family associated with K which cuts-off the fractional point.

In order to provide cliques to the separation routines for MKI or MCKI, we implemented a *clique pool* in which cliques are stored during the execution of the branch and cut. This pool is fed by the cliques found by the separation routine for the weak clique inequalities (WKI). We shall note that this implementation decision forbid the use of the MKI or MCKI family without the use of WKI.

4 COMPUTATIONAL EXPERIMENTATION

In this section we present our results on a computational experimentation to assess the practical contribution of the considered valid inequalities within a branch-and-cut algorithm.

The implementation is coded in Java by resorting to the CPLEX Java API framework version 12.8. The experimentation was carried out in a notebook running Windows with an AMD RYZEN 7 processor and 8Gb RAM.

Our test set is based on two groups of instances, one of them generated from the set of DIMACS instances for the standard graph coloring problem with up to 100 vertices (26 instances in total), while the second group is composed of randomly generated instances. Since there are no previous research of this problem, we build a new set of instances from the DIMACS's ones and we named this set as *DIMACS-based* instances. Its definition is motivated by the fact that MCVP involves a coloring of the graph $G - F$; a DIMACS-based instance consists on a graph $G = (V, E)$ and a set of weak edges F such that $G_{DIM} := G - F$ belongs to the above mentioned DIMACS family. For our experimentation, we considered different sets of weak edges F for each DIMAC's graph G_{DIM}, according to a probability q for each non-adjacent pair of vertices of G_{DIM}, of being chosen as a weak edge. The number q varies in the set $\{0.3, 0.5, 0.7, 0.9\}$. In this way we defined 104 DIMACS-based instances.

The group of random instances is generated as follows. Each instance involves a graph $G[n, p, q]$ with n nodes, where there is an edge between a pair of nodes with probability p and it is weak with probability q. We considered n belonging to the set $\{20, 25, 30, 35, \ldots 50\}$ and values of p and q in $\{0.3, 0.5, 0.7, 0.9\}$ for a total of 112 random instances.

For each graph G in these two groups of instances, we fix the number of colors at the chromatic number of the graph $G - F$, which was previously obtained for the random instances and it is known for most of the DIMACS graphs we considered [4]. In the case that the chromatic number is not known we use the best known upper bound for it [4]. By choosing the number of available colors this way, we ensure that there always exists a feasible solution for all of our MCVP instances. In our experimentation, we limit the running time to 900 seconds per instance.

4.1 Separation parameters for the WKI

We first perform a computational experimentation varying the different parameters involved in the separation routines developed for the WKI. Namely,

- **BACKTRACKING**: limiting the number of recursive calls to $|V|^2$.
- **GREEDY**: considering each of the three different orderings for the vertices, denoted as w, w_Z and w_N, according to the weights defined in Section 3.1.
- **GREEDYLS**: applying the greedy search and improving the solution by a local search. We proposed two different neighbourhoods and a combination of both, denoted as SWAP1, SWAP2 and SWAPB, respectively, as described in Section 3.1.

Table 1 shows the number of nodes in the B&B tree for each possible considered configuration, including the option of not using any of our cuts (column CPLEX). Each row represents a group of either DIMACS-based or random instances, which differ just in the parameter q (i.e., the probability for an edge to be weak). The reported value in each cell is the average result of the corresponding group of instances.

According to these results the best configuration is to apply a greedy algorithm considering the weights w_Z with local search performing $(1, 1)$-swaps on the clique obtained greedily.

Despite the fact that we have chosen the number of nodes as a measure of efficiency of the cuts, we also include in Table 2 the average times for each family of instances (DIMACS-based and random) in the given configuration to show that the time cost also behaves properly in our context.

Based on the obtained results, we keep the mentioned best configuration to perform a computational experimentation with the rest of the families of valid inequalities, with the goal to assess their practical contribution to the solution process.

We should remark the fact that results in Table 1 and Table 2 could be slightly misleading. The reason for this is that the average number of nodes is computed including also those instances achieving the time limit bound. Such instances report a number of nodes smaller than the real number of nodes which the method would require to solve the instance to optimality. For further analyzing this issue in a fair way, we computed the average node-counts restricted only to those instances which are solved to optimality by all methods. We could verify then that the behaviour is similar and so we hold to the conclusion stated above. We omit to give detailed results here but we support this in Section 4.2, where we compare the node-counts of different methods restricted only to those instances which were solved to optimality by all of them.

4.2 General results for WKI, MKI and MCKI

We conduct experiments to determine the strength of the cuts associated with the families WKI, MKI and MCKI when combined together. We report in Table 3 the average number of nodes and average times obtained by different combinations of the cuts. In order to make a fair comparison among the node-counts, we only consider those instances solved to optimality within the 900 seconds time limit, both by CPLEX and by every chosen configuration (i.e., "**GREEDYLS-w_N-SWAP1**", with every combinations of cuts). We should remark then that each row may involve now a different number of instances for the group, which we depict in column **#Instances**.

The column CPLEX, corresponds to the results obtained by running CPLEX without any of the cuts considered in this contribution. The rest of the columns of the table show the results of the chosen configuration **GREEDYLS-w_N-SWAP1** with the following combinations of cuts.

- WKI: weak clique inequalities;
- WKI+MCKI: weak clique inequalities and multicolor combinatorial clique inequalities;
- WKI+MKI: weak clique inequalities and multicolor clique inequalities;
- WKI+MKI+MCKI: weak clique inequalities, multicolor clique inequalities and multicolor combinatorial clique inequalities.

As the results in Table 3 evidence, the strength of the cuts is maximized when used alltogether (WKI+MCKI+MKI), exploring on average 305 nodes, which represents less than 7% of the nodes explored by CPLEX (i.e., 4878 nodes in average). When comparing this configuration with the one using just the WKI, we can see that it also reduces considerably the number of explored nodes from 1413 to 305.

On the other hand, we can see that the addition of all the cuts to the separation routine, may increase the running times. In fact, when comparing option WKI+MKI+MCKI against CPLEX, the average time is raised from 7,43 to 33,52 seconds.

This behaviour suggests that the implemented cuts are in fact effective and the added inequalities considerably help in strengthening up the linear relaxation of the original formulation. However, the increase of the execution times may indicate that the implemented separation routines in this work may be revised with the goal of improving their efficiency (e.g., by resorting to better data structures and/or algorithms from the vertex coloring literature). Indeed, we analyzed the dual bounds obtained at the root node of the branch-and-cut tree with and without using the proposed cuts. Our results showed that the gap of this bound against the optimal solution is decreased (in average) from 16,08% (without cuts) to 10,04%, when including all the considered separation routines.

5 CONCLUSIONS AND FUTURE WORK

In this contribution we consider a generalization of the k-coloring problem, referred to as the minimum chromatic violation problem (MCVP). Taking into account several exponentially sized families of valid inequalities for the MCVP introduced in [1], we address the associated separation problems with the goal of assessing the practical contribution of these inequalities when incorporated as cutting planes within the computational solution of the MCVP. The families of valid inequalities considered in this work are associated with cliques in the graph, thus we develop different heuristic procedures for finding cliques in the given graph.

We conducted a computational experimentation on a test set composed of 216 instances (104 DIMACS-based and 112 random) with different sizes and characteristics. At a first stage of the experimentation, we deduced a proper parameter configuration for the separation routine of one of the families, namely the *weak clique inequalities* (WKI). Afterwards, we fix these parameters and experiment with the rest of the families addressed in this work, as their separation procedures depend on a *clique pool* fed by the cliques generated by the WKI separation routine. In this last stage of the experimentation, we analyse the results obtained by the different possible combinations of WKI with MKI (*multicolor*

Table 1: Node report for different configurations for WKI.

Group	CPLEX	BACKTRACKING	GREEDY			GREEDYLS								
						w			w_Z			w_N		
			w	w_Z	w_N	SWAP1	SWAP2	SWAPB	SWAP1	SWAP2	SWAPB	SWAP1	SWAP2	SWAPB
1-FullIns_3	7321	2486	12451	7233	10506	9086	9699	5436	10485	6176	5791	9744	8967	12715
1-FullIns_4	50250	4259	625	257	5634	3	626	337	7	224	190	1857	5037	4040
1-Insertions_4	78330	6324	6093	4186	11771	3129	6552	4692	2132	3431	2803	5088	10426	8598
2-FullIns_3	111575	8468	13104	10702	24278	7834	10467	7650	5234	8962	6259	12776	22185	16450
2-Insertions_3	161930	8180	46750	26825	46553	23473	40695	25821	10327	25033	15733	21155	43313	35396
3-FullIns_3	53025	3811	1936	1033	4686	1119	1090	753	692	779	581	287	4104	2308
3-Insertions_3	112189	8043	21596	11581	22321	9044	16016	11886	6017	10481	7802	10162	19714	16549
4-Insertions_3	102356	5229	10888	3993	13846	3253	8481	6664	1753	3220	1807	5071	11006	9253
david	37460	9881	3848	2927	5678	3356	3391	2435	2851	2251	1698	3274	4861	3648
huck	53210	7632	662	605	4311	605	840	1072	309	803	684	1562	3592	3493
jean	74989	7848	2309	2223	3929	3015	2048	960	2625	2684	1215	3341	5383	2295
mug100_1	75300	3312	6950	1054	6175	2011	5687	2022	0	928	222	2205	5886	5092
mug100_25	72125	4243	7135	1589	6798	2001	6259	4119	302	1366	547	2649	6343	5131
mug88_1	89325	4463	9301	2472	10509	2341	7407	4684	209	2172	1272	3118	9571	7643
mug88_25	85013	4109	8073	3436	10682	2942	7033	4819	1139	2144	1021	4285	9817	8242
myciel3	13	14	16	16	19	16	14	14	16	15	15	19	19	19
myciel4	13367	10951	14828	12660	15767	16587	12978	14735	13868	14651	14428	13496	14582	14188
myciel5	183391	18085	32795	20757	52794	20751	30858	15173	16437	26284	16596	25527	51343	38811
myciel6	34473	5282	281	587	4005	36	44	15	436	15	9	489	3590	2571
queen10_10	12259	4294	1515	1199	3097	1725	1852	1940	1184	1118	985	919	2317	1614
queen5_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
queen6_6	2093	2644	2418	2882	2184	2008	2506	2839	2147	2037	1674	2282	2278	2171
queen7_7	69	44	45	38	62	38	30	34	44	61	48	82	60	64
queen8_12	7991	3246	3378	1505	2824	2579	3405	2531	1214	1328	1403	968	1973	1394
queen8_8	53708	15519	14687	6277	17102	6843	14535	10153	2650	5257	4060	5852	14613	10635
queen9_9	23015	9006	7179	2945	8039	4951	6775	5408	1637	2762	1782	2582	7297	5241
$G[20, 0.3, *]$	2	2	3	1	1	3	2	2	1	2	2	1	1	0
$G[20, 0.5, *]$	11	11	13	8	10	9	7	12	6	9	4	11	8	9
$G[20, 0.7, *]$	109	34	123	55	80	82	76	106	37	33	33	102	70	101
$G[20, 0.9, *]$	25	12	34	15	21	40	34	40	15	15	13	50	20	25
$G[25, 0.3, *]$	1	6	6	5	6	5	6	5	5	3	1	1	3	5
$G[25, 0.5, *]$	678	335	888	282	497	271	672	313	171	227	301	344	664	486
$G[25, 0.7, *]$	752	511	587	460	1086	378	459	461	626	262	434	503	752	950
$G[25, 0.9, *]$	3206	2138	1753	2397	2836	1924	3579	1919	2456	3435	2710	1782	1733	4775
$G[30, 0.3, *]$	144	65	80	69	120	71	63	63	69	63	66	47	66	68
$G[30, 0.5, *]$	797	743	428	293	535	523	788	420	526	493	209	485	663	686
$G[30, 0.7, *]$	3601	2282	2413	2666	2393	2674	2067	2302	2116	1714	1776	2551	2347	2183
$G[30, 0.9, *]$	5469	3963	5273	6190	5859	10126	6592	7233	5829	4538	12104	5838	5672	6436
$G[35, 0.3, *]$	349	207	281	207	642	259	222	245	189	191	213	254	212	316
$G[35, 0.5, *]$	2734	1736	3535	1628	2860	2147	2418	2265	694	1587	1511	2321	2696	2951
$G[35, 0.7, *]$	20465	11009	14241	12579	13995	6052	9606	10466	5880	10473	8624	13877	22242	17990
$G[35, 0.9, *]$	4669	4209	4238	3773	4032	4688	4754	3935	4853	3905	5085	4201	3493	3866
$G[40, 0.3, *]$	84	78	74	114	110	92	70	69	81	74	93	95	94	71
$G[40, 0.5, *]$	6319	3375	4325	3217	4910	3161	2741	3737	1725	4077	1738	2818	4866	4209
$G[40, 0.7, *]$	32951	17273	16897	18231	29006	12684	17411	22552	7906	21629	10240	20027	17820	21933
$G[40, 0.9, *]$	12650	5016	7242	4366	3987	5309	3792	6192	5647	6218	5204	6475	6276	7029
$G[45, 0.3, *]$	17417	3708	12846	3473	13231	4890	12176	5836	2315	3700	2038	7422	12358	8653
$G[45, 0.5, *]$	132353	15432	49556	25873	40826	13366	41065	29486	13717	19856	18977	24236	40140	45825
$G[45, 0.7, *]$	26723	10929	8922	23577	14271	11128	14693	17150	8614	12260	9664	14613	10325	23154
$G[45, 0.9, *]$	29116	22902	9866	20114	29972	11363	9603	11493	5236	12209	8169	13046	7375	8469
$G[50, 0.3, *]$	134894	29668	48925	17396	72277	14296	67752	35312	7566	22337	15452	30288	52433	39224
$G[50, 0.5, *]$	141635	38901	26045	18946	51960	13879	23120	17775	11136	19995	17839	16940	55500	36362
$G[50, 0.7, *]$	54647	11981	4139	3493	5876	2191	4077	2968	1472	3076	2462	2505	5228	4133
$G[50, 0.9, *]$	77962	9452	16059	14416	15551	8889	15939	13804	6888	14384	8864	12384	14470	18115
Average	**40825**	**6543**	**8660**	**5793**	**11306**	**4801**	**8205**	**6155**	**3324**	**5388**	**4119**	**5963**	**9922**	**8807**

Table 2: Time report for different configurations for WKI.

INSTANCES	CPLEX	BACKTRACKING	GREEDY			GREEDYLS								
						w			w_Z			w_N		
			w	w_Z	w_N	SWAP1	SWAP2	SWAPB	SWAP1	SWAP2	SWAPB	SWAP1	SWAP2	SWAPB
DIMACS-BASED	663,2	682,1	676,6	675,4	677,1	678,9	681,3	689,3	679,6	680,1	677,9	681,1	680,2	679,3
RANDOM	92,7	158,3	135,8	136,6	129,6	159,3	137,8	139,4	168,2	139,0	151,6	158,1	118,2	141,0
GENERAL AVERAGE	**367,4**	**410,5**	**396,2**	**396,0**	**393,2**	**409,5**	**399,5**	**404,2**	**414,4**	**399,5**	**405,0**	**410,0**	**388,8**	**400,2**

Table 3: Time-node report for combinations of WKI, MCKI and MKI

Group	CPLEX		GREEDYLS-w_N-SWAP1								#Instances
			WKI		WKI + MCKI		WKI + MKI		WKI + MKI + MCKI		
	time	nodes	time	nodes	time	nodes	time	nodes	time	nodes	
1-FullIns_3	3,45	4013	48,19	4817	53,70	4867	67,22	678	68,15	678	2
2-Insertions_3	29,70	47419	20,11	3837	39,68	5061	41,59	2604	44,36	2604	1
huck	2,51	40	11,08	173	10,73	112	2,75	0	2,79	0	1
myciel3	0,14	13	0,16	16	0,16	16	0,16	6	0,16	6	4
myciel4	1,17	2834	10,10	1914	7,27	1043	2,61	41	2,66	41	2
queen5_5	0,27	0	0,27	0	0,27	0	0,29	0	0,29	0	4
queen6_6	7,20	1943	19,81	1377	21,26	1377	73,47	769	74,56	769	2
queen7_7	5,07	69	15,39	44	15,97	44	28,30	31	29,01	31	4
$G[20, 0.3, *]$	0,15	2	0,16	1	0,16	2	0,16	0	0,16	0	4
$G[20, 0.5, *]$	0,25	11	0,23	6	0,21	8	0,28	3	0,25	3	4
$G[20, 0.7, *]$	0,23	109	0,48	37	0,53	50	0,42	2	0,42	8	4
$G[20, 0.9, *]$	0,29	25	0,64	15	0,64	15	0,66	29	0,67	29	4
$G[25, 0.3, *]$	0,25	1	0,17	5	0,19	6	0,19	6	0,20	6	4
$G[25, 0.5, *]$	0,45	678	1,30	171	1,12	132	0,98	45	0,96	48	4
$G[25, 0.7, *]$	0,80	752	4,29	626	1,57	90	1,98	24	2,11	17	4
$G[25, 0.9, *]$	3,06	3206	5,19	2456	5,76	2456	48,05	1753	49,68	1753	4
$G[30, 0.3, *]$	0,26	144	0,31	69	0,33	66	0,35	46	0,36	46	4
$G[30, 0.5, *]$	0,42	243	0,58	120	0,62	117	1,11	32	1,15	25	3
$G[30, 0.7, *]$	2,66	3601	14,90	2116	13,56	1088	7,82	181	9,41	195	4
$G[30, 0.9, *]$	1,67	547	3,55	686	3,88	686	42,82	524	43,49	524	3
$G[35, 0.3, *]$	0,49	320	0,78	223	0,87	196	0,85	116	0,87	100	3
$G[35, 0.5, *]$	2,55	2734	9,17	694	9,00	403	7,31	143	8,25	152	4
$G[35, 0.7, *]$	5,59	2524	18,18	2674	31,12	1301	60,61	772	48,51	393	3
$G[35, 0.9, *]$	3,61	135	6,12	737	6,44	737	7,97	99	8,19	99	2
$G[40, 0.3, *]$	0,59	84	1,28	81	1,25	84	1,36	82	1,43	82	4
$G[40, 0.5, *]$	12,32	7933	58,33	2371	114,76	2686	70,25	797	59,13	494	2
$G[40, 0.7, *]$	1,87	67	28,23	18	28,91	18	8,37	14	8,42	14	1
$G[40, 0.9, *]$	45,33	14999	58,56	8050	83,30	8050	467,66	572	463,01	572	2
$G[45, 0.3, *]$	12,33	17417	15,73	2315	19,42	1641	13,47	1014	13,80	964	4
$G[45, 0.5, *]$	2,08	227	16,57	231	34,52	267	14,30	70	14,95	77	2
$G[45, 0.7, *]$	68,12	49093	432,00	15868	297,58	6369	653,39	1488	446,66	727	2
$G[45, 0.9, *]$	4,49	66	35,27	208	36,25	208	42,56	102	43,06	102	1
$G[50, 0.3, *]$	4,35	1057	6,60	583	6,33	507	48,27	1926	50,24	1926	2
$G[50, 0.5, *]$	265,84	149398	850,26	19392	233,27	2370	144,27	924	101,27	492	1
$G[50, 0.7, *]$	1,73	44	12,12	37	8,61	23	18,42	13	26,00	26	1
Total average	**7,43**	**4878**	**26,18**	**1413**	**19,98**	**909**	**38,31**	**343**	**33,52**	**305**	**100**

clique inequalities) and/or MCKI (*multicolor combinatorial clique inequalities*).

As a strong point on our contribution, we highlight that the number of nodes in the branching tree is significantly decreased when the proposed cutting planes are used, thus evidencing their effectiveness. Indeed, we obtained good preliminary results showing that the dual bound given by the linear relaxation improves considerably when using our cuts in the root node without performing any branching.

The drawback, on the other hand, is the fact that execution times are increased when the separation routines are included in the algorithm, and so we shall conclude that this first implementation may not be as efficient as it needs to be. Indeed, as a future/ongoing line of work we are working in several ideas to improve the efficiency of our implementation. Our main focus is to improve our backtracking algorithm, since our implementation was quite straightforward and it didn't make profit of important characteristics of this type of algorithms. In particular, we are inspecting bounding procedures which may help to significantly reduce the size of the backtracking tree, allowing to find good solutions in less time. Our current work in this direction consists in adapting to our specific problem, the branch-and-bound procedure introduced in [8] for the classic maximum weight clique problem.

REFERENCES

[1] Mónica Braga, Diego Delle Donne, Mariana Escalante, Javier Marenco, María E. Ugarte, and María del C. Varaldo. 2020. The minimum chromatic violation problem: A polyhedral approach. *Discrete Applied Mathematics* 281 (2020), 69–80.

[2] Pablo Coll, Javier Marenco, Isabel Méndez-Díaz, and Paula Zabala. 2002. Facets of the graph coloring polytope. *Annals of Operations Research* 116-12 (2002), 79–90.

[3] Diego Delle Donne and Javier Marenco. 2016. Polyhedral studies of vertex coloring problems: The standard formulation. *Discrete Optimization* 21 (2016), 1–13.

[4] Stefano Gualandi and Federico Malucelli. 2012. Exact Solution of Graph Coloring Problems via Constraint Programming and Column Generation. *INFORMS Journal on Computing* 24 (2012), 81–100.

[5] Isabel Méndez-Díaz and Paula Zabala. 2006. A branch-and-cut algorithm for graph coloring. *Discrete Applied Mathematics* 154-5 (2006), 826–847.

[6] Isabel Méndez-Díaz and Paula Zabala. 2008. A cutting plane algorithm for graph coloring. *Discrete Applied Mathematics* 156-2 (2008), 159–179.

[7] Manfred W. Padberg. 1973. On the facial structure of set packing polyhedral. *Mathematical Programming* 5 (1973), 199–215.

[8] Pablo San Segundo, Fabio Furini, and Jorge Artieda. 2019. A new branch-and-bound algorithm for the Maximum Weighted Clique Problem. *Computers & Operations Research* 110 (2019), 18–33. https://doi.org/10.1016/j.cor.2019.05.017