Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 346 (2019) 199–208

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Facet-generating Procedures for the Maximum-impact Coloring Polytope

Mónica Braga[a,1] Javier Marenco[a,2]

[a] *Sciences Institute*
National University of General Sarmiento
Buenos Aires, Argentina

Abstract

Given two graphs $G = (V, E_G)$ and $H = (V, E_H)$ over the same set of vertices and given a set of colors C, the *impact on* H of a coloring $c : V \to C$ of G, denoted $\mathcal{I}(c)$, is the number of edges $ij \in E_H$ such that $c(i) = c(j)$. In this setting, the *maximum-impact coloring* problem asks for a proper coloring c of G maximizing the impact $\mathcal{I}(c)$ on H. This problem naturally arises in the context of assigning classrooms to courses, where it is desirable –but not mandatory– to assign lectures from the same course to the same classroom. We are interested in an integer programming approach for this problem. In this work we present two procedures that construct valid inequalities from existing inequalities, based on extending individual colors to sets of colors and on extending edges of G to cliques, respectively. If the original inequality defines a facet and additional technical hypotheses are satisfied, then the obtained inequality also defines a facet. We present a generic separation algorithm based on these procedures, and report computational experiments showing that this approach is effective.

Keywords: coloring, integer programming, facet-generating procedures.

1 Introduction

A recurring problem in course scheduling consists in determining which classrooms are to be assigned to each lecture of each course, in such a way that overlapping lectures receive different classrooms [4], where the starting and ending times of each lecture are given as part of the input. This situation is usually modeled by an undirected graph $G = (V, E_G)$, whose vertices represent the lectures and whose edges join pairs of lectures that cannot receive the same classroom since the corresponding time intervals have nonempty intersection, and by a set C of classrooms. The graph G is usually referred to as the *conflict graph* associated with the lectures. This problem corresponds to the classical graph vertex coloring problem, as any C-coloring

[1] Email: mbraga@campus.ungs.edu.ar
[2] Email: jmarenco@campus.ungs.edu.ar

https://doi.org/10.1016/j.entcs.2019.08.018

c (i.e., an assignment $c : V \to C$ such that $c(i) \neq c(j)$ whenever $ij \in E_G$) corresponds to a feasible assignment of classrooms to lectures. Note that this problem is feasible if and only if $|C| \geq \chi(G)$, where the *chromatic number* $\chi(G)$ represents the minimum number of colors in any feasible coloring of G.

A usual requirement in practical environments asks for all the lectures from the same course to be assigned to the same classroom. However, this requirement is not strict, and it can be violated if not enough classrooms are available. In order to take this requirement into account, the following combinatorial optimization problem was proposed in [1]. In addition to G, we have a second graph $H = (V, E_H)$ defined over the same set of vertices, in such a way that $ij \in E_H$ if and only if i and j are lectures from the same course. We assume $E_G \cap E_H = \emptyset$. If c is a coloring of G, we define the *impact of* c *on* H to be $\mathcal{I}(c) = |\{ij \in E_H : c(i) = c(j)\}|$, i.e., the number of edges from H whose endpoints receive the same color. Given two graphs $G = (V, E_G)$ and $H = (V, E_H)$ with $E_G \cap E_H = \emptyset$ and a set C of colors, the *maximum-impact coloring* problem (MICP) consists in finding a C-coloring of G maximizing the impact on H. MICP is NP-hard [5], even when restricting G to be an interval graph and H to be the union of disjoint cliques [1].

In the context of course timetabling, many real-life requirements and rules arise and particular constraints are imposed in order to obtain practical solutions. Several formulations and techniques for the classical vertex coloring problem were applied in a wide range of publications (see, e.g., [2,3,6,12]). Additionally, some of these applications have given place to scheduling software as in [7,8]. In [9,10] a method based on constraint branching is developed with the aim of reducing the problem size while attempting to retain the best solutions, and it is applied to real instances. In [11], the performance of SAT solvers is compared with standard integer programming techniques for a classroom assignment problem.

Since integer programming techniques have shown to be quite successful for the classical vertex coloring problem and for similar scheduling and timetabling problems, in [1] we proposed to tackle MICP with such techniques. We presented a natural integer programming formulation for MICP and identified several families of facet-inducing inequalities that turned out to be successful at enhancing the performance of a branch and cut procedure. Many of these families of valid inequalities are based on similar ideas and, consequently, the corresponding proofs of facetness resort to similar arguments. Moreover, similar ideas are present in the separation procedures associated with each family. These observations suggest the existence of general results explaining the facetness properties of the identified inequalities, and the design of a unified separation framework for them.

In this work we explore these issues, by presenting two validity- and facetness-preserving procedures, that construct valid inequalities from existing inequalities, enlarging their supports in the process. We also introduce a generic separation algorithm based on these procedures, which starts from a set of inequalities with small supports and seeks to apply these procedures in order to obtain cuts with supports as large as possible. We present computational experiments suggesting that this approach may be competitive with respect to the application of individual

separation algorithms for each family of valid inequalities.

This paper is organized as follows. Section 2 presents the integer programming formulation for MICP. Section 3 presents the two procedures for constructing valid inequalities. Finally, Section 4 reports our computational experiments, and Section 5 includes concluding remarks and ideas for future work.

2 Integer programming formulation

The following integer programming formulation for MICP, introduced in [1], is based on the *standard model* for vertex coloring. For $i \in V$ and $c \in C$, we define the binary *assignment variable* x_{ic} to be $x_{ic} = 1$ if the vertex i is assigned the color c, and $x_{ic} = 0$ otherwise. For every $ij \in E_H$ with $i < j$ we define the binary *impact variable* y_{ij} to be $y_{ij} = 0$ if the vertices i and j are assigned different colors. For $ij \in E_H$, $i < j$, we define $y_{ji} = y_{ij}$ as a notational convenience. In this setting, MICP can be formulated as follows.

$$\max \sum_{ij \in E_H, i<j} y_{ij}$$

$$\text{(1)} \quad \text{s.t.} \quad \sum_{c \in C} x_{ic} = 1 \qquad \forall i \in V$$

$$\text{(2)} \quad x_{ic} + x_{jc} \leq 1 \qquad \forall ij \in E_G,\ \forall c \in C$$

$$\text{(3)} \quad y_{ij} \leq 1 + x_{ic} - x_{jc} \qquad \forall ij \in E_H,\ i < j,\ \forall c \in C$$

$$\text{(4)} \quad y_{ij} \leq 1 + x_{jc} - x_{ic} \qquad \forall ij \in E_H,\ i < j,\ \forall c \in C$$

$$\text{(5)} \quad x_{ic} \in \{0, 1\} \qquad \forall i \in V,\ c \in C$$

$$\text{(6)} \quad y_{ij} \in \{0, 1\} \qquad \forall ij \in E_H,\ i < j$$

The objective function asks for the total impact to be maximized. Constraints (1) and (2) ensure that the x-variables define a proper vertex coloring of G, whereas constraints (3) and (4) force $y_{ij} = 0$ if i and j receive different colors (if, e.g., $x_{jc} = 1$ and $x_{ic'} = 1$ for $c' \neq c$, then (3) implies $y_{ij} = 0$). We do not impose constraints forcing y_{ij} to take value 1 if i and j get the same color, since in any optimal solution this situation is guaranteed, and this property makes the resulting polytope much easier to study. Finally, constraints (5) and (6) ask the variables to be binary.

Definition 2.1 [maximum-impact coloring polytope] Given two graphs $G = (V, E_G)$ and $H = (V, E_H)$ with $E_G \cap E_H = \emptyset$ and a finite set C, we define $\mathcal{P}_{\text{MIC}}(G, H, C) \subseteq \mathbb{R}^{|V||C|+|E_H|}$ to be the convex hull of the points $(x, y) \in \mathbb{R}^{|V||C|+|E_H|}$ satisfying constraints (1)-(6).

3 Facet-preserving procedures

We introduce in this section the two validity- and facetness-preserving procedures for $\mathcal{P}_{\text{MIC}}(G, H, C)$. As mentioned before, several families of facet-inducing inequalities for $\mathcal{P}_{\text{MIC}}(G, H, C)$ are based on similar ideas, and the following family provides an example of this situation.

Definition 3.1 [1] Let $ij \in E_H$ and let D be a proper nonempty subset of colors. We define

$$y_{ij} \leq \sum_{d \notin D} x_{id} + \sum_{d \in D} x_{jd} \tag{7}$$

to be the *partitioned inequality* associated with ij and D.

Inequality (7) asserts that y_{ij} must take value 0 if i is assigned a color from D (hence the first summation in the RHS is null) and j is assigned a color from $C \backslash D$ (hence the second summation in the RHS is null). The partitioned inequalities are facet-defining if $|C| \geq \chi(G) + 1$, and provide a generalization of the model constraints $y_{ij} \leq 1 - x_{ic} + x_{jc}$, for $ij \in E_H$ and $c \in C$, by considering the set D of colors instead of a single color c. This construction appears in several families of facet-inducing inequalities found in [1], suggesting that it could be turned into a general procedure that generates a valid inequality from a previous inequality by replacing the x-variables associated with a color c by the x-variables associated with a set of colors D. Procedure 1 tackles this issue.

If $(x, y) \in \mathcal{P}_{\text{MIC}}(G, H, C \cup D)$ is a feasible solution, where C and D are two disjoint sets, we define x_C to be the projection of x onto the variables associated with colors in C, i.e., $x_C = (x_{vc})_{v \in V, c \in C}$. We say that the colors in C are *consecutive* if there is a linear ordering among them. This is satisfied if, e.g., $C = \{1, \ldots, |C|\}$.

Procedure 1 *Let $\pi x + \mu y \leq \pi_0$ be a valid inequality for $\mathcal{P}_{MIC}(G, H, C)$. Fix $c \in C$, let D be a nonempty set of consecutive colors such that $C \cap D = \emptyset$, and define $C' = (C \backslash \{c\}) \cup D$. Define A and B to be the sets $A = \{v \in V : \pi_{vc} \neq 0\}$ and $B = \{v \in V : \mu_{vw} \neq 0$ for some $w \in V\}$. Finally, for any feasible solution $(x, y) \in \mathcal{P}_{MIC}(G, H, C \cup D)$, define $I(x, y) = \{i \in A \cup B : y_{ij} = 1$ for some $j \in B\}$. Assume that*

(i) *for every $(x, y) \in \mathcal{P}_{MIC}(G, H, C') \cap \mathbb{Z}^{|V||C'|+|E_H|}$, $I(x, y)$ induces a stable set in G,*

(ii) *for every $i \in A \cup B$, if there exist $(x, y) \in \mathcal{P}_{MIC}(G, H, C') \cap \mathbb{Z}^{|V||C'|+|E_H|}$ and a maximal independent set I' in $A \cup B$ such that $I(x, y) \subseteq I'$ and $i \notin I'$, then $\pi_{ic} \leq 0$, and*

(iii) *$\pi x_C + \mu y \leq \pi_0$ is valid for $\mathcal{P}_{MIC}(G, H, C \cup D)$.*

In this setting, the procedure generates the inequality

$$\sum_{i \in A} \sum_{d \in D} \pi_{ic} x_{id} + \sum_{i \in V} \sum_{d \notin D} \pi_{id} x_{id} + \sum_{ij \in E_H} \mu_{ij} y_{ij} \leq \pi_0, \tag{8}$$

for the instance (G, H, C').

The application of Procedure 1 to the model constraint $y_{ij} \leq 1 - x_{ic} + x_{jc}$, for $ij \in E_H$ and $c \in C$ provides the partitioned inequality (7), when replacing c by the set D of colors and combining the obtained inequality with the model constraint (1). In this case we have $A = B = \{i, j\}$ with $ij \in E_H$, hence the hypotheses (i) and (ii) of the procedure are trivially satisfied. Note that under the construction specified by Procedure 1, the inequality (7) is valid for an instance whose color set

includes D. The following results also imply that the obtained inequality (7) is valid and facet-inducing.

Theorem 3.2 *If the hypotheses of Procedure 1 hold, then the inequality (8) is valid for $\mathcal{P}_{MIC}(G, H, C')$.*

The proof of Theorem 3.2 takes an arbitrary feasible solution of $\mathcal{P}_{\text{MIC}}(G, H, C')$ and constructs an associated feasible solution in $\mathcal{P}_{\text{MIC}}(G, H, C \cup D)$ keeping the LHS of (8) either unchanged or at a smaller value. The validity of the original inequality is then applied in order to ensure that the new point satisfies (8).

Theorem 3.3 *If the hypotheses of Procedure 1 hold, $\chi(G) < |C|$, and $\pi x + \mu y \leq \pi_0$ induces a facet of $\mathcal{P}_{MIC}(G, H, C)$, then (8) induces a facet of $\mathcal{P}_{MIC}(G, H, C')$.*

The proof of Theorem 3.3 consists in taking a set of affinely independent points showing that the face of $P(G, H, C)$ induced by $\pi x + \mu y \leq \pi_0$ is indeed a facet, and constructs a suitable number of affinely independent points satisfying (8) with equality.

The second procedure generates valid inequalities from inequalities with smaller supports by replacing an edge of the graph by a clique. Specifically, if $ij \in E_G$ and $c \in C$, then we replace the variables x_{ic} and x_{jc} by the variables $\{x_{kc}\}_{k \in K}$, where K is a clique including ij, and we perform a similar operation on the y-variables incident to i and j. This allows, e.g., to obtaing the following inequalities.

Definition 3.4 [1] Let $K \subseteq V$ be a clique in G and let $i \in V \setminus K$ such that $ik \in E_H$ for all $k \in K$. Let $D \subseteq C$ such that $|D| \leq |C| - |K|$. We define

$$\sum_{k \in K} y_{ik} \leq \sum_{d \in D} \sum_{k \in K} x_{kd} + \sum_{d \notin D} x_{id} \tag{9}$$

to be the *clique-partitioned inequality* associated with the clique K, the vertex i, and the color set D.

Since K is a clique, at most one vertex from K can be assigned the same color as i, so at most one variable in $\{i_{ik}\}_{k \in K}$ can take value 1. The clique-partitioned inequality (9) asserts that $y_{ik} = 0$ for every $k \in K$ if i is assigned a color from D (hence the second summation in the RHS of (9) is null) and every vertex in K is assigned a color from $C \setminus D$ (hence the first summation in the RHS is null). This inequality is valid, and is also facet-inducing under simple technical conditions [1].

We first provide some definitions. If $i \in V$, define $N_G(i) = \{j \in V : ij \in E_G\}$ and $N_H(i) = \{j \in V : ij \in E_H\}$. For $p \geq 1$, we define $G[i, p] = (V \cup \{i_1, \ldots, i_p\}, E'_G)$ to be the graph obtained from G by adding p new vertices (i.e., $i_1, \ldots, i_p \notin V$) in such a way that the vertices $i, i_1, \ldots i_p$ are true twins. In other words, E'_G is obtained by adding an edge between i_t and j, for every $j \in N_G(i) \cup \{i\}$ and $t = 1, \ldots, p$ and between the new vertices, i.e., $E'_G = E_G \cup \{i_t j : j \in N_G(i) \cup \{i\}$ and $t = 1, \ldots, p\} \cup \{i_j i_k : j, k \in \{1, \ldots, p\}\}$. We also define $H(i, p) = (V \cup \{i_1, \ldots, i_p\}, E'_H)$ to be the graph obtained from G by adding p new vertices (i.e., $i_1, \ldots, i_p \notin V$) in such a way that the vertices $i, i_1, \ldots i_p$ are false twins. In other words, E'_H is

obtained by adding an edge between i_t and j, for every $j \in N_H(i)$ and $t = 1, \ldots, p$, i.e., $E'_H = E_H \cup \{i_t j : j \in N_H(i) \text{ and } t = 1, \ldots, p\}$.

Procedure 2 *Let $\pi x + \mu y \leq \pi_0$ be a valid inequality for $\mathcal{P}_{MIC}(G, H, C)$. Fix $ij \in E$ such that i and j are true twins in G and such that $N_H(i) = N_H(j) = \{\ell\}$. Fix also $c \in C$. Assume that*

(i) *$\pi_{ic} = \pi_{jc} \neq 0$ and $\pi_{id} = \pi_{jd} = 0$ for every $d \neq c$,*

(ii) *$\mu_{i\ell} = \mu_{j\ell}$, and*

(iii) *$\pi x_C + \mu y \leq \pi_0$ is valid for $\mathcal{P}_{MIC}(G, H, C \cup D)$, where $D = \{d_1, \ldots, d_{p+1}\}$ and $D \cap C = \emptyset$.*

Define $K = \{i, j\} \cup \{i_1, \ldots, i_p\}$. In this setting, the procedure generates the inequality

$$\sum_{k \in K} \pi_{ic} x_{kc} + \sum_{t \in V \setminus K} \sum_{d \in C} \pi_{td} x_{td} + \sum_{k \in K} \mu_{k\ell} y_{k\ell} + \sum_{uv \in E_H \setminus \{i\ell, j\ell\}} \mu_{uv} y_{uv} \leq \pi_0, \tag{10}$$

for the instance $(G[i, p], H(i, p), C \cup D)$.

Again, Procedure 2 preserves validity and facetness of the original inequality. The proofs of Theorem 3.5 and Theorem 3.6 follow similar lines as the corresponding statements for Procedure 1.

Theorem 3.5 *If the hypotheses of Procedure 2 hold, then the inequality (10) is valid for the polytope $\mathcal{P}_{MIC}(G[i, p], H(i, p), C \cup D)$.*

Theorem 3.6 *If the hypotheses of Procedure 2 hold, $\chi(G) < |C|$, and $\pi x + \mu y \leq \pi_0$ induces a facet of $P(G, H, C)$, then the inequality (10) defines a facet of $P(G[i, p], H(i, p), C \cup D)$.*

The clique-partitioned inequality (9) is obtained by applying Procedure 1 and Procedure 2 to the inequality $x_{ic} + y_{ij} + y_{ik} \leq 1 + x_{jc} + x_{kc}$, for $ij, ik \in E_H$ and $jk \in E_G$, which is valid and facet-inducing if $|C| > \chi(G) + 1$. To this end, we first apply Procedure 2 in order to replace the edge jk by a clique K, thus obtaining

$$x_{ic} + \sum_{t \in K} y_{it} \leq 1 + \sum_{t \in K} x_{tc}. \tag{11}$$

We next apply Procedure 1 to (11) in order to replace the color c by the set D of colors. By combining the obtained inequality with the model constraint (1), we get the clique-partitioned inequality (9), which is thus valid and facet-inducing for $P(G[i, p], H(i, p), C \cup D)$ if $|C| + |D| > \chi(G) + p + 1$.

4 Computational experiments

Procedure 1 and Procedure 2 provide tools for constructing valid inequalities with potentially large supports starting from inequalities with small supports, and also preserve facetness when the right hypotheses are satisfied. This suggests a simple heuristic for trying to find violated valid inequalities within a cutting plane environment: start from a violated or "almost violated" small inequality, and then greedily

try to enlarge the support of the inequality by using the procedures presented in the previous section. If the resulting inequality is violated, then it can be added as a cut. In this section, we explore the design of a cut-generating computational procedure based on these ideas.

The separation procedure has a pool of generic valid inequalities for $P(G, H, C)$, which we propose to call *templates*. In this setting, templates are very simple inequalities with small supports, that are used as starting points of the search for cuts. In our implementation, we resort to the following pool of templates:

(i) the model constraint $y_{ij} \leq 1 + x_{ic} - x_{jc}$, for $ij \in E_H$ and $c \in C$,

(ii) the *edge inequality* $x_{ic} + x_{jc} \leq 1$, for $ij \in E_G$ and $c \in C$,

(iii) the *vertex-clique inequality* $y_{ij} + y_{ik} \leq 1$ for $i, j, k \in V$ such that $jk \in E_G$ and $ij, ik \in E_H$,

(iv) the *semi-triangle inequality* $y_{ij} \leq 2 + (x_{jc} - x_{ic}) + (x_{id} - x_{jd}) - (x_{kc} - x_{kd})$ for $i, j, k \in V$ such that $ik, jk \in E_G$ and $ij \in E_H$, and for $c, d \in C$, $c \neq d$,

(v) the *semi-diamond inequality* $y_{ij} + 2y_{ik} \leq 3 + (x_{kc} + x_{jc} - x_{ic}) + (x_{id} - x_{kd}) + (x_{ke} - x_{ie}) - (x_{\ell d} + x_{\ell e})$ for $i, j, k, \ell \in V$ such that $jk, j\ell, k\ell \in E_G$ and $ij, ik \in E_H$, and for $c, d, e \in D$, $c \neq d$, $c \neq e$, $d \neq e$.

Given a fractional solution $(x^*, y^*) \in P(G, H, C)$, the separation procedure first employs a backtracking algorithm in order to detect violated and "almost violated" instances of these templates. To this end, the vertices and the colors present in the template definition are matched against all possible vertices and colors satisfying the template hypotheses, and all violated and "almost violated" inequalities are identified. Since the templates involve small supports, such a bracktracking procedure is not computationally expensive. An inequality $\pi x + \mu y \leq \pi_0$ is *almost violated* by (x^*, y^*) if $\pi_0 - \varepsilon \leq \pi x^* + \mu y^* \leq \pi_0$, for some small ε. We have used $\varepsilon = 0.25$ in our experiments.

Each valid inequality thus found is then subjected to Procedures 1 and 2. We greedily apply these procedures in order to enlarge the support of the obtained inequalities. The theoretical formulation of both procedures generates a valid inequality for a modified instance of the problem, namely the instance with color set $(C \backslash \{c\}) \cup D$ in Procedure 1 and the instance $(G[i, p], H(i, p), C \cup D)$ in Procedure 2. However, in our implementation we keep the instance fixed and execute the procedures with properly-constructed sub-instances of the original instance as follows. When applying both procedures, we take C to be the set of colors present in the support of the inequality, and we take D to be the remaining set of colors. Furthermore, when applying Procedure 2 we take as the input graph the subgraph of G induced by the support of the inequality, and instead of enlarging an edge into a clique, we search for cliques (in the whole graph) including existing edges. This allows for a fast implementation with no need of modifying the graph and the variable set.

The resulting procedure can potentially generate cuts coming from the families of valid inequalities presented in [1] (although it is not guaranteed that cuts coming

					Cplex 12.5		Cplex + cuts [1]		This work	
Instance	$\|V\|$	$\|E_G\|$	$\|E_H\|$	$\|C\|$	Time (s)	Nodes	Time (s)	Nodes	Time (s)	Nodes
2010.01.I	235	1894	124	21	1.40	1	1.45	1	1.47	1
2010.01.II	267	2593	299	22	16.21	42	12.06	1	11.43	1
2010.02.I	256	1884	118	18	4.15	116	1.96	1	1.85	1
2010.02.II	279	2914	164	26	2.63	9	2.09	1	2.12	1
2011.01.I	265	2092	137	16	33.82	228	2.59	1	2.19	1
2011.01.II	255	2295	218	20	2.36	3	2.12	1	2.11	1
2012.01.I	182	1381	113	18	7.45	51	2.11	1	1.90	1
2012.01.II	235	2220	189	23	3.78	8	2.75	1	2.55	1
2012.02.I	253	1974	162	20	4.58	28	2.31	1	2.23	1
2012.02.II	254	2368	186	22	1.52	1	1.61	1	1.54	1
2014.02.I	172	1201	266	20	* 0.06%	15624	4.02	1	3.23	1
2014.02.II	238	2294	498	20	** 0.36%	210	392.68	16	199.59	93

Table 1
Time to optimality and nodes in the enumeration tree for Cplex as a black-box solver (columns labeled "Cplex 12.5"), for the branch and cut presented in [1] (columns labeled "Cplex + cuts [1]"), and for a branch and cut using the cut-generating procedure presented in this section (columns labeled "This work"), respectively. For the instance marked with "*", Cplex reached the memory limit of 1GB. For the instance marked with "**", Cplex reached the time limit of 1 hour.

from every such family will eventually be generated). In contrast, the branch and cut procedure presented in [1] resorts to a tailored separation procedure for each family of valid inequalities. We compared both approaches within the same implementation, in order to assess the computational effectiveness of the single procedure proposed in this section.

Table 1 shows running times for a set of instances coming from a real setting. The implementation was performed within the Cplex 12.5 environment, and the experiments were carried out on a computer with an Intel Core 2 Duo CPU, with two T8100 cores running at 2 GHz, and 2 GB of RAM memory. We have kept all Cplex parameters at their default values. This table shows the improvement achieved when the separation procedures considered in [1] are employed, and also shows that the performance of the single procedure presented in this work is quite competitive with respect to these results. The branch and cut procedure employed in [1] is not able to solve the last two instances, due either to time limits or memory constraints.

Table 2 compares the number of cuts generated in each case. It is interesting to note that the procedure presented in this work finds a much smaller number of cuts, while obtaining a similar performance. This is due to the fact that the separation procedure for the semi-diamond inequalities used in [1] generates many inequalities. A better tuning of this procedure may generate a smaller number of cuts coming from this family, thus making the difference in the number of cuts less important. Nevertheless, it is not clear whether such a better tuning may be attained without resorting to the techniques presented in this work.

					Cuts		
Instance	$\|V\|$	$\|E_G\|$	$\|E_H\|$	$\|C\|$	Cplex 12.5	Cplex + cuts [1]	This work
2010.01.I	235	1894	124	21	0	0	0
2010.01.II	267	2593	299	22	22	6020	7
2010.02.I	256	1884	118	18	58	4264	9
2010.02.II	279	2914	164	26	0	3652	5
2011.01.I	265	2092	137	16	0	5254	17
2011.01.II	255	2295	218	20	0	2440	4
2012.01.I	182	1381	113	18	29	5472	16
2012.01.II	235	2220	189	23	32	4531	7
2012.02.I	253	1974	162	20	47	4736	12
2012.02.II	254	2368	186	22	0	0	0
2014.02.I	172	1201	266	20	135	26304	55
2014.02.II	238	2294	498	20	277	247029	181

Table 2
Number of cuts found by Cplex (column labeled "Cplex 12.5"), found by individual separation procedures for each family of valid inequalities (column labeled "Cplex + cuts [1]"), and by the procedure presented in this section (column labeled "This work"), respectively.

5 Conclusions

From a theoretical point of view, it is interesting to provide a unified framework explaining many families of facet-inducing inequalities. The facetness proofs provided in [1] contain similar ideas that are repeated many times and that are applied almost with no variations in different proofs, so the facet-preserving procedures presented in this work allow for more elegant proofs of these results. From a practical point of view, our computational experiments show that –at least for the instances considered in this work– it is not necessary to resort to a particular separation procedure for each family of valid inequalities, and that a single cut-generating procedure based on the ideas presented in this work allows to obtain similar computational results instead. This is interesting when implementing a branch and cut procedure, since only one separation procedure must be implemented, and the templates considered in such a procedure can be easily configured. As future work, we plan to extend this experimentation to further instances of MICP.

The facet-generating procedures and the separation algorithm presented in this work can be applied to the standard formulation of the classical vertex coloring formulation. In this case, we must ignore the y-variables in the procedures. It would be interesting to explore whether these ideas can be further refined for the classical vertex coloring problem, in order to get more precise procedures and a more efficient (unified) separation heuristic. It would also be interesting to consider similar facet-preserving procedures for other problems, as these ideas may contribute to provide simpler explanations of existing facets and simpler implementations of cutting-plane-based procedures.

References

[1] Braga, M., D. Delle Donne, R. Linfati, and J. Marenco, *The maximum-impact coloring polytope*, Intl. Trans. in Op. Res. **24** (2017), 303–324, URL: https://doi.org/10.1111/itor.12265.

[2] Burke, E., J. Mareček, A. Parkes, and H. Rudová, *A branch-and-cut procedure for the Udine course timetabling problem*, Annals of Operations Research **194** (2012), 71–87.

[3] Daskalaki, S., T. Birbas, and E. Housos, *An integer programming formulation for a case study in university timetabling*, European Journal of Operational Research **153** (2004), 117–135.

[4] De Werra, D., *An introduction to timetabling*, European Journal of Operational Research **19-2** (1985), 151–162.

[5] Garey, M., and D. Johnson, "Computers and intractability: A guide to the theory of NP-completeness", W. H. Freeman, 1979.

[6] Lach, G., and M. Lübbecke, *Curriculum based course timetabling: new solutions to Udine benchmark instances,* Annals of Operations Research, vol. 194, pp. 255–272, 2012.

[7] Miranda, J., *eClasSkeduler: A course scheduling system for the Executive Education Unit at the Universidad de Chile*, Interfaces **40-3** (2010), 196–207.

[8] Mooney, E., R. Rardin, and W. Parmenter, *Large-scale classroom scheduling*, IIEE transactions **28-5** (1996), 369–378.

[9] Phillips, A., D. Ryan, and M. Ehrgott, *Solving the classroom assignment problem using integer programming,* Proceedings of the 2013 Joint NZSA+ORSNZ Conference, 2013.

[10] Phillips, A., H. Waterer, M. Ehrgott, and D. Ryan, *Integer programming methods for large-scale practical classroom assignment problems*, Computers & Operations Research **53** (2015), 42–53.

[11] Wasfy, A., and F. Aloul, *Solving the university class scheduling problem using advanced ILP techniques*, Proceedings of the 4th IEEE GCC Conference, 2007.

[12] Waterer, H., *A zero-one integer programming model for room assignment at the University of Auckland*, Proceedings of the 1995 ORSNZ Conference, 1995.