Computational experience with general cutting planes for the Set Covering problem

Pasquale Avella [*] Maurizio Boccia [†] Igor Vasiliev [‡]

Abstract

In this paper we present a cutting plane algorithm for the Set Covering problem. Cutting planes are generated by a "general" (i.e. not based on the "template paradigm") separation algorithm based on the following idea:

i) identify a suitably small subproblem defined by a subset of the constraints of the formulation;

ii) run an exact separation algorithm over the subproblem to produce a violated cutting plane, if any exists.

Computational results on difficult small-medium size instances are reported.

1 Introduction

Let $E = \{e_1, \dots, e_m\}$ be a finite set and let $S = \{S_1, \dots, S_n\}$ be a given collection of subsets of E. Let $F \subset \{1, \dots, n\}$ be an index subset. F is said to cover E if $E = \bigcup_{j \in F} S_j$.

The *Set Covering Problem* (SCP) is to find a minimum weighted cover of E. SCP is generally NP-hard, and has relevant applications in Crew Scheduling, Vehicle Routing, Machine Learning.

Let $\boldsymbol{c} = (c_1, c_2, \dots, c_n)$ be a set of weights associated with the elements of E. Let $\boldsymbol{A} = (a_{ij})$ be a matrix with entries $a_{ij} \in \{0, 1\}$, where $a_{ij} = 1$ if $e_i \in S_j$, 0 otherwise, and let $\mathbf{1}$ denote a vector of ones of appropriate size. *SCP* can be formulated as:

$$\begin{aligned} \min \quad & \boldsymbol{c}^T \boldsymbol{x} \\ & \boldsymbol{A}\boldsymbol{x} \geq \mathbf{1} \\ & \boldsymbol{x} \in \{0,1\}^E \end{aligned} \tag{1}$$

Let $\gamma(\boldsymbol{A}) = \{\boldsymbol{x} \in \{0,1\}^E : \boldsymbol{A}\boldsymbol{x} \geq \mathbf{1}\}$ denote the set of the feasible solutions of *SCP*. We denote by $P(\boldsymbol{A}) = conv(\gamma(\boldsymbol{A}))$ the *Set Covering polytope*. All the non-dominated valid inequalities for $P(\boldsymbol{A})$ are of the form $\boldsymbol{\alpha}^T \boldsymbol{x} \geq \beta$, with $\boldsymbol{\alpha}, \beta \geq 0$ [17].

Most of the literature on Set Covering algorithms focused on heuristics for large-scale instances [6, 4, 5].

[*] RCOST - Research Center on Software Technology, Università del Sannio, Viale Traiano, 82100 Benevento, Italy, e-mail: avella@unisannio.it

[†] RCOST - Research Center on Software Technology, Università del Sannio, Viale Traiano, 82100 Benevento, Italy, e-mail: maboccia@unisannio.it

[‡] Institute of System Dynamics and Control Theory, Siberian Branch of Russian Academy of Sciences, Lermontov Srt., 134, 664033 Irkutsk, Russia, e-mail: vasilievigor@gmail.com

Much less attention has been paid to the exact solution of difficult instances. The only recent approaches we are aware of are the Mannino and Sassano [15] enumerative algorithm for the Steiner Triples and the disjunctive cutting plane algorithm proposed by Ferris, Pataki and Schmieta [10] to solve to optimality the well-known "*seymour*" instance.

The structure of the Set Covering polytope has been deeply investigated in Balas and Ng [2, 3], Cornuejols and Sassano [7], Sassano [17], Nobili and Sassano [16], Saxena [18, 19, 20], but these relevant theoretical results have not led yet to a successful cutting plane algorithm. In our opinion this is due to the difficulty of designing efficient separation algorithms.

In this paper we report on a computational experience with a separation procedure for general (i.e. not based on the "template paradigm") cutting planes - *SepGcuts* - based on the following idea:

i) identify a suitably small subproblem defined by a subset of the constraints of the formulation;

ii) run an exact separation algorithm over the subproblem to produce a violated cutting plane, if any exists.

SepGcuts combines MIP separation of rank-1 Chvatal-Gomory cuts to find a "separating subproblem", whose investigation ensures to return a violated valid inequality, and a brute-force separation routine for "suitably small" subproblems, to produce violated facets of the Set Covering polytope $P(\boldsymbol{A})$.

The remainder of the paper is organized as follows. In Section 2 we outline *SepGcuts*. In Section 3 we describe the exact separation procedure for $P(\boldsymbol{A}_S)$. In Section 4 we report on a computational experience on difficult small-medium size instances. In the Appendix we give the implementation details of the MIP separation procedure for rank-1 Chvátal-Gomory cuts.

2 Outline of the separation procedure *SepGcuts*

SepGcuts consists of three basic steps. First we look for a submatrix $\boldsymbol{A}_S$ of $\boldsymbol{A}$ with the property that $\hat{x} \notin P(\boldsymbol{A}_S)$.

Then we run an exact separation procedure to identify facet-inducing inequalities of $P(\boldsymbol{A}_S)$ which cut-off $\hat{x}$. By "exact separation" we mean a brute-force separation algorithm which guarantees to return a hyperplane separating $\hat{x}$ and $P(\boldsymbol{A}_S)$.

In the third step, sequential lifting is used to convert the facets of $P(\boldsymbol{A}_S)$ into facets of the Set Covering polytope $P(\boldsymbol{A})$.

For the success of the procedure it is crucial to choose sets S such that $\hat{x} \notin P(\boldsymbol{A}_S)$ and S is suitably small. In this paper we test rank-1 Chvátal-Gomory cutting planes as drivers to "promising" S subsets, because they are suitably "sparse", so leading to small S subsets. The separation procedure *SepGcuts* can be summarized as follows:

Procedure *SepGcuts*

1. **Choose $\boldsymbol{A}_S$**

   1a. Let $\boldsymbol{B} \geq \boldsymbol{b}$ the current formulation of the Set Covering problem, after the addition of some valid inequalities. We recall that both $\boldsymbol{B}$ and $\boldsymbol{b}$ have nonnegative entries. Run a MIP separation routine for rank-1 Chvátal-Gomory cuts [11, 13] to generate a valid inequality $\boldsymbol{c}^T\boldsymbol{x} \geq d$ for $P(\boldsymbol{A})$ which cuts-off $\hat{x}$, i.e. $\boldsymbol{c}^T\hat{\boldsymbol{x}} < d$

1b. Let $S = \{j \in N : c_j > 0\}$ be the *support of* $\boldsymbol{c}$ and let $\boldsymbol{c}_S = [c_j : j \in S]$ be the *support vector of* $\boldsymbol{c}$.

1c. Set $x_j = 1$ for each $j \in N \setminus S$. Let $\boldsymbol{A}_S$ be the submatrix of $\boldsymbol{A}$ defined by the columns of $\boldsymbol{A}$ with index in S and by the rows of $\boldsymbol{A}$ containing only variables in S (all the inequalities in $\boldsymbol{Ax} \geq \boldsymbol{1}$ including variables in $N \setminus S$ become redundant and can be removed).

2. **Exact separation over** $P(\boldsymbol{A}_S)$

2a. Let $\boldsymbol{x}_S = [x_i : i \in S]$ be the support vector of $\boldsymbol{x}$ and let $\gamma(\boldsymbol{A}_S) = \{\boldsymbol{x}_S \in \{0,1\}^S : \boldsymbol{A}_S\boldsymbol{x}_S \geq \boldsymbol{1}\}$ be the set of the feasible solutions of the reduced Set Covering problem defined by the submatrix $\boldsymbol{A}_S$. Let $P(\boldsymbol{A}_S) = conv(\gamma(\boldsymbol{A}_S))$ be the Set Covering polytope associated with $\boldsymbol{A}_S$.

2b. The inequality $\boldsymbol{c}_S^T\boldsymbol{x}_S \geq d$ is valid for $P(\boldsymbol{A}_S)$ and cuts-off $\hat{x}_S$.

2c. Run an exact separation procedure over $P(\boldsymbol{A}_S)$ to generate an inequality of the form $\boldsymbol{\alpha}_S^T\boldsymbol{x}_S \geq \beta$, $(\boldsymbol{\alpha}_S, \beta) \geq 0$, facet-inducing for $P(\boldsymbol{A}_S)$ and violated by $\hat{\boldsymbol{x}}$.

3. **Lifting**

3a. Any valid inequality for $P(\boldsymbol{A}_S)$ is also valid for $P(\boldsymbol{A})$ [17]. But in general it is not true that the facets of $P(\boldsymbol{A}_S)$ are facet-defining for $P(\boldsymbol{A})$ too, and sequential lifting is required to convert $\boldsymbol{\alpha}_S^T\boldsymbol{x}_S \geq \beta$ into a facet of $P(\boldsymbol{A})$.

3 Exact separation of $P(\boldsymbol{A}_S)$

Let $P(\boldsymbol{A}_S)$ be the Set Covering polytope associated with the reduced matrix $\boldsymbol{A}_S$. To find an inequality $\boldsymbol{\alpha}_S^T\boldsymbol{x}_S \geq \beta$ valid for $P(\boldsymbol{A}_S)$ and violated by the current fractional solution $\hat{\boldsymbol{x}}$, we solve the following Linear Programming problem:

$$\begin{aligned} \theta^* \quad = \quad & \min \hat{\boldsymbol{x}}_S^T\boldsymbol{\alpha}_S - \beta \\ & \boldsymbol{y}^T\boldsymbol{\alpha}_S \geq \beta, \qquad y \in ext(P(\boldsymbol{A}_S)) && (2) \\ & \boldsymbol{1}^T\boldsymbol{\alpha}_S - \beta = 1 && (3) \end{aligned}$$

where $ext(P(\boldsymbol{A}_S))$ denotes the set of the extreme points of $P(\boldsymbol{A}_S)$, and the normalization constraint (3) is introduced to prevent unboundedness. With this normalization we get an inequality $P(\boldsymbol{A}_S)$ minimizing the ratio $\frac{\hat{\boldsymbol{x}}_S^T\boldsymbol{\alpha}_S-\beta}{\beta}$.

By projecting the β, it is easily seen that the problem (2)-(3) provides an extreme point optimal solution for any $\hat{\boldsymbol{x}}$. Since any extreme point is defined by $|S|$ active constraints, it follows that the $\boldsymbol{\alpha}_S^T\boldsymbol{x}_S \geq \beta$ induces a facet of $P(\boldsymbol{A}_S)$.

3.1 The row generation algorithm

the LP (2)-(3) must be solved by row generation, since it contains a huge number of rows.

Step 0. Initialize the LP as:

$$\begin{align}
\theta^* \quad = \quad & \min \hat{\boldsymbol{x}}_S^T \boldsymbol{\alpha}_S - \beta \notag\\
& \mathbf{1}^T \boldsymbol{\alpha}_S - \alpha_j \geq \beta, \quad j \in S \tag{4}\\
& \mathbf{1}^T \boldsymbol{\alpha}_S - \beta = 1 \notag
\end{align}$$

where the constraints (4) are introduced to prevent unboundedness.

Step 1. Let F be a subset of $ext(P(\boldsymbol{A}_S))$. Solve the *partial separation LP* over F:

$$\begin{align}
\theta^* \quad = \quad & \min \hat{\boldsymbol{x}}_S^T \boldsymbol{\alpha}_S - \beta \notag\\
& \mathbf{1}^T \boldsymbol{\alpha}_S - \alpha_j \geq \beta, \quad j \in S \notag\\
& \boldsymbol{y}^T \boldsymbol{\alpha}_S \geq \beta, \qquad y \in F \tag{5}\\
& \mathbf{1}^T \boldsymbol{\alpha}_S - \beta = 1 \tag{6}
\end{align}$$

Let $(\overline{\boldsymbol{\alpha}}_S, \overline{\beta})$ be the optimal solution of the partial separation LP 5.

Step 3 Solve the Set Covering problem

$$\begin{align}
\underline{\lambda} \quad = \quad & \min \quad \overline{\boldsymbol{\alpha}}_S^T \boldsymbol{w}_S \notag\\
& \boldsymbol{A}_S \boldsymbol{w}_S \geq \mathbf{1} \tag{7}\\
& \boldsymbol{w} \in \{0,1\}^S \notag
\end{align}$$

looking for a solution $\underline{w} \in ext(P(\boldsymbol{A}_S))$ violating the inequality $\overline{\boldsymbol{\alpha}}^T \boldsymbol{w} \geq \overline{\beta}$, i.e. $\overline{\boldsymbol{\alpha}}^T \underline{\boldsymbol{w}} < \overline{\beta}$. Let $\underline{\boldsymbol{w}}$ and $\underline{\lambda}$ be the optimal solution of the Set Covering problem (7) and its value, respectively.

Step 4 If $\underline{\lambda} < \overline{\beta}$ then $F = F \cup \{\underline{\boldsymbol{w}}\}$ and goto **Step 1**.

Step 5 $(\overline{\boldsymbol{\alpha}}, \overline{\beta})$ is the optimal solution of the separation LP (2)-(3) and the inequality $\overline{\boldsymbol{\alpha}}^T \boldsymbol{x} \geq \overline{\beta}$ is valid for $P(\boldsymbol{A}_S)$.

3.2 Lifting

In the exact separation procedure, the number of row generation iterations grows exponentially with the size of the variables involved, so it is crucial to run the separation routine over the subset defined the fractional variables. After a valid inequality on the fractional space has been found, we use two different lifting procedures to make it valid and facet-defining for $P(\boldsymbol{A})$. One - named *local lifting* - is used to make cutting planes generated over the fractional subspace facet-defining for $P(\boldsymbol{A}_S)$. The second - termed as *global lifting* - is used to convert the facets of $P(\boldsymbol{A}_S)$ into facets of $P(\boldsymbol{A})$.

3.2.1 Local lifting

Let $\overline{x}$ be the optimal solution of the current LP relaxation and let $H_S = \{j \in S : 0 < \overline{x}_j < 1\}$ be the index subset of the fractional variables in $\overline{x}$.

The exact separation procedure runs over the set H. After a valid inequality $\boldsymbol{\alpha}_H^T \overline{\boldsymbol{x}}_H \geq \beta$ has been found for the restricted problem, we use standard sequential lifting to make it valid for $P(\boldsymbol{A}_S)$. Computing the lifting coefficients involves solving a Set Covering problem for each variable to be lifted.

It is well-known that the resulting inequalities depend on the order in which the variables are lifted, i.e. on the *lifting sequence*. In our experiments, down lifting is performed first. To define the lifting sequence we consider the reduced costs in the current LP-relaxation. Variables with smaller reduced costs are lifted first.

3.2.2 Global lifting

We term "global lifting" the sequential lifting procedure which converts a facet $\boldsymbol{\alpha}_S^T \boldsymbol{x}_S \geq \beta$ of $P(\boldsymbol{A}_S)$ into a facet $\boldsymbol{\alpha}^T \boldsymbol{x} \geq \beta$ of the Set Covering polytope $P(\boldsymbol{A})$.

It easy to prove that if $\alpha_S^T \overline{x}_S \geq \beta$ defines a facet of $P(\boldsymbol{A}_S)$, then it also defines a facet of $P(\boldsymbol{A})$ when all the variables in $N \setminus S$ are set to 1, so sequential down-lifting of the variables in $N \setminus S$ is required to make it facet-defining for $P(\boldsymbol{A})$.

Actually, only a subset of the variables in $N \setminus S$ needs to be lifted. Let $\boldsymbol{A}_{\overline{S}}$ be the submatrix of $\boldsymbol{A}$ defined by the rows of $\boldsymbol{A}$ containing at least one variable of S. It can be easily proved that the lifting coefficient is 0 for all the variables in $N \setminus \overline{S}$, so sequential lifting is required only for the variables in the set $\overline{S} \setminus S$.

3.3 Dealing with numerical troubles

It is well-known that rounding errors affect the solution of linear systems and hence of Linear Programming solvers. In a cutting plane procedure these errors in the computation of the coefficients can lead to *invalid* inequalities cutting-off a feasible solution. For further details on this crucial topic in Computational Mixed Integer Programming we refer the reader to [1, 9, 12].

In the exact separation procedure, rounding errors can occur in the solution of the partial separation LP and of the column generation problem. Such errors can impact on producing weak or even invalid cuts.

To reduce the possibility of generating invalid cutting planes, the obtained inequalities are post-processed to get equivalent cuts with integer coefficients to verify their validity. Let $(\boldsymbol{\alpha}^*, \beta^*)$ be the optimal solution of the partial separation LP, we post-process the inequality $\boldsymbol{\alpha}^{*T} x \geq \beta^*$ to get an equivalent inequality with integer coefficients. The following Integer Linear Programming problem:

$$\begin{aligned}
&min \quad t \\
&\boldsymbol{\gamma} = t\boldsymbol{\alpha}^* \\
&\gamma_0 = t\beta^* \\
&\boldsymbol{\gamma} \in \mathbb{Z}^E \\
&\gamma_0 \in \mathbb{Z} \\
&t \geq 1
\end{aligned}$$

returns an "integer" inequality $\boldsymbol{\gamma}^T \boldsymbol{x} \geq \gamma_0$, which is equivalent to the original cut.

Name	LB_{LP}	LB_D	$\%Gap_D$	$LB_{G/D}$	$\%Gap_{G/D}$	BestUB
SCPNRG1	159,89	160,36	3,11	161,75	12,31	176
SCPNRG2	142,04	142,95	7,61	143,2	9,7	155
SCPNRG3	148,27	149,27	5,98	150,13	11,12	166
SCPNRG4	148,94	149,9	5,31	151,08	11,85	168
SCPNRG5	148,23	149,25	5,43	150,05	9,7	168
SCPNRH1	48,12	48,5	2,55	48,72	4,03	64
SCPNRH2	48,63	49,11	3,34	49,25	4,31	64
SCPNRH3	45,19	45,71	4,05	45,96	6,01	59
SCPNRH4	44,04	44,65	4,04	44,8	5,08	59
SCPNRH5	42,37	42,84	4,04	43,23	7,39	55

Table 1: Computational results for NRG and NRH instances

A new Set Covering problem is then solved check the validity of the resulting inequality. We observe that the integrality of the lifting coefficients γ determines that lifting coefficients will be integer too.

4 Computational results

The algorithm was tested on two different groups of instances. The first group consists of the *SCPNRG* and of the *SCPNRH* instances of the OR-Library. Actually the second group is a singleton, as it consists of the well-known "seymour" instance. We also tested the *railxxx* OR-Library instances, which after preprocessing were easily solved to optimality by Cplex 10.1 with no need for cutting planes.

All the experiments ran on an Intel Xeon CPU 3.06GHz workstation with 1 Gb RAM.

4.1 *SCPNRG* and *SCPNRH* instances

The *SCPNRG* and *SCPNRH* instances are considered to be the hardest randomly generated instances in the OR-Library [6]. They have all 10000 columns and 1000 rows and are refractary to preprocessing. Here we tested the improvement on the lower yielded by disjunctive: first we test disjunctive cuts on the original formulation. Then we test disjunctive cuts after using the separation procedure of section 2.

Computational results are reported in Table 4.1, where columns LB_D and $\%GAP_D$ report on the lower bound and on percentage of the integrality gap closed by generating disjunctive cuts from the original formulation, respectively. Columns $LB_{G/D}$ and $\%Gap_{G/D}$ report on the lower bound and on the percentage of the integrality gap closed by generating disjunctive cuts after *SepGcuts*. Column $BestUB$ reports on the best know upper bound for each instance.

We can observe the the percentage of the gap closed by running *SepGcuts* first and then generating disjunctive cuts is on the average more than 80% larger than the gap closed by generating disjunctive cuts from the original formulation. However it is still too small (in particular for the subset of the NRH instances) to solve such problems to optimality within reasonable amounts of time.

4.2 *seymour*

seymour is a well-known difficult Set-Covering instance, posed by Paul Seymour as a by-product of a new proof of the Four Color Theorem (FCT) by Neil Robertson, Daniel Sanders, Paul Seymour, and

Robin Thomas. It is one of the most difficult problems in the MIPLIB. The value of the LP-relaxation is 403.84 and the optimal solution is 423.

In 2000, Ceria, Ferris and Pataki [10] obtained a remarkable result, proving 423 as the optimal solution of *seymour*. The problem was preprocessed at the root node by deleting all dominated rows and columns as usual in Set Covering problems. The reduced problem has 4323 rows, and 882 columns. The value of 423 in the original problem corresponds to the value of 238 in the preprocessed problem.

The approach described in [10] consisted of two steps. First disjunctive cuts are generated at the search tree nodes of level ≤ 8. At this point, i.e. after 256 nodes, the best lower bound increased by 15.17, closing the 79% of the gap.

In a second step, they submitted each of the 256 subproblems as a separate task, to be solved by a MIP solver (they used Cplex 6.6 and Xpress 11.25). The overall enumeration took more than 400 days of CPU time.

The instance we tested was preprocessed by P. Nobili and C. Mannino [14]. The reduced instance has 745 columns, 4053 rows and 27950 nonzeroes, with an offset of 236, so the value of its optimal solution is 187 instead of 423 [1].

The value of the LP-relaxation is 169.61. First we ran a brute-force separation of Chvátal-Gomory cuts over all the triples of inequalities of the initial formulation. After this "treatment", the lower bound raised to 180.01, closing the 63.4% of the gap. This phase took around 3600 secs of CPU time.

Then we switched to *SepGCuts*, which took about 36 hours of CPU time to raise the lower bound to 182.09, closing the 76.2% of the gap.

After the cutting phase, we passed the final formulation to Cplex 10.1, which took less than 17 hours to prove the optimality of solution of value 187.

Actually, it would be completely unfair to compare these results with the experience reported in [10]. We intend our experience as a clear showing of the general progress in the ability to solve difficult problems.

References

[1] D. Applegate, W. Cook, S. Dash, and D. Espinoza, Exact solutions to linear programming problems, Operations Research Letters (2007)

[2] E. Balas, S.M. Ng, On the Set Covering Polytope: I. All the Facets with Coefficients in 0,1,2, Mathematical Programming Ser A, 43, 1989, 57-69.

[3] E. Balas, S.M. Ng, On the Set Covering Polytope: II. Lifting the Facets with Coefficients in 0,1,2. Mathematical Programming Ser A, 45, 1989, 1-20.

[4] A. Caprara, M. Fischetti, P. Toth, A Heuristic Method for the Set Covering Problem, Operations Research 47 (1999) 730-743.

[5] A. Caprara, M. Fischetti, P. Toth, Algorithms for the Set Covering Problem, Annals of Operations Research 98 (2000) 353-371.

[6] S. Ceria, P. Nobili, A. Sassano, A Lagrangian-based Heuristic for Large-scale Set Covering Problems, Mathematical Programming 81 (1998) 215-228.

[1] The preprocessed instance can be downloaded at www.ing.unisannio.it/boccia

[7] G. Cornuejols, A. Sassano, On the 0,1 facets of the set-covering problem. Mathematical Programming Ser A, 43 (1989) 45-55.

[8] CPLEX Optimization Inc., Using the CPLEX callable library, Reference Manual, 20055.

[9] D. G. Espinoza, On Linear Programming, Integer Programming and Cutting Planes, Ph.D. Thesis, Georgia Institute of Technology, School of Industrial and Systems Engineering, 2006.

[10] M.C. Ferris, G. Pataki and S. Schmieta, Solving the seymour problem, Optima, 66:1-7, 2001.

[11] M. Fischetti, A. Lodi, Optimizing over the first Chvátal closure, Mathematical Programming Ser. B, 110(1), 3-20, 2007.

[12] M.G. Goycoolea, Cutting Planes for Large Mixed Integer Programming Models, Ph.D. Thesis, Georgia Institute of Technology, School of Industrial and Systems Engineering, 2006.

[13] A.M.C.A. Koster, A. Zymolka, M.Kutschka, Algorithms to Separate 0,1/2-Chvátal-Gomory Cuts, ZIB-report 07-10.

[14] C. Mannino, P. Nobili. Solving hard Set Covering instances I: preprocessing, Technical Report DIS

[15] C. Mannino, A. Sassano, "Solving Hard Set-Covering Problems". Operations Research Letters, Vol. 18 (1995) 1-5.

[16] P. Nobili, A. Sassano, Facets and Lifting Procedures for the Set-Covering Polytope. Mathematical Programming Ser B, 45 (1989) 111-137.

[17] A. Sassano, On the facial structure of the set-covering problem. Mathematical Programming Ser A, 44 (1989) 181-202.

[18] A. Saxena, On the Set-Covering Polytope: I. All the facets with coefficients in 0,1,2,3 , GSIA Working Paper 2004-E37, 2004.

[19] A. Saxena, On the Set-Covering Polytope: II. Lifting facets with coefficients in 0,1,2,3 , GSIA Working Paper 2004-E30, 2004.

[20] A. Saxena, On the Set-Covering Polytope III: Complete characterization of 3-critical matrices , GSIA Working Paper 2004-E31, 2004.

5 Appendix: MIP separation of rank-1 Chvátal-Gomory cuts: implementation details

Let

$$\begin{array}{ll} \min & \boldsymbol{c}^T \boldsymbol{x} \\ & \boldsymbol{B}\boldsymbol{x} \geq \boldsymbol{b} \\ & \boldsymbol{x} \in \{0,1\}^E \end{array} \tag{8}$$

be the current formulation of the Set Covering problem, after the addition of cutting planes.

The separation problem for rank-1 Chvatal-Gomory cuts is formulated as a Mixed-Integer Programming (MIP) problem [11]. Let $\boldsymbol{u}$ be the Chvatal-Gomory multipliers associated with the rows of $\boldsymbol{B}$ (based on our experiments, we did not consider the lower and upper bounds associated with each variables). Let $\boldsymbol{c}$ and d be the l.h.s. and the r.h.s. coefficient of the $\boldsymbol{c}^T\boldsymbol{x} \geq d$ inequality, respectively. The MIP formulation is:

$$\begin{aligned} &\min \hat{\boldsymbol{x}}^T\boldsymbol{c} - d \\ &\boldsymbol{c} \geq \boldsymbol{B}^T\boldsymbol{u} \\ &d \leq \boldsymbol{b}^T\boldsymbol{u} + (1-\varepsilon) \\ &\boldsymbol{c} \in \mathbb{Z}_+^E \\ &d \in \mathbb{Z}_+ \\ &0 \leq \boldsymbol{u} \leq 1-\varepsilon \end{aligned} \tag{9}$$

5.1 Looking for stronger cutting planes

Finding a quality measure to evaluate the effectiveness of a cut is still an open problem in cutting plane algorithms. Some authors observe that looking for cutting planes which maximize the ratio between violation and the size of the l.h.s. (i.e. a "normalized" violation) leads to more effective cutting planes than searching cutting planes with maximum violation.

For Set Covering problems, since the l.h.s. and r.h.s. coefficients are nonnegative, we can fix the r.h.s. of the cut to $\overline{d}$. Then minimizing $\hat{\boldsymbol{x}}^T\boldsymbol{c}$ corresponds to minimize the ratio $\frac{\hat{\boldsymbol{x}}^T\boldsymbol{c}}{\overline{d}}$:

$$\begin{aligned} z^* = \quad &\min \hat{\boldsymbol{x}}^T\boldsymbol{c} \\ &\boldsymbol{c} \geq \boldsymbol{B}^T\boldsymbol{u} \\ &\boldsymbol{b}^T\boldsymbol{u} \geq \overline{d} - (1-\varepsilon) \\ &\boldsymbol{c} \in \mathbb{Z}_+^E \\ &0 \leq \boldsymbol{u} \leq 1-\varepsilon \end{aligned} \tag{10}$$

The problem (10) is solved by a MIP solver, with a limit on the size of the search tree. A violated cut is found if $\hat{\boldsymbol{x}}^T\boldsymbol{c} < \overline{d}$. Initially we set $\overline{d} = 2$ and increase $\overline{d}$ if no violated cuts have been found within the time limit. An upper bound on $\overline{d}$ is imposed as an exit condition.

5.2 Preprocessing

To reduce computation times, the MIP problem (10) runs over the subproblem defined by the fractional variables, since any variable x_j with $\hat{x}_j = 0$ gives no contribution to the cut violation. Given the optimal set $\boldsymbol{u}$ of multipliers, the corresponding coefficient can be easily computed as $c_j = \boldsymbol{u}^T\boldsymbol{A}_j$ and the resulting inequality can be promptly made globally valid by using the "optimal" multipliers. In order to keep the size of the problem (10) reasonably small, we consider only the inequalities in the current formulation whose slack is in absolute value smaller than a given threshold δ. In our experiments we set $\delta = 0.2$.

5.3 Getting more cuts from the same MIP

Since the solution of the problem (10) is generally time-consuming, it is convenient to retrieve not only the most violated cut, but also the other “sub-optimal” violated cutting planes discovered during the search tree.

To this purpose, after a violated cut $\overline{\boldsymbol{c}}^T \boldsymbol{x} \geq \overline{d}$ is found during the search tree, the following operations are performed:

i) the violated cut $\overline{\boldsymbol{c}}^T \boldsymbol{x} \geq \overline{d}$ is stored into a cut pool;

ii) the MIP solver does not update the incumbent;

iii) The inequality

$$\overline{\boldsymbol{c}}^T \boldsymbol{c} \leq 0.99 \sum_{j \in N} \overline{c}_j \sum_{j \in N} c_j \tag{11}$$

is added to the problem (10) in order to cut-off $(\overline{\boldsymbol{c}}, \overline{d})$ from the set of the feasible solutions.