Improving Cutting Plane Generation with 0-1 Inequalities by Bi-criteria Separation

E. Amaldi, S. Coniglio, and S. Gualandi

Politecnico di Milano, Dipartimento di Elettronica e Informazione, Italy
{amaldi,coniglio,gualandi}@elet.polimi.it

Abstract. In cutting plane-based methods, the question of how to generate the "best possible" cuts is a central and critical issue. We propose a bi-criteria separation problem for generating valid inequalities that simultaneously maximizes the cut violation and a measure of the diversity between the new cut and the previously generated cut(s). We focus on problems with cuts having 0-1 coefficients, and use the 1-norm as diversity measure. In this context, the bi-criteria separation amounts to solving the standard single-criterion separation problem (maximizing violation) with different coefficients in the objective function. We assess the impact of this general approach on two challenging combinatorial optimization problems, namely the Min Steiner Tree problem and the Max Clique problem. Computational experiments show that the cuts generated by the bi-criteria separation are much stronger than those obtained by just maximizing the cut violation, and allow to close a larger fraction of the gap in a smaller amount of time.

1 Introduction

Cutting planes are a central component of modern methods for solving Integer or Mixed-Integer Linear Programs (IPs or MILPs). In theory, a pure cutting plane method, where the separation is carried out w.r.t. an appropriate family of valid inequalities (e.g., Gomory cuts), converges in a finite number of iterations. However, convergence is difficult to achieve, often due to numerical issues. The relaxation of the problem, which is iteratively enriched with newly found violated inequalities and re-optimized, often becomes numerically ill-conditioned [BFZ08]. Cuts which are not valid can also be found [Mar09]. In practice, cutting planes are typically used within a Branch-and-Cut framework.

A recurrent and important issue in cutting plane-based methods is the generation, at each iteration, of the "best possible" cut to be added to the current relaxation. The fundamental underlying question is to define a quantitative measure that favors (possibly strong) valid inequalities that are "better" than others [PR91]. The usual criterion for cutting plane

generation is the maximization of the cut violation (or depth), i.e., the amount by which the cut is violated by the optimal solution of the current relaxation. The advantage of this criterion is that it leads to optimizing a linear objective function. Alternative criteria have been proposed, e.g., the maximization of the Euclidean distance from the optimal solution of the relaxation to the cut.

In several computational studies [BCC93,ACF07], a large number of cuts is first generated, and then a subset is selected to be added to the current relaxation. Usually, cuts are first ranked with respect to the Euclidean distance between the optimal solution and the cut and then with respect to their pairwise angles. As pointed out in [BCC96], the purpose is to select, on the one hand, valid inequalities that cut away as much as possible the feasible region of the relaxation and, on the other hand, cuts that are as orthogonal as possible, typically to avoid similar cuts to be added. In the same paper, it is observed that, for Lift-and-Project cuts, when inequalities that cut the polyhedron of the relaxation from different angles are introduced, new tighter cuts are likely to be found in the next cutting plane iterations. In [FL06], when optimizing over the rank 1 Chvátal-Gomory closure, the presence of multiple maximally violated cuts is exploited by solving a modified separation problem, where an additional penalty term favors the generation of undominated cuts. The resulting solution, though, is no longer optimal w.r.t. the violation of the cut. In the same work, it is also observed that cuts which are as *diverse* as possible turn out to be computationally beneficial. These ides are systematically applied, for several families of cuts, in the solver SCIP [Ach07].

In this work, we propose a bi-criteria separation problem for generating valid inequalities with 0-1 coefficients. This problem amounts to finding, among all maximally violated cuts, one that also maximizes a diversity measure between this cut and the previously generated one(s). As diversity measure, we use the 1-norm of the difference between two successive cuts. This clearly differs from what has been previously done in the literature, where a (fast) generation of a large number of valid inequalities is followed by a cut selection phase. We show that, whenever the cuts have 0-1 coefficients, the bi-criteria separation problem amounts to solving the standard single-criterion one (where the violation is maximized) with different coefficients in the objective function. We assess the impact of this general approach on two challenging combinatorial optimization problems, namely the Min Steiner Tree problem and the Max Clique problem.

2 Preliminaries

Consider the general Mixed-Integer Linear Program

$$\begin{aligned}(P) \quad & \min cx : Ax \leq b \\ & s.t.\ x_i \in \mathbb{R},\ \text{for } i = 1, \ldots, p \\ & \quad x_i \in \mathbb{Z},\ \text{for } i = p+1, \ldots, n \\ & \quad \alpha x \leq \alpha_0,\ \text{for } (\alpha, \alpha_0) \in \mathcal{C},\end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and $\mathcal{C}$ is the set of valid inequalities w.r.t. we want to separate. We suppose that $\mathcal{C}$ is known only implicitly, i.e., as the set of feasible solutions to some combinatorial problem, possibly described as another Mixed-Integer Linear Program.

In this work, we assume that $\mathcal{C}$ only contains inequalities with 0-1 coefficients, i.e. $\alpha \in \{0,1\}^n$. This encompasses a number of families of inequalities that are valid for many combinatoral optimization problems, e.g., cut, clique, stable set, rank, cycle, and cover inequalities [MMWW02]. For the sake of simplicity, here we assume $\alpha_0 = 1$, but the proposed approach is valid for any α_0.

Let t be the index of the cutting plane generation iteration. Let

$$(P^t) \qquad \min\{cx : Ax \leq b, x \in \mathbb{R}^n, \alpha x \leq 1 \text{ for } (\alpha, 1) \in \mathcal{C}^t\}$$

be the current relaxation of *(P)*, where only a subset $\mathcal{C}^t \subseteq \mathcal{C}$ of the inequalities is considered, and the integrality constraints are neglected. Let x^* be the corresponding optimal solution. The standard single-criterion separation problem, where a maximally violated cut is found, is

$$(SEP) \qquad \max\{\alpha x^* - 1 : (\alpha, 1) \in \mathcal{C}\}.$$

3 Bi-criteria Separation Problems

Let t be the iteration index of the current iteration, and let $\alpha^t x \leq 1$ be the cut generated at the previous iteration. Motivated by the considerations in Sec. 1, we would like to generate a cut $\alpha x \leq 1$ that not only maximizes the violation w.r.t. x^* but that also, among all maximally violated cuts, is as diverse as possible from $\alpha^t x \leq 1$.

As diversity measure, we propose the 1-norm between successive cuts, namely the function $||\alpha - \alpha^t||_1$, where $||v||_1 = \sum_{i=1}^n |v_i|$ for any $v \in \mathbb{R}^n$.

Since all α_i and α_i^t take 0-1 values, we clearly have

$$\sum_{i=1}^{n} \mid \alpha_i - \alpha_i^t \mid = \sum_{i=1}^{n} \alpha_i(1-\alpha_i^t) + \sum_{i=1}^{n}(1-\alpha_i)\alpha_i^t = \sum_{i=1}^{n} \alpha_i - 2\sum_{i=1}^{n} \alpha_i\alpha_i^t + \sum_{i=1}^{n} \alpha_i^t. \tag{1}$$

By letting e denote the all ones vector, and neglecting the last term, which is constant, this diversity measure can be expressed as $e\alpha - 2\alpha\alpha^t$. This is of practical importance, since we aim at a separation problem which is, computationally, not too hard to solve. From an empirical point of view, $||\alpha - \alpha^t||_1$ also nicely captures the differences between the angles of 0-1 vectors.

The cut violation $\alpha x^* - 1$ and the diversity measure $||\alpha - \alpha^t||_1$ can be combined into the following bi-criteria separation problem, where they are optimized with priority:

$$(\textit{BC-SEP}) \qquad \max\left\{\alpha x^* - 1 + \epsilon(e\alpha - 2\alpha\alpha^t) : (\alpha, 1) \in \mathcal{C}\right\}.$$

A small enough value of the proportionality parameter $\epsilon > 0$ can always be found so as to guarantee that the priority between $\alpha x^* - 1$ and $e\alpha - 2\alpha\alpha^t$ is respected. It can be derived as follows. Let $M := n$ be the maximum of $e\alpha - 2\alpha\alpha^t$, and $m := e\alpha^t$ its minimum. Let also $\delta := \min_{i=1,\ldots,n}\{x_i^*\}$ be the smallest variation in $\alpha x^* - 1$, corresponding to a variation in the variables α. Then, it suffices to select ϵ such that $\epsilon(M - m) < \delta$.

We also propose the variant of *(BC-SEP)* in which the diversity w.r.t. the whole set of previously generated cuts is taken into account. This is achieved by defining the multi-cut bi-criteria separation problem as

$$(\textit{MC-BC-SEP}) \qquad \max\left\{\alpha x^* - 1 + \epsilon(e\alpha - 2\alpha\bar{\alpha}^t) : (\alpha, 1) \in \mathcal{C}\right\},$$

where $\bar{\alpha}^t := \frac{1}{t}\sum_{l=1}^{t} \alpha^l$ is the average of all the previously generated cuts. Since (1) also holds when $\bar{\alpha}^t \in [0,1]^n$, the objective function has the same structure of that of *(BC-SEP)*.

4 Solving the Bi-criteria Separation problems

Since the objective functions of *(BC-SEP)* and *(MC-BC-SEP)* are linear functions of the variable α, any of the two proposed separation problems is obtained by just taking *(SEP)*, and changing its objective function vector x^* into a new vector $\hat{x}^*$. For *(BC-SEP)*, $\hat{x}_i^* := x_i^* + \epsilon(2\alpha_i^t - 1)$, for all $i = 1, \ldots, n$. For *(MC-BC-SEP)*, $\hat{x}_i^* := x_i^* + \epsilon(2\bar{\alpha}_i^t - 1)$, for all $i = 1, \ldots, n$. In this sense, the bi-criteria separation problem just amounts to solving

the standard *(SEP)* to separate $\hat{x}^*$ instead of x^*. Note, however, that even if $x^* \geq 0$, $\hat{x}^*$ can be negative.

The bi-criteria separation problem is here adapted to two challenging combinatorial problems: Min Steiner Tree and Max Clique. For both of them, we consider a single family $\mathcal{C}$ of facet defining valid inequalities.

Min Steiner Tree. Given a graph $G = (V, E)$, a set $T \subset V$ of terminals and a cost function $c : E \to \mathbb{R}^+$, find a Steiner tree of G, i.e., a tree that spans all the nodes in T, of minimum total cost. Let $G' = (V, A)$ be the directed version of G, with a pair of arcs $(i, j), (j, i)$ for each edge $\{i, j\} \in E$. Let $r \in T$ be an arbitrary root node. As in [KM98], we consider the following directed formulation of the problem, where the integrality constraints are relaxed:

$$\min \quad \sum_{(ij)\in A} c_{ij} x_{ij} \tag{2}$$

$$\text{s.t.} \quad \sum_{(ij)\in\nu^+(S)} x_{ij} \geq 1, \ \text{ for } S \subset V : r \in S, V \setminus S \cap T \neq \emptyset, \tag{3}$$

$$0 \leq x_{ij} \leq 1, \qquad \text{for } (i, j) \in A. \tag{4}$$

In this case, we take as $\mathcal{C}$ the set of all cut inequalities (3). The separation problem amounts to finding, for each terminal $t \in T \setminus \{r\}$, a Min $s - t$ Cut [1] on G' with $s = r$, where the values of the solution x^*_{ij} of the current relaxation are used as arc capacities. This separation problem can be formulated as the following Linear Program

$$\min \quad \sum_{(ij)\in A} x^*_{ij} \alpha_{ij} \tag{5}$$

$$\text{s.t.} \ \ \alpha_{ij} \geq \pi_i - \pi_j, \ \text{ for } (ij) \in A, \tag{6}$$

$$\pi_r - \pi_\tau \geq 1, \tag{7}$$

$$\alpha_{ij} \geq 0, \qquad \text{for } (i, j) \in A, \tag{8}$$

where α is the incidence vector of an $s-t$ cut and π is the vector of node potentials. Since the cut inequalities can be separated in polynomial time, due to the equivalence between separation and optimization [NW88], (2)–(4) can be also solved in polynomial time.

[1] To comply with the standard notation, in this subsection, the index t denotes a terminal node in T, instead of the index of the cutting plane algorithm iteration.

Max Clique. Given an undirected graph $G = (V, E)$, find a maximum clique of G, i.e., a largest set of nodes that are pairwise connected. We consider the following formulation of the problem:

$$\max \quad \sum_{i \in V} x_i \tag{9}$$

$$\text{s.t.} \sum_{i \in S} x_i \leq 1, \quad \text{for } S \in \mathcal{S}, \tag{10}$$

$$0 \leq x_{ij} \leq 1, \ \text{for } \{i,j\} \in E, \tag{11}$$

where the integrality constraints are relaxed. In this case, we take as $\mathcal{C}$ the set $\mathcal{S}$ of all maximal Stable Sets of G. The separation problem, which amounts to finding a maximum weighted stable set in G, where the weights are given by the components of x^*, and is formulated as the following 0-1 Integer Program

$$\max \left\{ \alpha x^* - 1 : \alpha_i + \alpha_j \leq 1, \text{ for } \{i,j\} \in E, \alpha \in \{0,1\}^n \right\},$$

where $\alpha \in \{0,1\}^n$ is the incidence vector of a stable set. Note that, since this separation problem is $\mathcal{NP}$-hard again because of the equivalence between separation and optimization, solving (9)-(11) to optimality is also $\mathcal{NP}$-hard.

For the separation of cut inequalities, the following result holds. Remember that, because of the Linear Programming duality, an $s-t$ cut of minimum total capacity can be found by solving the corresponding Max Flow problem.

Proposition 1. *Solving* (BC-SEP) *or* (MC-BC-SEP) *for cut inequalities amounts to solving a Max Flow problem where* $\alpha_{ij} = 0$ *for all* $(ij) : \hat{x}^*_{ij} < 0$.

Proof. Consider the Linear Programming dual of (5)–(8), with the additional upper bounds $\mu_{ij} \leq 1$ for $(ij) \in A$. *(BC-SEP)* or *(MC-BC-SEP)* amounts to

$$\max \ \phi - \sum_{(ij) \in A} h_{ij} \tag{12}$$

$$\text{s.t.} \sum_{ij} x_{ij} - \sum_{ji} x_{ij} = \begin{cases} \phi & \text{if } i = r \\ -\phi & \text{if } i = t \\ 0 & \text{else} \end{cases} \tag{13}$$

$$0 \leq x_{ij} \leq \hat{x}^*_{ij} + h_{ij}, \qquad \text{for } (i,j) \in A, \tag{14}$$

$$h_{ij} \geq 0, \qquad \text{for } (i,j) \in A, \tag{15}$$

which is a Max Flow problem with the extra set of variables h_{ij}, where ϕ is the value of the flow. If $\hat{x}^*_{ij} \geq 0$ for all $(i,j) \in A$, by letting any $h_{ij} = q$, the resulting capacity on the arc (i,j) is increased by q, and q units of cost are paid in the objective function, while ϕ is increased by q only if (i,j) belongs to the unique optimal cut of the network. If it is the case, the objective function value is unaltered and hence this solution is equivalent to the corresponding one in which $h_{ij} = 0$. Otherwise, if the cut containing (i,j) is not unique or not optimal, in any optimal solution, h_{ij} will be 0 for all $(i,j) \in A$. It follows that an optimal solution can always be achieved by letting $h_{ij} = 0$ for all $(i,j) \in A$. Whenever $\hat{x}^*_{ij} < 0$ for any (i,j), $h_{ij} \geq -\hat{x}^*_{ij}$ is implied, and (12)–(15) can be reformulated by adding the term $-\sum_{(i,j)\in A:\hat{x}^*_{ij}<0} \hat{x}^*_{ij}$ to (12) and substituting 0 for every $\hat{x}^*_{ij} < 0$, thus obtaining a problem with capacities $\hat{x}^*_{ij} \geq 0$, where the h_{ij} variables can be set to 0. □

For the separation of stable set inequalities, a similar results holds.

Fact 1 *Solving* (BC-SEP) *or* (MC-BC-SEP) *for stable set inequalities amounts to solving a Max Weighted Stable Set problem where* $\alpha_i = 0$ *for all* $i \in V : \hat{x}^*_i < 0$.

5 Computational Results

In this section, we assess the impact of the proposed bi-criteria separation problems *(BC-SEP)* and *(MC-BC-SEP)*, in the context of a pure cutting plane algorithm, when compared to *(SEP)*. The experiments are carried out for the Min Steiner Tree and Max Clique problems.

The algorithms are implemented in C++, using the gnu-g++-4.3 compiler. For both problems, the relaxations are solved with CPLEX 11 (with default parameters). The Boost Graph Library is used for the graph algorithms and data structures. The experiments are carried out on a standard desktop computer with an Intel Core2Duo processor and 2.0 GB of RAM.

5.1 Min Steiner Tree

We compare *(SEP)* and *(BC-SEP)* for the Min Steiner Tree problem on two set of instances taken from the SteinLib [KMV00]: I640 and PUC. Those sets are among the hardest in the library. We consider the following setting. At each iteration, we generate a round of cuts containing as many violated inequalities as they are found, by solving a Max Flow problem for each pair (r,t) with $t \in T \setminus \{r\}$. When solving *(BC-SEP)* for terminal t,

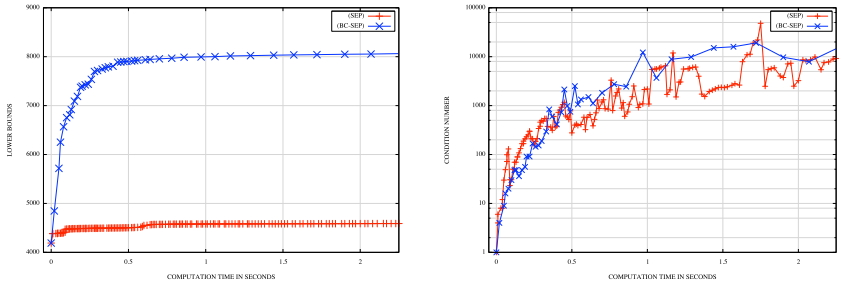
we consider, as previous cut, the last cut generated when solving *(BC-SEP)* w.r.t. the same terminal. The separation problems are solved using the Edmonds-Karp's algorithm that has complexity $O(|V|\,|E|^2)$. The root node $r \in T$ is chosen as the terminal with the largest node degree, i.e. the number of neighbors in G. Experimentally, we observed that this choice allows to close a much larger fraction of the gap when using both *(SEP)* and *(BC-SEP)*. For each instance, we derive an initial pool $\mathcal{C}^0$ of cuts by solving, for each pair of source $s = r$ and sink $t \in T \setminus \{r\}$, a Min $s - t$ Cut problem with unit capacity on every arc.

For all the experiments, illustrated in Tab. 1 and Tab. 2, the time limit was set to 1000 seconds. We report the fraction of gap closed (Gap), the CPU time in seconds (Time), the number of rounds of cuts generated (Rnds), and the exact condition number of the scaled basis matrix (see `ExactKappa` in CPLEX User Guide), averaged of the over the last 20 iterations (Avgκ). The gap is evaluated w.r.t. the value of an optimal solution, if known, or w.r.t. the best known upper bound. The condition number is averaged to mitigate the oscillations that we observed along the runs. The best values for *(SEP)* and *(BC-SEP)* are reported in bold. For the columns Time, Cuts, and Avgκ of *(BC-SEP)*, we computed the ratios, over all instances, of the corresponding value and that obtained with *(SEP)*. The inverse of their geometric mean is reported in the last line (Aggregate). In the same line, for the Gap column, we report the arithmetic mean for both *(SEP)* and *(BC-SEP)*.

Table 1 reports the results for the I640 instances. With *(BC-SEP)*, we managed to solve to optimality 7 instances out of 13. None is solved, within the time limit, with *(SEP)*. On average, with *(BC-SEP)* 94% of the gap was closed, as opposed to the 64% closed with *(SEP)*, generating only 20% of the cuts and in just 4% of the CPU time. The average condition number is also 20% smaller. Figure 1.a and 1.b show the relation between the improvement of the bounds and the increase in the condition numbers for *(SEP)* and *(BC-SEP)*, on two I640 instances. Note that, although the growth rates of the condition number are similar, the improvement in the bound for *(BC-SEP)*, with respect to that for *(SEP)*, is substantial. Table 2 shows the computational results obtained for the PUC instances. With *(BC-SEP)*, 11 instances are solved to optimality out of 18, while with *(SEP)* only 3 are solved. On average, with *(BC-SEP)* 95% of the gap is closed, as opposed the 32.8% closed with *(SEP)*, in 10% of the time, and also generating 20% of the rounds of cuts.

Table 1. *(SEP)* vs. *(BC-SEP)*, on the I640 SteinLib instances.

				(SEP)				*(BC-SEP)*			
Instance	$\|V\|$	$\|E\|$	$\|T\|$	Gap	Time	Rnds	Avgκ	Gap	Time	Rnds	Avgκ
i640-001	640	1920	9	47.4	1000.0	1024	2.9E+5	**100.0**	**0.5**	**75**	**5.1E+3**
i640-011	640	8270	9	81.4	1000.0	1005	2.9E+6	**100.0**	**5.7**	**146**	**1.2E+4**
i640-031	640	2560	9	47.9	1000.0	943	6.9E+5	**100.0**	**0.9**	**79**	**8.5E+3**
i640-041	640	40896	9	**78.7**	1000.0	2228	9.0E+0	**78.7**	1000.0	**1985**	**6.0E+0**
i640-101	640	1920	25	48.4	1000.0	779	3.9E+5	**100.0**	**4.0**	**57**	**1.1E+5**
i640-131	640	2560	25	58.6	1000.0	729	3.9E+5	**100.0**	**5.7**	**55**	**1.1E+5**
i640-141	640	40896	25	91.1	1000.0	333	1.5E+5	**97.5**	1000.0	**138**	**4.4E+3**
i640-201	640	1920	50	48.2	1000.0	600	2.6E+6	**100.0**	**2.6**	**48**	**1.5E+5**
i640-211	640	4135	50	80.4	1000.0	515	**2.5E+5**	**97.6**	1000.0	**114**	4.7E+5
i640-231	640	2560	50	56.2	1000.0	588	2.0E+6	**99.7**	**243.9**	**119**	**6.0E+5**
i640-301	640	1920	160	55.7	1000.0	394	**7.0E+4**	**100.0**	**10.1**	**28**	7.8E+4
i640-311	640	8270	160	84.5	1000.0	**1335**	**9.0E+0**	**86.4**	1000.0	1376	1.2E+1
i640-331	640	2560	160	61.5	1000.0	**2338**	**9.0E+0**	**62.6**	1000.0	2482	**9.0E+0**
Aggregate				64.6				94.0	0.04	0.2	0.2



(a) Lower Bound (b) Condition Number κ

Fig. 1. Growth rate of the lower bound and the condition number as a function of the CPU time for the instance `i640-131`. The vertical axis in (b) is in logarithmic scale.

5.2 Max Clique

We compare *(SEP)*, *(BC-SEP)*, and *(MC-BC-SEP)* for the Max Clique problem on a set of instances taken from the Second DIMACS Implementation Challenge [JT96]. We consider a setting where the pool of cuts $\mathcal{C}^0$ is initialized as the empty set. Therefore, the initial relaxation *(P^0)* only contains the box constraints (11). Since we consider a single family of inequalities, and solve each separation problem to optimality, a single cut is added at each round. The separation problems are solved with CPLEX.

Table 3 reports, for each of *(SEP)*, *(BC-SEP)*, and *(MC-BC-SEP)* and for each instance, the CPU time in seconds (Time), the number of cuts generated (Cuts) and the condition (κ) of the basis matrix of the solution of the last relaxation, as computed by CPLEX. The time limit is set to 1000 seconds. When *(BC-SEP)* or *(MC-BC-SEP)* outperform

Table 2. *(SEP)* vs. *(BC-SEP)*, on the PUC SteinLib instances.

				(SEP)				*(BC-SEP)*			
Instance	$\|V\|$	$\|E\|$	$\|T\|$	Gap	Time	Rnds	Avgκ	Gap	Time	Rnds	Avgκ
cc3-4p	64	288	8	**91.9**	334.5	871	1.6E+5	91.2	**10.6**	**148**	**4.7E+4**
cc3-4u	64	288	8	79.5	1000.0	903	5.5E+4	**90.5**	**31.7**	**240**	**5.1E+4**
cc3-5p	125	750	13	30.5	1000.0	872	**3.5E+5**	**91.8**	**374.6**	**352**	6.2E+5
cc3-5u	125	750	13	29.9	1000.0	843	3.4E+5	**90.3**	1000.0	**305**	**1.3E+5**
cc5-3p	243	1215	27	8.5	1000.0	521	**1.2E+5**	**97.9**	1000.0	**163**	4.1E+5
cc5-3u	243	1215	27	8.2	1000.0	510	**1.2E+5**	**97.3**	1000.0	**157**	3.5E+5
cc6-2p	64	192	12	**94.1**	96.6	531	9.4E+4	93.7	**1.5**	**61**	**2.3E+4**
cc6-2u	64	192	12	66.6	1000.0	866	1.4E+5	**92.7**	**2.5**	**74**	**5.3E+4**
hc10p	1024	5120	512	0.6	1000.0	307	**1.1E+5**	**97.4**	1000.0	**24**	1.2E+6
hc10u	1024	5120	512	0.5	1000.0	409	**2.5E+3**	**97.3**	1000.0	**19**	6.1E+4
hc6p	64	192	32	**96.6**	188.2	431	7.7E+4	96.5	**0.1**	**15**	**7.5E+2**
hc6u	64	192	32	41.9	1000.0	497	4.2E+4	**95.2**	**8.2**	**55**	**1.5E+4**
hc7p	128	448	64	17.4	1000.0	346	9.6E+4	**96.6**	**1.2**	**21**	**1.0E+4**
hc7u	128	448	64	14.2	1000.0	399	**1.2E+5**	**95.2**	**406.9**	**90**	1.3E+5
hc8p	256	1024	128	3.7	1000.0	228	**2.1E+5**	**98.6**	**30.9**	**33**	2.6E+5
hc8u	256	1024	128	3.2	1000.0	459	**1.1E+5**	**98.0**	1000.0	**47**	1.5E+5
hc9p	512	2304	256	1.4	1000.0	261	**2.2E+5**	**98.6**	**153.3**	**35**	4.6E+5
hc9u	512	2304	256	1.0	1000.0	571	**4.9E+4**	**98.2**	1000.0	**25**	9.6E+4
Aggregate				32.8				95.4	0.1	0.1	0.9

(SEP), or *(SEP)* outperforms both, w.r.t. either of the quantities that we report, the corresponding value is highlighted in bold. The last line (Aggregate) reports, for each column of *(BC-SEP)* and *(MC-BC-SEP)*, the inverse of the geometric means of the ratios, over all instances, of the corresponding column value and that obtained with *(SEP)*. The instance *hamming8-2*, reported in italics, is neglected, since with *(SEP)* the cutting plane algorithm did not terminate within the time limit. Since, within the time limit, all problems were solved to optimality, we omit the column Gap. With *(BC-SEP)*, when compared to *(SEP)*, we solve all the instances in, on average, 90% of the time, generating 70% of the cuts and yielding a final relaxation with 70% the condition number. *(MC-BC-SEP)* yields even better results. When compared to *(SEP)*, we manage, on average, to solve all the problem in 60% of the time, generating 50% of the cuts and yielding a final condition number 30% smaller.

Figure 2 shows the growth rate of the condition number for *(SEP)*, *(BC-SEP)*, and *(MC-BC-SEP)*, plotted as a function of the CPU time, for two instances.

6 Concluding Remarks

We have proposed a bi-criteria separation problem for the generation of cutting panes. This problem amounts to generating, by solving a single

Table 3. *(SEP)* vs. *(BC-SEP)* and *(MC-BC-SEP)*, on DIMACS Max Clique instances.

			(SEP)			*(BC-SEP)*			*(MC-BC-SEP)*		
Instance	$\|V\|$	$\|E\|$	Time	Cuts	κ	Time	Cuts	κ	Time	Cuts	κ
c-fat200-1	200	1534	2.0	222	1314	**1.6**	**183**	**1079**	**1.2**	**162**	**785**
c-fat200-2	200	3235	0.5	79	103	0.8	**77**	**73**	**0.5**	**45**	**2**
c-fat200-5	200	8473	6.6	260	299	6.8	249	290	**5.4**	**206**	**192**
c-fat500-10	500	4459	50.2	410	194	59.1	**409**	330	**34.9**	**249**	**2**
c-fat500-1	500	46627	22.3	450	3976	**11.9**	**329**	**2851**	0.4	**27**	**12**
c-fat500-2	500	9139	59.4	544	4397	64.7	551	4626	**7.2**	**182**	**1494**
c-fat500-5	500	23191	12.8	247	401	18.3	258	418	**8.7**	**127**	**69**
hamming6-2	64	1824	**0.6**	89	35	1.2	101	**30**	0.7	**62**	**12**
hamming6-4	64	704	22.6	232	483	**4.3**	**70**	**234**	**3.5**	**60**	**156**
hamming8-2	256	31616	> *1000*	*57*	*233*	189.5	497	**75**	73.2	236	**40**
johnson16-2-4	120	5460	97.4	515	993	**31.6**	**49**	**38**	**12.8**	**22**	**11**
johnson8-2-4	28	210	0.1	47	43	**0.0**	**12**	**6**	**0.0**	**12**	**6**
johnson8-4-4	70	1855	**3.2**	110	181	3.3	**59**	**76**	3.5	**53**	**52**
MANN_a9	45	918	**0.3**	**49**	51	0.9	**49**	**25**	0.9	**49**	**24**
myciel3	11	20	**0.0**	23	28	0.1	**16**	**19**	0.1	**16**	**19**
myciel4	23	71	1.1	67	136	**0.7**	**39**	**83**	**0.6**	**31**	**59**
myciel5	47	236	21.9	228	782	**8.2**	**90**	**564**	**4.9**	**71**	**298**
queen10_10	100	2940	41.4	391	1524	**41.1**	**339**	**1330**	43.4	**338**	**1365**
queen11_11	121	3960	40.5	433	2194	**32.9**	**351**	3058	**35.1**	**353**	6804
queen12_12	144	5192	54.2	488	1999	**46.3**	**407**	**1056**	**42.1**	**372**	2974
queen13_13	169	6656	71.0	555	2651	**62.0**	**456**	**2631**	**65.4**	**453**	**2527**
queen14_14	196	8372	121.0	648	3553	**120.4**	**538**	**3235**	137.0	**542**	**3106**
queen15_15	225	10360	234.7	713	4211	**217.1**	**608**	**4179**	**193.8**	**600**	**4080**
queen16_16	256	12640	351.7	809	4910	**335.6**	**673**	**4905**	**350.6**	**689**	**4887**
Aggregate						0.9	0.7	0.7	0.6	0.5	0.3

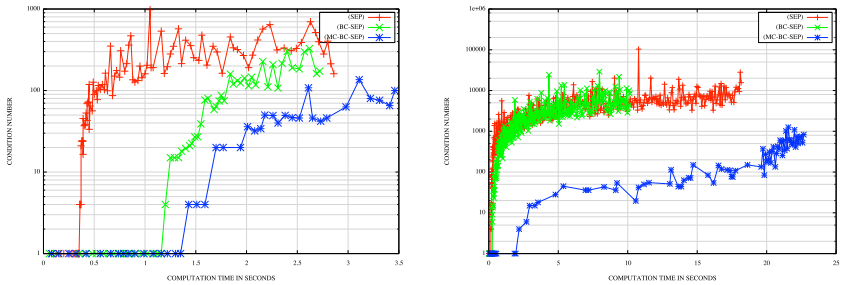
optimization problem, a maximally violated cut which also maximizes, among all the maximally violated ones, a measure of the diversity between the new cut and the previously generated one(s). For cuts with 0-1 coefficients, the bi-criteria separation problem reduces to a standard single-criterion separation problem with a different objective function vector.

Computational results for the Min Steiner Tree and Max Clique problems show that, by solving the bi-criteria separation problem, not only tighter bounds are obtained in a shorter time, but also significantly less cuts are generated.

Currently, we are extending the proposed bi-criteria separation to general classes of inequalities, e.g., Chvátal-Gomory cuts.

References

[ACF07] G. Andreello, A. Caprara, and M. Fischetti. Embedding 0,1/2-cuts in a branch-and-cut framework: a computational study. *J. on Computing*, 19(2):229–238, 2007.



(a) johnson8-4-4 (b) c-fat-500-1

Fig. 2. Growth rate of the condition number as a function of the CPU time for the instances `johnson8-4-4` and `c-fat-500-1`. The vertical axis is in logarithmic scale.

[Ach07] T. Achterberg. *Constrained Integer Programming*. PhD thesis, Technische Universitat at Berlin, 2007.

[BCC93] E. Balas, S. Ceria, and G. Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Math. Program.*, 58(1–3):295–324, 1993.

[BCC96] E. Balas, S. Ceria, and G. Cornuéjols. Mixed 0–1 programming by lift-and-project in a branch-and-cut framework. *Manag. Sci.*, 42(9):1229–1246, 1996.

[BFZ08] E. Balas, M. Fischetti, and A. Zanette. Can pure cutting plane algorithms work? In *Proc IPCO*, volume 5035 of *LNCS*, pages 416–434. Springer, 2008.

[FL06] M. Fischetti and A. Lodi. Optimizing over the first Chvatal closure. *Math. Program.*, 110(1):3–20, 2006.

[JT96] D.S. Johnson and M.A. Trick. Cliques, Coloring, and Satisfiability. Second DIMACS Implementation Challenge, vol 26 of DIMACS Series in Disc. Math. and Theo. Comp. Scie. *Amer. Math. Soc.*, 1996.

[KM98] T. Koch and A. Martin. Solving Steiner Tree Problems in Graphs to Optimality. *Networks*, 32(3):207–232, 1998.

[KMV00] T. Koch, A. Martin, and S. Voß. SteinLib: An updated library on steiner tree problems in graphs. Technical Report ZIB-Report 00-37, ZIB Berlin, Takustr. 7, Berlin, 2000.

[Mar09] F. Margot. Testing Cut Generators for mixed-integer linear programming. *Math. Prog. Comp.*, 1:69–95, 2009.

[MMWW02] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey. Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics 123*, 123:397–446, 2002.

[NW88] G. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.

[PR91] M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Reviews*, 33:60–100, 1991.