Mathematical Programming manuscript No.
(will be inserted by the editor)

Matteo Fischetti · Domenico Salvagnin · Arrigo Zanette

Minimal Infeasible Subsystems and Benders cuts

Received: date / Accepted: date

Abstract There are many situations in mathematical programming where cutting planes can be generated by solving a certain "cut generation linear program" whose feasible solutions define a family of valid inequalities for the problem at hand. Disjunctive cuts and Benders cuts are two familiar examples.

In this paper we concentrate on classical Benders cuts, that belong to the basic toolbox for mixed-integer programming. It is a common experience that the use of Benders cuts is not always as effective as hoped, especially if the impact of simple yet fundamental design issues are underestimated and the method is implemented "as in its textbook description".

In this paper we propose alternative selection criteria for Benders cuts, and analyze them computationally. Our approach is based on the correspondence between minimal infeasible subsystems of an infeasible LP, and the vertices of the so-called *alternative polyhedron*. The choice of the "most effective" violated Benders cut then correspond to the selection of a suitable vertex of the alternative polyhedron, hence a clever choice of the dual objective function is crucial—whereas the textbook Benders approach uses a completely random selection policy, at least when the so-called feasibility cuts are generated.

Computational results on a testbed of MIPLIB instances are presented, where the quality of Benders cuts is measured in terms of "percentage of gap closed" at the root node, as customary in cutting plane methods. We show

M. Fischetti
DEI, University of Padova E-mail: fisch@dei.unipd.it

D. Salvagnin
DMPA, University of Padova E-mail: dominiqs@gmail.com

A. Zanette
DMPA, University of Padova E-mail: zanettea@math.unipd.it

that the proposed methods allow for a speedup of 1 to 2 orders of magnitude with respect to the textbook one.

Keywords Mixed-Integer Linear Programming, Cutting Planes, Decomposition, Benders cuts, Computational Analysis

1 Introduction

There are many situations in mathematical programming where cutting planes can be generated by solving a certain Cut Generation Linear Program (CGLP) whose feasible solutions define a family of valid inequalities for the problem at hand. Disjunctive cuts and Benders cuts are two familiar examples.

Benders cuts were originally proposed in [4] as a machinery to convert a generic mixed-integer program involving integer variables x and continuous variable y into an integer program involving the x variables only, possibly plus a single continuous variable η taking into account the overall contribution to the objective function of the continuous variables (say $d^T y$). The continuous y variables are projected away by a standard projection technique based on dynamic cutting-plane generation. At each iteration, one solves the current *master problem* relaxation in the (x, η) space, and sends the optimal solution (x^*, η^*) to the so-called *slave problem*. This is an LP in the y space that tries to define suitable y-variables y^* such that (x^*, y^*) is feasible for the original problem, and $\eta^* = d^T y^*$. If the slave problem is feasible, we are done. Otherwise, a so-called (feasibility or optimality) *Benders cut* in the (x, η) space is generated by using Farkas' characterization of infeasible LPs, the cut is added to the master problem, and the method is iterated.

The definition of the CGLP is however only the first step for the effective use of the associated cuts, as three main topics need to be addressed:

i) *When to cut?* Possible answers range from "only when an integer super-optimal solution is available" (as in the original proposal of Benders, where cuts are applied only to cut the optimal solution of the current master problem) to "whenever a fractional (or integer infeasible) solution is available" (as in modern branch-and-cut frameworks).
ii) *What to cut?* The usual choice in integer programming is to cut the optimal solution of an LP relaxation. However this may lead to unstable behavior and slow convergence, so stabilization through box constraints or quadratic penalty functions may be needed–this is not usually done in standard branch-and-cut algorithms, but it is common practice e.g. in bundle methods.
iii) *How to choose the cut?* Given the point x^* to be separated, choose the "best possible" cut(s) among the violated ones.

All three points above play an important role in the design of an effective solution method. In this paper we focus on (iii), and in particular we address the topic of selecting in an effective way Benders cuts for general Mixed-Integer Linear Programs (MIPs). As we aim at understanding the properties that make a single Benders cut "a good cut" in a branch-and-cut context, in the present study we do not address alternative solution approaches that generate rounds of Benders cuts (as opposed to just one or two cuts) at each separation call—though we believe that the idea of generating a large number of simultaneous cuts can play an important role in speeding up the overall convergence of any cutting plane method, including the Benders' one.

We propose alternative selection criteria for Benders cuts, and analyze them computationally. As customary in mixed-integer programming, the effectiveness of the generated cuts is measured by the quality of the root node bound.

Our approach is based on the correspondence between minimal infeasible subsystems of an infeasible LP, and the vertices of the so-called *alternative polyhedron*. The choice of the "most effective" violated Benders cut then corresponds to the selection of a suitable vertex of the alternative polyhedron, hence a clever choice of the dual objective function is crucial—whereas the textbook Benders approach uses a completely random selection policy, at least when feasibility cuts are generated.

Computational results on a testbed of MIPLIB instances are presented, where the quality of Benders cuts is measured in terms of "percentage of gap closed" at the root node, as customary in cutting plane methods. We show that the proposed methods allow for a speedup of 1 to 2 orders of magnitude with respect to the textbook one.

2 Benders cuts: theory ...

Suppose we are given a MIP problem

$$\begin{aligned} \min c^T x + d^T y \\ Ax \geq b \\ Tx + Qy \geq r \\ x \geq 0,\ x \text{ integer} \\ y \geq 0 \end{aligned} \tag{1}$$

where $x \in \Re^n$, $y \in \Re^t$, and matrix Q has m rows.

Classical Benders decomposition states that solving such a problem is equivalent to solving

$$\begin{aligned} \min c^T x + \eta \\ Ax \geq b \\ \eta \geq u^T(r - Tx), \quad u \in \text{VERT} \\ v^T(r - Tx) \leq 0, \quad v \in \text{RAY} \\ x \geq 0,\ x \text{ integer} \end{aligned} \tag{2}$$

where the additional variable η takes into account the objective function term $d^T y$, while sets VERT and RAY contain the vertices and extreme rays (respectively) of the polyhedron D defined by:

$$\begin{aligned} \pi^T Q \leq d^T \\ \pi \geq 0 \end{aligned} \tag{3}$$

The above formulation has exponentially many inequalities, so an iterative solution approach based on cutting planes is needed, that can be outlined as follows.

1. Solve the so-called *master problem*:

$$\begin{aligned} \min c^T x + \eta & \\ Ax &\geq b \\ \{\text{previously generated Benders cuts}\} & \\ x \geq 0,\ x \text{ integer} & \end{aligned} \tag{4}$$

   including (some of) the Benders cuts generated so far (none at the very beginning). Let (x^*, η^*) be an optimal solution of the master problem.
2. Solve the so-called *dual slave problem*:

$$\begin{aligned} \max \pi^T (r - Tx^*) & \\ \pi^T Q &\leq d^T \\ \pi \geq 0 & \end{aligned} \tag{5}$$

3. If the dual slave problem is unbounded, choose any unbounded extreme ray $\overline{v}$, and add the so-called *Benders feasibility cut*

$$\overline{v}^T (r - Tx) \leq 0$$

   to the master and go to Step 1. Otherwise, let the optimal value and an optimal vertex be z^* and $\overline{u}$ respectively. If $z^* \leq \eta^*$ then stop. Otherwise, add the so-called *Benders optimality cut*

$$\eta \geq \overline{u}^T (r - Tx)$$

   to the master problem, and go to Step 1.

The distinction between *optimality cuts* (involving the η variable) and *feasibility cuts* (that assert some property of the feasible x vector) is very important in practice, and will be analyzed in greater detail in the sequel.

As already noted by other authors, but seldom applied in practice, Benders cuts can be generated to separate any solution (integer or not) of the master problem. As a consequence, these cuts can easily be embedded into a modern branch-and-cut scheme where Benders cuts (among others) are generated at each node of the branching tree.

Note that:

– Although presented for the MIP case, the Benders framework is by no means limited to it. In particular, any problem of the form

$$\begin{aligned} \min c(x) + d^T y & \\ g(x) &\geq 0 \\ F(x) + Qy &\geq r \\ y \geq 0 & \end{aligned} \tag{6}$$

with arbitrary $c()$, $g()$ and $F()$ is suitable to be solved with this method, provided that we have a solver for the master problem (see [6]). This also means that, given any arbitrary partition of the variables, any linear programming problem can be casted into the Benders framework, by projecting away a subset of the variables. This is indeed done in practice with problems that simplify considerably (e.g., decompose) after fixing a subset of their decision variables—this is the case, e.g., in Stochastic Linear Programs (SLPs).

- The Benders method is in fact a pure cutting plane approach in which, given a solution (x^*, η^*) of a problem relaxation (the master), we look for a violated valid inequality. In particular, the search for such an inequality is done by solving an LP problem (the dual slave), which acts as a *Cut Generating LP* akin to the one used in disjunctive programming (as a matter of fact, disjunctive cuts can be viewed as Benders cuts derived from a compact extended formulation).
- The set of Benders cuts corresponds to the vertices and extreme rays of D and is independent of the current master solution (x^*, η^*), which is used only to decide which is next cut to add. For this purpose a suboptimal (or even infeasible) master solution can be used as well, as e.g. in the recent proposals by Rei e al. [12] and by Poojari and Beasley [11].

Given the considerations above, in the following we focus on a generic LP of the form

$$\begin{aligned} \min c^T x + d^T y & \\ Ax \geq b & \\ Tx + Qy \geq r & \\ x \geq 0 & \\ y \geq 0 & \end{aligned} \tag{7}$$

This LP may be the root relaxation of a MIP problem, or just a large-scale LP problem suitable for Benders decomposition (e.g., a SLP problem).

3 ... and practice

The first question we asked ourselves was: What can be considered a modern, yet classical, implementation of Benders decomposition to be used for benchmarking purposes? As a matter of fact, any implementation of the Benders approach has to face a number of implementation issues that affect heavily the overall performance of the method, and many authors using Benders cuts tend to classify their methods as just "standard implementations" without giving sufficient details.

A first issue is how to obtain a good, yet easily computable, initial lower bound on η, so as to prevent the generation of several dominated (and thus useless) optimality cuts. From a theoretical point of view, we are interested in the best-possible optimality cut of the form

$$\eta \geq \pi^T r - 0^T x$$

so $\pi^T r$ can be obtained by just solving the LP:

$$\begin{aligned} \max \pi^T r \\ \pi^T Q \le d^T \\ \pi^T T = 0^T \\ \pi \ge 0 \end{aligned} \tag{8}$$

However, if the slave problem does not have a special structure (i.e., if it does not decompose nicely), the introduction of the coupling matrix T yields an LP problem of the same size as the original LP, so this approach is not always viable computationally. Therefore, in our tests we prefer to calculate a trivial bound on $d^T y$ based only on the lower and upper bounds on the y variables (if no bounds are given, we just write $\eta \ge -M$ for a suitably large M).

Then we addressed the relative contribution of optimality and feasibility cuts to the convergence of the method. Indeed, according to our computational experience these two classes of cuts behave quite differently in many important respects:

– For many problems where term $d^T y$ gives a significant contribution to the overall optimal value, optimality cuts can be much more effective in moving the bound than feasibility cuts, because they involve the η variable explicitly.
– Optimality cuts are typically quite bad from a numerical point view. In particular, optimality cuts tend to exhibit an higher *dynamism* than feasibility cuts, i.e., a higher ratio between the maximum and minimum absolute value of the cut coefficients. This was somewhat expectable, because optimality cuts have to take into account the objective function, which may be of a completely different magnitude (and precision) with respect to the constraints.
– Optimality cuts tend to be much denser than the feasibility ones. Again, this is not surprising since the role of optimality cuts is to provide a lower bound on the objective function term η that is based on the value of the variables x of the master problem, and it is unlikely that just a few master variables can succeed in producing a tight bound.

As a consequence, it is important to have some control on the kind (and quality) of Benders cuts generated at each iteration. Unfortunately, Benders decomposition—as it is typically implemented in the literature—is heavily biased toward feasibility cuts. As a matter of fact, as long as a violated feasibility cut exists, the dual slave is unbounded and hence no optimality cut is generated. As noted by Benders himself [4], however, if we solve the dual slave with the primal simplex method, then when we discover an unbounded ray we are "sitting on a vertex" of polyhedron D, and thus we can generate also an optimality cut with no additional computational effort. A main drawback of this approach is that optimality cut produced is not guaranteed to be violated, and in any case its discovery was quite "random" as the corresponding vertex is by no mean a one maximizing a certain quality index such as cut violation, depth, etc.

The lack of control on the *quality* of the Benders cuts is even more striking when feasibility cuts are generated, since the textbook method does not give any rule to choose among the unbounded rays. To illustrate this important (and often underestimated) point, suppose that we want to apply a textbook Benders decomposition approach to the well-known Asymmetric Traveling Salesman Problem (ATSP). Our compact MIP formulation then involves binary variables x_{ij} associated with the arcs of digraph $G = (V, A)$, and continuous flow variables y^k_{ij} that describe a flow of value 1 from a fixed source node (say node 1) to sink node k, for all $k \in V \setminus \{1\}$. In this example, system $Ax \geq b$ corresponds to in- and out-degree restrictions, whereas system $Tx + Qy \geq r$ is made by $|V| - 1$ independent blocks corresponding to the flow-conservation equations for each k, plus the coupling constraints $y^k_{ij} \leq x_{ij}$ for all $k \in V \setminus \{1\}$ and $(i, j) \in A$. It is not hard to see that, in this case, Benders cuts are of the feasibility type only, and correspond to the classical Subtour Elimination Constraints (SECs) of the form $\sum_{(i,j)\in\delta^+(S)} x_{ij} \geq 1$. These cuts are known to be facet-defining (assuming G is complete digraph), hence they are very strong in practice—so we can conclude that "Benders cuts make a wonderful job". What is clearly inefficient is instead the way these cuts would be handled by the standard Benders method. First of all, SECs would be generated only after having solved to proven optimality the current master, and used to cut integer points only. This is clearly inefficient, since SECs should be generated at each node of the branching tree, or at least whenever the incumbent solution is updated (as in the old-day method by Miliotis [9,10]). But even if SECs were generated within a modern branch-and-cut framework, what is completely missing in the Benders method is a sensible cut selection criterion—once a violated SEC exists, the dual slave becomes unbounded and *any* violated SEC can be returned by the separation procedure—whereas we know that SEC density (among other characteristics) plays a crucial role in speeding-up convergence.

The considerations above prompted us to introduce an effective criterion for choosing among violated (optimality or feasibility) Benders cuts, very much in the spirit of disjunctive cut generation that is also based on CGLPs (see Balas, Ceria, and Cornuéjols [3], and also Fischetti, Lodi and Tramontani [5]). As far as we know, no research effort was devoted to this particular topic in the literature, with one notable exception—the acceleration procedure by Magnanti and Wong [8]. This procedure provides a criterion to choose, among equivalent optimal vertices of the dual slave polyhedron, a "Pareto-optimal" one that corresponds to a maximally-violated optimality cut that is not strictly dominated (within the master feasible solution set) by any other maximally-violated cut. The procedure has however some drawbacks:

– According to its original definition, the procedure would require the dual slave to have a bounded optimal value, hence it could not be applied in a completely general context involving feasibility cuts—this drawback can however be partially overcome by introducing artificial dual bounds.
– The user has to provide a point in the relative interior of the master feasible set. This is quite a simple task if the the master has a very special structure, as in the cases addressed by Magnanti and Wong in their study, but is NP-hard in general if the master is a MIP, since we need a point in

the relative interior of the convex hull of the integer feasible points, which is usually not known. Moreover, the outcome of the procedure depends on the choice of the interior point.

- The method may be computationally heavy, as it requires to solve two LPs to generate a single cut, the second LP being often quite time-consuming due to the presence of an additional equation that fixes the degree of violation to the cut—this equation is in fact very dense and numerically unstable.
- The Magnanti-Wong criterion benefits from the existence of several equivalent optimal solutions of the dual slave problem (i.e., several maximally-violated optimality cuts), which is however not very frequent when fractional (as opposed to integer) points of the master are cut.

4 Benders cuts and Minimal Infeasible Subsystems

The CGLP of a Benders cut can always be seen as a feasibility problem: given a master solution (x^*, η^*), it is possible to generate a violated cut if and only if the following primal slave problem is infeasible:

$$\begin{aligned} d^T y &\le \eta^* \\ Qy &\ge r - Tx^* \\ y &\ge 0 \end{aligned} \tag{9}$$

or equivalently, by LP duality, if the following dual slave problem is unbounded:

$$\begin{aligned} \max \pi^T(r - Tx^*) - \pi_0 \eta^* & \\ \pi^T Q &\le \pi_0 d^T \\ \pi, \pi_0 &\ge 0 \end{aligned} \tag{10}$$

If the separation is successful, given the dual solution (extreme ray) $(\overline{\pi}, \overline{\pi}_0)$ the generated cut is

$$\overline{\pi}^T(r - Tx) - \overline{\pi}_0 \eta \le 0$$

In practice, one is interested in detecting a "minimal source of infeasibility" of (9), so as to detect a small set of rows that allow to cut the master solution. According to Gleeson and Ryan [7], the rows of any *Minimal* (with respect to set inclusion) *Infeasible Subsystem* (MIS) of (9) are indexed by the support of the vertices of the following polyhedron, sometimes called the *alternative polyhedron*:

$$\begin{aligned} \pi^T Q &\le \pi_0 d^T \\ \pi^T(r - Tx^*) - \pi_0 \eta^* &= 1 \\ \pi, \pi_0 &\ge 0 \end{aligned} \tag{11}$$

where the unbounded objective function—namely, the cut violation to be maximized—has been fixed to a normalization positive value (if the alternative polyhedron is empty, we are done). By choosing an appropriate objective function it is therefore possible to optimize over the alternative polyhedron, thus selecting a violated cut corresponding to a MIS of (9) with certain useful properties. Therefore, the choice of the objective function is a main issue to be addressed when designing a separation procedure based on a CGLP, as in the Benders method.

A natural objective function whose purpose is to try to minimize the cardinality of the support of the optimal vertex (and hence to find a small-cardinality MIS [1]) is

$$\min \sum_{i=1}^{m} \pi_i + \pi_0 \tag{12}$$

As we are only interested in solutions with a positive cut violation, and since $\{(\pi, \pi_0) \geq 0 : \pi^T Q \leq \pi_0 d^T\}$ is a cone, we can swap the role of the objective function (12) and of the normalization condition in (11), yielding the following equivalent CGLP akin to the one used for disjunctive cuts by Balas, Ceria, and Cornuéjols [3]:

$$\begin{aligned} \max \pi^T (r - Tx^*) - \pi_0 \eta^* \\ \pi^T Q \leq \pi_0 d^T \\ \sum_{i=1}^{m} \pi_i + \pi_0 = 1 \\ \pi, \pi_0 \geq 0 \end{aligned} \tag{13}$$

It is worth noting that the feasible solution set of the above CGLP is never empty nor unbounded, so a violated cut can be generated if and only if the CGLP has a strictly positive optimal value. The latter formulation is preferable from a computational point because the normalization constraint $\sum_{i=1}^{m} \pi_i + \pi_0 = 1$, though very dense, is numerically more stable than its "cut violation" counterpart $\pi^T(r - Tx^*) - \pi_0 \eta^* = 1$. Moreover, at each iteration only the CGLP objective function is affected by the change in the master solution, hence its re-optimization with the primal simplex method is usually quite fast.

A geometric interpretation of (13) is as follows. The CGLP feasible set is now defined as the intersection of the homogenization of the dual polyhedron D with the normalization hyperplane $\sum_{i=1}^{m} \pi_i + \pi_0 = 1$. It is not difficult to see that there is a one-to-one correspondence between the vertices of this feasible set and the extreme rays (if $\pi_0 = 0$) and vertices (if $\pi_0 \neq 0$) of D. Therefore, the reformulation does not actually change the set of Benders cuts that can be generated, but it is nevertheless useful in that it allows for a more clever choice of the violated cut to be separated.

5 Computational results

The effectiveness of our CGLP formulation has been tested on a collection of problems from the MIPLIB 2003 library [1]. Among the instances in this

[1] Finding a minimum-cardinality MIS is an NP-hard problem in general; see, e.g., Amaldi et al. [2]

Problem	# variables	# integer	# continuous	# constraints
10teams	2025	1800	225	230
a1c1s1	3648	192	3456	3312
aflow40b	2728	1364	1364	1442
danoint	521	56	465	664
fixnet6	878	378	500	478
modglob	422	98	324	291
momentum1	5174	2349	2825	42680
pp08a	240	64	176	136
timtab1	397	171	226	171
timtab2	675	294	381	294
tr12-30	1080	360	720	750

Table 1 Testbed characteristics

testbed, we have chosen the mixed-integer cases with a meaningful number of integer and continuous variables. Moreover, we discarded some instances with numerical instability and which, after the variables were partitioned, were too easy to solve even by the classical Benders method [2]. Table 1 shows our final testbed with the main characteristics of each instance.

Standard variable partitioning has been applied—integer (and binary) variables are viewed as master variables x, and the continuous variables are viewed as slave variables y.

We implemented two variants of the classical (textbook) Benders method, as well as two variants of our MIS-based CGLP, namely:

tb: This is the original method as proposed by Benders [4]. If the dual slave problem is bounded, we generate one optimality cut, otherwise we generate both a feasibility and an optimality cut (the optimality cut being added to the master problem only if it is violated by the current master solution).

tb_noopt: This is a standard Benders implementation method as often seen on textbooks. This method always generate only one cut per iteration—in case of unboundedness, only the feasibility cut associated with the unbounded dual-slave ray detected by the LP solver is added to the master.

mis: This is our basic MIS-based method. It uses the CGLP (13) to solve the separation problem, hence it generates only one cut per iteration.

mis2: This is a modified version of *mis*: after having solved the CGLP, if the generated cut is an optimality one, we enforce the generation of an additional feasibility cut by imposing the condition $\pi_0 = 0$.

In our experiments, we handled the equations in the MIP model (if any) explicitly, without replacing them with pairs of inequalities; this implies the presence of free dual multipliers and the use of their absolute value in the normalization condition.

The implementation was done in C++ on a Linux 2.6 platform and all tests were performed on an Intel Core2 Quad CPU Q6600 with 4GB of RAM. We used ILOG Cplex 11.0 as the black-box LP solver; we disabled the LP

[2] A couple of instances exhibit a block structure of the slave problem and just a few iterations where enough to terminate the method.

presolver and forced the use of the primal simplex method for the solution of the dual slaves so as to be able to get a meaningful output even in case of unbounded problems. Before solving an instance, we performed a standard bound shifting in order to reduce the number of slave variable bounds to dualize. For this reason, the optimal LP value reported in our tables may differ from the value reported in the literature.

The quality of the generated Benders cuts is measured in terms of "percentage gap closed" at the root node, as customary in cutting plane methods. The results are shown in Tables 2 and 3. Results with *tb_noopt* are not reported since this method was never better (and often much worse) than *tb*: a typical behavior is illustrated in Figures 1 and 2.

		Time				Iterations							
Problem	Method	80%	90%	95%	99%	80%	90%	95%	99%	bestBound	optimum	totTime	totIter
	tb	0.08	0.21	0.22	0.26	24	35	38	43	897.00	897.00	0.51	71
10teams	mis	0.04	0.05	0.06	0.07	9	11	14	19	897.00	897.00	0.21	66
	mis2	0.04	0.05	0.05	0.07	9	11	14	19	897.00	897.00	0.22	66
	tb	-	-	-	-	-	-	-	-	707.61	997.53	1000.45	3714
a1c1s1	mis	3.68	6.01	10.62	19.39	144	218	296	482	997.53	997.53	39.95	914
	mis2	86.58	189.93	280.78	-	62	118	173	-	982.38	997.53	451.98	296
	tb	0.03	0.03	0.04	0.09	1	2	5	13	1005.66	1005.66	0.24	44
aflow40b	mis	0.05	0.05	0.07	0.16	1	1	2	7	1005.66	1005.66	0.89	44
	mis2	0.04	0.04	0.07	0.17	1	1	2	7	1005.66	1005.66	0.90	44
	tb	21.22	24.30	29.16	29.16	595	654	766	766	62.64	62.64	36.15	1251
danoint	mis	0.18	0.18	0.18	1.03	43	43	43	87	62.64	62.64	1.74	186
	mis2	3.00	3.00	3.00	5.42	43	43	43	72	62.64	62.64	12.46	167
	tb	0.75	1.19	1.61	2.14	183	254	310	368	1200.88	1200.88	3.20	523
fixnet6	mis	0.05	0.12	0.16	0.34	39	65	83	139	1200.88	1200.88	0.70	230
	mis2	0.28	0.43	0.64	1.02	26	39	56	87	1200.88	1200.88	1.79	161
	tb	-	-	-	-	-	-	-	-	-	-	-	-
modglob*	mis	0.34	0.34	1.38	2.01	62	62	303	473	20430900.00	20430947.62	50.31	3573
	mis2	0.58	0.87	3.00	6.06	34	61	274	613	20430900.00	20430947.62	44.83	3079
	tb	-	-	-	-	-	-	-	-	-	-	-	-
momentum1*	mis	0.35	0.59	0.73	1.60	0	3	5	18	72793.30	72793.35	26.21	207
	mis2	-	-	-	-	-	-	-	-	-	-	-	-
	tb	0.01	0.01	0.01	1.03	9	14	16	339	2748.35	2748.35	4.11	825
pp08a	mis	0.13	0.28	0.33	0.52	125	195	213	280	2748.35	2748.35	1.71	696
	mis2	0.03	0.04	0.04	0.68	9	13	14	179	2748.35	2748.35	2.20	540
	tb	60.15	61.70	67.09	77.27	676	705	778	963	28655.10	28694.00	83.70	1046
timtab1	mis	1.51	2.33	3.08	5.03	601	831	978	1294	28694.00	28694.00	6.13	1431
	mis2	2.81	3.48	4.13	5.09	362	433	494	575	28694.00	28694.00	5.83	635
	tb	444.26	517.35	663.03	898.50	1162	1388	1731	2165	83269.00	83592.00	1003.04	2327
timtab2	mis	17.96	35.51	52.70	119.58	1091	1493	1812	2965	83592.00	83592.00	204.90	4080
	mis2	14.36	21.33	29.74	46.28	536	682	827	1131	83592.00	83592.00	64.14	1395
	tb	1.22	1.95	3.14	19.14	146	188	222	254	14210.43	14210.43	357.80	518
tr12-30	mis	18.83	36.41	59.49	88.98	547	669	768	860	14210.43	14210.43	123.18	1015
	mis2	0.63	0.66	0.66	12.69	17	18	18	249	14210.43	14210.43	13.42	272

Table 2 Comparison of the effectiveness of various separation methods in moving the lower bound at the root node. We report the computing time and number of iterations needed to reach 80%, 90%, 95% and 99% of the optimal root relaxation value, as well as the total running times and number of iterations needed for convergence (within a time limit of 2,000 seconds). Times are given in CPU seconds. (*) indicates failed cut generation due to numerical problems.

As reported in Table 2 *tb* is the most efficient method only in 1 out of 11 instances, namely `aflow40`, and only with little advantage over the competitors. On the other hand, *mis* and *mis2* are much more effective on 10 out of 11 instances, with speedups of 1 to 2 orders of magnitude. As expected, the average density of the cuts generated by *mis* and *mis2* is considerably smaller than *tb*, see Table 3. This has a positive effect on the rate of growth of the master solution time as a function of the number of iterations, as reported in column *Master Rate* in the table.

A closer analysis of instance `a1c1s1` provides some insights on the strength of the proposed methods: at each iteration, while *tb* generates weak feasibility and optimality cuts, with no selection criteria for both, *mis* is able to cut the current master solution with just a good optimality cut. This is however not always the best strategy: for example, in `timtab1`, `timtab2` and `tr12-30`, feasibility cuts are really crucial for the effectiveness of the method and should be preferred—hence *mis2* becomes the leading method.

A comparison between *mis* and *mis2* shows that *mis* candidates as the method of choice, as it is usually faster due to the extra computing time that *mis2* spends in generating the additional feasibility cut (at least, in our present implementation); see Table 3. Nevertheless, as already mentioned, there are instances such that `timtab2` and `tr12-30` where the extra separation effort is rewarded by a significant improvement of the overall performance.

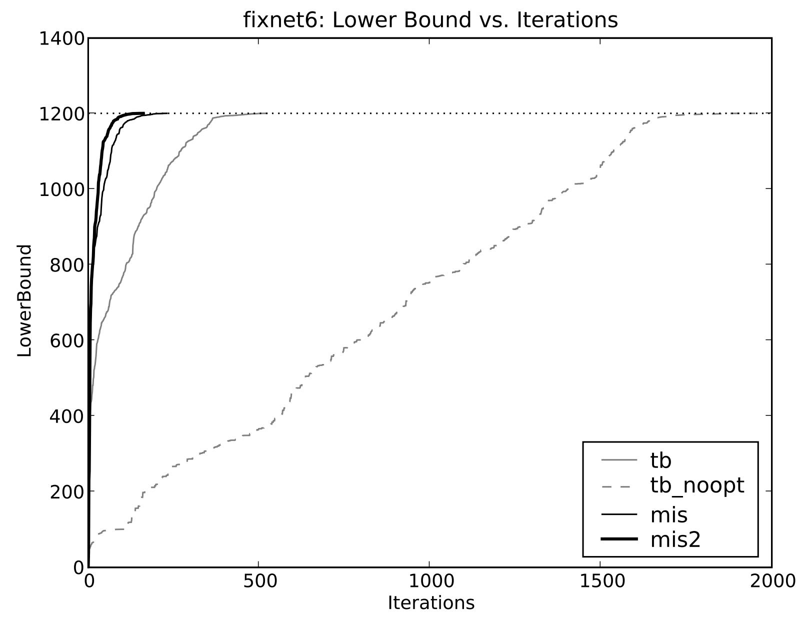
6 Conclusions

We have investigated alternative cut selection criteria for Benders cuts. By using the correspondence between minimal infeasible subsystems of an infeasible LP and the vertices of a so-called *alternative polyhedron*, we were able to define a simple yet effective cut-generation LP allowing for the selection of strong Benders cuts. Computational results on a set of MIPLIB instances show that the proposed method allows for a speedup of 1 to 2 orders of magnitude with respect to the textbook one.

Acknowledgements This work was supported by the Future and Emerging Technologies unit of the EC (IST priority), under contract no. FP6-021235-2 (project "ARRIVAL") and by MiUR, Italy (PRIN 2006 project "Models and algorithms for robust network optimization").

References

1. Achterberg, T., Koch, T., Martin, A.: MIPLIB 2003. Operations Research Letters **34**(4), 1–12 (2006). DOI 10.1016/j.orl.2005.07.009. URL `http://www.zib.de/Publications/abstracts/ZR-05-28/`. See `http://miplib.zib.de`
2. Amaldi, E., Pfetsch, M.E., Jr, L.T.: On the maximum feasible subsystem problem, IISs and IIS-hypergraphs. Mathematical Programming **95**(3), 533–554 (2003). URL `http://dx.doi.org/10.1007/s10107-002-0363-5`
3. Balas, E., Ceria, S., Cornuéjols, G.: Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. Management Science **42**, 1229–1246 (1996)



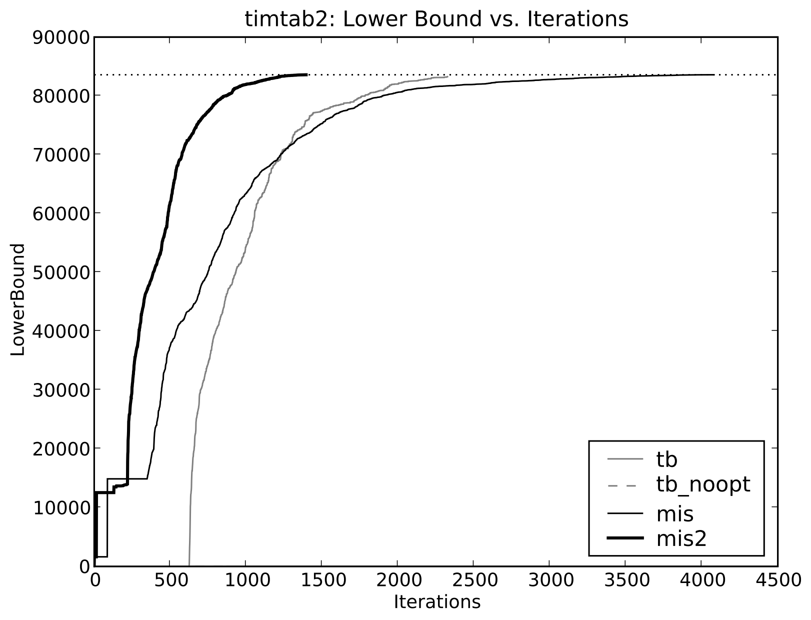
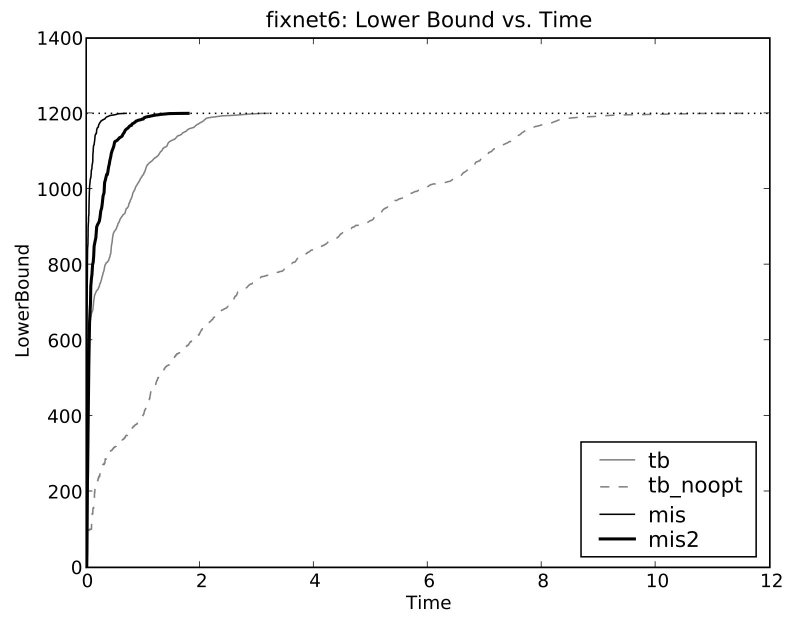


Fig. 1 Lower bound growth vs. iterations with different separation methods. The dotted line is the known optimal value. For timtab2, *tb_noopt* was not able to improve its initial null lower bound.



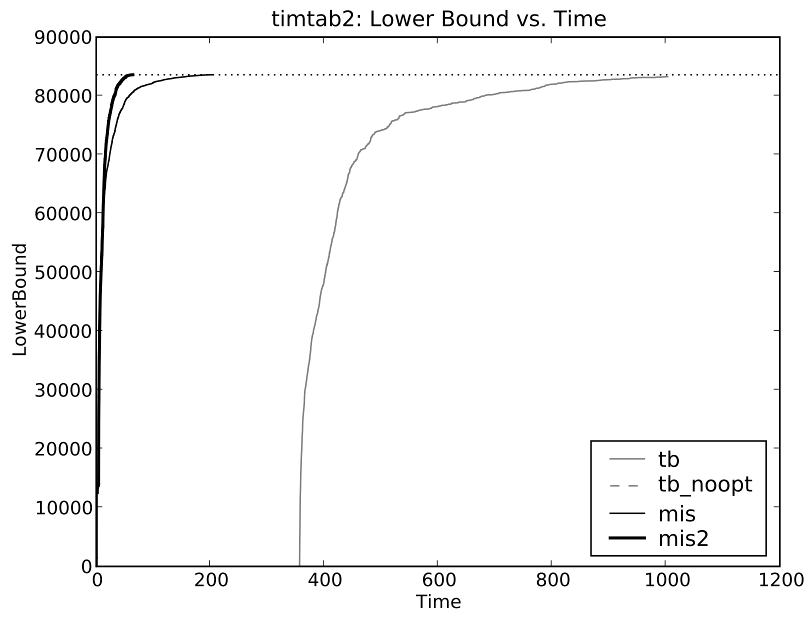


Fig. 2 Lower bound growth vs. time with different separation methods. The dotted line is the known optimal value. For timtab2, *tb_noopt* was not able to improve its initial null lower bound.

4. Benders, J.: Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik **4**, 238–252 (1962)
5. Fischetti, M., Lodi, A., Tramontani, A.: Experiments with disjunctive cuts. Tech. rep. (2007). In preparation
6. Geoffrion, A.M.: Generalized benders decomposition. Journal of Optimization Theory and Applications **10**, 237–260 (1972)
7. Gleeson, J., Ryan, J.: Identifying minimally infeasible subsystems of inequalities. ORSA Journal on Computing **2**(1), 61–63 (1990)
8. Magnanti, T., Wong, R.: Accelerating Benders decomposition: algorithmic enhancement and model selection criteria. Operations Research **29**, 464–484 (1981)
9. Miliotis, P.: Integer programming approaches to the travelling salesman problem. Mathematical Programming **10**, 367–378 (1976)
10. Miliotis, P.: Using cutting planes to solve the symmetric travelling salesman problem. Mathematical Programming **15**, 177–178 (1978)
11. Poojari, C., Beasley, J.: Improving Benders decomposition using a genetic algorithm. Tech. rep. (2006). URL `http://www.optimization-online.org/DB_FILE/2007/02/1591.pdf`. Submitted to INFORMS Journal on Computing
12. Rei, W., Cordeau, J.F., Gendreau, M., Soriano, P.: Accelerating Benders decomposition by local branching. Tech. rep. (2006). To appear in INFORMS Journal on Computing, January 2006.

Problem	Method	# cuts	# opt.	# feas.	Avg Dens.	Master Rate	Avg T. Sep (s)
10teams	tb	107	36	71	383	1.14E-04	1.94E-04
	mis	66	0	66	53	3.97E-05	1.94E-04
	mis2	66	0	66	53	3.44E-05	2.39E-04
a1c1s1	tb	5893	3714	2179	76	1.65E-04	6.01E-03
	mis	914	906	8	26	2.63E-05	3.05E-02
	mis2	577	296	281	14	4.13E-05	1.52E+00
aflow40b	tb	44	0	44	252	4.74E-06	4.24E-03
	mis	44	0	44	242	-5.93E-06	1.86E-02
	mis2	44	0	44	242	1.04E-05	1.89E-02
danoint	tb	1412	1251	161	48	3.09E-06	2.02E-02
	mis	186	186	0	37	5.25E-06	8.72E-03
	mis2	180	167	13	35	4.99E-06	7.38E-02
fixnet6	tb	806	523	283	46	1.05E-05	1.24E-03
	mis	230	210	20	22	9.38E-06	2.01E-03
	mis2	321	160	161	24	1.60E-05	9.59E-03
modglob*	tb	-	-	-	-	-	-
	mis	3573	3557	16	31	6.74E-06	2.30E-03
	mis2	3088	3077	11	29	5.84E-06	6.25E-03
momentum1*	tb	-	-	-	-	-	-
	mis	414	383	31	143	3.18E-04	2.61E-02
	mis2	-	-	-	-	-	-
pp08a	tb	901	825	76	40	7.97E-06	3.14E-04
	mis	696	688	8	17	3.52E-06	5.91E-04
	mis2	613	540	73	16	3.45E-06	2.46E-03
timtab1	tb	2083	1042	1041	56	2.52E-06	4.37E-04
	mis	1431	1354	77	17	4.31E-06	1.10E-03
	mis2	1268	633	635	10	8.82E-06	4.88E-03
timtab2	tb	4609	2316	2293	103	8.98E-05	5.98E-04
	mis	4080	3918	162	45	1.90E-05	3.29E-03
	mis2	2783	1388	1395	23	3.79E-05	1.31E-02
tr12-30	tb	1026	513	513	144	4.13E-03	7.69E-04
	mis	1015	999	16	44	3.20E-04	4.55E-03
	mis2	544	272	272	19	6.75E-05	4.02E-02

Table 3 Statistics on the Benders cuts generated by the different methods. We report the number of generated (optimality and feasibility) cuts, their average density, the rate of growth of the master solution time as a function of the number of iterations (standard linear regression on the master-problem running times vs. iterations), and the average separation time in CPU seconds. (*) indicates failed cut generation due to numerical problems.