Cutting plane methods for integer and combinatorial optimization

Pierre Bonami, Gérard Cornuéjols, Andrea Lodi

Hammamet, March 22-23, 2010

April 10, 2010

Part I

Cutting Planes for Mixed-Integer Linear Programs

1. Mixed-Integer Linear Programming (MILP): notation.
   The Linear Programming (LP) relaxation.
   Strengthening the LP relaxation by cutting planes.
   How much cuts are important in the MILP software?

2. Cutting Planes for MILPs.
   Families of cutting planes and their relationships.

3. Advanced topics.
   Closures and separation.

3. Violated split cuts from the simplex tableau [Fischetti, Lodi & Tramontani 2010]

- a violated split cut can be obtained for free from the tableau basis
- such a split cut is an optimal solution of the CGLP truncated with the "trivial" normalization

$$u_0 + v_0 = 1 \tag{1}$$

- such a split cut is obtained by using only constraints which are tight at x^* (only tight constraints play a role in the cut definition)

3. MIG vs. split cuts: further connections

A different normalization

$$\sum_{i=1}^{m+n} u_i + \sum_{i=1}^{m+n} v_i + u_0 + v_0 = 1 \qquad (2)$$

has been used in the literature [Balas & Perregaard (2002,2003)].

- The CGLP optimal solution with normalization (1) corresponds to a CGLP basic solution truncated with (2).
- By choosing an elementary disjunction of the form

  $$x_j \leq \lfloor x_j^* \rfloor \quad OR \quad x_j \geq \lceil x_j^* \rceil \quad (j \in I, x_j^* \text{ fractional}),$$

  and computing the corresponding CGLP solution (u^*, v^*, u_0^*, v_0^*), then the Balas-Jeroslow strengthening procedure applied to (u^*, v^*, u_0^*, v_0^*) yields precisely the MIG cut associated with the tableau row corresponding to the basic variable x_j.

3. MIG vs. split cuts: further connections (cont.d)

In practice:

- The GMI from the tableau (related to the basic variable x_j with $j \in I$, x_j^* fractional) is a basic solution of CGLP
- Such a solution is optimal if CGLP is truncated with the trivial normalization (1)
  $\Rightarrow$ there is no need to solve the CGLP with (1)
- Such a solution is non-optimal (in general) if CGLP is truncated with normalization (2)

3. MIG vs. split cuts: further connections (cont.d)

[Balas and Perregaard 2003]:

- Warm starting the CGLP with the solution associated with the MIG from the tableau and solving the CGLP with normalization (2) to optimality allows in practice to strengthen the MIG.
  $\Rightarrow$ MIG from tableau can be strengthened by using normalization (2)
- There exists a precise correspondence between the CGLP bases and the bases of the simplex tableau in the original space of the variables (x, s):
  $\Rightarrow$ the CGLP can be implicitly solved by working in the space (x, s) instead of working in the double-sized CGLP space (u, v, u_0, v_0)
  $\Rightarrow$ Consistent speed-up

3. MIG vs. split cuts: further connections (cont.d)

Questions:

- Which is the impact of normalization (2) in practice?
- How much can we gain?

3. Comparing of normalizations

Table: 10 iterations of cuts. At each iteration one cut is generated from any fractional variable.

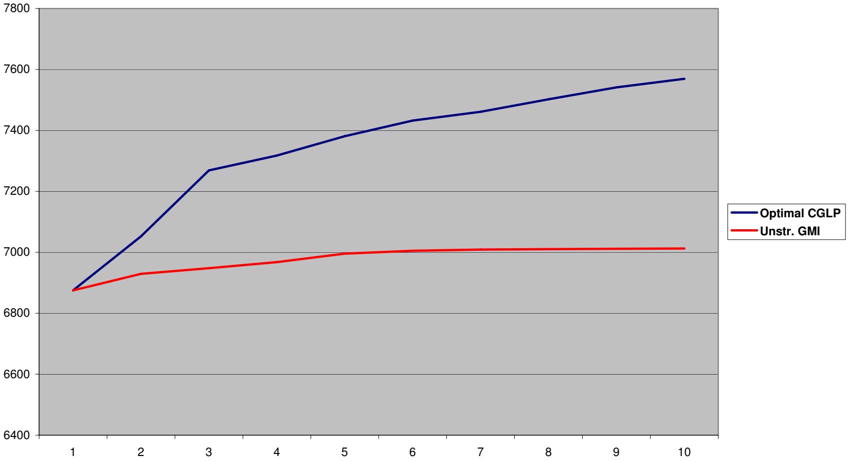
Unstrengthened MIG vs. Optimal CGLP cuts						
	Unstrengthened MIG			Optimal CGLP		
Instance	n.cuts	gap%	$\#S(u,v)$	n.cuts	gap%	$\#S(u,v)$
bell3a	137	70.74	59.49	71	70.74	43.72
bell5	202	28.18	31.20	178	94.29	11.75
blend2	156	28.73	11.70	192	30.51	8.10
flugpl	93	15.15	7.57	92	18.36	5.85
gt2	191	98.71	14.52	196	93.46	10.28
lseu	152	32.94	14.34	196	41.33	9.17
*m.share1	68	0.00	1.00	74	0.00	1.39
mod008	104	12.09	10.40	139	17.05	12.41
p0033	103	58.33	5.72	113	67.86	4.81
p0201	574	18.58	56.03	767	93.82	13.43
rout	445	8.52	135.39	434	24.26	68.07
*stein27	235	0.00	19.74	252	0.00	6.53
vpm1	255	36.95	9.03	263	55.84	5.39
vpm2	424	42.08	71.72	403	74.96	17.27
avg.	236.333	37.583	35.593	253.667	56.873	17.521

3. An example: instance p0201

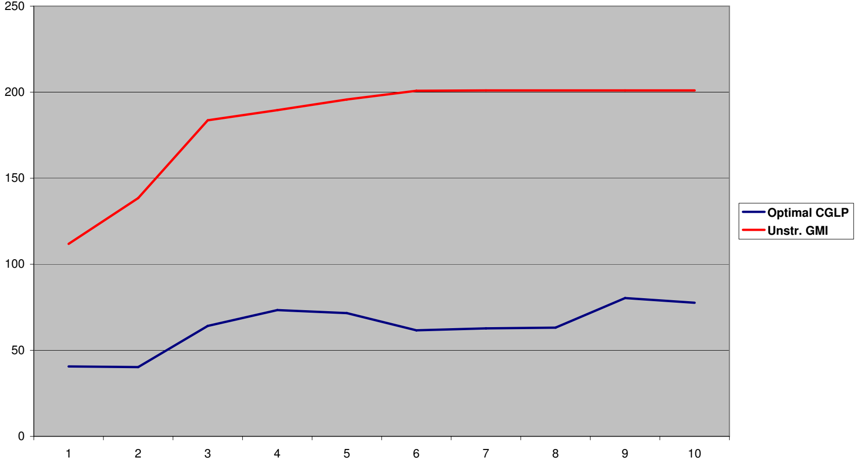
Within 10 rounds of cuts, we report the following indicators:

1. quality of the lower bound
2. average cuts' density
3. cuts' rank
4. average cardinality of the *dual support* $S(u, v)$,
   i.e., how many constraints used on average to generate a cut

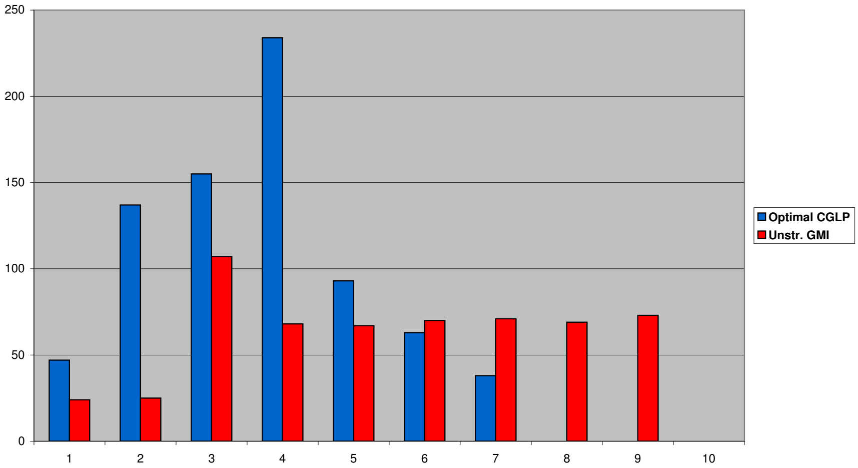
3. Instance p0201: lower bound



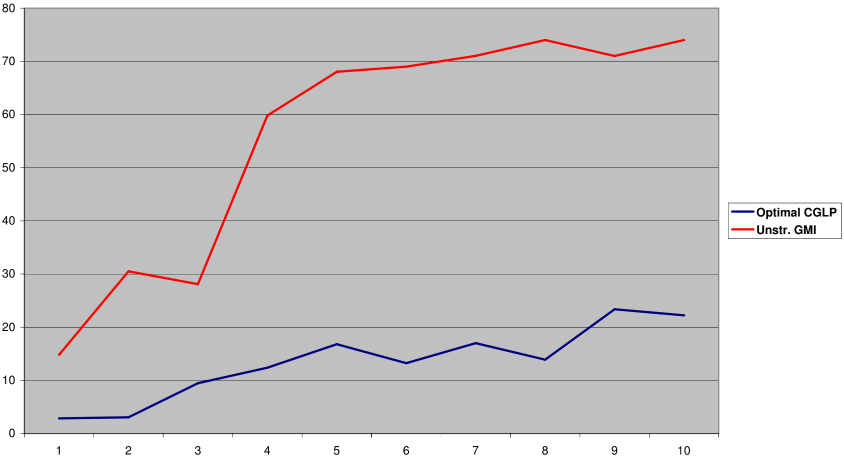
3. Instance p0201: average cuts' density



3. Instance p0201: cuts' rank



3. Instance p0201: average cardinality of $S(u, v)$



3. Why is normalization (2) much better than (1)?

The normalization (1) takes care only of the disjunction:

1. only constraints which are tight at x^* are used in the cut derivation
   $\Rightarrow$ the rank of the cuts becomes higher and higher very quickly:
   - weak cuts are more likely to be used as generating constraints for other cuts
   - round by round, cuts tend to be less effective.
2. constraints' multipliers are not under control:
   2.1 many constraints are used in the cut derivation
   $\Rightarrow$ cuts becomes denser and denser;
   2.2 multipliers can assume huge values and there is no control outside the support of x^*
   $\Rightarrow$ cut coefficients of the variables outside the support are not under control.

3. Why is normalization (2) much better than (1)?

The normalization (2) has the following very nice properties:

1. the norm of the separated cuts tends to be smaller wrt the constraints used for their generation (each multiplier < 1), and small-norm constraints are implicitly penalized by the normalization itself (sum of multipliers $= 1$) $\Rightarrow$ low-rank inequalities are separated.
   - weak cuts are unlikely to be used
   - round by round, cuts tend to remain more effective.
2. since low-rank inequalities are preferred and since the original inequalities (rank-0) are generally sparse, the separated cuts remain sparse.
3. all the multipliers are under control,
   $\Rightarrow$ cut coefficients of the variables outside the support are kept under control.