A Polynomial Procedure For Generating Generalized Intersection Cuts

Selvaprabu Nadarajah

(Joint work with Egon Balas and Francois Margot)

Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA

Full Hyperplane Activation

Focus: Non recursive cut generation for mixed integer programming (MIP) problems

MIP: $\min\{cx : Ax \geq b, x \geq 0,\ x_j \in \mathbb{Z},\ j \in \mathcal{N}_I \subseteq \mathcal{N} := [0,1,\ldots,N]\}$

Linear relaxation feasible set, P: $\{x : Ax \geq b,\ x \geq 0\}$

Integer hull, P_I: $P \cap \{x : x_j \in \mathbb{Z}, j \in \mathcal{N}_I\}$

Issue: Standard recursive cutting plane procedures suffer from numerical issues caused by dual degeneracy (Balas et al. 2010; Zanette et al. 2010)

Solution paradigm: Non-recursive cut generation using intersection points

1. Generate and store the collection of intersection points created by intersecting edges of a relaxation C of P with the boundary of a convex set S such that $int(S) \cap P_I = \emptyset$
2. Use these intersection points to generate deeper generalized intersection cuts (GICs) without recursion

Full hyperplane activation:

Let: C^1 be the cone defined by the optimal simplex tableau

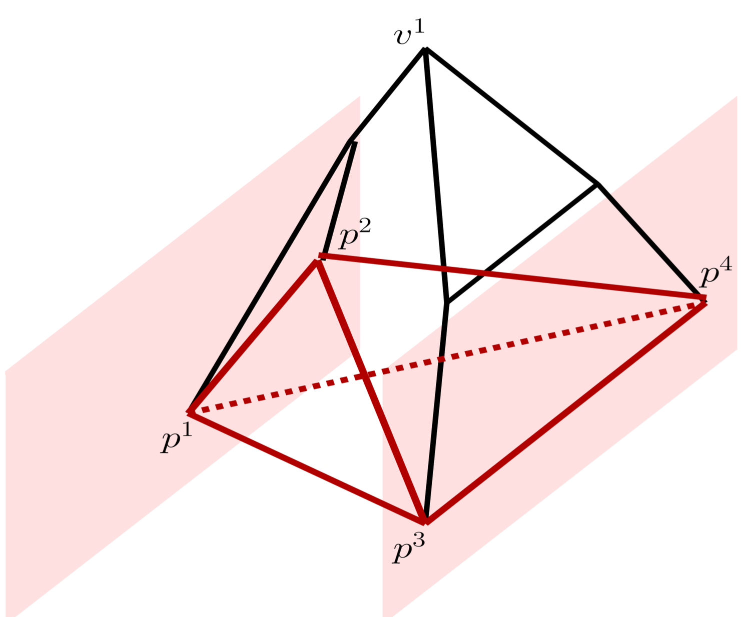
$q_j = r^j \cap \mathrm{bd}S$, where r^j is a ray of C^1

0. Initialize: $\mathcal{P} = \{q^j\}_{j=1}^N$ and $C = C^1$
1. While $n \leq n_h$
   - (a) **Select** halfspace H^+
   - (b) **Remove** from $\mathcal{P}$ all intersection points cut off by H^+
   - (c) **Add** intersection points created by edges of $C \cap H$ with bd$S$
   - (d) **Update** $C = C \cap H^+$
   - (e) $n = n + 1$

Valid cuts: Basic feasible solution $\bar{\alpha}$ to

$$\alpha p^j \geq \bar{\beta},\ \ \forall j \in Q$$

for some $\bar{\beta} \in \{-1, 0, 1\}$ that cuts off a point "above" $\mathrm{conv}(\{p^j\}_{j \in \mathcal{P}})$



Computational issues with full hyperplane activation:

1. Requires a description of the finite and infinite edges of a general polyhedron C
2. The number of edges of C, and hence the number of intersection points, **grow exponentially** with the number of hyperplane activations

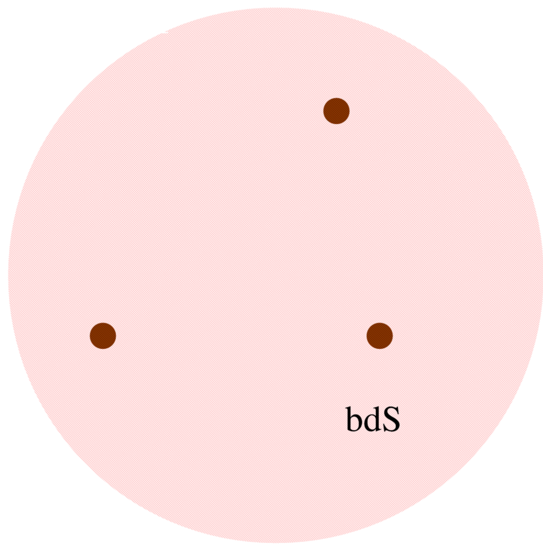
Partial Hyperplane Activation

A streamlined procedure that generates at most $n_h^2 N$ intersection points after n_h partial hyperplane activations

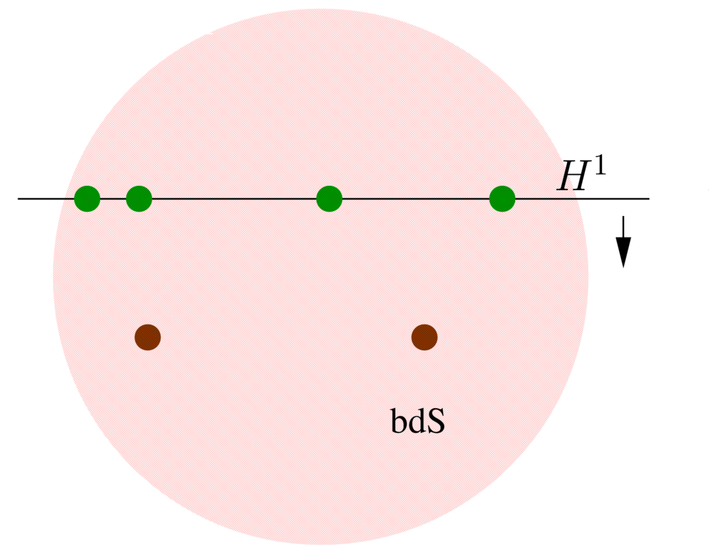
Main ideas:

1. Use a polyhedron C that is a relaxation of P but not defined by the facets of P
2. Activate hyperplanes on C^1 alone: Replace $C \cap H$ in step 1(c) with $C^1 \cap H$ and remove step 1(d)

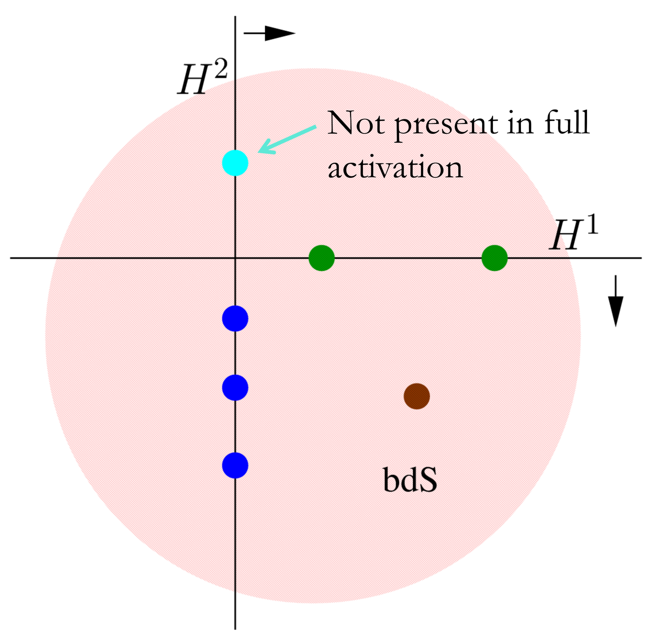
Patch on the bdS with intersection points created by edges of C^1



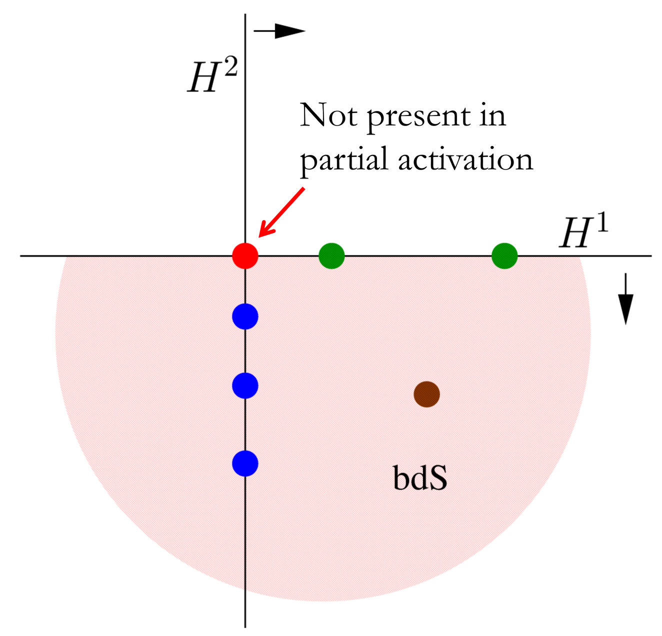
Intersection points after first full or partial hyperplane activation



Intersection points after second partial hyperplane activation



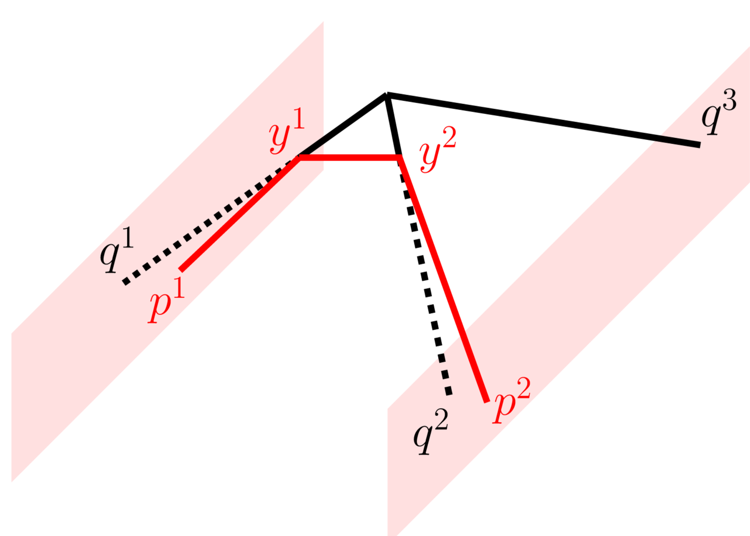
Intersection points after second full hyperplane activation



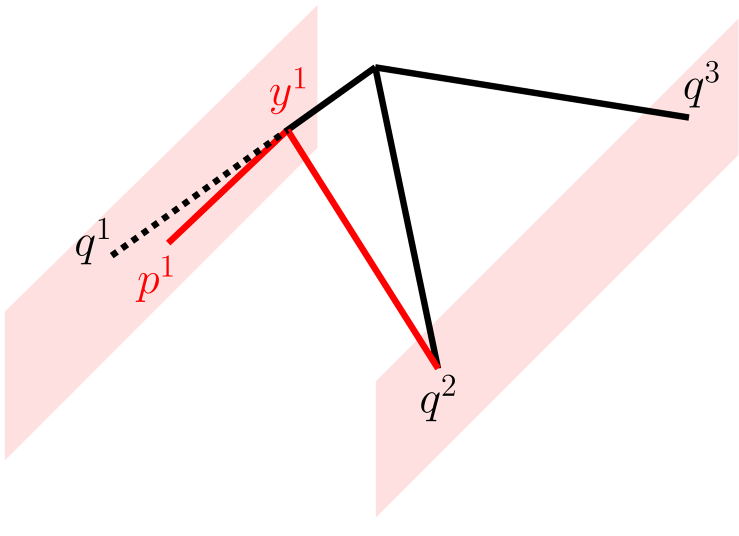
Why "partial"?

When activating a hyperplane H on C^1, we only generate new intersection points from a subset of rays intersected by H

Full activation



Partial activation



Computational Results

- σ denotes the index set of all integer variables fractional in the optimal LP solution
- One round of standard intersection cuts (SICs): $|\sigma|$ SICs, one from each row $x_i, i \in \sigma$
- One round of GICs: $|\sigma|$ GICs generated using simple splits for each $i \in \sigma$
- We compare the gap closed by adding one round of SICs with one round of GICs

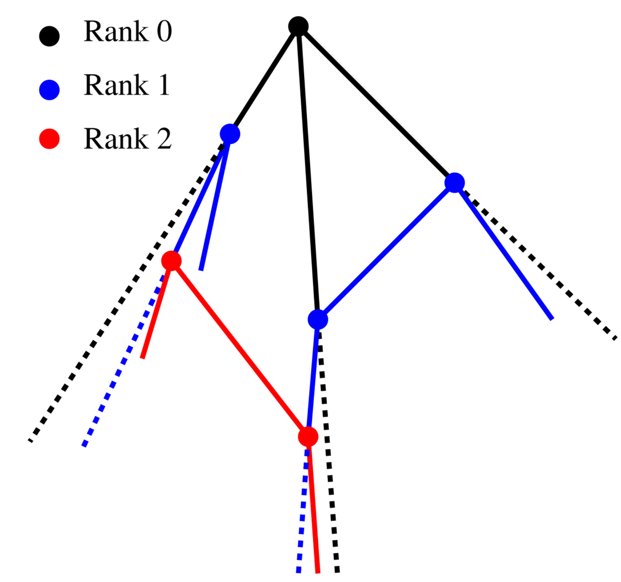
Table 1: Percentage gap closed and CPU time

Instance	$\lvert\sigma\rvert$	n^h	SIC	GIC	GIC − SIC	CPU Time (s)
p0201	13	11	0.00	90.29	90.29	2.95
egout	40	4	51.93	65.09	13.16	0.90
bell3a	32	18	44.74	56.85	12.11	5.11
mod013	5	26	4.40	10.82	6.42	0.11
stein15*	5	3	50.00	54.78	4.78	0.07
bell5	25	13	14.53	18.25	3.72	2.61
bell4	46	13	23.37	25.68	2.31	3.68
bm23	6	3	5.93	7.03	1.10	0.02
misc05	15	17	3.60	4.48	0.87	12.60
lseu	12	5	4.57	5.30	0.73	0.19
p0033	6	7	1.83	2.51	0.68	0.04
bell3b	36	19	44.57	44.91	0.34	6.60
mas74	12	20	3.30	3.55	0.25	19.96
sentoy	8	3	10.38	10.58	0.19	0.13
mas76	11	20	2.37	2.38	0.01	24.42
gt2	11	25	83.13	83.13	0	1.37
sample2	12	25	5.86	5.86	0	1.07

*: objective function $\sum_{i=1}^{15} i x_i$

Conclusions

- Our partial hyperplane activation procedure generates at most $n_h^2 N$ intersection points after n_h partial hyperplane activations
- Computational results show that GICs generated using these points improve on standard intersection cuts in a significant manner
- We are currently testing a substantially faster implementation: O(N) reduction in number of operations required to generate an intersection point.
- **Extension:** A hierarchy of hyperplane activation procedures, where procedures of **higher rank** generate deeper intersection points than ones of lower rank
- **Vertex rank:** Length of shortest path from the apex of C^1
- **Our partial activation procedure is a rank 1 procedure.**
- **Rank 2 procedure:** Repeatedly activate a pair of hyperplanes on C^1



References:

1. Balas, E., M. Fischetti, A. Zanette. 2010. On the enumerative nature of Gomory's dual cutting plane method. Mathematical Programming B, 125(2): 325-351.
2. Balas, E., F. Margot. 2011. Generalized intersection cuts and a new cut generating paradigm. Mathematical Programming A, DOI: 10.1007/s10107-011-0450-6.
3. Zanette, A., M. Fischetti, and E. Balas. 2011. Lexicography and degeneracy: Can a pure cutting plane algorithm work? Mathematical Programming B, 130(1):153-176.