Product and factor filtering for RLT for bilinear and mixed-integer problems

Ksenia Bestuzheva [1], Ambros Gleixner [1,2], Tobias Achterberg [3]

[1]Zuse Institute Berlin and [2]HTW Berlin and [3]Gurobi Optimization

Annual conference of the Society for Operations Research in Germany
September 1, 2023

SPONSORED BY THE
Federal Ministry of Education and Research

Mixed-Integer Programs with Bilinear Products

$$
\begin{aligned}
\min\ & \mathbf{c}^T\mathbf{x} \\
\text{s.t. } & A\mathbf{x} \le \mathbf{b}, \\
& g(\mathbf{x},\mathbf{w}) \le 0, \\
& x_i x_j \lesseqgtr w_{ij}\ \forall (i,j) \in \mathcal{I}^w, \qquad (*) \\
& \underline{\mathbf{x}} \le \mathbf{x} \le \overline{\mathbf{x}},\ \underline{\mathbf{w}} \le \mathbf{w} \le \overline{\mathbf{w}}, \\
& x_j \in \mathbb{R} \text{ for all } j \in \mathcal{I}^c,\ x_j \in \{0,1\} \text{ for all } j \in \mathcal{I}^b,
\end{aligned}
$$

where

g - nonlinear function,
$(*)$ - bilinear product relations.

- We aim to improve the performance of **spatial branch and bound** for MI(N)LPs with bilinear products
- We focus on efficiently constructing **tight linear programming (LP) relaxations**

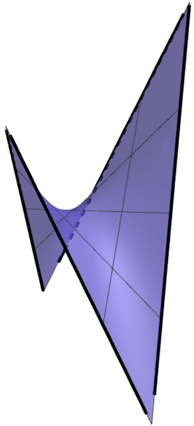
Bilinear Products

We are interested in constraints

$$x_i x_j \lesseqgtr w_{ij} \ \forall (i,j) \in \mathcal{I}^w.$$

These constraints are **nonlinear** and **nonconvex**.

Applications: pooling, packing, wastewater treatment, power systems optimisation, portfolio optimisation, etc.



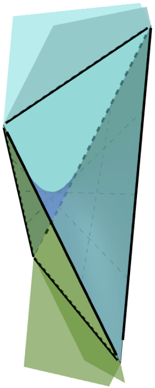
Relaxations of Bilinear Products

The convex hull of $x_i x_j = w_{ij}$ is given by the well-known **McCormick envelopes**:

$$\begin{aligned}
w_{ij} &\geq \underline{x}_i x_j + x_i \underline{x}_j - \underline{x}_i \underline{x}_j, \\
w_{ij} &\geq \overline{x}_i x_j + x_i \overline{x}_j - \overline{x}_i \overline{x}_j, \\
w_{ij} &\leq \underline{x}_i x_j + x_i \overline{x}_j - \underline{x}_i \overline{x}_j, \\
w_{ij} &\leq \overline{x}_i x_j + x_i \underline{x}_j - \overline{x}_i \underline{x}_j.
\end{aligned}$$

This is often a **weak relaxation**! Use other constraints to strengthen it.

RLT (Reformulation Linearization Technique): derive cuts from product relation + combinations of linear constraints/bounds.



RLT Cuts for Bilinear Products

We focus on RLT cuts derived by multiplying a constraint with a variable bound.

For example, multiply constraints of the problem by the lower bound factor of x_j (**reformulation** step):

$$\sum_{i=1}^{n} a_i x_i (x_j - \underline{x}_j) \leq b(x_j - \underline{x}_j).$$

Apply linearizations to each term $x_i x_j$ (**linearization** step):

- if relation $x_i x_j \lesseqgtr w_{ij}$ exists with the appropriate sign, replace $x_i x_j$ with w_{ij}
  - if the relation is violated in the right direction, this will increase cut violation
- otherwise, use a suitable reformulation or relaxation

Motivation and Contributions

- RLT cuts can provide strong dual bounds
- However, a large number of cuts is generated
  - Difficult to select which cuts to apply
  - LP sizes may increase dramatically
  - Even separation itself can be prohibitively expensive

Contributions:

- A method for detecting implicit bilinear products in MILPs $\rightarrow$ can apply bilinear RLT also to MILPs
- A filtering technique for choosing promising implicit products
- An efficient separation algorithm that considers only potentially relevant factor combinations

Implicit Bilinear Products

A bilinear product $w_{ij} = x_i x_j$, where x_i is binary, can be modeled by linear constraints:

Product	Implied relation	Big-M constraint
$w_{ij} \geq x_i x_j$	$x_i = 0 \;\Rightarrow\; w_{ij} \geq 0,$ $x_j = 1 \;\Rightarrow\; w_{ij} \geq x_j.$	$-w_{ij} + \underline{x}_j x_i \leq 0,$ $-w_{ij} + x_j + \overline{x}_j x_i \leq \overline{x}_j$
$w_{ij} \leq x_i x_j$	$x_i = 0 \;\Rightarrow\; w_{ij} \leq 0,$ $x_i = 1 \;\Rightarrow\; w_{ij} \leq x_j.$	$w_{ij} - \overline{x}_j x_i \leq 0,$ $w_{ij} - x_j - \underline{x}_j x_i \leq -\underline{x}_j.$

Deriving product relations from linear constraints:

- Perform the reformulation in the reverse direction: derive product relations from linear constraints
- Generalize to pairs of general linear constraints with at most three variables and at least one binary variable

Redundancy Filtering

$$
\begin{aligned}
&\text{(constraint 1)} && w_{ij} \le \ell_1(x_i, x_j) = d_1/b_1 - (a_1/b_1)x_i - (c_1/b_1)x_j \\
&\text{(constraint 1)} && w_{ij} \le \ell_2(x_i, x_j) = d_2/b_2 - (a_2/b_2)x_i - (c_2/b_2)x_j \\
&\text{(implied product relation)} && w_{ij} \le q(x_i, x_j) = \frac{\gamma}{b_1 b_2} x_i x_j - \left(\frac{a_1 - d_1}{b_1} + \frac{d_2}{b_2}\right) x_i - \frac{c_2}{b_2} x_j + \frac{d_2}{b_2}
\end{aligned}
$$

The product relation is non-redundant if:

$$
q(x_i, x_j) \le \ell_i(x_i, x_j), \;\; i = 1, 2 \;\; \Leftrightarrow
$$

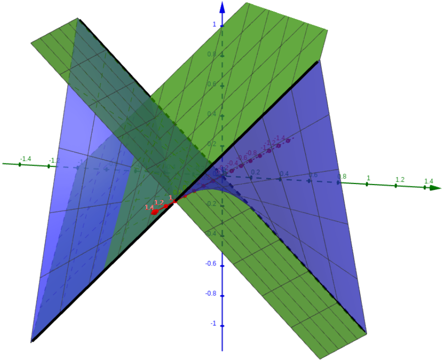
$$
\begin{cases}
x_j \in (1/\gamma(b_1 d_2 - b_2 d_1 + a_1 b_2 - a_2 b_1), 1/\gamma(b_1 d_2 - b_2 d_1)) \text{ if } b_1 \gamma < 0, \\
x_j \in (1/\gamma(b_1 d_2 - b_2 d_1), 1/\gamma(b_1 d_2 - b_2 d_1 + a_1 b_2 - a_2 b_1)) \text{ if } b_1 \gamma > 0.
\end{cases}
$$

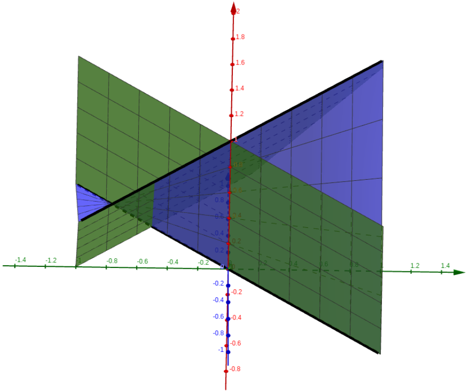
We only use products such that:

$$
\frac{\min\{\overline{x}_j^*, \overline{x}_j\} - \max\{\underline{x}_j^*, \underline{x}_j\}}{\overline{x}_j - \underline{x}_j} \ge 0.3,
$$

where $\underline{x}_j^*$ and $\overline{x}_j^*$ are the smallest and largest values s.t. the product relation is non-redundant.

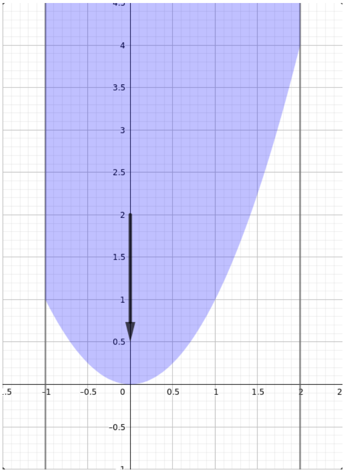
Implicit Product Example





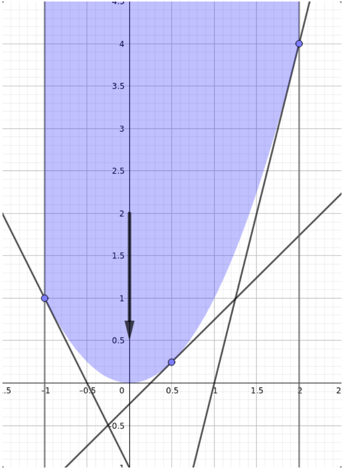
Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node subproblems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i, j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



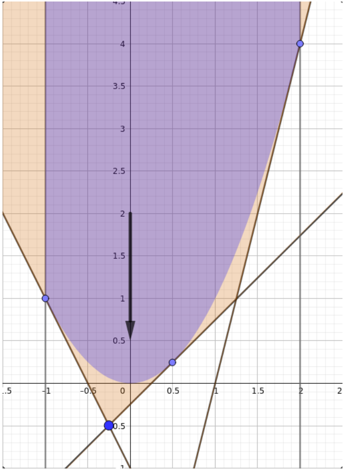
Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node subproblems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i, j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



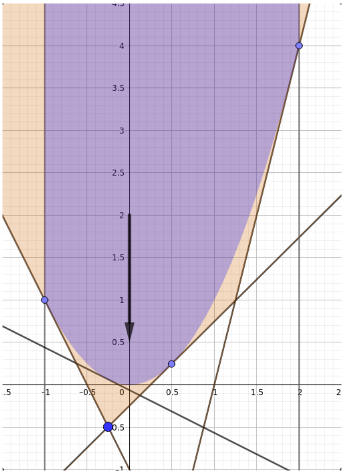
Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node subproblems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i,j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



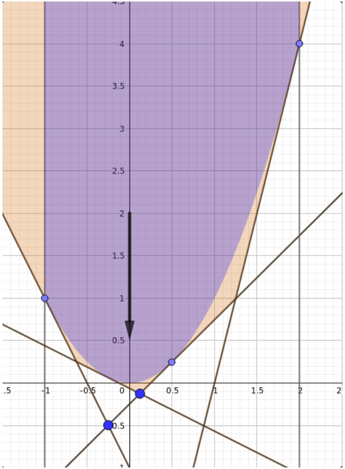
Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node sub-problems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i, j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



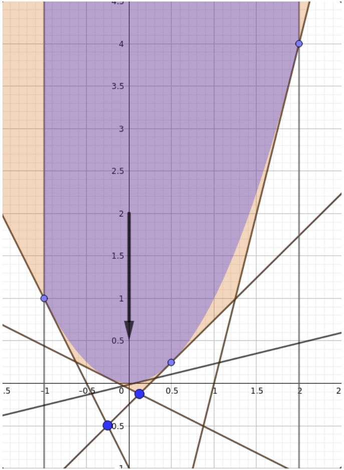
Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node subproblems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i,j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



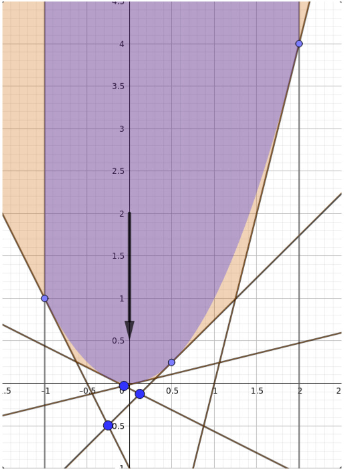
Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node subproblems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i, j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



Separation in spatial BB solvers

- LP-based BB builds LP relaxations of node sub-problems
- $(\mathbf{x}^*, \mathbf{w}^*)$ - solution of an LP relaxation
- Suppose that $(\mathbf{x}^*, \mathbf{w}^*)$ violates the relation $x_i x_j \lesseqgtr w_{ij}$ for some $(i, j) \in \mathcal{I}^w$
- Need to generate cuts that separate $(\mathbf{x}^*, \mathbf{w}^*)$ from the feasible region



How to Separate RLT Cuts More Efficiently

Issue: too many RLT cuts, even separation can become extremely expensive.

Consider:

- A reformulated constraint $\sum_i a_i x_i x_j \leq b x_j$ (always satisfied at $(\mathbf{x}^*, \mathbf{w}^*)$)
- Product relations $x_i x_j = w_{ij}$

Perform the **linearization step**:

- $a_i x_j x_j \to a_i w_{ij}$
- For simplicity assume that all product relations exist with equality
- RLT cut: $\sum_i a_i w_{ij} \leq b x_j$

The cut can be violated at the current LP solution only if $a_i x_i^* x_j^* < a_i w_{ij}^*$ for some i

The key idea is to process as few as possible constraint + factor combinations.

Separation Algorithm

The standard algorithm considers every linear constraint in the problem for each variable that participates in bilinear products. This may become very expensive!

Improved algorithm

- For each multiplier x_j (that participates in bilinear products)
  - For each x_i appearing in violated products with x_j (data structures must allow efficient access)
    - Mark each linear constraint r_k containing x_i with *le* if $a_i x_i^* x_j^* < a_i w_{ij}^*$ and with *ge* otherwise
  - For each marked linear constraint r_k, construct an RLT cut:
    using the lower bound factor if r_k has an *le* mark
    using the upper bound factor if r_k has a *ge* mark (both at the same time is possible)

How the marks work

- Multiply by $x_i - \underline{x}_i$: term $+a_i x_i x_j$ exists in the reformulated constraint
- If there is an *le* mark, then $a_i x_i^* x_j^* < a_i w_{ij}^* \rightarrow$ replacement increases violation

- Multiply by $\overline{x}_i - x_i$: term $-a_i x_i x_j$ exists in the reformulated constraint
- If there is a *ge* mark, then $-a_i x_i^* x_j^* < -a_i w_{ij}^* \rightarrow$ replacement increases violation

Computational Setup

- Using a development version of SCIP (`https://scipopt.org`)
- Time limit one hour
- Testsets: subsets where (either explicit or implicit) bilinear products exist chosen from
  - MINLP: 1846 MINLPLib instances
  - MILP: a testset comprised of 666 instances from MIPLIB3, MIPLIB 2003, 2010 and 2017, and Cor@l
- Frequency: every 10 nodes

- Performed experiments for implicit products with Gurobi: same RLT algorithm, different implementation
- Results not directly comparable, but consistent with SCIP results

Impact of RLT Cuts: MILP

Settings:

- Off: RLT cuts are disabled
- IERLT: RLT cuts are added for both explicit and implicit products

		Off			IERLT			IERLT/Off	
Subset	instances	solved	time	nodes	solved	time	nodes	time	nodes
All	971	905	**45.2**	1339	**909**	46.7	**1310**	1.03	0.98
Affected	581	571	**48.8**	1936	**575**	51.2	**1877**	1.05	0.97
[100,tilim]	329	319	439.1	9121	**323**	**430.7**	**8333**	0.98	0.91
[1000,tilim]	96	88	1436.7	43060	**92**	**1140.9**	**31104**	0.79	0.72

Impact of RLT Cuts Derived From Explicit Products: MINLP

Settings:

- Off: RLT cuts are disabled
- ERLT: RLT cuts are added only for products that exist explicitly in the problem
- IERLT: RLT cuts are added for both explicit and implicit products

		Off			ERLT			ERLT/Off	
Subset	instances	solved	time	nodes	solved	time	nodes	time	nodes
All	6622	4434	67.5	3375	**4557**	**57.5**	**2719**	0.85	0.81
Affected	2018	1884	18.5	1534	**2007**	**10.6**	**784**	0.57	0.51
[100,tilim]	861	727	519.7	35991	**850**	**196.1**	**12873**	0.38	0.36
[1000,tilim]	284	150	2354.8	196466	**273**	**297.6**	**23541**	0.13	0.12

		ERLT			IERLT			ERLT/IERLT	
Subset	instances	solved	time	nodes	solved	time	nodes	time	nodes
All	6622	4565	**57.0**	2686	**4568**	57.4	**2638**	1.01	0.98
Affected	1738	1702	**24.2**	1567	**1705**	24.8	**1494**	1.02	0.95
[100,tilim]	706	670	**359.9**	**22875**	**673**	390.4	24339	1.09	1.06
[1000,tilim]	192	156	**1493.3**	**99996**	**159**	1544.7	107006	1.03	1.07

Impact of the Separation Algorithm

Settings:

- RLT cuts for both explicit and implicit products are enabled
- Marking-off: a straightforward separation algorithm is used
- Marking-on: the new separation algorithm is used

			Marking-off			Marking-on			M-on/M-off	
Test set	subset	instances	solved	time	nodes	solved	time	nodes	time	nodes
MILP	All	949	780	124.0	**952**	**890**	**45.2**	1297	0.37	1.37
	Affected	728	612	156.6	**1118**	**722**	**46.4**	1467	0.30	1.31
	All-optimal	774	774	58.4	**823**	774	**21.2**	829	0.36	1.01
MINLP	All	6546	4491	64.5	**2317**	**4530**	**56.4**	2589	0.88	1.12
	Affected	3031	2949	18.5	**1062**	**2988**	**14.3**	1116	0.78	1.05
	All-optimal	4448	4448	9.1	**494**	4448	**7.4**	502	0.81	1.02

Impact of Redundancy Filtering

Settings:

- IERLT: similar to IERLT in other tables
- RedFilter: redundancy filtering enabled

MILP:		IERLT			RedFilter			RedFilter/IERLT	
Subset	instances	solved	time	nodes	solved	time	nodes	time	nodes
All	1640	1004	561.7	5564	**1011**	**555.2**	**5557**	0.99	1.00
Affected	501	479	227.1	6160	**486**	**266.6**	**6052**	0.96	0.98
[100,tilim]	669	647	552.1	5328	**654**	**539.9**	**5240**	0.98	0.98
[1000,tilim]	223	201	1902.8	25264	**208**	**1783.8**	**23732**	0.94	0.94

MINLP:		IERLT			RedFilter			RedFilter/IERLT	
Subset	instances	solved	time	nodes	solved	time	nodes	time	nodes
All	713	**602**	**53.3**	2455	597	53.5	**2428**	1.00	0.99
Affected	238	**235**	**54.9**	2469	230	55.6	**2374**	1.01	0.96
[100,tilim]	166	**163**	419.1	15704	158	**417.9**	**15197**	1.00	0.97
[1000,tilim]	53	**50**	**1419.8**	55143	45	1477.9	**52918**	1.04	0.96

Summary

- Implicit product relations are detected by analysing MILP constraints
- An algorithm based on row marking efficiently separates RLT cuts
- Redundancy filtering removes implicit products that are (almost) redundant

- RLT cuts improve performance for difficult MILP instances ([1000,timelim])
- RLT cuts for explicit products considerably improve MINLP performance
- RLT cuts derived from implicit products are slightly detrimental to MINLP performance
- The separation algorithm is crucial and enables the improvements yielded by RLT
- Redundancy filtering speeds up MILP solving, but is almost performance-neutral on MINLP