Computing with the Mixed Integer Rounding Cut.

Marcos Goycoolea.
Universidad Adolfo Ibañez
Santiago, Chile.

(joint work with Sanjeeb Dash, IBM Research, NY)

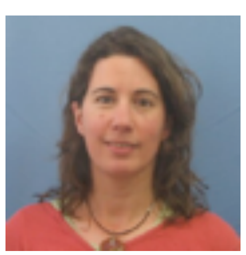
ANILLO ACT-88

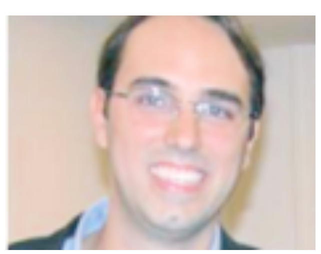
IMSA

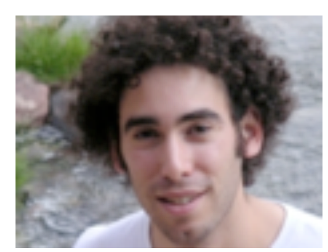
Mathematical modeling for Industrial and Management Science Applications

UNIVERSIDAD ADOLFO IBÁÑEZ








A quick history ...

- Gomory's Mixed Integer cut (1960).

  Introduces the GMI procedure.

- Nemhauser and Wolsey (1990).

  Introduces the MIR cut as a generalization of the GMI cut to non-tableau rows.

- Balas, Ceria, Cornuejols, Nataraj (1996).

  Showed how to use GMI cuts in effective manner.

- Bixby et al (2000,2004).

  Report importance of GMI/MIR cuts in modern software (CPLEX).

An overview of my research on computing with MIR cuts (the last 5 years).

- Effectiveness of generalized MIR inequalities.
- On the strength of GMI and MIR cuts.
- On numerically accurate GMI cuts.
- On GMI cuts from non-optimal bases.

In this talk:

- A more detailed look at the MIR cut.
- On GMI cuts from non-optimal bases.

A closer look at the MIR inequality

A simple mixed-integer set:

$$\sum_{i \in I} a_i x_i + \sum_{j \in C} a_j w_j = b$$

$$x_i \in \mathbb{Z} \;\; \forall i \in I$$
$$x_i, w_j \geq 0 \;\; \forall i, j \in 1, \ldots, n$$

A simple mixed-integer set:

$$\sum_{i \in I} a_i x_i + \sum_{j \in C} a_j w_j = b \qquad \begin{array}{l} x_i \in \mathbb{Z} \;\; \forall i \in I \\ x_i, w_j \geq 0 \;\; \forall i, j \in 1, \ldots, n \end{array}$$

Define $\hat{a} = a - \lfloor a \rfloor$ $\qquad S = \{ i \in I : \hat{a}_i \leq \hat{b} \}$

The Mixed-Integer-Rounding Cut (MIR) :

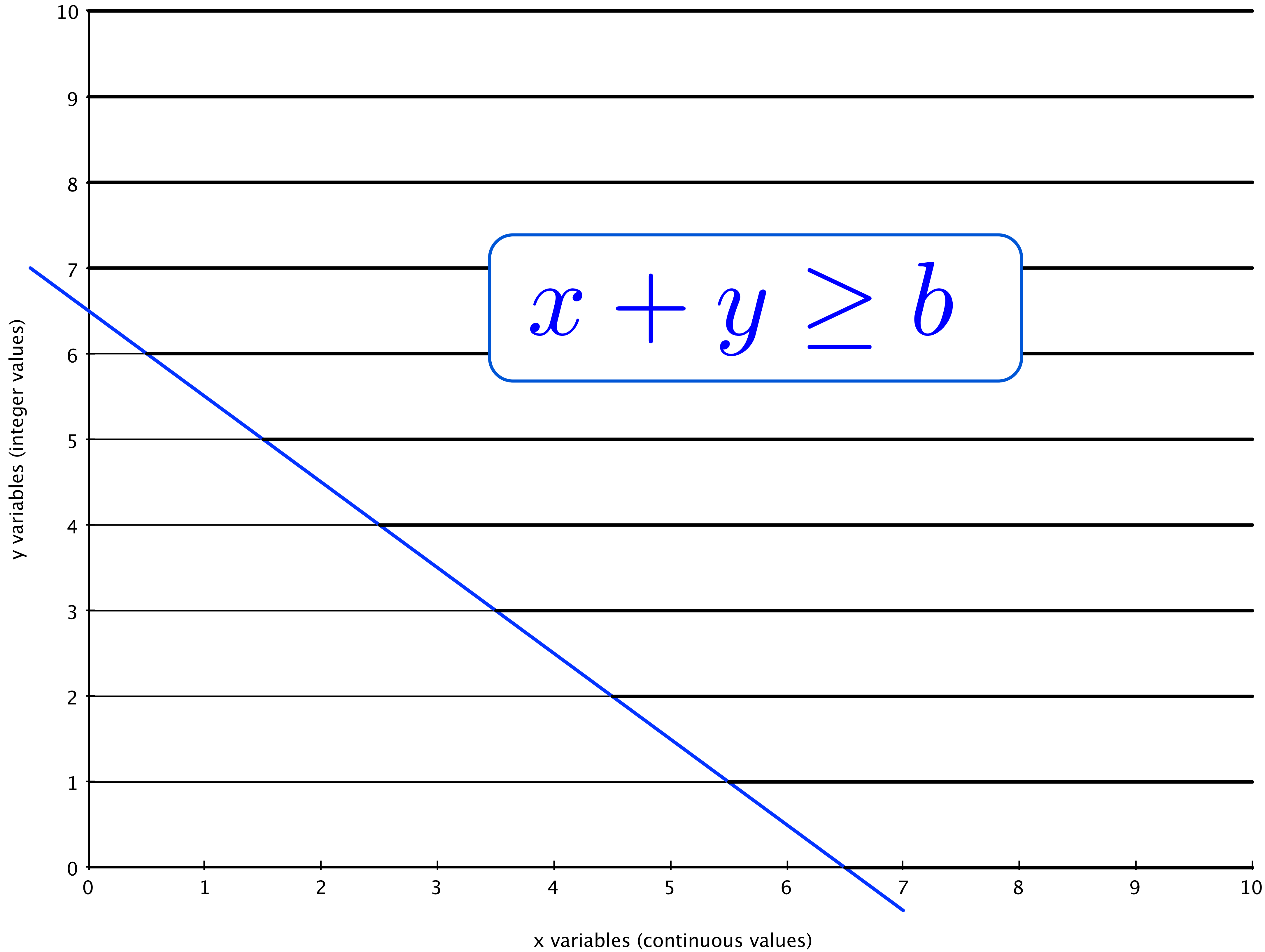
$$\sum_{i \in S} (\hat{a}_i + \hat{b} \lfloor a_i \rfloor) x_i + \sum_{i \in I \setminus S} (\hat{b} + \hat{b} \lfloor a_i \rfloor) x_i + \sum_{a_j > 0} a_j w_j \geq \hat{b} \lceil b \rceil$$

A simple mixed-integer set:

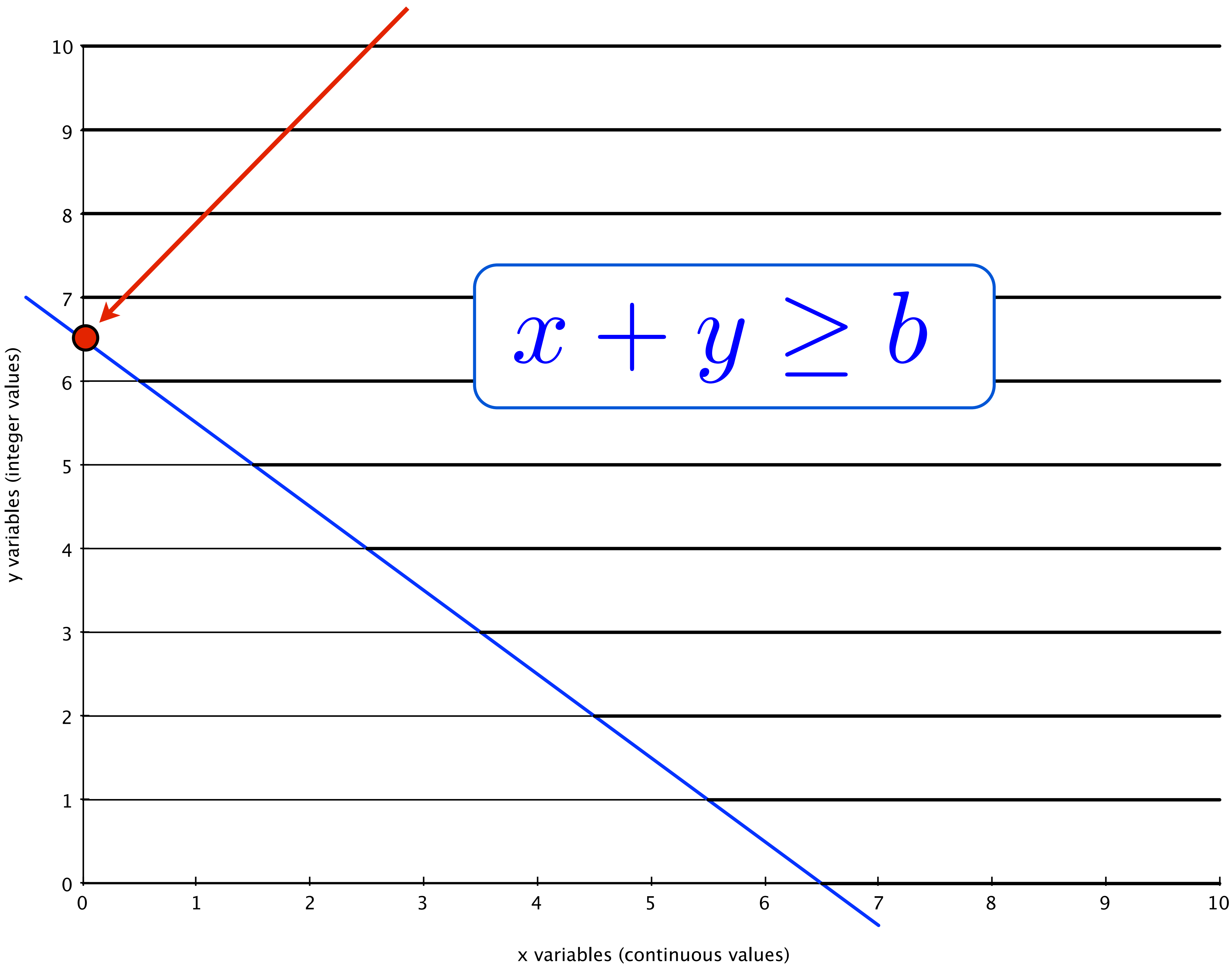
$$x + y \geq b \qquad y \in \mathbb{Z}, x \geq 0$$

The Mixed-Integer-Rounding Cut (MIR) :

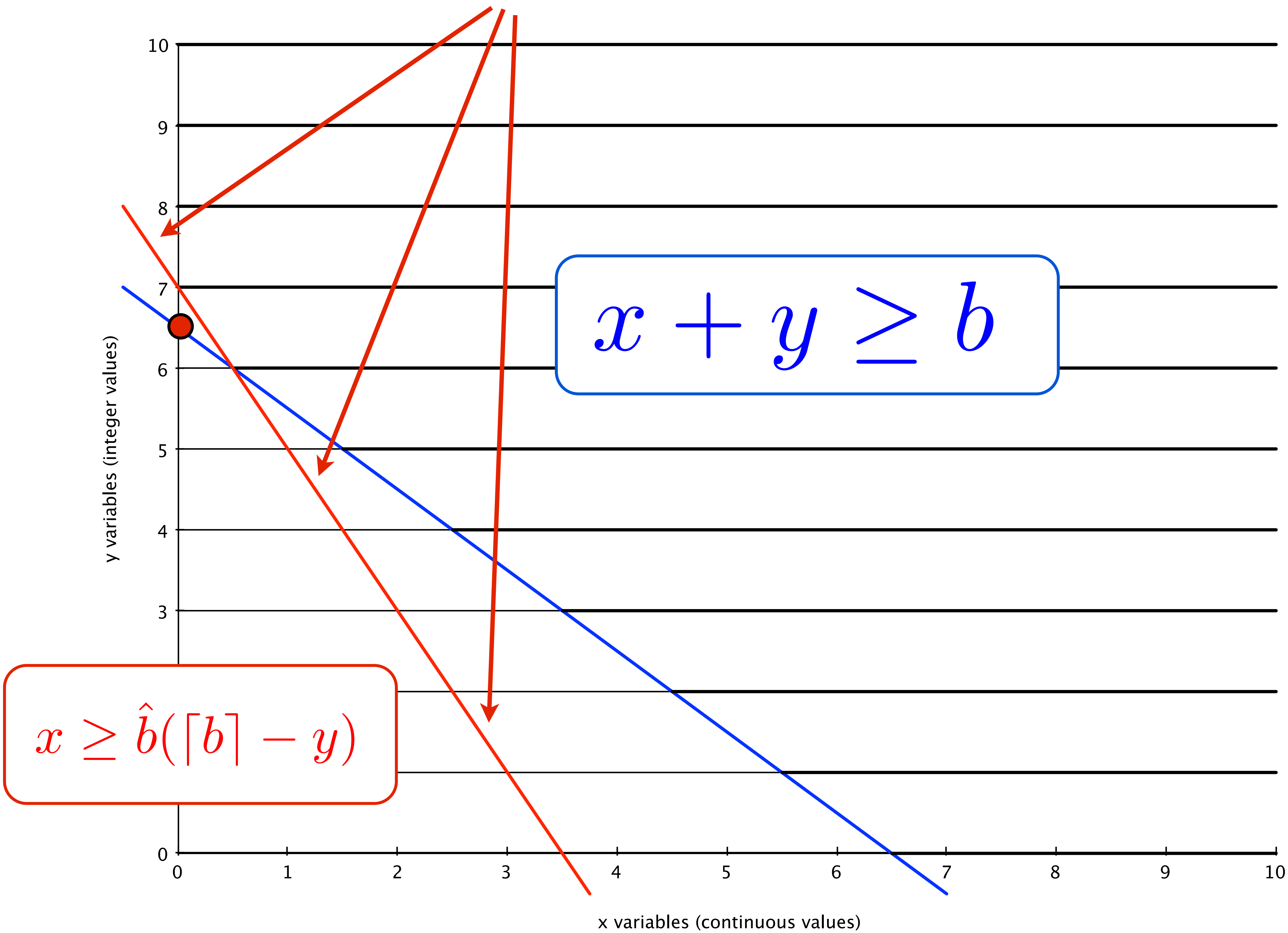
$$x + \hat{b}y \geq \hat{b}\lceil b \rceil$$


$x + y \geq b$
y variables (integer values)
x variables (continuous values)
0
1
2
3
4
5
6
7
8
9
10

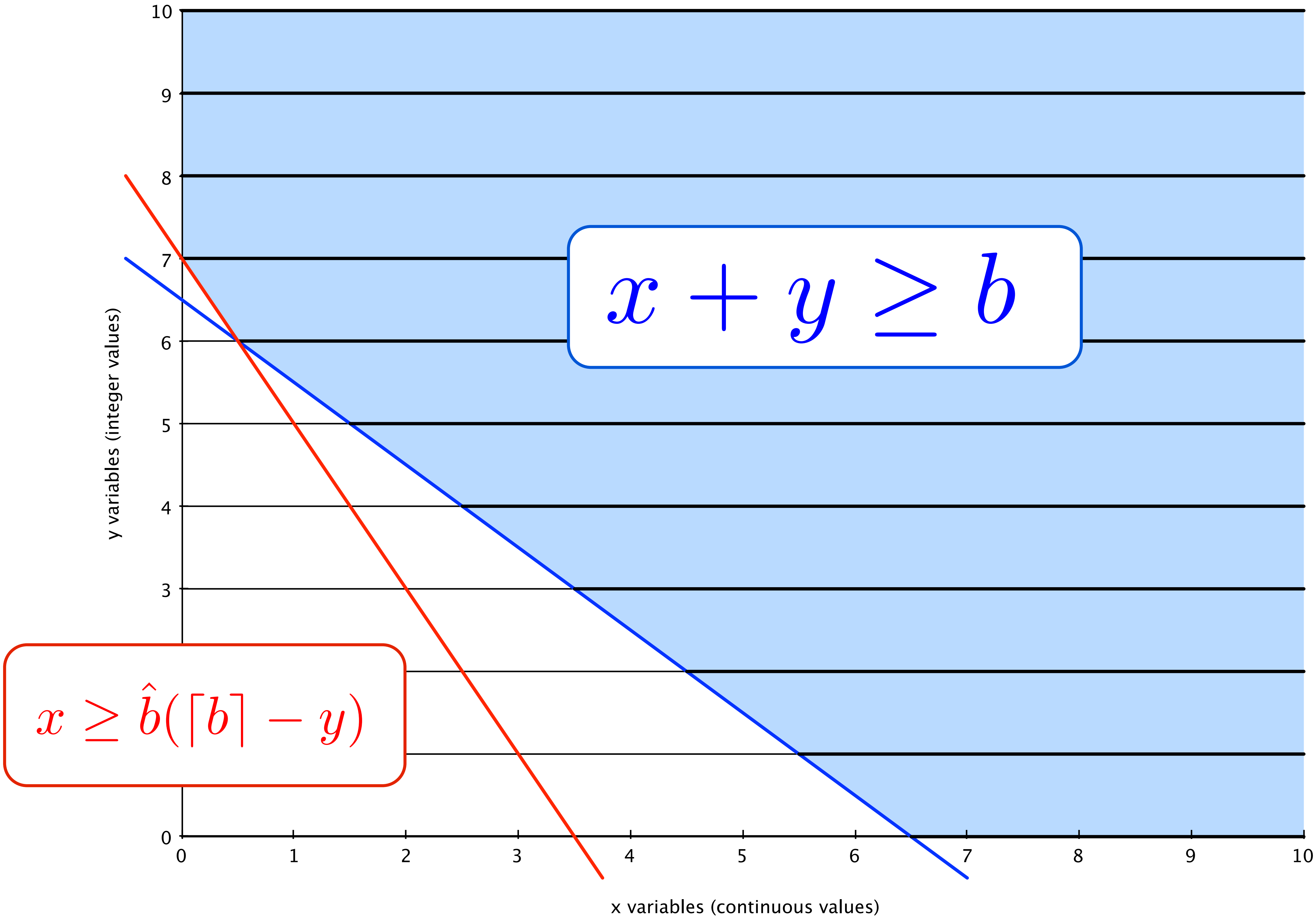
A fractional extreme point

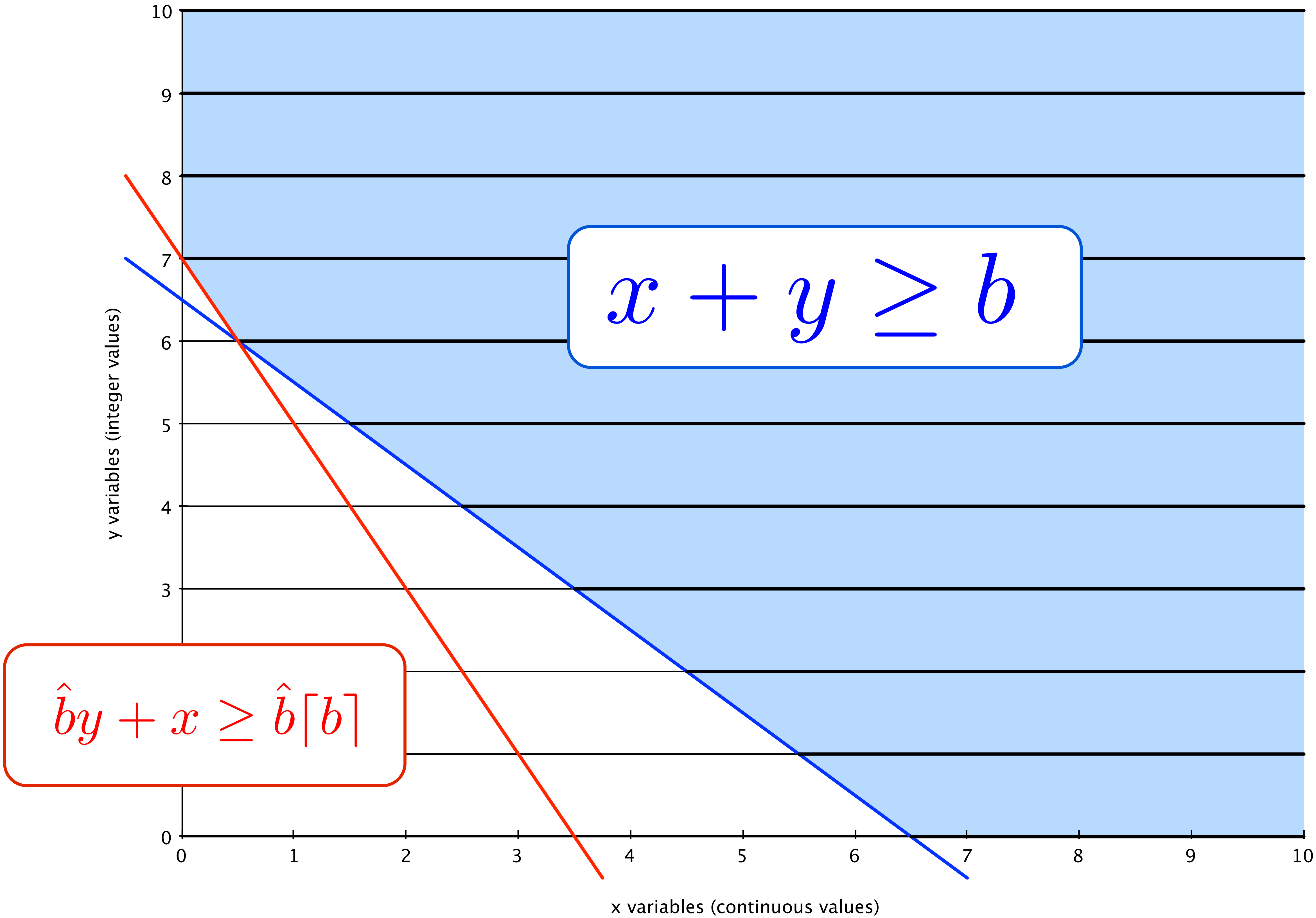


MIR inequality cuts it off!



And defines convex hull of feasible region..





$x + y \geq b$
$\hat{b}y + x \geq \hat{b}\lceil b \rceil$
y variables (integer values)
x variables (continuous values)
0
1
2
3
4
5
6
7
8
9
10

How to derive the general inequality?

$$\sum_{i \in I} a_i x_i + \sum_{j \in C} a_j w_j = b$$

$$\sum_{i \in S \setminus I} \lceil a_i \rceil x_i + \sum_{i \in S \cap I} a_i x_i + \sum_{a_j > 0} a_j w_j \geq b$$

$$\left(\sum_{i \in S \setminus I} \lceil a_i \rceil x_i + \sum_{i \in S \cap I} \lfloor a_i \rfloor x_i \right) + \left(\sum_{i \in S \cap I} \hat{a}_i x_i + \sum_{a_j > 0} a_j w_j \right) \geq b$$

$$\hat{b} \left(\sum_{i \in S \setminus I} \lceil a_i \rceil x_i + \sum_{i \in S \cap I} \lfloor a_i \rfloor x_i \right) + \left(\sum_{i \in S \cap I} \hat{a}_i x_i + \sum_{a_j > 0} a_j w_j \right) \geq \hat{b} \lceil b \rceil$$

GMI cuts and optimal LP relaxation bases

When the base system is defined by an equality, and b is fractional, the MIR is equivalent to:

$$\sum_{i \in I, \hat{a}_i < \hat{b}} \frac{\hat{a}_i}{\hat{b}} x_i + \sum_{i \in I, \hat{a}_i \geq \hat{b}} \frac{1 - \hat{a}_i}{1 - \hat{b}} x_i + \frac{1}{\hat{b}} \sum_{j \in C : a_j > 0} a_j w_j + \frac{1}{1 - \hat{b}} \sum_{j \in C : a_j < 0} a_j w_j \geq 1$$

This is the GMI cut (Gomory, 1960).
Base system optimal tableau row => violated!

GMI-based algorithms.

1. Solve LP relaxation L(i)
2. Apply MIR procedure to optimal tableau rows
3. Add GMI cuts to L(i) and obtain L(i+1)
4. Go to step 1 and repeat.

GMI-based algorithms.

1. Solve LP relaxation L(i)
2. Apply MIR procedure to optimal tableau rows
3. Add GMI cuts to L(i) and obtain L(i+1)
4. Go to step 1 and repeat.

+ Very good bound improvements
+ Relatively cheap (computational time)
- Dense high-rank cuts slow down LP solver
- Numerical problems lead to invalid cuts!

Overcoming these difficulties

1. Carefully control the floating point arithmetic operations to ensure numerical correctness.

2. Stick to generating rank-1 cuts.

Generating multiple rounds of rank-1 GMI inequalities

GMI-based algorithms.

1. Solve LP relaxation L(i)
2. Apply MIR procedure to optimal tableau rows
3. Add GMI cuts to L(i) and obtain L(i+1)
4. Go to step 1 and repeat.

Generating multiple rounds of rank-1 cuts.

1. Solve LP relaxation L(i)
2. Get x*, optimal solution of L(i)
3. Find a set of linear inequalities implied by L
4. Apply MIR procedure to these inequalities, and keep those that are violated by x*
5. Add MIR cuts to L(i) and obtain L(i+1)
6. Go to step 1 and repeat

Generating multiple rounds of rank-1 cuts.

1. Solve LP relaxation L(i)
2. Get x*, optimal solution of L(i)
3. Find a set of linear inequalities implied by L
4. Apply MIR procedure to these inequalities, and keep those that are violated by x*
5. Add MIR cuts to L(i) and obtain L(i+1)
6. Go to step 1 and repeat

Finding a set of linear inequalities implied by L:

First approach: Use IP to compute appropriate linear combination of rows.

Dash, Gunluk and Lodi (2010).
Balas and Saxeena (2008).

Finding a set of linear inequalities implied by L:

First approach: Use IP to compute appropriate linear combination of rows.

Dash, Gunluk and Lodi (2010).
Balas and Saxeena (2008).

Method	GAP closed
1-GMI	26.09%
DGL	62.53%
BS	76.52%

< 1h (w/ time limit)

< 10 days

Our approach: Use heuristics to compute appropriate bases of original LP, and apply GMI procedure.

We start with a system:

$$L_1 = \min\{cx : Ax = b, x \geq 0\}$$

After several rounds of cuts we obtain:

$$\begin{aligned} L_i &= \min\{\bar{c}\bar{x} : \bar{A}\bar{x} = \bar{b}, \bar{x} \geq 0\} \\ &= \min\{cx : Ax = b, Cx \geq d, x \geq 0\} \end{aligned}$$

If we solve Li, we obtain:

$\bar{B}$ an optimal basis

$\bar{x}^*$ an optimal LP solution

We start with a system:

$$L_1 = \min\{cx : Ax = b, x \geq 0\}$$

After several rounds of cuts we obtain:

$$L_i = \min\{\bar{c}\bar{x} : \bar{A}\bar{x} = \bar{b}, \bar{x} \geq 0\}$$
$$= \min\{cx : Ax = b, Cx \geq d, x \geq 0\}$$

If we solve Li, we obtain:

$\bar{B}$ an optimal basis

$\bar{x}^*$ an optimal LP solution

can we use these to find a "good" basis for our original system??

$\bar{B}$ is an optimal basis of

$$L_i = \min\{\bar{c}\bar{x} : \bar{A}\bar{x} = \bar{b}, \bar{x} \geq 0\}$$
$$= \min\{cx : Ax = b, Cx \geq d, x \geq 0\}$$

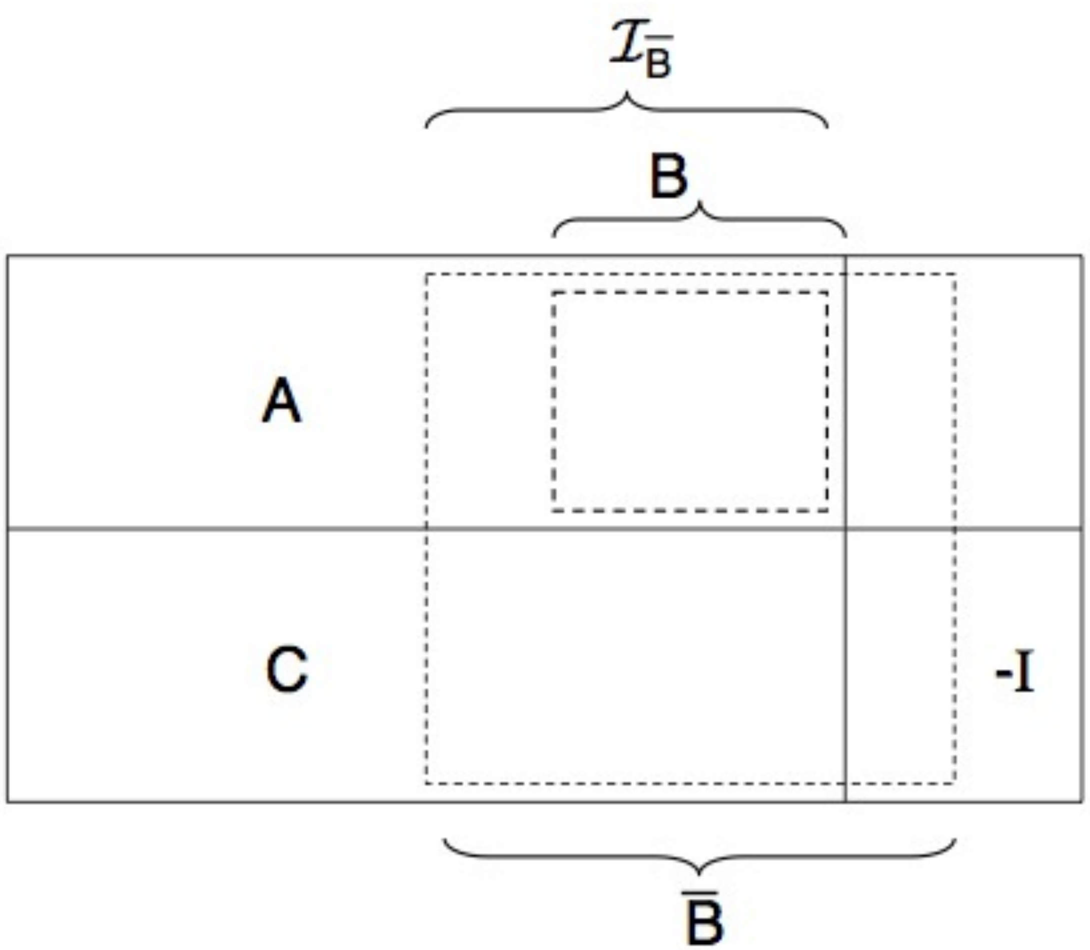
This basis contains slack variables corresponding to rows defined by C!

Does the optimal basis of Li "contain" an optimal basis of L1 ?

(among the structural variables)

$$L_1 = \min\{cx : Ax = b, x \geq 0\}$$

$$\begin{aligned} L_i &= \min\{\bar{c}\bar{x} : \bar{A}\bar{x} = \bar{b}, \bar{x} \geq 0\} \\ &= \min\{cx : Ax = b, Cx \geq d, x \geq 0\} \end{aligned}$$



$$\bar{B} = \begin{bmatrix} A_{\bar{B}} & 0 \\ C_{\bar{B}} & -I' \end{bmatrix}$$

$A_{\bar{B}}$ has full rank!

The basic idea

Consider $\bar{B}$ an optimal basis of

$$L_i = \min\{\bar{c}\bar{x} : \bar{A}\bar{x} = \bar{b}, \bar{x} \geq 0\}$$
$$= \min\{cx : Ax = b, Cx \geq d, x \geq 0\}$$

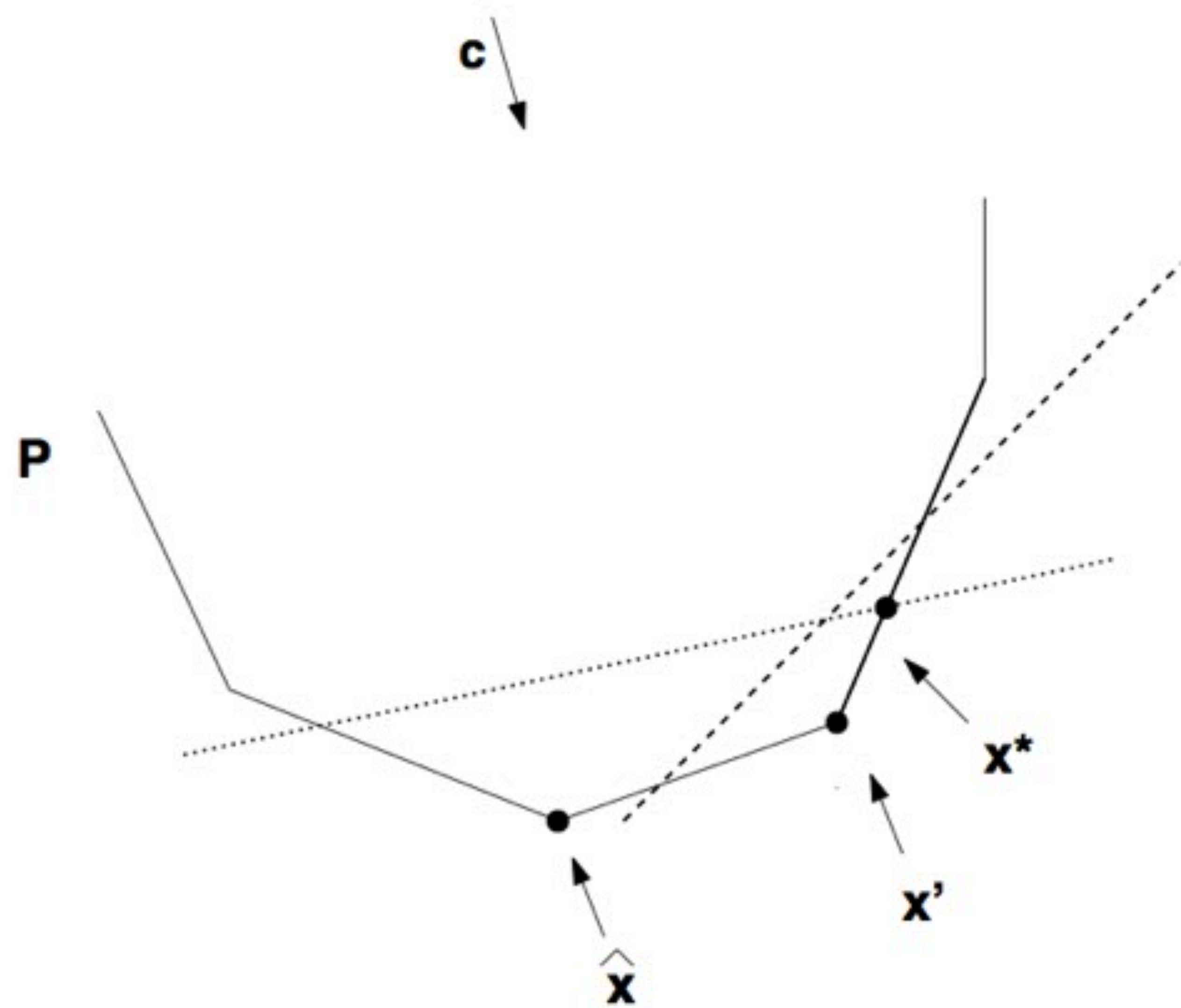
Look for a basis of

$$L_1 = \min\{cx : Ax = b, x \geq 0\}$$

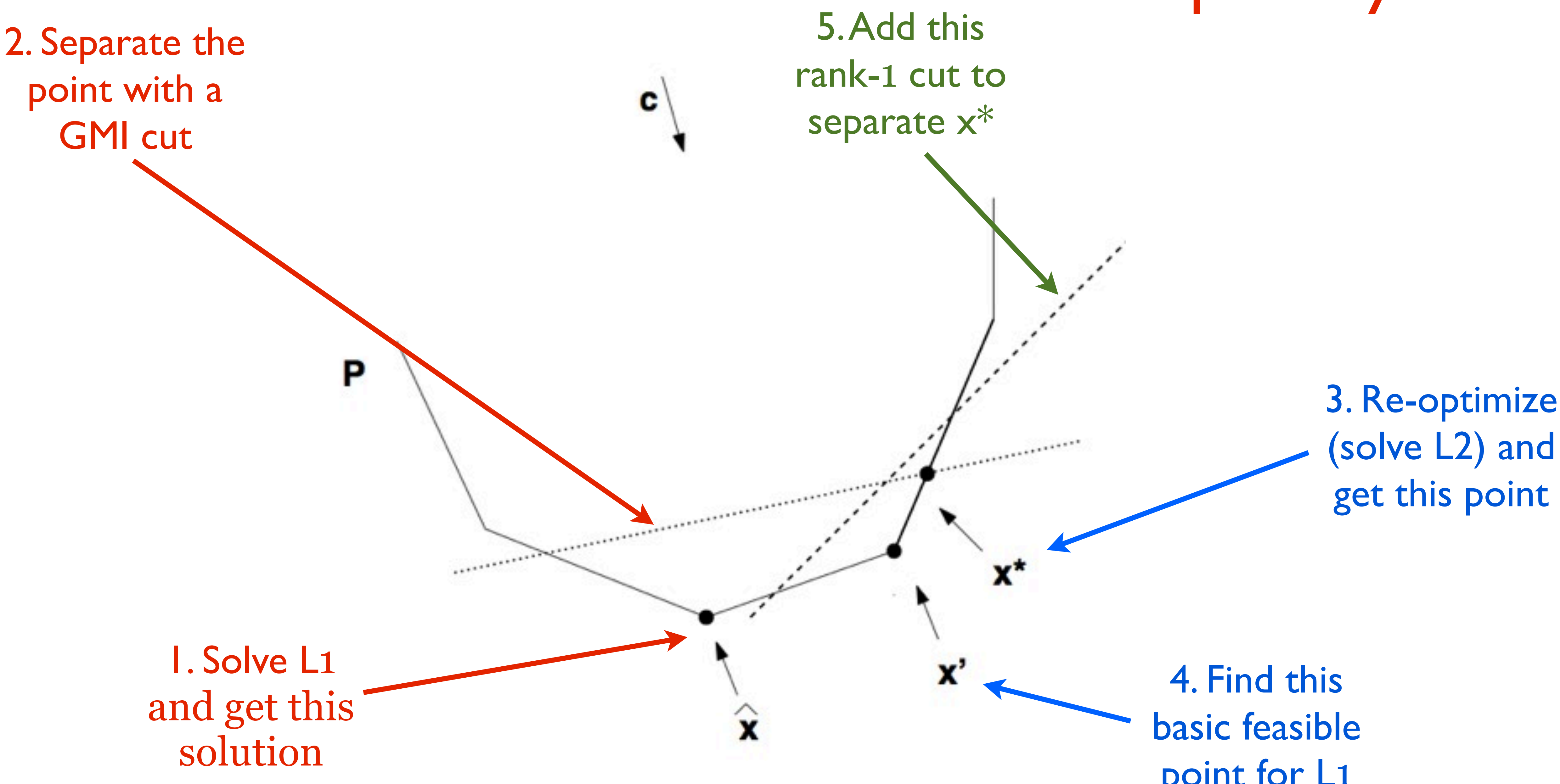
Among the columns of $I_{\bar{B}}$

That is, ignore non-basic variables (fix them at their corresponding bounds) and consider the remaining subproblem.

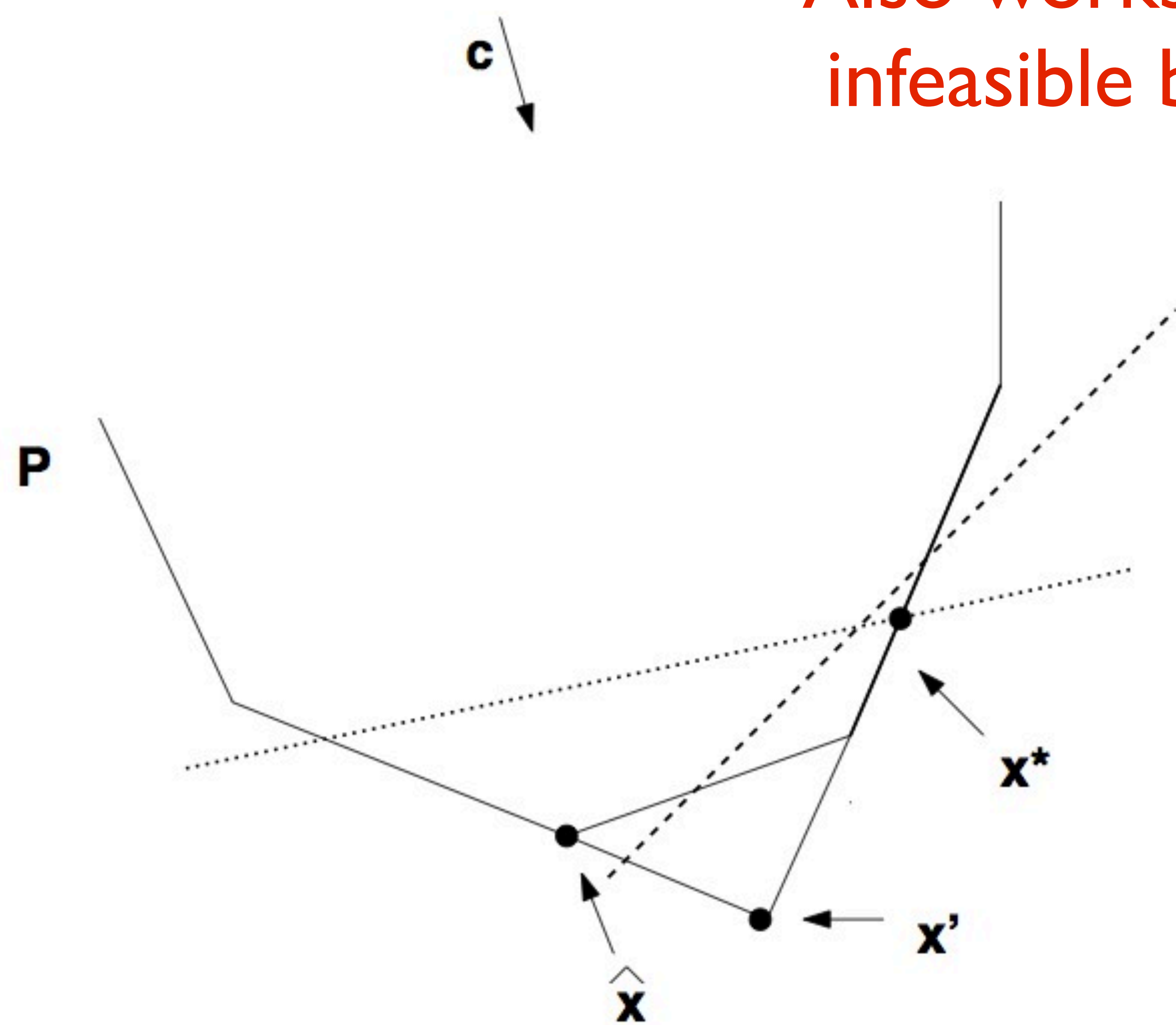
Graphically



Graphically



Also works with infeasible bases



3 techniques to find the basis itself

1. FEAS.
   Solve LP on "unfixed" variables with original objective.

2. MATROID (greedy and random objectives).
   Use greedy matroid algorithm to find max set.

3. SPARSE.
   Refactorize a basis so as to achieve sparsity.

Computational results

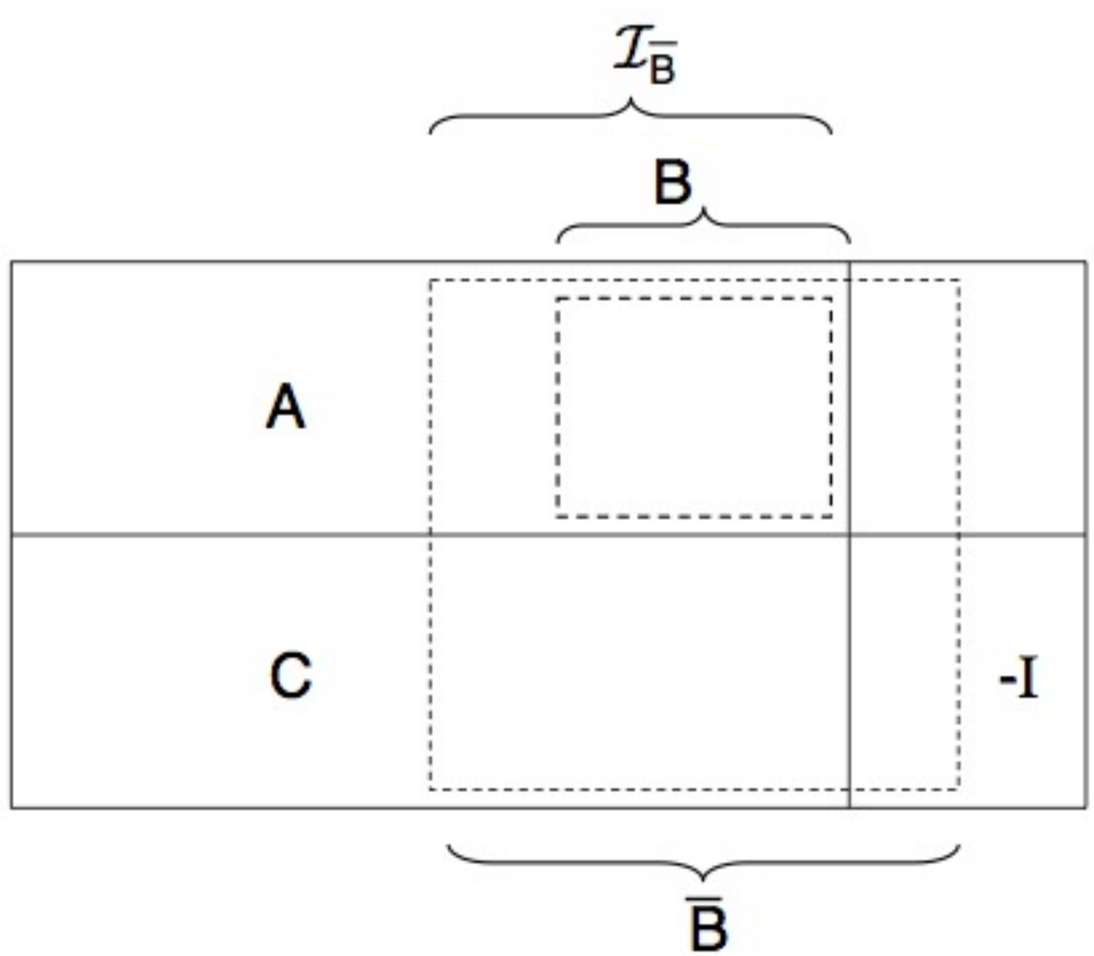
	MIPLIB 3.0	MIPLIB 2003
1-GMI	26.09	18.37
DGL	62.53	-
B-S	76.52	-
L&P	30.21	-
FEAS	43.96	27.64
SPARSE	38.56	29.25
GREEDY	42.62	31.03
RANDOM	41.67	25.78
RANDOM5	48.10	29.75
ALL	52.39	35.51

Avg. Performance on MIPLIB Problems

We never assumed Cx >= 0 defined valid inequalities!

$$L_1 = \min\{cx : Ax = b, x \geq 0\}$$

$$L_i = \min\{\bar{c}\bar{x} : \bar{A}\bar{x} = \bar{b}, \bar{x} \geq 0\}$$
$$= \min\{cx : Ax = b, Cx \geq d, x \geq 0\}$$



$$\bar{B} = \begin{bmatrix} A_{\bar{B}} & 0 \\ C_{\bar{B}} & -I' \end{bmatrix}$$

$A_{\bar{B}}$ has full rank!

Branch and Gather

1. Input: x*

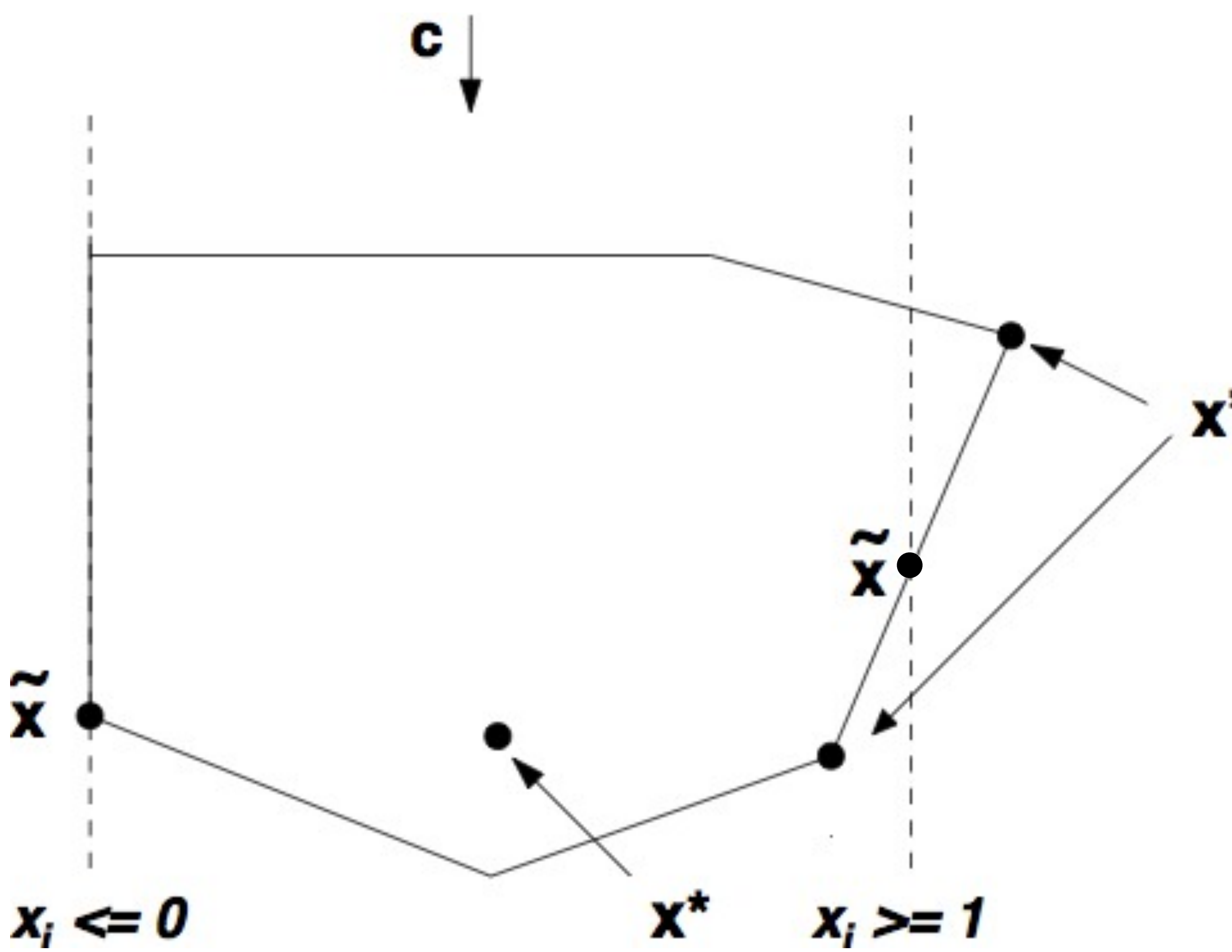
2. Start branch-and-bound algorithm.

3. At each node of the tree, let C represent branching constraints and locally valid cuts.

4. At each node of tree, apply cut procedure.

5. Collect cuts for some number of nodes.

Branch and Gather



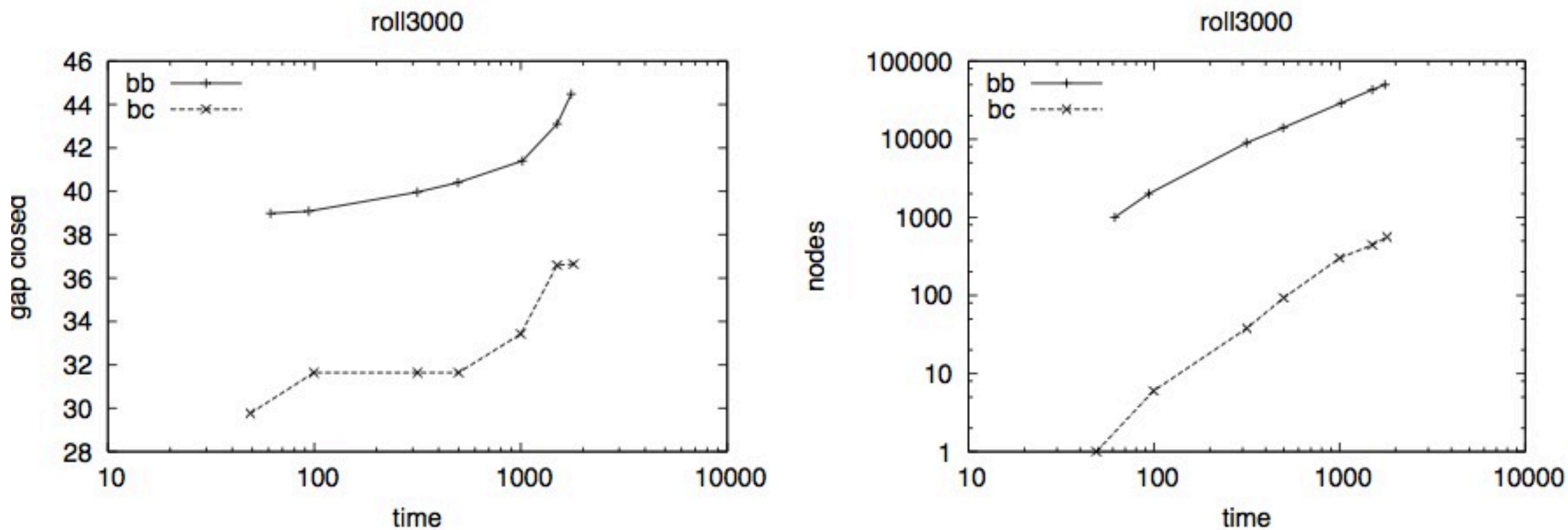
Computational results

	MIPLIB 3.0	MIPLIB 2003
1-GMI	26.09	18.37
DGL	62.53	-
B-S	76.52	-
L&P	30.21	-
ALL	52.39	35.51
ALL+BG5	62.16	39.68
ALL+BG100	64.58	40.82

Avg. Performance on MIPLIB Problems. We use Branch-and-Gather algorithm on root node of each instance.

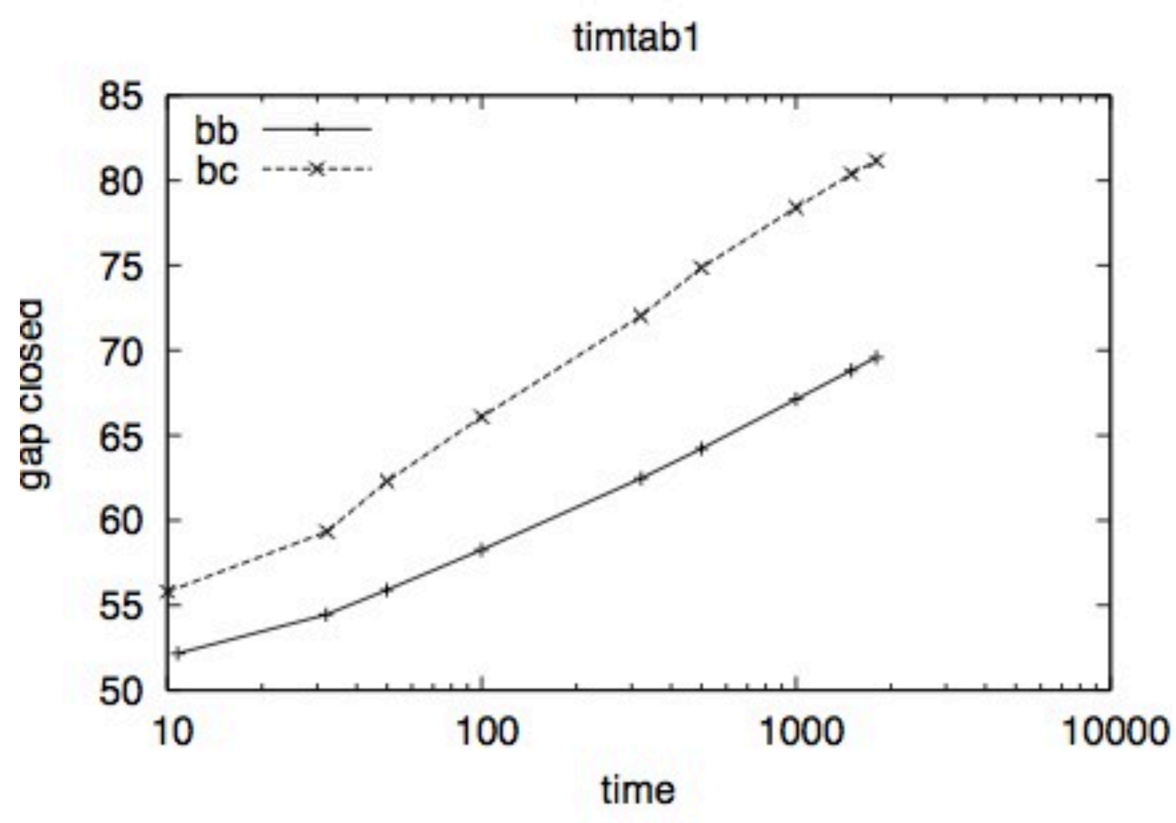
instance	GMI	% gap closed	time DEF	% gap closed	time MIR	% gap split	time split
cap6000	41.65	62.41	14.65	49.77	3,600	65.17	1,260
flugpl	11.74	11.74	0.01	80.23	3,600	100.00	26
gt2	76.50	96.61	0.29	98.38	2,618	98.37	599
p0033	56.82	83.18	0.10	87.42	2,552	87.42	429
pp08a	52.36	93.75	3.12	95.76	3,600	97.03	12,482
qiu	1.76	25.89	3715.85	29.19	3,600	77.51	200,354
set1ch	38.11	83.59	30.66	76.47	3,600	89.74	10,768

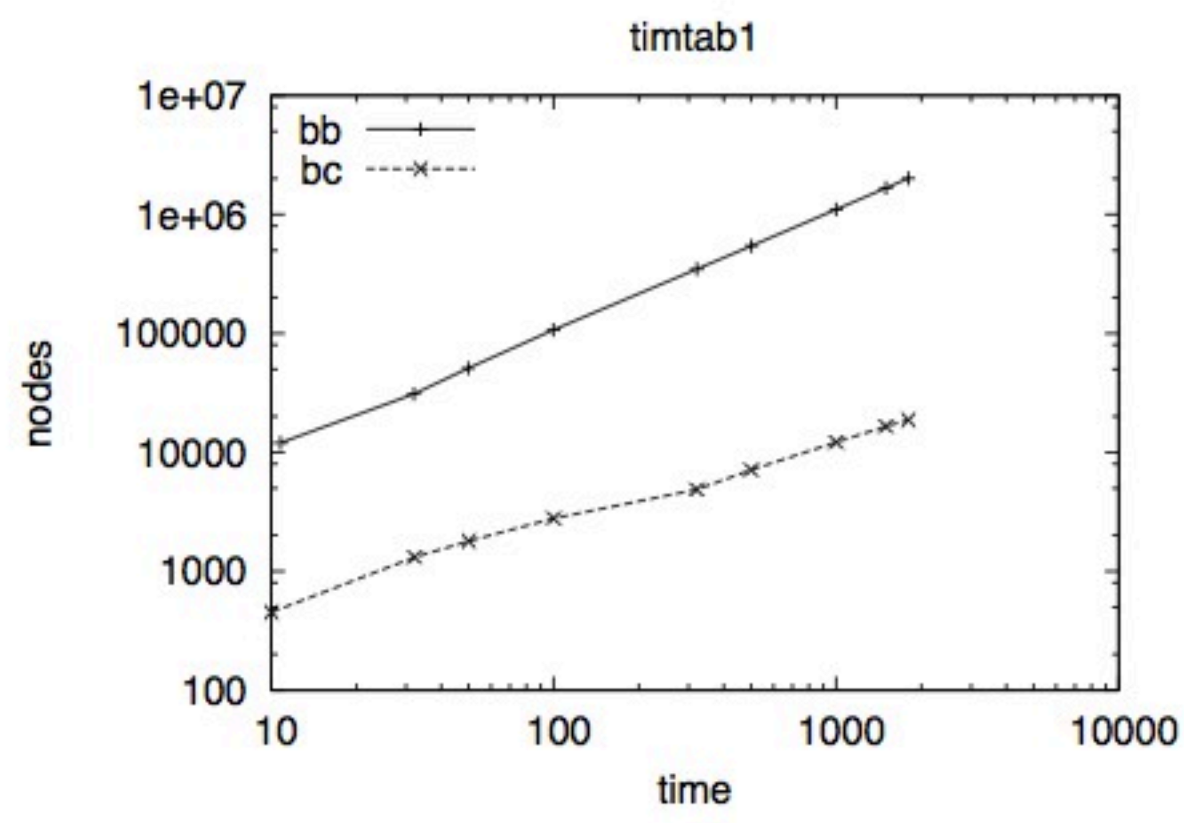
Branch-and-Cut



If we add cuts at every node of the branch-and-bound tree we dont always do better.

Branch-and-Cut

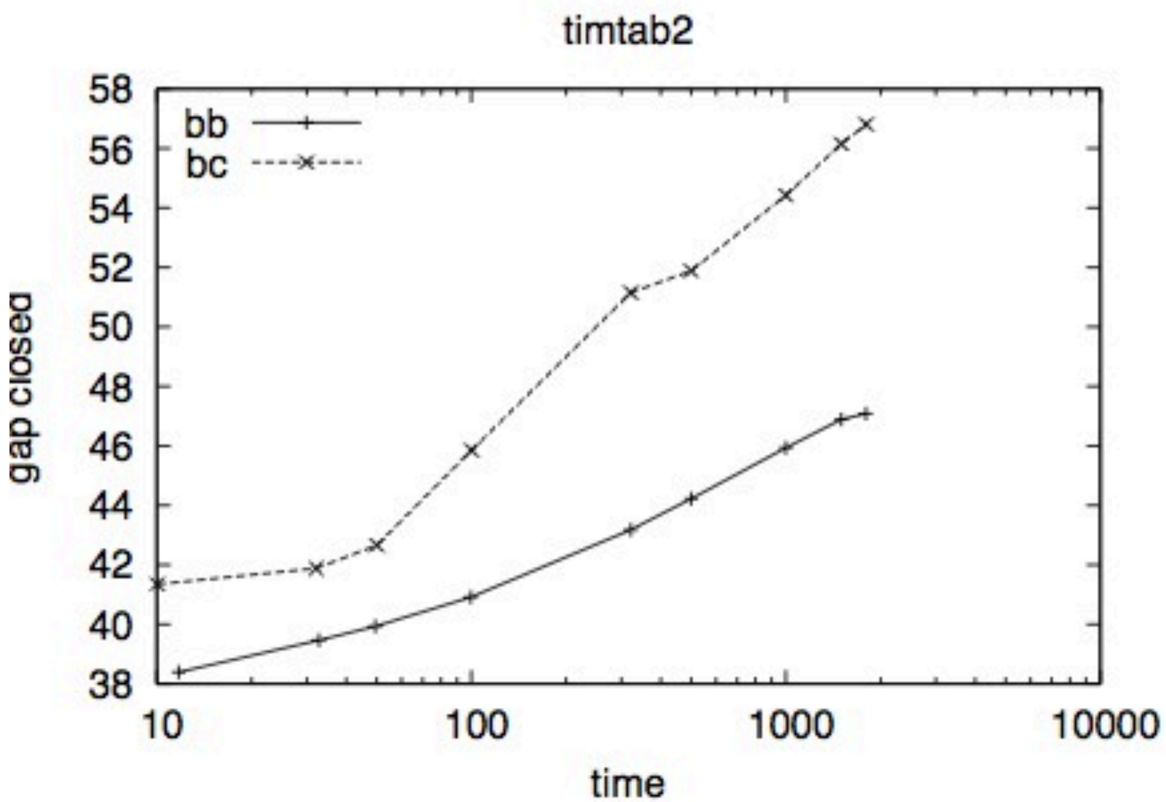


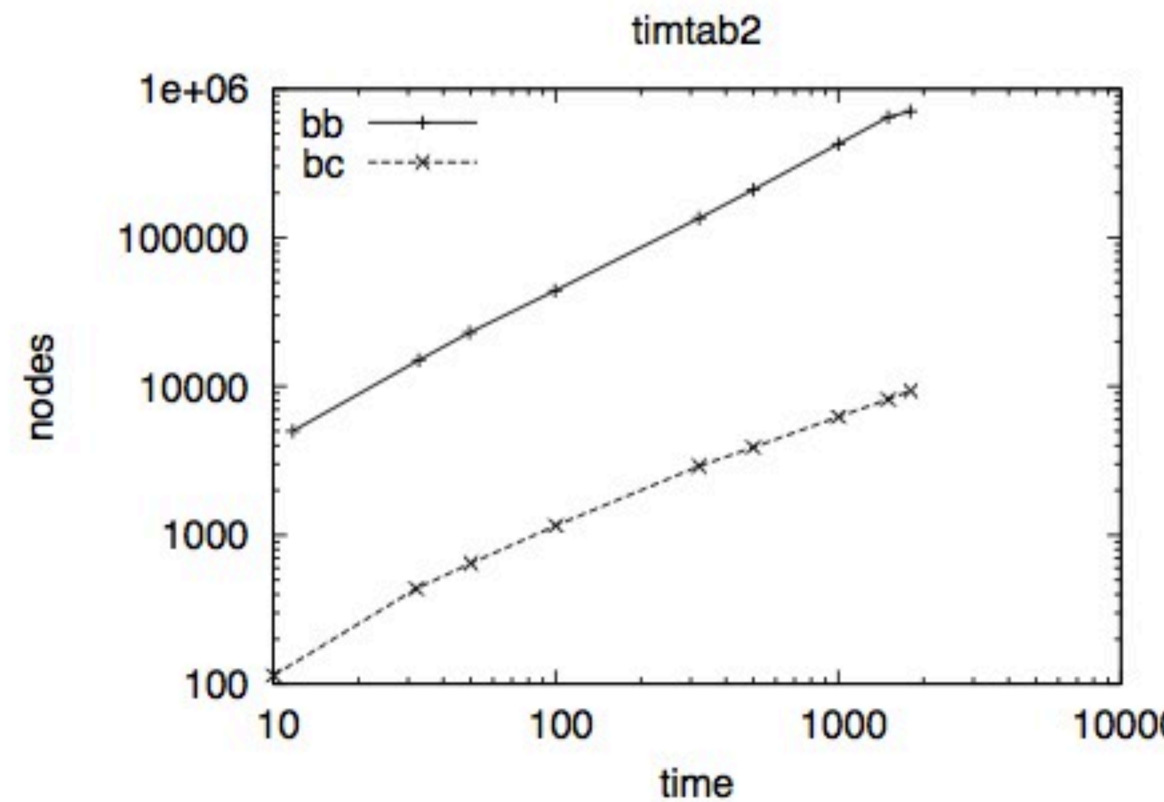


Solved in 26 minutes, using 7500 nodes.

CPLEX takes 1 hour, using 2,245,200 nodes.

Branch-and-Cut





Solved in 92 hours, using 265,900 nodes.

First solved in 114 days of computing time, and 17 million nodes (problem specific cuts + grid-computing), by Buscieck et al. 2009.

Later solved in 22 hours of computing time by a problem-specific branch-and-cut method (see MIPLIB 2003 webpage).

We don't even need a system $Cx >= d$. Just a point x^*.

We don’t even need a system Cx >= d. Just a point x*.

We can use this for non-linear MIPs as well!

We don't even need a system Cx >= d. Just a point x*.

We can use this for non-linear MIPs as well!

ibienst1 -- MINLP from Mittleman's test set.
quadratic objective, linear constraints

Given x* , linear relaxation optimum. Use FEAS to try to separate. Guaranteed to find a feasible solution -- not necessarily basic.

We don't even need a system $Cx >= d$. Just a point x^*.

We can use this for non-linear MIPs as well!

ibienst1 -- MINLP from Mittleman's test set.
quadratic objective, linear constraints

Given x^* , linear relaxation optimum. Use FEAS to try to separate. Guaranteed to find a feasible solution -- not necessarily basic.

In 3 seconds, using our heuristics, we obtain the same bound CPLEX 11,2 obtains in 30 seconds after using 100+ nodes of branching.

CONCLUSIONS

Given a nonbasic solution x* of an m-row system Ax = b, x >= 0 with m+t non-zeros:

1. One can find a basis such that the GMI cuts from this basis are violated by x* if A is sparse and t is small.

3. The time to find these cuts is comparable to generating a round of GMI cuts.

4. Such cuts help to improve the bound.

5. This procedure can be used to obtain GMI cuts from nodes of a branch-and-bound tree or in nonlinear optimization problems.

CONCLUSIONS

Given a nonbasic solution x* of an m-row system Ax = b, x >= 0 with m+t non-zeros:

1. One can find a basis such that the GMI cuts from this basis are violated by x* if A is sparse and t is small.

3. The time to find these cuts is comparable to generating a round of GMI cuts.

4. Such cuts help to improve the bound.

5. This procedure can be used to obtain GMI cuts from nodes of a branch-and-bound tree or in nonlinear optimization problems.

Questions?

Numerically accurate GMI inequalities

Floating point arithmetic: basic properties.