Dantzig-Wolfe Reformulations of general Mixed Integer Programs

Enrico Malaguti *DEIS - University of Bologna*

Joint work with:

Martin Bergner *RWTH - Aachen University*

Alberto Caprara *DEIS - University of Bologna*

Fabio Furini *DOOR - Politecnico di Milano*

Marco Lübbecke *RWTH - Aachen University*

Emiliano Traversi *TU Dortmund*

(Mixed) Integer Linear Programming

$$(P) \qquad \min \boldsymbol{c}'\boldsymbol{x}$$

$$\boldsymbol{D}\boldsymbol{x} \leq \boldsymbol{d}$$
$$\boldsymbol{B}\boldsymbol{x} \leq \boldsymbol{b}$$

$$\boldsymbol{x} \text{ integer}$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}; \boldsymbol{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}; \boldsymbol{d} = \begin{bmatrix} d_1 \\ \vdots \\ d_m \end{bmatrix}; \boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_p \end{bmatrix}; \boldsymbol{D} = \begin{bmatrix} \delta_{11} & & \delta_{1n} \\ & \ddots & \\ \delta_{m1} & & \delta_{m1} \end{bmatrix}; \boldsymbol{B} = \begin{bmatrix} \beta_{11} & & \beta_{1n} \\ & \ddots & \\ \beta_{p1} & & \beta_{p1} \end{bmatrix};$$

Traditional cut generation

lp(P) $\min \boldsymbol{c}'\boldsymbol{x}$

$$\boldsymbol{D}\boldsymbol{x} \leq \boldsymbol{d}$$
$$\boldsymbol{B}\boldsymbol{x} \leq \boldsymbol{b}$$

$$\boldsymbol{x} \geq 0$$

Given a fractional solution to the *lp(P)*, choose a subset of constraints, e.g. $\boldsymbol{B}\boldsymbol{x} \leq \boldsymbol{b}$, and derive valid inequalities implied by these constraints and the integrality of $\boldsymbol{x}$.

Dantzig-Wolfe decomposition (of IPs)

Partial convexification of a subset of the constraints, e.g., $\boldsymbol{Bx} \leq \boldsymbol{b}$.
By defining $Z = \{\boldsymbol{x}: \boldsymbol{Bx} \leq \boldsymbol{b}, \boldsymbol{x}\ integer\}$, we define $DW(lp(P))$ as:

$$\begin{gathered} \min \boldsymbol{c}'\boldsymbol{x} \\ \boldsymbol{Dx} \leq \boldsymbol{d} \\ \boldsymbol{x} \in conv\{Z\} \\ \boldsymbol{x} \geq 0 \end{gathered}$$

This is equivalent to (implicitly) impose all valid inequalities for the subset of constraints $\boldsymbol{Bx} \leq \boldsymbol{b}$.

Traditionally applied to "well" structured problems.
Problem defined by the subset of constraints can be "easily" managed.

Dantzig-Wolfe decomposition

Assuming that $conv\{Z\}$ is bounded and denoting as V the set of its vertices, constraint $\boldsymbol{x} \in conv\{Z\}$ means that $\boldsymbol{x}$ can be expressed as a convex combination of the vertices $z \in V$ (Minkowski theorem):

$$x = \sum_{z \in V} z\lambda^z$$

$$\sum_{z \in V} \lambda^z = 1$$

$$\lambda^z \geq 0 \quad z \in V$$

Thus we have an extended formulation in the exponentially many λ^z, $z \in V$ variables.

Block-diagonal structure

$$\min \boldsymbol{cx}$$

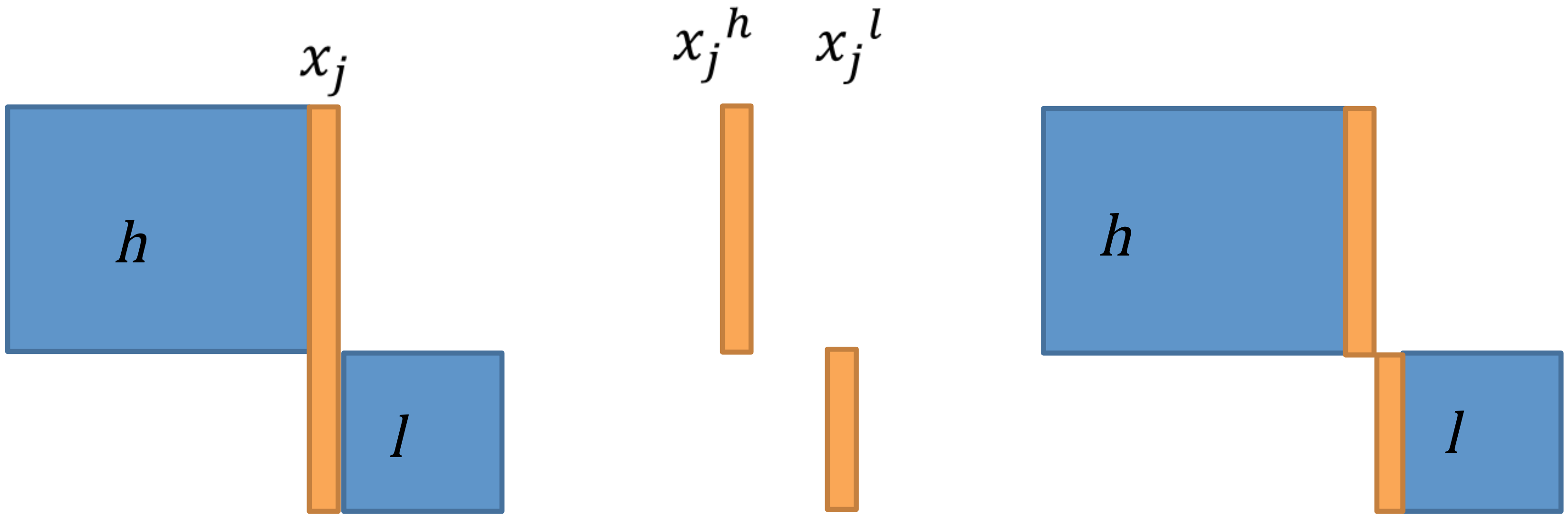
$$\begin{bmatrix} & \boldsymbol{Dx} & \end{bmatrix} \leq \boldsymbol{d}$$

$$\begin{bmatrix} [\boldsymbol{B^1x^1}] & & \\ & \ddots & \\ & & [\boldsymbol{B^kx^k}] \end{bmatrix} \leq \boldsymbol{b}$$

$$\boldsymbol{x} \geq 0$$

Where $\boldsymbol{x} = \boldsymbol{x^1} + \boldsymbol{x^2} + \cdots + \boldsymbol{x^k}$.
Letting $Z^h = \{\boldsymbol{x^h}: \boldsymbol{B^h x^h} \leq \boldsymbol{b^h}, \boldsymbol{x^h}\ integer\}$,
the problem is: $\{\min \boldsymbol{cx}, \boldsymbol{Dx} \leq \boldsymbol{d}, \boldsymbol{x^h} \in \boldsymbol{Z^h}, h = 1, \dots, k\}$.
E.g. Bin Packing Problem.

Partial block-diagonal structure



$$x_j = x_j{}^h \quad x_j = x_j{}^l$$

One additional linking constraint every time a variable is duplicated.

Partial block-diagonal structure

$$\min \boldsymbol{cx}$$

$$\boldsymbol{Dx} \leq \boldsymbol{d}$$

$$x_j = x_j{}^h, \qquad j = 1, \ldots, n, \forall h : j \in G_h$$

$$\boldsymbol{x^h} \in conv\{Z_h\}, \qquad \forall h$$

Assuming that sets $Z^h = \left\{\boldsymbol{x^h}: \boldsymbol{B^h x^h} \leq \boldsymbol{b^h}, \boldsymbol{x^h}\ integer\right\}$ are bounded, and applying Mincowksi theorem,$\boldsymbol{x^h}$ can be expressed as convex combination of the vertices $z^h \in V^h$ of $conv\{Z_h\}$.
G_h - index set of the variables that appear in group h with a non-zero coefficient.

DW reformulation

$$\min \boldsymbol{cx}$$

$$\boldsymbol{Dx} \leq \boldsymbol{d}$$

$$x_j = \sum_{z^h \in V^h}^{p_h} {z_j}^h \lambda^{z^h} \quad j = 1, \ldots, n, \qquad \forall h: j \in G_h$$

$$\sum_{z^h \in V^h}^{p} \lambda^{z^h} = 1 \quad \forall h$$

$$\lambda^{z^h} \geq 0, \qquad \forall h, z^h \in V^h$$

G_h - index set of the variables that appear in group h with a non-zero coefficient.

Column Generation

$$x_j = \sum_{z^h \in V^h}^{p_h} z_j{}^h \lambda^{z^h} \quad j = 1, \dots, n, \forall h: j \in G_h \qquad \longrightarrow \pi_j$$

$$\sum_{z^h \in V^h}^{p} \lambda^{z^h} = 1 \quad \forall h \qquad \longrightarrow \gamma_h$$

We consider the LP relaxation of the reformulated problem and initialize it with a subset of the λ variables.

For every block *h*, new variables with negative reduced cost can be generated through the following (M)ILP, by introducing integer variables $z_j, j \in G_h$:

$$\begin{gathered} max\ \boldsymbol{\pi^h z} \\ \boldsymbol{B^h z} \leq \boldsymbol{b^h} \\ \boldsymbol{z}\ integer \end{gathered}$$

Branch and Price

The solution to the LP relaxation of the DW reformulation can be fractional. Integrality can be enforced via branching.

Branch and Price can be conveniently implemented by branching directly on the original *x* variables.

$$\min \boldsymbol{c}\boldsymbol{x}$$

$$\boldsymbol{D}\boldsymbol{x} \leq \boldsymbol{d}$$

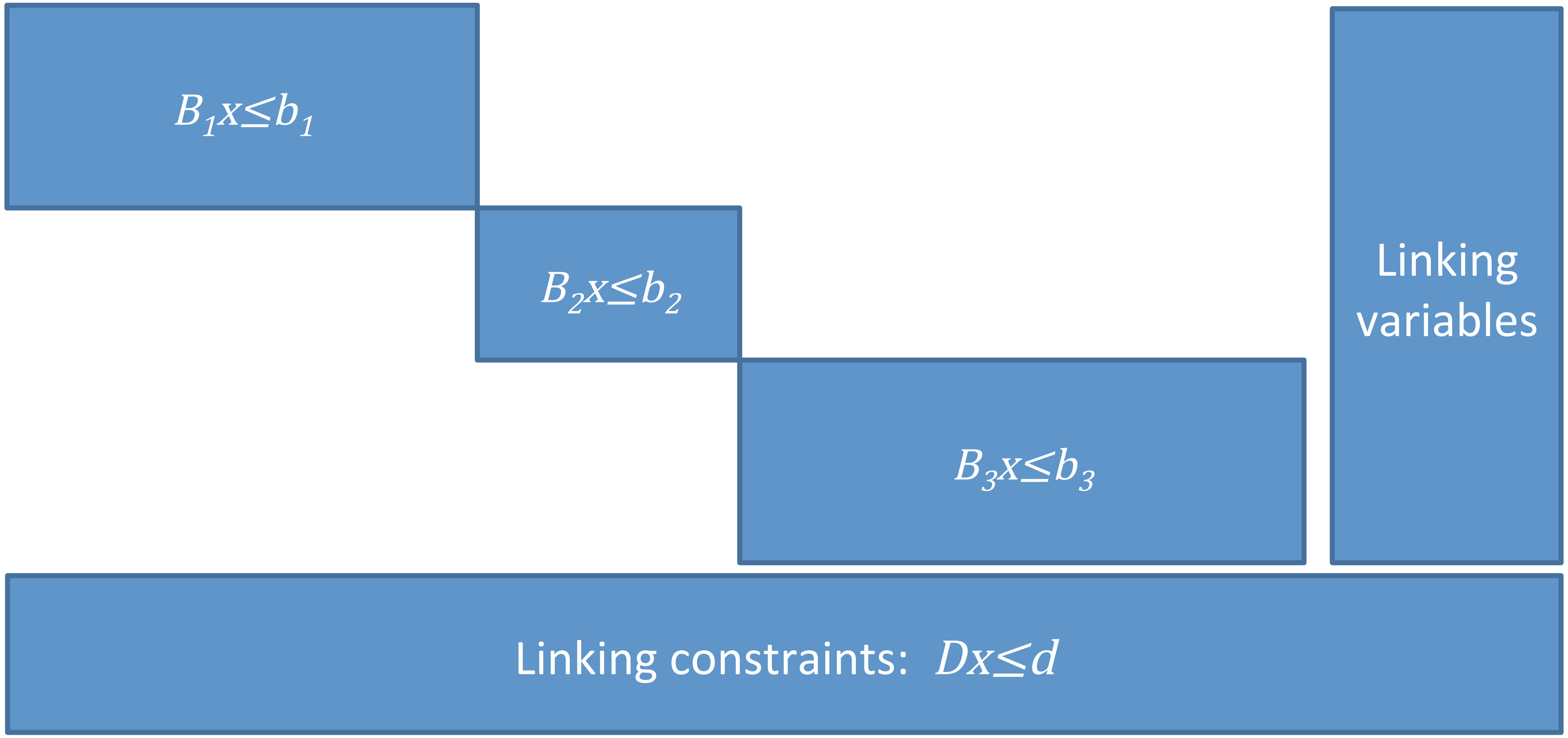
$$x_j = \sum_{z^h \in V^h}^{p_h} z_j{}^h \lambda^{z^h} \quad j = 1, \dots, n, \qquad \forall h{:}\, j \in G_h$$

$$\sum_{z^h \in V^h}^{p} \lambda^{z^h} = 1 \quad \forall h$$

$$\lambda^{z^h} \geq 0, \qquad \forall h, z^h \in V^h$$

How should the constraint Matrix be decomposed?

arrowhead form



How should the constraint Matrix be decomposed?

- How many constraints shold be kept as linking constraints? Which ones?

- How many groups should be considered for the contraints to be convexified?

- How should the constraints be grouped toghether?

- How to reduce the number of linking variables (duplicated variables)?

How should the constraint Matrix be decomposed?

Once the number of groups k for the contraints to be convexified is decided (*this is a big issue!*), the remaining question is:

- which constraint goes in which group?
- which constraints are kept as linking constraints?

-> solve a suited ILP model minimizing the number of duplicated variables;

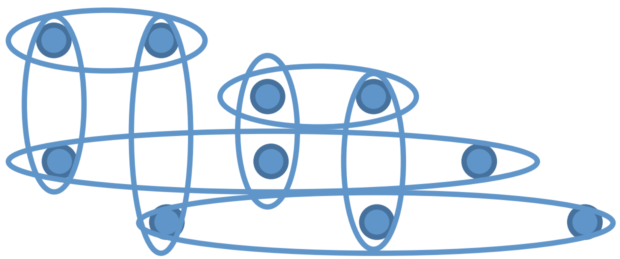
-> solve a partitioning problem for the vertices of a suited (hyper)-graph.

How should the constraint matrix be decomposed?

Define an hyper-graph *H* where:

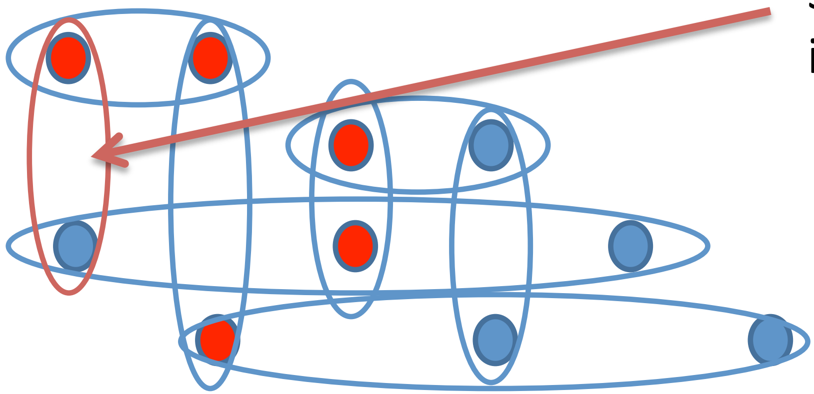
- there is one node for every non-zero coefficient of the constraint matrix;
- there is a weighted hyper-edge connecting all nodes for non-zero entries in a given row, and a weighted hyper-edge connecting all nodes for non-zero entries in a given column.

Solve a min-cut (equi) *k*-partitioning on *H*



How should the constraint Matrix be decomposed?

Solve a min-cut (equi) *k*-partitioning on *H*

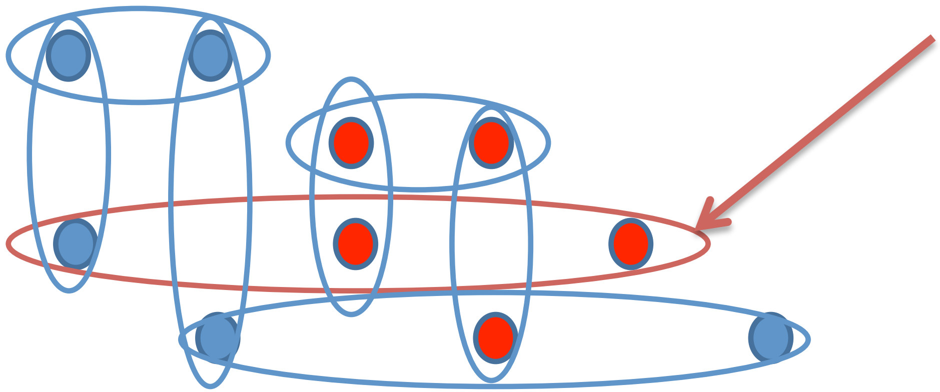


Split variable hyper-edge is a linking variable

2-partitioning

How should the constraint Matrix be decomposed?

Solve a min-cut (equi) *k*-partitioning on *H*



Split constraint hyper-edge is a linking constraint

2-partitioning

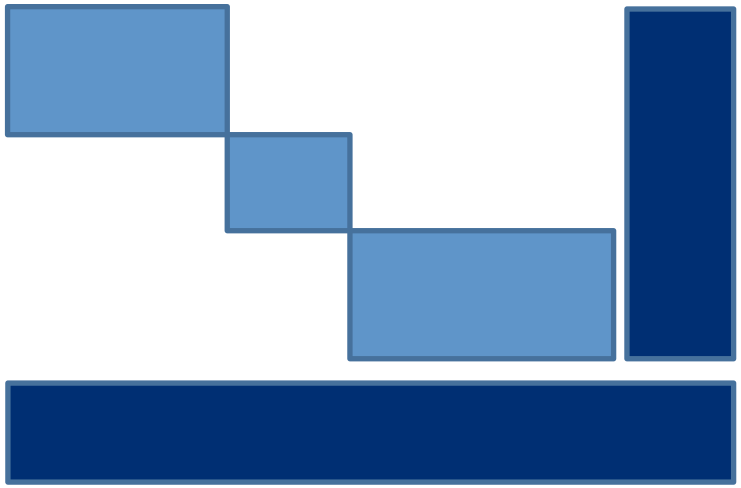
How to compare two decompositions?

Suppose we have several decompositions, which one is the best promising one, i.e., the one that we espect to give the best bound without being “too difficult“ to solve?

Less linking variables, less linking constraints, high density of the blocks:

Border measure *B*

Average block density *D*



An Algorithm for the Automatic DW Decomposition of MIPS

- Construct the hypergraph *H* associated with the constraint matrix;
- Produce several decompositions of the constraint matrix by computing a *min k-equicut* of *H* through a heuristic algorithm (*H-Metis*). Each decomposition is obtained by specifying:

  - the number of blocks *k*;
  - weights for row and column hyper-edges in *H*.
- Compare the decompositions and select the one having the smallest *B.(1-D)* value;
- Duplicate the linking variables and obtain a block-diagonal constraint matrix;
- Apply DW-decomposition.

Computational experiments

The aim of these experiments is to show that DW reformulation of general MIPs can produce strong bounds, comparable with those produced by cutting plane procedures embedded in state-of-the-art MIP solvers.

In particular, the method can be very effective on some instances where cutting planes are weak, that is, it can be used in as an alternative to cutting plane methods.

We considered 23 problems from the miplib 2003 and 16 additional problems from the benchmark subset of the miplib 2010, having:

- less than 20,000 non-zero coefficients;
- density is between 0.05% and 5%;
- At least 20% of integer or binary variables.

Computational experiments

Experiments were performed on one core of a i7 computer with 4 GB ram under linux operating system, with a time limit of 1 hour.

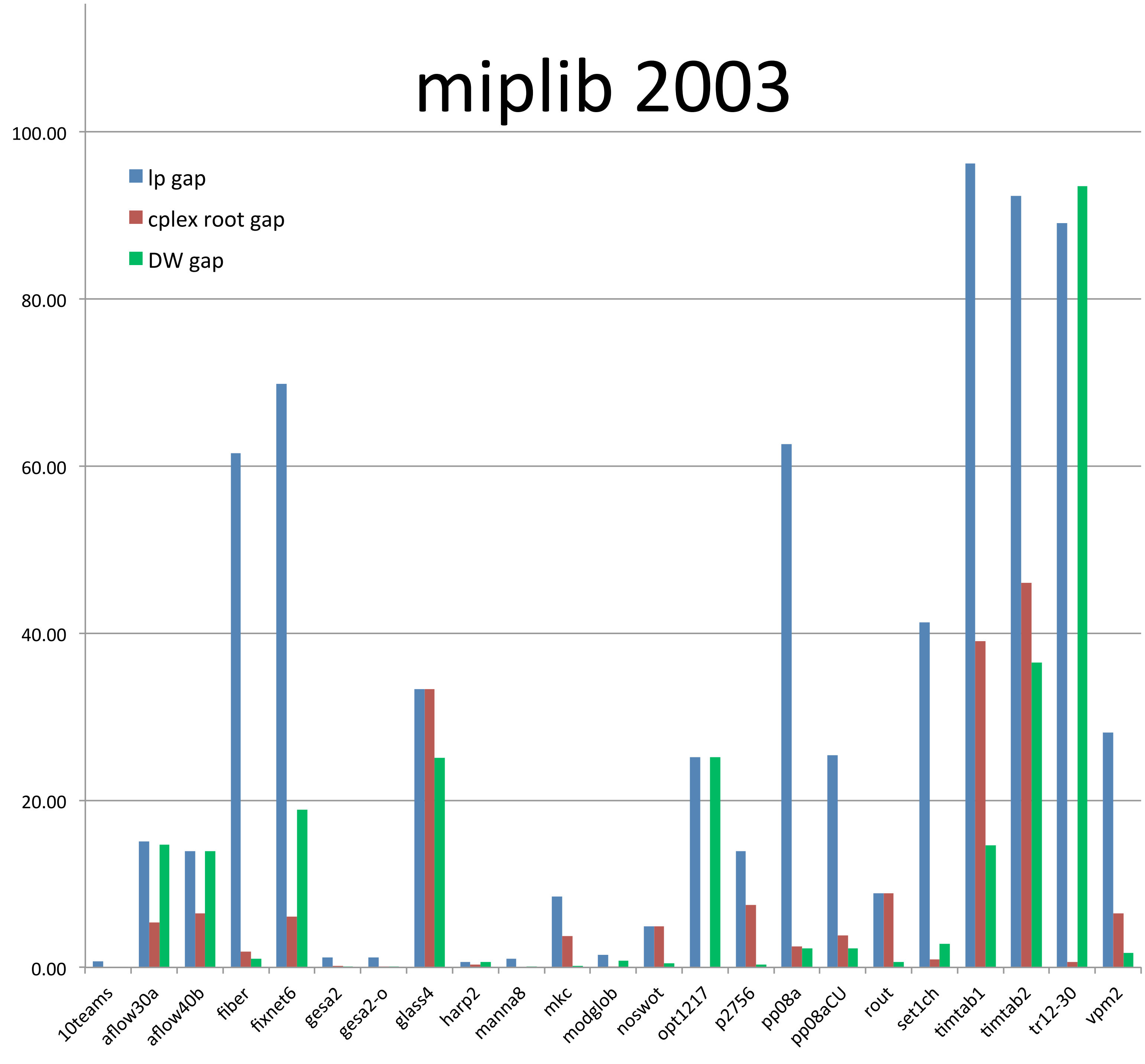
LPs and column generation problems (MIPS) where solved with Cplex12.2

We compare the bound obtained by the DW reformulation (*DW*) and the bound obtained by CPLEX12.2 at the root node of its Branch and Cut (*Root*), that is, after that cuts to strengthen the LP relaxation of the original problem have been added.

When the column generation does not converge within time limit, where are still able to compute a (weaker) valid dual bound via the violation of the slave problems.

Computational experiments - miplib 2003

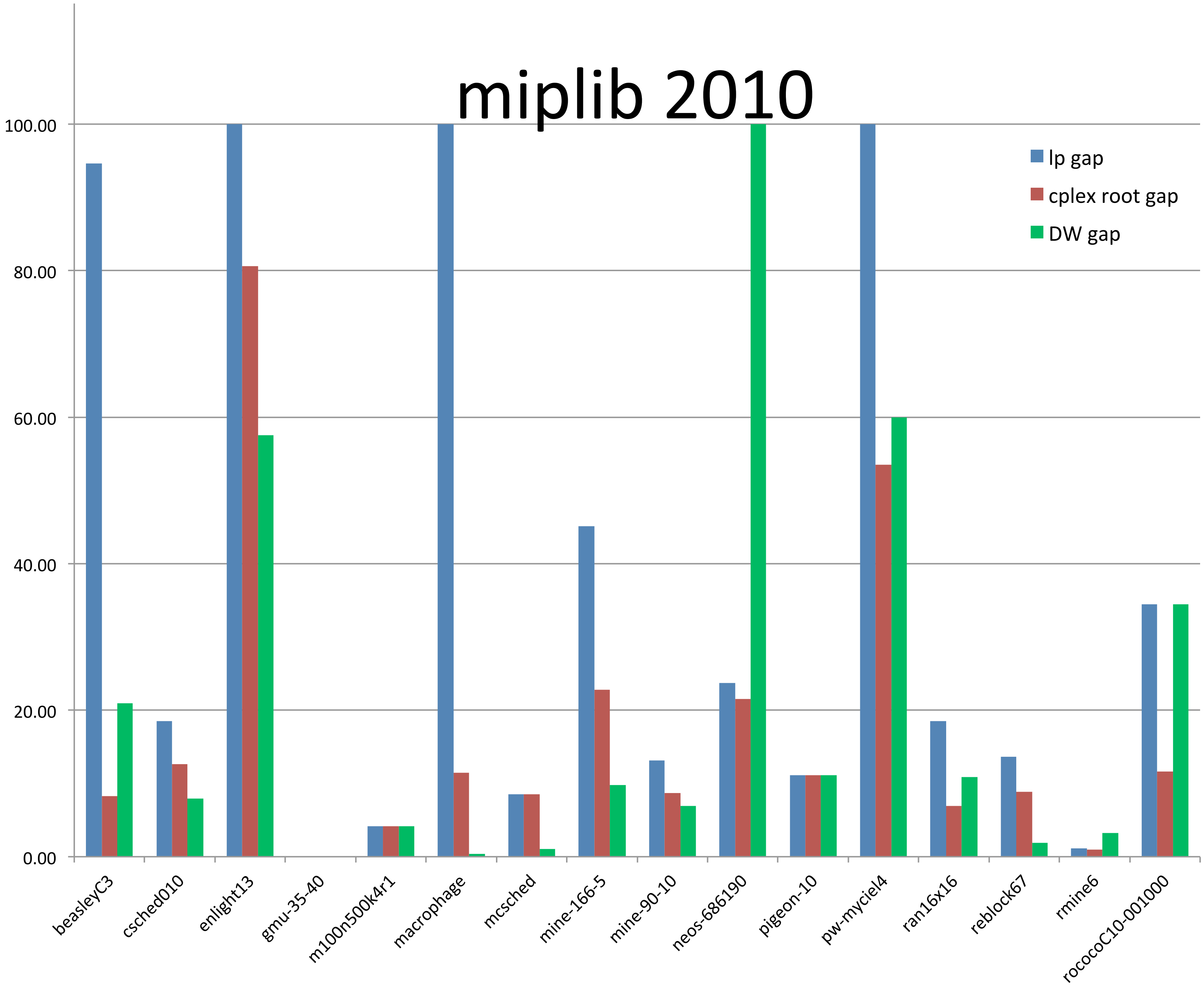
	k	opt	lp gap	cplex root gap	DW gap
10teams	9	924.00	0.76	0.00	0.00
aflow30a	2	1158.00	15.10	5.35	14.71
aflow40b	8	1168.00	13.90	6.47	13.91
fiber	2	405935.00	61.55	1.89	**1.07**
fixnet6	2	3983.00	69.85	6.06	18.89
gesa2	2	25779900.00	1.18	0.21	**0.06**
gesa2-o	2	25779900.00	1.18	0.10	**0.07**
glass4	2	1200010000.00	33.33	33.33	**25.12**
harp2	10	-73899800.00	0.61	0.37	0.66
manna81	2	-13164.00	1.01	0.00	0.08
mkc	2	-563.85	8.51	3.78	**0.16**
modglob	2	20740500.00	1.49	0.14	0.77
noswot	2	-41.00	4.88	4.88	**0.49**
opt1217	10	-16.00	25.13	0.00	25.13
p2756	3	3124.00	13.93	7.47	**0.34**
pp08a	2	7350.00	62.61	2.52	**2.27**
pp08aCUTS	2	7350.00	25.43	3.82	**2.27**
rout	5	1077.56	8.88	8.86	**0.68**
set1ch	2	54537.80	41.31	0.92	2.83
timtab1	2	764772.00	96.25	39.05	**14.60**
timtab2	2	1096560.00	92.38	46.00	**36.50**
tr12-30	2	130596.00	89.12	0.68	93.47
vpm2	2	13.75	28.08	6.44	**1.71**



miplib 2003
100.00
80.00
60.00
40.00
20.00
0.00
lp gap
cplex root gap
DW gap
10teams
aflow30a
aflow40b
fiber
fixnet6
gesa2
gesa2-o
glass4
harp2
manna8
mkc
modglob
noswot
opt1217
p2756
pp08a
pp08aCU
rout
set1ch
timtab1
timtab2
tr12-30
vpm2

Computational experiments - miplib 2010

	k	opt	lp gap	cplex root gap	DW gap
beasleyC3	2	754.00	94.64	8.24	20.95
csched010	2	408.00	18.52	12.62	**7.91**
enlight13	2	71.00	100.00	80.66	**57.55**
gmu-35-40	2	-2406677.75	0.01	0.01	0.03
m100n500k4r1	2	-24.00	4.17	4.17	4.17
macrophage	2	374.00	100.00	11.48	**0.37**
mcsched	2	211913.00	8.56	8.54	**1.03**
mine-166-5	5	-566395707.87	45.09	22.79	**9.78**
mine-90-10	10	-784302337.63	13.12	8.73	**6.90**
neos-686190	3	6730.00	23.70	21.54	100.00
pigeon-10	2	-9000.00	11.11	11.11	11.11
pw-myciel4	2	10.00	100.00	53.54	60.00
ran16x16	2	3823.00	18.48	6.93	10.86
reblock67	2	-34627142.08	13.61	8.87	**1.87**
rmine6	2	-457.19	1.12	0.98	3.22
rococoC10-001000	2	11460.00	34.42	11.62	34.42


miplib 2010
100.00
80.00
60.00
40.00
20.00
0.00
lp gap
cplex root gap
DW gap
beasleyC3
csched010
enlight13
gmu-35-40
m100n500k4r1
macrophage
mcsched
mine-166-5
mine-90-10
neos-686190
pigeon-10
pw-myciel4
ran16x16
reblock67
rmine6
rococoC10-001000

Conclusions

- Proof of concept: DW reformulation of general MIPs che be very effective in producing strong bounds, in particular where Cutting Plane is not effective.
- We provided an authomatic framework for computing the DW reformulation of general MIPs, which does not need ad hoc tuning.
- Future work should consider the automatic detection of those instances where to applying DW reformulation instead of Cutting Planes methods is more effective.