A *Simplicial Decomposition - Branch and Price* for convex quadratic mixed binary problems

Enrico Bettiol *TU Dortmund*

joint work with

Lucas Létocart *LIPN - Université Paris 13*,

Francesco Rinaldi *University of Padova*,

Emiliano Traversi *LIPN - Université Paris 13*

14 September 2020

CTW 2020

technische universität dortmund

Outline

The model

Problem formulation

$$\begin{aligned} \min \; & x^\top Q x + q^\top x = f(x) \\ \text{s. t.} \; & x \in X \\ & x_i \in \{0, 1\} \quad \forall i \in \mathcal{I}. \end{aligned}$$

$$X = \{x \in \mathbb{R}^n \,|\, Ax \geq b, Cx = d, \ell \leq x \leq u\}.$$

$$x, \ell, u \in \mathbb{R}^n,$$
$$Q \in \mathbb{R}^{n \times n},\; q \in \mathbb{R}^n,$$
$$A \in \mathbb{R}^{m_1 \times n},\; b \in \mathbb{R}^{m_1},$$
$$C \in \mathbb{R}^{m_2 \times n},\; d \in \mathbb{R}^{m_2},$$
$$n, m_1, m_2 \in \mathbb{N},$$
$$\mathcal{I} \subseteq \{1, \ldots, n\}.$$

Hypothesis:

$$Q \succeq 0$$

The model

Problem formulation

$$\begin{aligned} \min\ & x^\top Q x + q^\top x = f(x) \\ \text{s. t.}\ & x \in X \\ & x_i \in \{0,1\} \quad \forall i \in \mathcal{I}. \end{aligned}$$

$$X = \{x \in \mathbb{R}^n \mid Ax \geq b, Cx = d, \ell \leq x \leq u\}.$$

$$x, \ell, u \in \mathbb{R}^n,$$
$$Q \in \mathbb{R}^{n \times n},\ q \in \mathbb{R}^n,$$
$$A \in \mathbb{R}^{m_1 \times n},\ b \in \mathbb{R}^{m_1},$$
$$C \in \mathbb{R}^{m_2 \times n},\ d \in \mathbb{R}^{m_2},$$
$$n, m_1, m_2 \in \mathbb{N},$$
$$\mathcal{I} \subseteq \{1, \ldots, n\}.$$

Hypothesis:

$Q \succeq 0$

Further assumptions:

- Q dense.

Features

Our approach

We develop an algorithm based on two main blocks:

- the Branch and Bound (B&B), based on the continuous relaxation of the binary problem;
- the Simplicial Decomposition (SD) algorithm: a *column generation method* to solve the continuous relaxation at each node.

Features

Our approach

We develop an algorithm based on two main blocks:

- the Branch and Bound (B&B), based on the continuous relaxation of the binary problem;
- the Simplicial Decomposition (SD) algorithm: a *column generation method* to solve the continuous relaxation at each node.

Further assumptions

There exists an efficient oracle to solve the linear problem:

$$\begin{aligned} \min\ & c^\top x \\ \text{s. t.}\ & x \in X. \end{aligned}$$

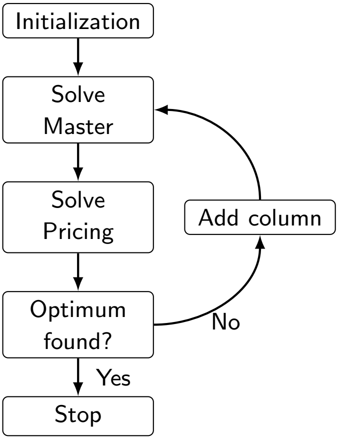
e.g. Shortest Path, Spanning Tree problems.

Column Generation

Column generation methods

- Originally developed for LPs
- Problems with *large* number of variables
- Solve with a subset of variables and add new variables iteratively.

Column Generation

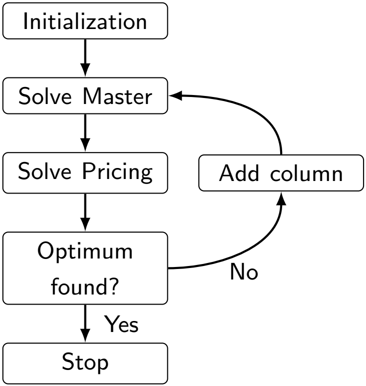


Column generation: the Simplicial Decomposition (SD)

SD: extension of Frank-Wolfe method

- Master problem: original objective function, optimized over a simplex.
- Pricing problem: linearized objective function, original domain.
- All the original constraints are in the pricing.
- Finite convergence.

(Holloway, 1974)

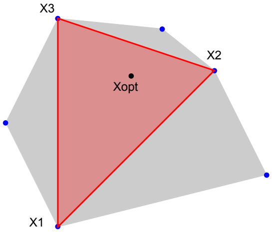


The master problem

At a k-th iteration, k vertices $x_1, \dots x_k \in X$ are provided. $k \ll n$.

Master problem

$$\begin{aligned} \min \quad & x^T Q x + q^T x \\ \text{s. t.} \quad & x = \sum_{i=1}^{k} \lambda_i x_i \\ & \sum_{i=1}^{k} \lambda_i = 1 \\ & \lambda_i \geq 0, \quad \forall i = 1, \dots, k. \end{aligned}$$

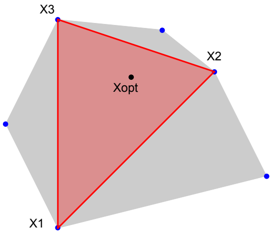


The master problem

At a k-th iteration, k vertices $x_1, \dots x_k \in X$ are provided. $k \ll n$.

Master problem

$$\begin{aligned} \min \quad & \lambda^\top \tilde{Q} \lambda + \tilde{q}^\top \lambda \\ \text{s. t.} \quad & \sum_{i=1}^{k} \lambda_i = 1 \\ & \lambda_i \geq 0, \quad \forall i = 1, \dots, k. \end{aligned}$$



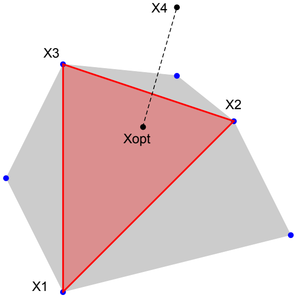
$$B = \{x_1, \dots x_k\}$$
$$\tilde{Q} = B^\top Q B$$
$$\tilde{q} = B^\top q.$$

The pricing problem

The pricing consists in the linearization of the objective function in the optimal point of the master.

Pricing problem

$$\begin{aligned} \min \quad & \nabla f(x_{opt})^\top x \\ s.\ t. \quad & Ax \geq b \\ & Cx = d \\ & \ell \leq x \leq u. \end{aligned}$$



It provides a new feasible extreme point x_{k+1}.

Duality in Quadratic Programming

Primal problem

$$\begin{aligned} \min \quad & \frac{1}{2}x^\top Qx + q^\top x \\ s.\ t. \quad & Ax = b && [\pi] \\ & x \geq 0 && [s] \end{aligned}$$

Duality in Quadratic Programming

Primal problem

$$\begin{aligned} \min \quad & \frac{1}{2}x^\top Q x + q^\top x \\ s.\ t. \quad & Ax = b && [\pi] \\ & x \geq 0 && [s] \end{aligned}$$

Dual problem

$$\begin{aligned} \max \quad & -\frac{1}{2}x^\top Q x + b^\top \pi \\ s.\ t. \quad & -Qx + A^\top \pi \leq q \end{aligned}$$

SD as DWD

(Restricted) Master Problem

$$
\begin{aligned}
\min \quad & \frac{1}{2}x^\top Qx + q^\top x \\
s.\ t. \quad & x - B\lambda = 0 && [\pi] \\
& e^\top \lambda = 1 && [\pi_0] \\
& \lambda \geq 0
\end{aligned}
$$

$$x \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^k$$

$$B = (x_1 \dots x_k) \in \mathbb{R}^{n\times k}$$

$$e = (1, \dots, 1)^\top \in \mathbb{R}^n$$

SD as DWD

(Restricted) Master Problem

$$
\begin{aligned}
\min \quad & \frac{1}{2}x^\top Qx + q^\top x \\
s.\ t. \quad & x - B\lambda = 0 && [\pi] \\
& e^\top \lambda = 1 && [\pi_0] \\
& \lambda \geq 0
\end{aligned}
$$

Dual problem

$$
\begin{aligned}
\max \quad & -\frac{1}{2}x^\top Qx + \pi_0 \\
s.\ t. \quad & -Qx + \pi = q \\
& -\pi^\top x_j + \pi_0 \leq 0, \quad \forall j = 1, \dots, k
\end{aligned}
$$

$$x \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^k$$

$$B = (x_1 \dots x_k) \in \mathbb{R}^{n \times k}$$

$$e = (1, \dots, 1)^\top \in \mathbb{R}^n$$

SD as DWD

(Restricted) Master Problem

$$\begin{aligned} \min \quad & \frac{1}{2}x^\top Q x + q^\top x \\ s.\ t. \quad & x - B\lambda = 0 && [\pi] \\ & e^\top \lambda = 1 && [\pi_0] \\ & \lambda \geq 0 \end{aligned}$$

$$x \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^k$$

$$B = (x_1 \dots x_k) \in \mathbb{R}^{n \times k}$$

$$e = (1, \dots, 1)^\top \in \mathbb{R}^n$$

Dual problem

$$\begin{aligned} \max \quad & -\frac{1}{2}x^\top Q x + \pi_0 \\ s.\ t. \quad & -Qx + \pi = q \\ & -\pi^\top x_j + \pi_0 \leq 0, \quad \forall j = 1, \dots, k \end{aligned}$$

Pricing problem

$$\begin{aligned} \min \quad & {\pi^*}^\top x - \pi_0^* \\ s.\ t. \quad & Ax \geq b \\ & Cx = d \\ & \ell \leq x \leq u \end{aligned}$$

SD as DWD

(Restricted) Master Problem

$$\begin{aligned} \min \quad & \frac{1}{2}x^\top Qx + q^\top x \\ s.\ t. \quad & x - B\lambda = 0 && [\pi] \\ & e^\top \lambda = 1 && [\pi_0] \\ & \lambda \geq 0 \end{aligned}$$

$$x \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^k$$

$$B = (x_1 \dots x_k) \in \mathbb{R}^{n \times k}$$

$$e = (1, \dots, 1)^\top \in \mathbb{R}^n$$

Dual problem

$$\begin{aligned} \max \quad & -\frac{1}{2}x^\top Qx + \pi_0 \\ s.\ t. \quad & -Qx + \pi = q \rightarrow \pi^* = \nabla f(x^*) \\ & -\pi^\top x_j + \pi_0 \leq 0, \quad \forall j = 1, \dots, k \end{aligned}$$

Pricing problem

$$\begin{aligned} \min \quad & {\pi^*}^\top x - \pi_0^* \\ s.\ t. \quad & Ax \geq b \\ & Cx = d \\ & \ell \leq x \leq u \end{aligned}$$

SD as DWD

(Restricted) Master Problem

$$
\begin{aligned}
\min \quad & \frac{1}{2}x^\top Qx + q^\top x & \\
s.\ t. \quad & x - B\lambda = 0 & [\pi] \\
& e^\top \lambda = 1 & [\pi_0] \\
& \lambda \geq 0 &
\end{aligned}
$$

Dual problem

$$
\begin{aligned}
\max \quad & -\frac{1}{2}x^\top Qx + \pi_0 \\
s.\ t. \quad & -Qx + \pi = q \rightarrow \pi^* = \nabla f(x^*) \\
& -\pi^\top x_j + \pi_0 \leq 0, \quad \forall j = 1, \dots, k
\end{aligned}
$$

$$x \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^k$$

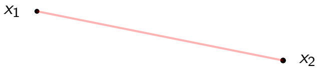
$$B = (x_1 \dots x_k) \in \mathbb{R}^{n \times k}$$

$$e = (1, \dots, 1)^\top \in \mathbb{R}^n$$

Pricing problem

$$
\begin{aligned}
\min \quad & \nabla f(x^*)^\top x + const \\
s.\ t. \quad & Ax \geq b \\
& Cx = d \\
& \ell \leq x \leq u
\end{aligned}
$$

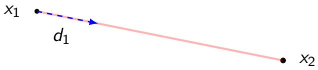
Master: Adapted Conjugate Direction Method (ACDM)



Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \ldots d_k$ ($d_i^\top Q d_j = 0 \; \forall i \neq j$) (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.

Master: Adapted Conjugate Direction Method (ACDM)

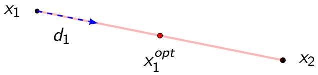


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ ($d_i^\top Q d_j = 0 \, \forall i \neq j$) (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)

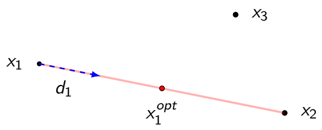


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ $(d_i^\top Q d_j = 0 \, \forall i \neq j)$ (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)

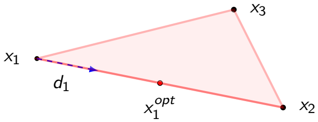


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ $(d_i^\top Q d_j = 0 \, \forall i \neq j)$ (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)

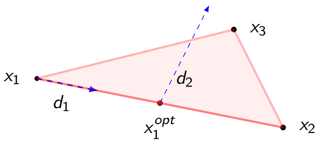


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ ($d_i^\top Q d_j = 0 \; \forall i \neq j$) (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)

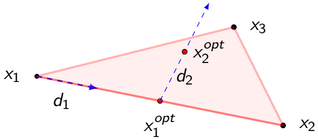


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ $(d_i^\top Q d_j = 0 \; \forall i \neq j)$ (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)

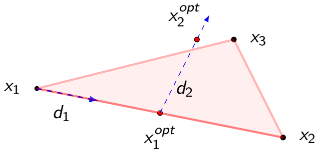


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \ldots d_k$ $(d_i^\top Q d_j = 0 \, \forall i \neq j)$ (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.
- PRO: Most of the times, only one step.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)

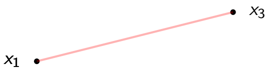


Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ ($d_i^\top Q d_j = 0 \; \forall i \neq j$) (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.
- PRO: Most of the times, only one step.
- CON: If the optimum is on a face, all the directions must be recalculated.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Master: Adapted Conjugate Direction Method (ACDM)



Exploit the structure of the simplices.

- We iteratively update a set of *conjugate directions* $d_1, \dots d_k$ $(d_i^\top Q d_j = 0 \; \forall i \neq j)$ (Reuse the informations from previous iteration).
- At each iteration we generate a new direction.
- Find new optimal point along this direction.
- PRO: Most of the times, only one step.
- CON: If the optimum is on a face, all the directions must be recalculated.

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Results for continuous problems

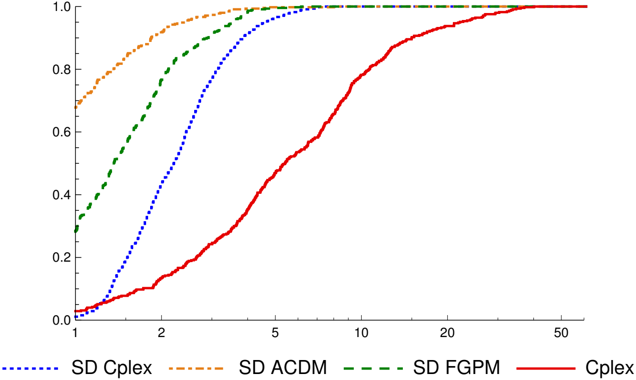


Figure: Performance profile, randomly generated Quadratic Problems

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

Results for continuous problems

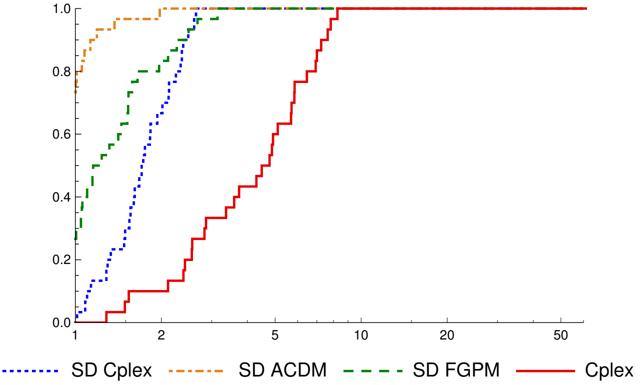


Figure: Performance profile, Quadratic Shortest Path Problems

(E. Bettiol, L. Létocart, F. Rinaldi, E. Traversi, *A conjugate direction based simplicial decomposition framework for solving a specific class of dense convex quadratic programs*, *Computational Optimization and Applications*, 2019)

SD for MIQPs: Branch and bound

Idea: solve the problem via a Branch and Bound (B&B) strategy and solve the continuous relaxation with SD.

Branching strategy: Depth first

- Suitable to take advantage of the SD framework;
- implemented as recursive algorithm;
- limited number of open nodes.

Branching rule: largest fractional value

The variable with the largest fractional part is fixed to 1.
If the optimal point is sparse $\rightarrow$ it is found rapidly.

SD exploitation

Feasibility of the nodes

In a (mixed) binary problem solved with B&B and SD, branching on a fractional variable produces feasible child nodes (if the extreme points are binary).

Indeed, if x_k is fractional, there must be feasible extreme points $\bar{x}^1$, $\bar{x}^2$ with $\bar{x}^1_k = 0$, $\bar{x}^2_k = 1$.

SD exploitation

Feasibility of the nodes

In a (mixed) binary problem solved with B&B and SD, branching on a fractional variable produces feasible child nodes (if the extreme points are binary).

Indeed, if x_k is fractional, there must be feasible extreme points $\bar{x}^1$, $\bar{x}^2$ with $\bar{x}^1_k = 0$, $\bar{x}^2_k = 1$.

Idea: reuse columns of previous iterations.

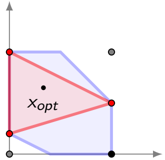
Warmstart

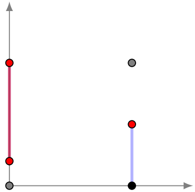
Column reused

At each node:

- the continuous relaxation is found;
- if the optimum is fractional: variable j is set to 1 or 0.
    - Some columns have j^{th} component integer;
    - Store columns x with $x_j = 1$ for the left child;
    - Store columns x with $x_j = 0$ for the right child.

This gives a warmstart at each node.





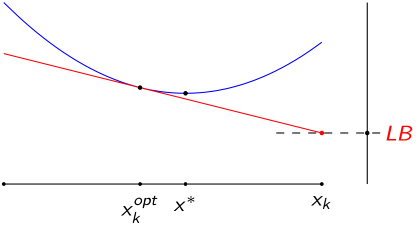
SD features: dualbound and early stopping

Dualbound

The pricing problem provides us with a lower bound (SD - DWD):

$$LB = f(x_k^{opt}) + \nabla f(x_k^{opt})^\top (x_k - x_k^{opt})$$

We use it to prune nodes.



Early stopping

Stop the computation of the pricing before reaching the optimum, but ensure a descent direction: generate a point x_k s. t.

$$\nabla f(x_k^{opt})^\top (x_k - x_k^{opt}) < -\varepsilon < 0.$$

Only if the solution LB is $LB < UB$.

Problem instances

Quadratic Shortest Path Problems (QSPP)

$$
\begin{aligned}
\min \; & f(x) = x^\top Q x + q^\top x \\
\text{s. t.} \; & \sum_{e \in \delta^+(s)} x_e = 1, \\
& \sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = 0, \quad \forall v \neq s, t \\
& \sum_{e \in \delta^-(t)} x_e = 1. \\
& x \in \{0, 1\}^n.
\end{aligned}
$$

$q \in \mathbb{R}^n$ and $Q \in \mathbb{R}^{n \times n}$

s, t: source and termination nodes;

$\delta^+(v)$: outgoing arcs from node v; $\delta^-(v)$: incoming arcs to node v.

Instances: two types of graph

- Grid Shortest Path Problem (QGSPP): 12 instances, square grid of side 10 to 15, # variables from 180 to 420, # constraints from 100 to 225;
- Random Shortest Path Problem (QRSPP): 72 instances, random graphs with 1000, 3000 or 5000 variables, 100 to 300 constraints.

Hardware and software

1. *IBM Ilog Cplex v.12.6.2*.
2. Performed on MAGI cluster at Université Paris 13:
   Processor Intel Xeon E5-2650 v3 (2,3GHz), 64 GB of RAM, one core used.

Results: QGSPP

Algorithm	Time (s)	# Nodes	# Nodes opt	T Master (s)	T Pricing (s)
Cplex	1036.8	163997	94633		
BBSD	575.1	265192	19848	147.3	350.2
BBSD+e	564.6	265196	19845	146.1	341.9
BBSD+col	560.9	265192	19845	142.9	354.8
BBSD+col+e	613.2	265194	19845	155.2	389.2

Table: Average results for the Quadratic Shortest Path Problem, grid graphs.

Algorithm	Time (s)	# Nodes	# Nodes opt	T Master (s)	T Pricing (s)
Cplex	138.8	194	5		
BBSD	12.2	163	31	2.9	8.3
BBSD+e	13.9	163	31	4.6	8.3
BBSD+col	11.1	163	31	2.5	7.2
BBSD+col+e	11.4	163	31	2.9	7.0

Table: Average results for the Quadratic Shortest Path Problem, random graphs.

Quadratic Minimum Spanning Tree Problems (QMSTP)

A formulation

$$\begin{aligned} \min\ & f(x) = x^\top Q x + q^\top x \\ \text{s. t.}\ & \sum_{e \in E(S)} x_e \leq |S| - 1, \quad \forall S \subseteq V, |S| > 2 \\ & \sum_{e \in E} x_e = n - 1. \\ & x \in \{0,1\}^m. \end{aligned}$$

$G(V, E)$ graph with n vertices and m edges;
$q \in \mathbb{R}^m$ and $Q \in \mathbb{R}^{m \times m}$

$E(S)$: set of edges entirely inside a subset of vertices S.

Preliminary results

Instances

Grid graphs: 18 instances, square grids of side 5 to 7.

# Vertices	# Edges	Time (s)		# nodes		# columns	
		SDBB	SDBB+col	SDBB	SDBB+col	SDBB	SDBB+col
25	40	0.27	0.15	1557	1553	13080	6523
36	60	54.35	33.48	190663	189746	1871141	1031868
49	84	5128.80	3276.50	9330521	9276077	124934508	69266606

Table: Quadratic Minimum Spanning Tree Problem.

Conclusions

Summary

- A column generation algorithm has been proposed for convex MBQPs;
- the algorithm performs well with two different sets of random instances;
- the properties of SD can be exploited in the B&B, in particular:
  - the dual bound;
  - the generated columns.

Perspectives

- Including strategies to improve the lower bounds, adding valid cuts;
- finding strategies to project the conjugate directions;
- using heuristics and different branching strategies;
- *projection of columns*.

Thank you for your attention!