Global Solution of General Quadratic Programs

Amélie Lambert

06 april 2016

MINO/COST PhdSchool

1. Presentation of problem (P)
2. Phase 1 : Computing an equivalent quadratic formulation (P^*)
3. Phase 2 : An algorithm for solving (P^*)
4. Some computational results

Presentation of problem (P)

Presentation of problem (P)

$$(P)\begin{cases} \min \quad f_0(x) = \langle Q_0, xx^T \rangle + c_0^T x \\ \text{s.t.} \\ \quad f_r(x) = \langle Q_r, xx^T \rangle + c_r^T x \leq b_r \\ \quad l_i \leq x_i \leq u_i \\ \quad x_i \in \mathbb{N} \quad i \in J \\ \quad x_i \in \mathbb{R} \quad i \in I \backslash J \end{cases}$$

- set I of n variables : integer (set J) or continuous,
- m quadratic constraints.

Presentation of problem (P)

$$(P) \begin{cases} \min & f_0(x) = \langle Q_0, xx^T \rangle + c_0^T x \\ \text{s.t.} & \\ & f_r(x) = \langle Q_r, xx^T \rangle + c_r^T x \leq b_r \\ & l_i \leq x_i \leq u_i \\ & x_i \in \mathbb{N} \quad i \in J \\ & x_i \in \mathbb{R} \quad i \in I \backslash J \end{cases}$$

- set I of n variables : integer (set J) or continuous,
- m quadratic constraints.

Two Difficulties :

1. Non convexity of quadratic functions $f_r(x)$,
2. Integrality of some of the variables x_i.

Outline of the solution method MIQCR

Phase 1 : Computing an equivalent quadratic formulation (P^*) :

i) Add new variables y and non-convex constraints $y = xx^T$,

ii) Rewrite quadratic functions $f_r(x)$ as **convex functions** of x and y.

$\Longrightarrow$ Non convexity only in $y = xx^T$ and $x_i \in \mathbb{N}$

Outline of the solution method MIQCR

Phase 1 : Computing an equivalent quadratic formulation (P^*) :

i) Add new variables y and non-convex constraints $y = xx^T$,

ii) Rewrite quadratic functions $f_r(x)$ as **convex functions** of x and y.

$\Longrightarrow$ Non convexity only in $y = xx^T$ and $x_i \in \mathbb{N}$

Phase 2 : An algorithm for solving (P^*) based on a relaxation $(\overline{P}^*)$:

Outline of the solution method MIQCR

Phase 1 : Computing an equivalent quadratic formulation (P^*) :

i) Add new variables y and non-convex constraints $y = xx^T$,

ii) Rewrite quadratic functions $f_r(x)$ as **convex functions** of x and y.

$\Longrightarrow$ Non convexity only in $y = xx^T$ and $x_i \in \mathbb{N}$

Phase 2 : An algorithm for solving (P^*) based on a relaxation $(\overline{P}^*)$:

- Integer case :
  i) Linearize $y = xx^T$ and relax $x_i \in \mathbb{N} \Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial
  ii) Solve (P^*) : B&B based on $(\overline{P}^*)$.

Outline of the solution method MIQCR

Phase 1 : Computing an equivalent quadratic formulation (P^*) :

i) Add new variables y and non-convex constraints $y = xx^T$,

ii) Rewrite quadratic functions $f_r(x)$ as **convex functions** of x and y.

$\Longrightarrow$ Non convexity only in $y = xx^T$ and $x_i \in \mathbb{N}$

Phase 2 : An algorithm for solving (P^*) based on a relaxation $(\overline{P}^*)$:

- Integer case :
  - i) Linearize $y = xx^T$ and relax $x_i \in \mathbb{N} \Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial
  - ii) Solve (P^*) : B&B based on $(\overline{P}^*)$.
- Mixed case :
  - i) Relax $y = xx^T$ and $x_i \in \mathbb{N} \Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial
  - ii) Solve (P^*) : spatial B&B based on $(\overline{P}^*)$.

Phase 1 : Computing an equivalent quadratic formulation (P^*)

Equivalent formulation of the quadratic functions

- We introduce up to n^2 new variables y that are meant to satisfy :

$$y = xx^T$$

Equivalent formulation of the quadratic functions

- We introduce up to n^2 new variables y that are meant to satisfy :

$$y = xx^T$$

- For each quadratic function $f_r(x)$, we consider $S_r \succeq 0$.

Equivalent formulation of the quadratic functions

- We introduce up to n^2 new variables y that are meant to satisfy :

$$y = xx^T$$

- For each quadratic function $f_r(x)$, we consider $S_r \succeq 0$.
- We replace each quadratic function $f_r(x) = \langle Q_r, xx^T \rangle + c_r^T x$ by :

$$f_{r,S_r}(x, y) = \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, y \rangle$$

Equivalent formulation of the quadratic functions

- We introduce up to n^2 new variables y that are meant to satisfy :

$$y = xx^T$$

- For each quadratic function $f_r(x)$, we consider $S_r \succeq 0$.
- We replace each quadratic function $f_r(x) = \langle Q_r, xx^T \rangle + c_r^T x$ by :

$$f_{r,S_r}(x, y) = \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, y \rangle$$

We have :

- $f_{r,S_r}(x, y) = f_r(x)$ when $y = xx^T$
- $f_{r,S_r}(x, y)$ is convex

An equivalent quadratic reformulation to (P)

$(P_{S_0,\ldots,S_m})$

$$\min \langle S_0, xx^T \rangle + c_0^T x + \langle Q_0 - S_0, y \rangle$$

$$\text{s.t.} \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, y \rangle \leq b_r$$

$$y_{ii} \leq x_i(u_i + l_i) - u_i l_i$$

$$y_{ii} \geq 2u_i x_i - u_i^2$$

$$y_{ii} \geq 2l_i x_i - l_i^2$$

McCormick's for i=j

An equivalent quadratic reformulation to (P)

$(P_{S_0,\dots,S_m})$

$$\min \langle S_0, xx^T \rangle + c_0^T x + \langle Q_0 - S_0, y \rangle$$

$$\text{s.t.} \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, y \rangle \leq b_r$$

$$y_{ii} \leq x_i(u_i + l_i) - u_i l_i$$

$$y_{ii} \geq 2u_i x_i - u_i^2$$

$$y_{ii} \geq 2l_i x_i - l_i^2$$

McCormick's for i=j

$$y_{ij} \leq u_j x_i + l_i x_j - u_j l_i$$

$$y_{ij} \leq u_i x_j + l_j x_i - u_i l_j$$

$$y_{ij} \geq u_j x_i + u_i x_j - u_i u_j$$

$$y_{ij} \geq l_j x_i + l_i x_j - l_i l_j$$

McCormick's for $i \neq j$

An equivalent quadratic reformulation to (P)

$(P_{S_0,\dots,S_m})$

$$\min \langle S_0, xx^T \rangle + c_0^T x + \langle Q_0 - S_0, y \rangle$$

$$\text{s.t.} \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, y \rangle \leq b_r$$

$$y_{ii} \leq x_i(u_i + l_i) - u_i l_i$$

$$y_{ii} \geq 2u_i x_i - u_i^2$$

$$y_{ii} \geq 2l_i x_i - l_i^2$$

McCormick's for i=j

$$y_{ij} \leq u_j x_i + l_i x_j - u_j l_i$$

$$y_{ij} \leq u_i x_j + l_j x_i - u_i l_j$$

$$y_{ij} \geq u_j x_i + u_i x_j - u_i u_j$$

$$y_{ij} \geq l_j x_i + l_i x_j - l_i l_j$$

McCormick's for $i \neq j$

$$y_{ij} = x_i x_j$$

non convex

$$x_i \in \mathbb{N} \quad i \in J$$

$$x_i \in \mathbb{R}$$

A quadratic convex relaxation to (P)

$$(\overline{P}_{s_0,\ldots,s_m})\begin{cases}
\min \langle S_0, xx^T\rangle + c_0^T x + \langle Q_0 - S_0, y\rangle \\
\text{s.t.} \langle S_r, xx^T\rangle + c_r^T x + \langle Q_r - S_r, y\rangle \le b_r \\
y_{ii} \le x_i(u_i + l_i) - u_i l_i \\
y_{ii} \ge 2u_i x_i - u_i^2 \\
y_{ii} \ge 2l_i x_i - l_i^2 \\
y_{ij} \le u_j x_i + l_i x_j - u_j l_i \\
y_{ij} \le u_i x_j + l_j x_i - u_i l_j \\
y_{ij} \ge u_j x_i + u_i x_j - u_i u_j \\
y_{ij} \ge l_j x_i + l_i x_j - l_i l_j \\
x_i \in \mathbb{R}
\end{cases}$$

McCormick's for i=j

McCormick's for $i \neq j$

A quadratic convex relaxation to (P)

$(\overline{P}_{S_0,\ldots,S_m})$

$$\min \langle S_0, xx^T \rangle + c_0^T x + \langle Q_0 - S_0, y \rangle$$

$$\text{s.t.} \langle S_r, xx^T \rangle + c_r^T x + \langle Q_r - S_r, y \rangle \leq b_r$$

$$y_{ii} \leq x_i(u_i + l_i) - u_i l_i$$

$$y_{ii} \geq 2u_i x_i - u_i^2$$

$$y_{ii} \geq 2l_i x_i - l_i^2$$

McCormick's for i=j

$$y_{ij} \leq u_j x_i + l_i x_j - u_j l_i$$

$$y_{ij} \leq u_i x_j + l_j x_i - u_i l_j$$

$$y_{ij} \geq u_j x_i + u_i x_j - u_i u_j$$

$$y_{ij} \geq l_j x_i + l_i x_j - l_i l_j$$

McCormick's for $i \neq j$

$$x_i \in \mathbb{R}$$

Question : Which $S_0^*, \ldots, S_m^*$ maximize the optimal value of $(\overline{P}_{S_0,\ldots,S_m})$?

Which $S_0^*, \ldots, S_m^*$ maximize $(\overline{P}_{S_0,\ldots,S_m})$?

Best $S_0^*, \ldots, S_m^*$ can be deduced from the dual optimal variable of the "Shor+RLT" SDP relaxation of (P) :

$$
(SDP) \begin{cases}
\min f(X,x) = \langle Q_0, X \rangle + c_0^T x & \\
\text{s.t. } \langle Q_r, X \rangle + c_r^T x \leq b_r & \alpha \\
X_{ii} \leq x_i(u_i + l_i) - u_i l_i & \lambda \\
X_{ii} \geq 2u_i x_i - u_i^2 & \\
X_{ii} \geq 2l_i x_i - l_i^2 & \\
X_{ij} \leq u_j x_i + l_i x_j - u_j l_i & \Phi \\
X_{ij} \leq u_i x_j + l_j x_i - u_i l_j & \\
X_{ij} \geq u_j x_i + u_i x_j - u_i u_j & \\
X_{ij} \geq l_j x_i + l_i x_j - l_i l_j & \\
\begin{pmatrix} 1 & x \\ x^T & X \end{pmatrix} \succeq 0 & \\
x_i \in \mathbb{R} \quad X \in \mathcal{S}_n &
\end{cases}
$$

Computation of $(S_0^*, \ldots, S_m^*)$

Theorem :

A best set of semi-definite matrices can be computed as follows :

i) For the objective function :

$$S_0^* = Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + diag(\lambda^*) + \Phi^*$$

ii) For the constraints : $S_r^* = \mathbf{0}_n$ (amounts to linearize the quadratic constraints)

Computation of $(S_0^*, \ldots, S_m^*)$

Theorem :

A best set of semi-definite matrices can be computed as follows :

i) For the objective function :

$$S_0^* = Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + diag(\lambda^*) + \Phi^*$$

ii) For the constraints : $S_r^* = \mathbf{0}_n$ (amounts to linearize the quadratic constraints)

Property :

$$v(\overline{P}_{S_0^*,\ldots,S_m^*}) = v(SDP)$$

Our best quadratic convex relaxation $(\overline{P}^*)$

$$(\overline{P}^*) = (\overline{P}_{S_0^*,\dots,S_m^*}) \begin{cases} \min \langle Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, xx^T \rangle + c_0^T x - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, y \rangle \\ \text{s.t.} \langle Q_r, y \rangle + c_r^T x \leq b_r \\ y_{ii} \leq x_i(u_i + l_i) - u_i l_i \\ y_{ii} \geq 2u_i x_i - u_i^2 \\ y_{ii} \geq 2l_i x_i - {l^2}_i \\ y_{ij} \leq u_j x_i + l_i x_j - u_j l_i \\ y_{ij} \leq u_i x_j + l_j x_i - u_i l_j \\ y_{ij} \geq u_j x_i + u_i x_j - u_i u_j \\ y_{ij} \geq l_j x_i + l_i x_j - l_i l_j \\ x_i \in \mathbb{R} \end{cases}$$

Our best quadratic convex relaxation $(\overline{P}^*)$

$$(\overline{P}^*) = (\overline{P}_{S_0^*,\dots,S_m^*}) \begin{cases} \min \langle Q_0 + \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, xx^T \rangle + c_0^T x - \langle \sum_{r=1}^{m} \alpha_r^* Q_r + \Phi^*, y \rangle \\ \text{s.t.} \langle Q_r, y \rangle + c_r^T x \leq b_r \\ y_{ii} \leq x_i(u_i + l_i) - u_i l_i \\ y_{ii} \geq 2u_i x_i - u_i^2 \\ y_{ii} \geq 2l_i x_i - l^2{}_i \\ y_{ij} \leq u_j x_i + l_i x_j - u_j l_i \\ y_{ij} \leq u_i x_j + l_j x_i - u_i l_j \\ y_{ij} \geq u_j x_i + u_i x_j - u_i u_j \\ y_{ij} \geq l_j x_i + l_i x_j - l_i l_j \\ x_i \in \mathbb{R} \end{cases}$$

$\Longrightarrow$ Solving $(\overline{P}^*)$ can be done in polynomial time

To sum up : Phase 1 of method MIQCR

1) Solve (SDP) and get the optimal dual values $(\alpha^*, \lambda^*, \Phi^*)$

2) Compute $(S_0^*, \ldots, S_m^*)$ as described in the Theorem, and get $(\overline{P}^*)$

To sum up : Phase 1 of method MIQCR

1) Solve (SDP) and get the optimal dual values $(\alpha^*, \lambda^*, \Phi^*)$

2) Compute $(S_0^*, \ldots, S_m^*)$ as described in the Theorem, and get $(\overline{P}^*)$

Difficulty :

(SDP) has a huge number of constraints.

How to solve (*SDP*)

Existing methods for solving (*SDP*) :

- Interior-point methods (implementations CSDP, SeDuMi, SDPA).
  In general, intractable for instances with matrix size larger than 1000, or with more than 10000 constraints.
- Spectral Bundle method (implementations SB)
- Conic Bundle method (callable Conic Bundle library)
- A variety of alternative methods like algorithms based on augmented Lagrangian methods,...

How to solve (*SDP*)

Existing methods for solving (*SDP*) :

- Interior-point methods (implementations CSDP, SeDuMi, SDPA).
  In general, intractable for instances with matrix size larger than 1000, or with more than 10000 constraints.
- Spectral Bundle method (implementations SB)
- Conic Bundle method (callable Conic Bundle library)
- A variety of alternative methods like algorithms based on augmented Lagrangian methods,...

We choose to use a dynamic bundle method to obtain a reasonable solution within short time.

A static bundle method for solving (*SDP*)

Perform Phase 1 amounts to solve (SDP)

We want to solve (SDP) to get its optimal dual values :

$$
(SDP)\begin{cases}
\min f(X,x) = \langle Q_0, X\rangle + c_0^T x & \\
\text{s.t. } \langle Q_r, X\rangle + c_r^T x \leq b_r & \leftarrow \alpha \\
X_{ii} \leq x_i(u_i + l_i) - u_i l_i & \leftarrow \lambda \\
X_{ii} \geq 2u_i x_i - u_i^2 & \leftarrow \lambda \\
X_{ii} \geq 2l_i x_i - l_i^2 & \leftarrow \lambda \\
\begin{pmatrix} 1 & x \\ x^T & X \end{pmatrix} \succeq 0 & \\
x \in \mathbb{R}^n \quad X \in \mathcal{S}_n & \\
X_{ij} \leq u_j x_i + l_i x_j - u_j l_i & \leftarrow \Phi \\
X_{ij} \leq u_i x_j + l_j x_i - u_i l_j & \leftarrow \Phi \\
X_{ij} \geq u_j x_i + u_i x_j - u_i u_j & \leftarrow \Phi \\
X_{ij} \geq l_j x_i + l_i x_j - l_i l_j & \leftarrow \Phi
\end{cases}
$$

Identifying the difficulty

$$
(SDP)\begin{cases}
\min f(X,x) = \langle Q_0, X\rangle + c_0^T x \\
\text{s.t. } \langle Q_r, X\rangle + c_r^T x \le b_r \\
X_{ii} \le x_i(u_i + l_i) - u_i l_i \\
X_{ii} \ge 2u_i x_i - u_i^2 \\
X_{ii} \ge 2l_i x_i - l_i^2 \\
\begin{pmatrix} 1 & x \\ x^T & X \end{pmatrix} \succeq 0 \\
x \in \mathbb{R}^n \quad X \in \mathcal{S}_n
\end{cases}
$$

$(x, X) \in S : \mathcal{O}(n+m)$ constraints.

Identifying the difficulty

$$(SDP)\begin{cases}
\min f(X,x) = \langle Q_0, X\rangle + c_0^T x \\
\text{s.t. } \langle Q_r, X\rangle + c_r^T x \le b_r \\
X_{ii} \le x_i(u_i + l_i) - u_i l_i \\
X_{ii} \ge 2u_i x_i - u_i^2 \\
X_{ii} \ge 2l_i x_i - l_i^2 \\
\begin{pmatrix} 1 & x \\ x^T & X \end{pmatrix} \succeq 0 \\
x \in \mathbb{R}^n \quad X \in \mathcal{S}_n \\
X_{ij} \le u_j x_i + l_i x_j - u_j l_i \\
X_{ij} \le u_i x_j + l_j x_i - u_i l_j \\
X_{ij} \ge u_j x_i + u_i x_j - u_i u_j \\
X_{ij} \ge l_j x_i + l_i x_j - l_i l_j
\end{cases}$$

$(x, X) \in S : \mathcal{O}(n+m)$ constraints.

Difficulty : $\mathcal{O}(n^2)$ McCormick's constraints $(i \neq j)$.

Rewriting (SDP)

We rewrite (SDP) as (SDP_T) using the following notation.

$$(SDP_T)\begin{cases} \max -\langle Q_0, X\rangle - c_0^T x \\ \text{s.t.} (X,x) \in S \\ \quad h_{ij}^t(X,x) \leq 0, \qquad\qquad (i,j,t) \in T \end{cases}$$

where $T = \{(i,j,t) : 1 \leq i < j \leq n, t = 1,\ldots,4\}$, and for all $(i,j,t) \in T$:

$$h_{ij}^t(X,x) = \begin{cases} X_{ij} - u_j x_i - l_i x_j + u_j l_i & t = 1 \\ X_{ij} - u_i x_j - l_j x_i + u_i l_j & t = 2 \\ -X_{ij} + u_j x_i + u_i x_j - u_i u_j & t = 3 \\ -X_{ij} + l_j x_i + l_i x_j - l_i l_j & t = 4 \end{cases}$$

The Lagrangian dual problem

With each constraint $h_{ij}^t(X, x) \leq 0$ we associate a Lagrange multiplier $\Phi_{ij}^t \geq 0$.

The Lagrangian dual problem

With each constraint $h_{ij}^t(X, x) \leq 0$ we associate a Lagrange multiplier $\Phi_{ij}^t \geq 0$.

We now consider the partial Lagrangian :

$$\mathcal{L}_T(X, x, \Phi) = -\langle Q_0, X \rangle - c_0^T x - \langle \Phi, h(X, x) \rangle$$

The Lagrangian dual problem

With each constraint $h_{ij}^t(X, x) \leq 0$ we associate a Lagrange multiplier $\Phi_{ij}^t \geq 0$.

We now consider the partial Lagrangian :

$$\mathcal{L}_T(X, x, \Phi) = -\langle Q_0, X \rangle - c_0^T x - \langle \Phi, h(X, x) \rangle$$

and we obtain the dual functional

$$g_T(\Phi) = \max_{(X,x) \in S} \quad \mathcal{L}_T(X, x, \Phi).$$

The Lagrangian dual problem

With each constraint $h_{ij}^t(X, x) \leq 0$ we associate a Lagrange multiplier $\Phi_{ij}^t \geq 0$.

We now consider the partial Lagrangian :

$$\mathcal{L}_T(X, x, \Phi) = -\langle Q_0, X \rangle - c_0^T x - \langle \Phi, h(X, x) \rangle$$

and we obtain the dual functional

$$g_T(\Phi) = \max_{(X,x) \in S} \quad \mathcal{L}_T(X, x, \Phi).$$

By minimizing $g_T(\Phi)$ we get the partial Lagrangian dual problem (LD_T) associated with (SDP_T), where $v(SDP_T) = v(LD_T)$:

$$(LD_T) = \{\min_{\Phi \geq 0} g_T(\Phi)\}$$

The Lagrangian dual problem

With each constraint $h_{ij}^t(X,x) \leq 0$ we associate a Lagrange multiplier $\Phi_{ij}^t \geq 0$.

We now consider the partial Lagrangian :

$$\mathcal{L}_T(X,x,\Phi) = -\langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle$$

and we obtain the dual functional

$$g_T(\Phi) = \max_{(X,x)\in S} \quad \mathcal{L}_T(X,x,\Phi).$$

By minimizing $g_T(\Phi)$ we get the partial Lagrangian dual problem (LD_T) associated with (SDP_T), where $v(SDP_T) = v(LD_T)$:

$$(LD_T) = \{\min_{\Phi\geq 0} g_T(\Phi)\}$$

We are going to solve (LD_T)

Evaluation of the dual functional

The dual functional is a semi-definite problem :

$$g_T(\Phi) \begin{cases} \max & -\langle Q_0, X \rangle - c_0^T x - \langle \Phi, h(X,x) \rangle \\ \text{s.t.} & (X,x) \in S \end{cases}$$

Evaluation of the dual functional

The dual functional is a semi-definite problem :

$$g_T(\Phi) \begin{cases} \max & -\langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle \\ \text{s.t.} & (X,x) \in S \end{cases}$$

For a given $\bar{\Phi}$, we can easily evaluate $g_T(\bar{\Phi})$ (e.g. by an interior-point method)

Evaluation of the dual functional

The dual functional is a semi-definite problem :

$$g_T(\Phi) \begin{cases} \max & -\langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle \\ \text{s.t.} & (X,x) \in S \end{cases}$$

For a given $\bar{\Phi}$, we can easily evaluate $g_T(\bar{\Phi})$ (e.g. by an interior-point method)

From this evaluation we get :

- A matching pair $((X^*, x^*), \bar{\Phi})$ where $g_T(\bar{\Phi}) = \mathcal{L}_T(X^*, x^*, \bar{\Phi})$
- The value of $g_T(\bar{\Phi})$

Computation of a subgradient

By definition, for a matching pair $((X^*, x^*), \bar{\Phi})$ we have :

$$g_T(\bar{\Phi}) = -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \bar{\Phi}, h(X^*, x^*) \rangle$$

Computation of a subgradient

By definition, for a matching pair $((X^*, x^*), \bar{\Phi})$ we have :

$$g_T(\bar{\Phi}) = -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \bar{\Phi}, h(X^*, x^*) \rangle$$

As $g_T(\Phi)$ is a maximization problem over (X, x), we have for any Φ :

$$g_T(\Phi) \geq -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \Phi, h(X^*, x^*) \rangle$$

Computation of a subgradient

By definition, for a matching pair $((X^*, x^*), \bar{\Phi})$ we have :

$$g_T(\bar{\Phi}) = -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \bar{\Phi}, h(X^*, x^*) \rangle$$

As $g_T(\Phi)$ is a maximization problem over (X, x), we have for any Φ :

$$g_T(\Phi) \geq -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \Phi, h(X^*, x^*) \rangle$$

Subtracting, we get :

$$g_T(\Phi) - g_T(\bar{\Phi}) \geq \langle \bar{\Phi} - \Phi, h(X^*, x^*) \rangle$$

$$g_T(\Phi) \geq g_T(\bar{\Phi}) + \langle \Phi - \bar{\Phi}, -h(X^*, x^*) \rangle$$

Computation of a subgradient

By definition, for a matching pair $((X^*, x^*), \bar{\Phi})$ we have :

$$g_T(\bar{\Phi}) = -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \bar{\Phi}, h(X^*, x^*) \rangle$$

As $g_T(\Phi)$ is a maximization problem over (X, x), we have for any Φ :

$$g_T(\Phi) \geq -\langle Q_0, X^* \rangle - c_0^T x^* - \langle \Phi, h(X^*, x^*) \rangle$$

Subtracting, we get :

$$g_T(\Phi) - g_T(\bar{\Phi}) \geq \langle \bar{\Phi} - \Phi, h(X^*, x^*) \rangle$$

$$g_T(\Phi) \geq g_T(\bar{\Phi}) + \langle \Phi - \bar{\Phi}, -h(X^*, x^*) \rangle$$

Thus, $s = -h(X^*, x^*) \in \partial g_T(\bar{\Phi})$ (i.e. s is a subgradient of g_T at $\bar{\Phi}$)

Which algorithm to solve (SDP_T) ?

We want to solve (SDP_T) or equivalently :

$$(LD_T) = \{\min_{\Phi \geq 0} g_T(\Phi) = \max_{(x,X)\in S} - \langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle\}$$

Which algorithm to solve (SDP_T) ?

We want to solve (SDP_T) or equivalently :

$$(LD_T) = \{\min_{\Phi \geq 0} g_T(\Phi) = \max_{(x,X)\in S} -\langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle\}$$

- For a given $\bar{\Phi}$, we can evaluate $g_T(\bar{\Phi})$, and get a matching pair $((x^*, X^*), \bar{\Phi})$.

Which algorithm to solve (SDP_T) ?

We want to solve (SDP_T) or equivalently :

$$(LD_T) = \{\min_{\Phi \geq 0} g_T(\Phi) = \max_{(x,X)\in S} -\langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle\}$$

- For a given $\bar{\Phi}$, we can evaluate $g_T(\bar{\Phi})$, and get a matching pair $((x^*, X^*), \bar{\Phi})$.
- We also can compute a subgradient $s = -h(X,x) \in \partial g_T(\Phi)$

Which algorithm to solve (SDP_T) ?

We want to solve (SDP_T) or equivalently :

$$(LD_T) = \{\min_{\Phi \geq 0} g_T(\Phi) = \max_{(x,X)\in S} -\langle Q_0, X\rangle - c_0^T x - \langle \Phi, h(X,x)\rangle\}$$

- For a given $\bar{\Phi}$, we can evaluate $g_T(\bar{\Phi})$, and get a matching pair $((x^*, X^*), \bar{\Phi})$.
- We also can compute a subgradient $s = -h(X,x) \in \partial g_T(\Phi)$

$\Longrightarrow$ Solving (LD_T) can be done by a subgradient algorithm here we choose to use a dynamic bundle method.

Principle of the static bundle method (1/3)

Start with some initial Φ^0 (e.g. $\Phi^0 = 0$), solve $g_T(\Phi^0)$, and get a matching pair $((x^0, X^0), \Phi^0)$.

Principle of the static bundle method (1/3)

Start with some initial Φ^0 (e.g. $\Phi^0 = 0$), solve $g_T(\Phi^0)$, and get a matching pair $((x^0, X^0), \Phi^0)$.

The algorithm is iterative and maintains at each iteration :

- a currently best approximation $\hat{\Phi}$ to the minimizer of (LD_T),

Principle of the static bundle method (1/3)

Start with some initial Φ^0 (e.g. $\Phi^0 = 0$), solve $g_T(\Phi^0)$, and get a matching pair $((x^0, X^0), \Phi^0)$.

The algorithm is iterative and maintains at each iteration :

- a currently best approximation $\hat{\Phi}$ to the minimizer of (LD_T),
- a sequence $\{x^i, X^i\}_{i=1,\dots,k}$ and $\hat{\Phi}$ where each $((x^i, X^i), \hat{\Phi})$ is a matching pair,

Principle of the static bundle method (1/3)

Start with some initial Φ^0 (e.g. $\Phi^0 = 0$), solve $g_T(\Phi^0)$, and get a matching pair $((x^0, X^0), \Phi^0)$.

The algorithm is iterative and maintains at each iteration :

- a currently best approximation $\hat{\Phi}$ to the minimizer of (LD_T),
- a sequence $\{x^i, X^i\}_{i=1,\dots,k}$ and $\hat{\Phi}$ where each $((x^i, X^i), \hat{\Phi})$ is a matching pair,
- a sequence $\{s^i \in \partial g_T(\Phi^i)\}_{i=1,\dots,k}$ of the past subgradients,

Principle of the static bundle method (1/3)

Start with some initial Φ^0 (e.g. $\Phi^0 = 0$), solve $g_T(\Phi^0)$, and get a matching pair $((x^0, X^0), \Phi^0)$.

The algorithm is iterative and maintains at each iteration :

- a currently best approximation $\hat{\Phi}$ to the minimizer of (LD_T),
- a sequence $\{x^i, X^i\}_{i=1,\dots,k}$ and $\hat{\Phi}$ where each $((x^i, X^i), \hat{\Phi})$ is a matching pair,
- a sequence $\{s^i \in \partial g_T(\Phi^i)\}_{i=1,\dots,k}$ of the past subgradients,
- a sequence $\{g_T(\Phi^i)\}_{i=1,\dots,k}$ of the past evaluations.

Principle of the static bundle method (2/3)

It combines two ideas to determine a new trial point Φ^{k+1} :

Principle of the static bundle method (2/3)

It combines two ideas to determine a new trial point Φ^{k+1} :

- Approximate the model :

  As $g_T(\Phi) \geq g_T(\Phi^i) + \langle s^i, \Phi - \Phi^i \rangle$, one can construct a piecewise-linear approximation of $g_T(\Phi)$:

$$\hat{g}_{T_k}(\Phi) = \max_{i=1,\ldots,k} g_T(\Phi^i) + \langle s^i, \Phi - \Phi^i \rangle$$

Principle of the static bundle method (2/3)

It combines two ideas to determine a new trial point Φ^{k+1} :

- Approximate the model :

  As $g_T(\Phi) \geq g_T(\Phi^i) + \langle s^i, \Phi - \Phi^i \rangle$, one can construct a piecewise-linear approximation of $g_T(\Phi)$:

$$\hat{g}_{T_k}(\Phi) = \max_{i=1,\ldots,k} g_T(\Phi^i) + \langle s^i, \Phi - \Phi^i \rangle$$

- The proximal point idea :
  The idea is to penalize the shift from the currently best approximation $\hat{\Phi}$ with a term proportional to $|| \Phi - \hat{\Phi} ||^2$

Principle of the static bundle method (3/3)

Difference with a classical subgradient algorithm :

- Store in a bundle past informations to compute the next iterate
- Work with a best currently approximation $\hat{\Phi}$
- Construct $\hat{g}_{T_i}$, a piecewise-linear approximation of g_T
- Determine the next center by optimizing close to the current center and update it if the progress is sufficient.

The static bundle method : algorithm

Given : a convex function $g_T(\hat{\Phi})$ by a first order oracle

1. Let $\hat{\Phi}$, $\Phi^0 = \hat{\Phi}$, $k = 0$.
   Compute $g_T(\hat{\Phi})$, (X^*, x^*) and $s^0 \in \partial g_T(\hat{\Phi})$.

The static bundle method : algorithm

Given : a convex function $g_T(\hat{\Phi})$ by a first order oracle

1. Let $\hat{\Phi}$, $\Phi^0 = \hat{\Phi}$, $k = 0$.
   Compute $g_T(\hat{\Phi})$, (X^*, x^*) and $s^0 \in \partial g_T(\hat{\Phi})$.

2. Determine a cutting plane model $\hat{g}_{T_k}(\Phi)$ based on the previous evaluations of sugradients.

The static bundle method : algorithm

Given : a convex function $g_T(\hat{\Phi})$ by a first order oracle

1. Let $\hat{\Phi}$, $\Phi^0 = \hat{\Phi}$, $k = 0$.
   Compute $g_T(\hat{\Phi})$, (X^*, x^*) and $s^0 \in \partial g_T(\hat{\Phi})$.

2. Determine a cutting plane model $\hat{g}_{T_k}(\Phi)$ based on the previous evaluations of sugradients.

3. Minimize $\hat{g}_{T_k}(\Phi) + \frac{u}{2} \mid\mid \Phi - \hat{\Phi} \mid\mid^2$ and determine a trial point Φ^{k+1}
   The augmented model with weight u is used to optimize in a sphere of radius u (keep candidate close to center $\hat{\Phi}$)

The static bundle method : algorithm

Given : a convex function $g_T(\hat{\Phi})$ by a first order oracle

1. Let $\hat{\Phi}$, $\Phi^0 = \hat{\Phi}$, $k = 0$.
   Compute $g_T(\hat{\Phi})$, (X^*, x^*) and $s^0 \in \partial g_T(\hat{\Phi})$.

2. Determine a cutting plane model $\hat{g}_{T_k}(\Phi)$ based on the previous evaluations of sugradients.

3. Minimize $\hat{g}_{T_k}(\Phi) + \frac{u}{2} \mid\mid \Phi - \hat{\Phi} \mid\mid^2$ and determine a trial point Φ^{k+1}
   The augmented model with weight u is used to optimize in a sphere of radius u (keep candidate close to center $\hat{\Phi}$)

4. Compute $g_T(\Phi^{k+1})$, and $s^{k+1} \in \partial g_T(\Phi^{k+1})$
   compare the actual to predicted progress :
   - SERIOUS STEP : move center $\hat{\Phi} = \Phi^{k+1}$
   - NULL STEP : keep center $\hat{\Phi}$ and improve the model at Φ^{k+1}

The static bundle method : algorithm

Given : a convex function $g_T(\hat{\Phi})$ by a first order oracle

1. Let $\hat{\Phi}$, $\Phi^0 = \hat{\Phi}$, $k = 0$.
   Compute $g_T(\hat{\Phi})$, (X^*, x^*) and $s^0 \in \partial g_T(\hat{\Phi})$.

2. Determine a cutting plane model $\hat{g}_{T_k}(\Phi)$ based on the previous evaluations of sugradients.

3. Minimize $\hat{g}_{T_k}(\Phi) + \frac{u}{2} \mid\mid \Phi - \hat{\Phi} \mid\mid^2$ and determine a trial point Φ^{k+1}
   The augmented model with weight u is used to optimize in a sphere of radius u (keep candidate close to center $\hat{\Phi}$)

4. Compute $g_T(\Phi^{k+1})$, and $s^{k+1} \in \partial g_T(\Phi^{k+1})$
   compare the actual to predicted progress :
   - SERIOUS STEP : move center $\hat{\Phi} = \Phi^{k+1}$
   - NULL STEP : keep center $\hat{\Phi}$ and improve the model at Φ^{k+1}

5. k=k+1 go to Step 2

A dynamic bundle method for solving (SDP_T)

A dynamic bundle method for solving (SDP_T)

- The number of McCormick constraints is $4\binom{n}{2}$.

A dynamic bundle method for solving (SDP_T)

- The number of McCormick constraints is $4\binom{n}{2}$.
- But, we are interested only in the subset of T for which these constraints are likely to be active at the optimum.

A dynamic bundle method for solving (SDP_T)

- The number of McCormick constraints is $4\binom{n}{2}$.
- But, we are interested only in the subset of T for which these constraints are likely to be active at the optimum.
- This set is not known in advance

A dynamic bundle method for solving (SDP_T)

- The number of McCormick constraints is $4\binom{n}{2}$.
- But, we are interested only in the subset of T for which these constraints are likely to be active at the optimum.
- This set is not known in advance

Dynamic approach :

in the course of the algorithm we dynamically add and remove elements in order to identify "important" constraints.

A dynamic bundle method for solving (SDP_T)

- We consider a subset $\mathcal{T} \subseteq T$ and work with the function

$$g_{\mathcal{T}}(\Phi) = \max_{(X,x)\in S} \quad \mathcal{L}_{\mathcal{T}}(X, x, \Phi).$$

A dynamic bundle method for solving (SDP_T)

- We consider a subset $\mathcal{T} \subseteq T$ and work with the function

$$g_{\mathcal{T}}(\Phi) = \max_{(X,x)\in S} \quad \mathcal{L}_{\mathcal{T}}(X, x, \Phi).$$

- We start with $\mathcal{T} = \emptyset$ and after a first function evaluation we separate violated inequalities and add the elements to set $\mathcal{T}$ accordingly.

A dynamic bundle method for solving (SDP_T)

- We consider a subset $\mathcal{T} \subseteq T$ and work with the function

$$g_{\mathcal{T}}(\Phi) = \max_{(X,x)\in S} \quad \mathcal{L}_{\mathcal{T}}(X, x, \Phi).$$

- We start with $\mathcal{T} = \emptyset$ and after a first function evaluation we separate violated inequalities and add the elements to set $\mathcal{T}$ accordingly.
- We keep on updating this set in course of the bundle iterations by :
  - removing elements with associated multiplier close to zero
  - separate newly violated constraints.

A dynamic bundle method for solving (SDP_T) - Experiments

k-cluster problem : finding a subset of k vertices of G such that the induced subgraph is as dense as possible :

$$(KC)\begin{cases} \max f(x) = \sum_{i=1}^{n}\sum_{j=i+1}^{n} \delta_{ij} x_i x_j \\ \text{s.t.} \sum_{i=1}^{n} x_i = k \\ x \in \{0,1\}^n \end{cases}$$

$\delta_{ij} = 1 \Longleftrightarrow$ an edge links vertices i and j, $x_i = 1 \Longleftrightarrow$ vertex i is selected.

A dynamic bundle method for solving (SDP_T) - Experiments

k-cluster problem : finding a subset of k vertices of G such that the induced subgraph is as dense as possible :

$$(KC)\begin{cases} \max f(x) = \sum_{i=1}^{n}\sum_{j=i+1}^{n} \delta_{ij} x_i x_j \\ \text{s.t.} \sum_{i=1}^{n} x_i = k \\ x \in \{0,1\}^n \end{cases}$$

$\delta_{ij} = 1 \Longleftrightarrow$ an edge links vertices i and j, $x_i = 1 \Longleftrightarrow$ vertex i is selected.

Experiments :

- `MIQCR` : We use SB to directly solve (SDP) : Intractable for interior-point solver (CSDP)
- `MIQCR-CB` : At each iteration, we use CSDP to evaluate $g_{\mathcal{T}}(\Phi)$ and the callable conic bundle library to get the next value of Φ.

MIQCR and MIQCR-CB for 45 k-cluster instances with $n = 80$

			MIQCR (with SB)					MIQCR-CB (with CSDP)				
n	*d*	*k*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*
80	25	20	3.39	1434	129	1563	2371	3.00	6	4	10	940
80	25	40	1.00	482	83	565	924	0.72	6	3	9	204
80	25	60	0.30	199	39	238	182	0.07	5	3	8	0
80	50	20	2.34	981	173	1154	3281	2.02	6	5	11	2013
80	50	40	0.76	373	143	516	1809	0.49	6	3	9	426
80	50	60	0.29	178	215	393	1748	0.08	5	3	8	8
80	75	20	1.49	1273	188	1461	3912	1.37	5	6	11	2081
80	75	40	0.59	411	988	1399	19621	0.49	6	6	12	5353
80	75	60	0.20	220	132	352	917	0.04	5	3	8	0
Mean			**1.57**	**617**	**232**	**849**	**3863**	**0.92**	**6**	**4**	**10**	**1225**

Observations :

MIQCR and MIQCR-CB for 45 k-cluster instances with $n = 80$

			MIQCR (with SB)					MIQCR-CB (with CSDP)				
n	*d*	*k*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*
80	25	20	3.39	1434	129	1563	2371	3.00	6	4	10	940
80	25	40	1.00	482	83	565	924	0.72	6	3	9	204
80	25	60	0.30	199	39	238	182	0.07	5	3	8	0
80	50	20	2.34	981	173	1154	3281	2.02	6	5	11	2013
80	50	40	0.76	373	143	516	1809	0.49	6	3	9	426
80	50	60	0.29	178	215	393	1748	0.08	5	3	8	8
80	75	20	1.49	1273	188	1461	3912	1.37	5	6	11	2081
80	75	40	0.59	411	988	1399	19621	0.49	6	6	12	5353
80	75	60	0.20	220	132	352	917	0.04	5	3	8	0
Mean			**1.57**	**617**	**232**	**849**	**3863**	**0.92**	**6**	**4**	**10**	**1225**

Observations :

- The gap obtained with MIQCR-CB is smaller

MIQCR and MIQCR-CB for 45 k-cluster instances with $n = 80$

			MIQCR (with SB)					MIQCR-CB (with CSDP)				
n	*d*	*k*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*
80	25	20	3.39	1434	129	1563	2371	3.00	6	4	10	940
80	25	40	1.00	482	83	565	924	0.72	6	3	9	204
80	25	60	0.30	199	39	238	182	0.07	5	3	8	0
80	50	20	2.34	981	173	1154	3281	2.02	6	5	11	2013
80	50	40	0.76	373	143	516	1809	0.49	6	3	9	426
80	50	60	0.29	178	215	393	1748	0.08	5	3	8	8
80	75	20	1.49	1273	188	1461	3912	1.37	5	6	11	2081
80	75	40	0.59	411	988	1399	19621	0.49	6	6	12	5353
80	75	60	0.20	220	132	352	917	0.04	5	3	8	0
Mean			**1.57**	**617**	**232**	**849**	**3863**	**0.92**	**6**	**4**	**10**	**1225**

Observations :

- The gap obtained with MIQCR-CB is smaller
- The time for Phase 1 is divided by 100 with MIQCR-CB

MIQCR and MIQCR-CB for 45 k-cluster instances with $n = 80$

			MIQCR (with SB)					MIQCR-CB (with CSDP)				
n	*d*	*k*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*
80	25	20	3.39	1434	129	1563	2371	3.00	6	4	10	940
80	25	40	1.00	482	83	565	924	0.72	6	3	9	204
80	25	60	0.30	199	39	238	182	0.07	5	3	8	0
80	50	20	2.34	981	173	1154	3281	2.02	6	5	11	2013
80	50	40	0.76	373	143	516	1809	0.49	6	3	9	426
80	50	60	0.29	178	215	393	1748	0.08	5	3	8	8
80	75	20	1.49	1273	188	1461	3912	1.37	5	6	11	2081
80	75	40	0.59	411	988	1399	19621	0.49	6	6	12	5353
80	75	60	0.20	220	132	352	917	0.04	5	3	8	0
Mean			**1.57**	**617**	**232**	**849**	**3863**	**0.92**	**6**	**4**	**10**	**1225**

Observations :

- The gap obtained with MIQCR-CB is smaller
- The time for Phase 1 is divided by 100 with MIQCR-CB
- The time for Phase 2 is divided by 58 with MIQCR-CB

MIQCR and MIQCR-CB for 45 k-cluster instances with $n = 80$

			MIQCR (with SB)					MIQCR-CB (with CSDP)				
n	*d*	*k*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*	*Gap*	*P1*	*P2*	*Tt*	*Nodes*
80	25	20	3.39	1434	129	1563	2371	3.00	6	4	10	940
80	25	40	1.00	482	83	565	924	0.72	6	3	9	204
80	25	60	0.30	199	39	238	182	0.07	5	3	8	0
80	50	20	2.34	981	173	1154	3281	2.02	6	5	11	2013
80	50	40	0.76	373	143	516	1809	0.49	6	3	9	426
80	50	60	0.29	178	215	393	1748	0.08	5	3	8	8
80	75	20	1.49	1273	188	1461	3912	1.37	5	6	11	2081
80	75	40	0.59	411	988	1399	19621	0.49	6	6	12	5353
80	75	60	0.20	220	132	352	917	0.04	5	3	8	0
Mean			**1.57**	**617**	**232**	**849**	**3863**	**0.92**	**6**	**4**	**10**	**1225**

Observations :

- The gap obtained with MIQCR-CB is smaller
- The time for Phase 1 is divided by 100 with MIQCR-CB
- The time for Phase 2 is divided by 58 with MIQCR-CB
- Consequence : the total time 2 is divided by 85 with MIQCR-CB

How to explain these improvements ?

Expected improvement :

- The time for Phase 1 is reduced with `MIQCR-CB`
  Why ? At each iteration we solve (SDP) which size in in $\mathcal{O}(n)$

How to explain these improvements ?

Expected improvement :

- The time for Phase 1 is reduced with `MIQCR-CB`
  Why ? At each iteration we solve (SDP) which size in in $\mathcal{O}(n)$

Good news :

- The gap obtained with `MIQCR-CB` is smaller
  Why ? The Spectral Bundle method does not reach the exact optimal solution to (SDP)

How to explain these improvements ?

Expected improvement :

- The time for Phase 1 is reduced with `MIQCR-CB`
  Why ? At each iteration we solve (SDP) which size in in $\mathcal{O}(n)$

Good news :

- The gap obtained with `MIQCR-CB` is smaller
  Why ? The Spectral Bundle method does not reach the exact optimal solution to (SDP)

- The time for Phase 2 is reduced with `MIQCR-CB`
  Why ? As we add and and remove dynamically the constraints, by construction Φ^* has less non-zero elements and less additional y variables.

How to explain these improvements ?

Expected improvement :

- The time for Phase 1 is reduced with `MIQCR-CB`
  Why ? At each iteration we solve (SDP) which size in in $\mathcal{O}(n)$

Good news :

- The gap obtained with `MIQCR-CB` is smaller
  Why ? The Spectral Bundle method does not reach the exact optimal solution to (SDP)
- The time for Phase 2 is reduced with `MIQCR-CB`
  Why ? As we add and and remove dynamically the constraints, by construction Φ^* has less non-zero elements and less additional y variables.

Idea : Use a parameter p to control the size of the convex relaxation.

A parameterized dual heuristic for solving (SDP_T)

Recall : Perform Phase 1 amounts to solve (SDP)

We want to solve :

$$
(SDP)\begin{cases}
\min f(X,x) = \langle Q_0, X\rangle + c_0^T x & \\
\text{s.t. } \langle Q_r, X\rangle + c_r^T x \leq b_r & \leftarrow \alpha \\
X_{ii} \leq x_i(u_i + l_i) - u_i l_i & \leftarrow \lambda^1 \\
X_{ii} \geq 2u_i x_i - u_i^2 & \leftarrow \lambda^2 \\
X_{ii} \geq 2l_i x_i - l_i^2 & \leftarrow \lambda^3 \\
\begin{pmatrix} 1 & x \\ x^T & X \end{pmatrix} \succeq 0 & \\
x \in \mathbb{R}^n \quad X \in \mathcal{S}_n & \\
X_{ij} \leq u_j x_i + l_i x_j - u_j l_i & \leftarrow \Phi^1 \\
X_{ij} \leq u_i x_j + l_j x_i - u_i l_j & \leftarrow \Phi^2 \\
X_{ij} \geq u_j x_i + u_i x_j - u_i u_j & \leftarrow \Phi^3 \\
X_{ij} \geq l_j x_i + l_i x_j - l_i l_j & \leftarrow \Phi^4
\end{cases}
$$

The dual of (SDP)

$$
(D)\begin{cases}
\max -\sum_{r=1}^{m} \alpha_r b_r - \langle diag(\lambda^1) + \Phi^1 + \Phi^2, ul^T \rangle - \langle diag(\lambda^2) + \Phi^3, uu^T \rangle - \langle diag(\lambda^3) + \Phi^4, ll^T \rangle \\
\text{s.t. } Q_0 + \sum_{r=1}^{m} \alpha_r Q_r + diag(\lambda) + \Phi \succeq 0 \\
c_0 + \sum_{r=1}^{m} \alpha_r c_r - (diag(\lambda^1 - \lambda^2) + \Phi^1 + \Phi^2 - 2\Phi^3)^T u - (diag(\lambda^1 - \lambda^3) + \Phi^1 + \Phi^2 - 2\Phi^4)^T l \geq 0 \\
\Phi = \Phi^1 + \Phi^2 - \Phi^3 - \Phi^4 \\
\lambda = \lambda^1 - \lambda^2 - \lambda^3 \\
\alpha \in \mathbb{R}^m_+,\ \Phi \in \mathcal{S}_n,\ \Phi^1, \Phi^2, \Phi^3, \Phi^4 \in \mathcal{S}^+_n,\ \lambda^1,\ \lambda^2,\ \lambda^2 \in \mathbb{R}^n_+
\end{cases}
$$

The dual of (SDP)

$$
(D)\begin{cases}
\max -\sum_{r=1}^{m} \alpha_r b_r - \langle diag(\lambda^1) + \Phi^1 + \Phi^2, ul^T \rangle - \langle diag(\lambda^2) + \Phi^3, uu^T \rangle - \langle diag(\lambda^3) + \Phi^4, ll^T \rangle \\
\text{s.t. } Q_0 + \sum_{r=1}^{m} \alpha_r Q_r + diag(\lambda) + \Phi \succeq 0 \\
c_0 + \sum_{r=1}^{m} \alpha_r c_r - (diag(\lambda^1 - \lambda^2) + \Phi^1 + \Phi^2 - 2\Phi^3)^T u - (diag(\lambda^1 - \lambda^3) + \Phi^1 + \Phi^2 - 2\Phi^4)^T l \geq 0 \\
\Phi = \Phi^1 + \Phi^2 - \Phi^3 - \Phi^4 \\
\lambda = \lambda^1 - \lambda^2 - \lambda^3 \\
\alpha \in \mathbb{R}^m_+,\ \Phi \in \mathcal{S}_n,\ \Phi^1, \Phi^2, \Phi^3, \Phi^4 \in \mathcal{S}^+_n,\ \lambda^1,\ \lambda^2,\ \lambda^2 \in \mathbb{R}^n_+
\end{cases}
$$

By the first constraint, we have :

$$Q_0 + \sum_{r=1}^{m} \alpha_r Q_r + diag(\lambda) + \Phi \succeq 0$$

A note on the dual values and convexity of $f_{\alpha,\lambda,\Phi}(x,y)$

Remark 1 :

For any feasible solution of (D), constraint $Q_0 + \sum_{r=1}^{m} \alpha_r Q_r + diag(\lambda) + \Phi \succeq 0$ is satisfied.

$\Longrightarrow$ the associated function $f_{\alpha,\lambda,\Phi}(x,y)$ is convex, i.e. the reformulation is valid even if we are not able to solve (SDP) to optimality.

A note on the dual values and convexity of $f_{\alpha,\lambda,\Phi}(x,y)$

Remark 1 :

For any feasible solution of (D), constraint $Q_0 + \sum_{r=1}^{m} \alpha_r Q_r + diag(\lambda) + \Phi \succeq 0$ is satisfied.

$\Longrightarrow$ the associated function $f_{\alpha,\lambda,\Phi}(x,y)$ is convex, i.e. the reformulation is valid even if we are not able to solve (SDP) to optimality.

Remark 2 :

From any $\Phi \geq 0$, one can build a PSD matrix $Q_0 + \sum_{r=1}^{m} \alpha_r Q_r + diag(\lambda) + \Phi$.

For instance, if $\Phi = 0$ one can take $\lambda = -\lambda_{min}(Q_0 + \sum_{r=1}^{m} \alpha_r Q_r)$.

It means that parameter Φ is not essential for convexification.

Limitation for solving (SDP_T) by dynamic bundle method

- The computation of $(\alpha^*, \lambda^*, \Phi^*)$ with the dynamic bundle method still can require much computational time.

Limitation for solving (SDP_T) by dynamic bundle method

- The computation of $(\alpha^*, \lambda^*, \Phi^*)$ with the dynamic bundle method still can require much computational time.

- **Idea :** consider a relaxation of (SDP_T) : drop some McCormick constraints and compute a dual optimal solution $(\bar{\alpha}, \bar{\lambda}, \bar{\Phi})$ of the reduced problem : some dual variables of $\bar{\Phi}$ are thus non considered.

Limitation for solving (SDP_T) by dynamic bundle method

- The computation of $(\alpha^*, \lambda^*, \Phi^*)$ with the dynamic bundle method still can require much computational time.
- **Idea :** consider a relaxation of (SDP_T) : drop some McCormick constraints and compute a dual optimal solution $(\bar{\alpha}, \bar{\lambda}, \bar{\Phi})$ of the reduced problem : some dual variables of $\bar{\Phi}$ are thus non considered.
- A feasible dual solution to (SDP_T) can be obtained by completing $(\bar{\alpha}, \bar{\lambda}, \bar{\Phi})$ with zeros for the dual variables corresponding to the dropped constraints.

A parameterized dual heuristic for solving (SDP_T)

- we consider a parameter p that is an upper bound on the cardinality of $\mathcal{T}$ ($|\mathcal{T}| \leq p$). (p is the maximum number of McCormick constraints considered in the reduced problem)

A parameterized dual heuristic for solving (SDP_T)

- we consider a parameter p that is an upper bound on the cardinality of $\mathcal{T}$ ($|\mathcal{T}| \leq p$). (p is the maximum number of McCormick constraints considered in the reduced problem)

- The proposed dual heuristic has two extreme cases :
  - if $p = 4\binom{n}{2}$, we solve (SDP_T) and get the associated dual solution.
  - if $p = 0$, we make a single iteration : we get the optimal solution of the reduced problem obtained from (SDP_T) where we drop all McCormick constraints.

A parameterized dual heuristic for solving (SDP_T)

- Finally, our algorithm for solving (SDP_T) returns a solution Φ^* having at most p positive components.

A parameterized dual heuristic for solving (SDP_T)

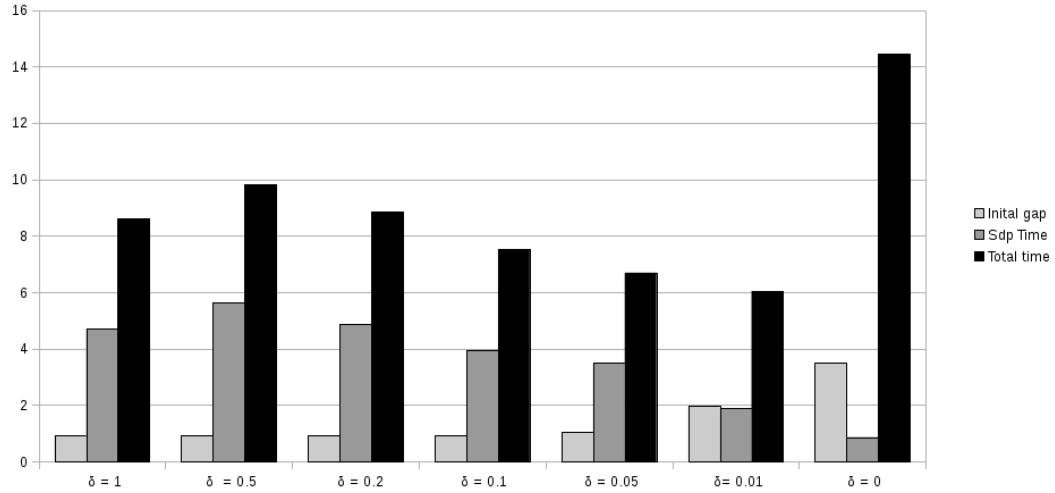
- Finally, our algorithm for solving (SDP_T) returns a solution Φ^* having at most p positive components.
- The number of variables y_{ij} of problem (P^*) is also at most p only
  $\Longrightarrow$ Phase 2 of `MIQCR` can be solved much faster

A parameterized dual heuristic for solving (SDP_T)

- Finally, our algorithm for solving (SDP_T) returns a solution Φ^* having at most p positive components.
- The number of variables y_{ij} of problem (P^*) is also at most p only
  $\Longrightarrow$ Phase 2 of `MIQCR` can be solved much faster
- This parameter p controls the size, and in a sense the tightness, of the semidefinite relaxation used for computing the convex relaxation of method `MIQCR`.

Computational study of the influence of parameter p

δ	p
1	3160
0.5	1580
0.2	632
0.1	316
0.05	158
0.01	31



Observations :

- if the number of active constraints at the optimum is strictly smaller than p, the optimal solution of (SDP) is obtained faster when δ is set to 1.
- Otherwise, the smaller the p, the faster the solution time of (SDP) is computed.

Phase 2 : An algorithm for solving (P^*)

The pure integer case

Outline of the algorithm

Integer case : Build $(\overline{P}^*)$ a quadratic convex relaxation to (P^*) :

i) Linearize $y = xx^T$ (previous talk) and relax $x_i \in \mathbb{N}$
$\Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial

Outline of the algorithm

Integer case : Build $(\overline{P}^*)$ a quadratic convex relaxation to (P^*) :

i) Linearize $y = xx^T$ (previous talk) and relax $x_i \in \mathbb{N}$
$\Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial

ii) Solve (P^*) : B&B based on $(\overline{P}^*)$ (i.e. the continuous relaxation of (P^*)) with a standard solver.

The mixed-integer case

Outline of the algorithm

Mixed case : Build $(\overline{P}^*)$ a quadratic convex relaxation to (P^*) :

i) Relax $y = xx^T$ and $x_i \in \mathbb{N} \Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial

Outline of the algorithm

Mixed case : Build $(\overline{P}^*)$ a quadratic convex relaxation to (P^*) :

i) Relax $y = xx^T$ and $x_i \in \mathbb{N} \Longrightarrow$ Solving $(\overline{P}^*)$ is polynomial

ii) Solve (P^*) : spatial B&B based on $(\overline{P}^*)$.

The variable selection Strategy

$(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$:

1. If $(\bar{x}, \bar{y})$ is feasible for (P^*), the branch is pruned.

The variable selection Strategy

$(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$:

1. If $(\bar{x}, \bar{y})$ is feasible for (P^*), the branch is pruned.
2. Else, if $y \neq xx^T$, take i^* such that :

$$(i^*, j^*) = \underset{\bar{x}_i \bar{x}_j \neq \bar{y}_{ij}}{\operatorname{argmax}} |s^*_{0ij}(\bar{x}_i \bar{x}_j - \bar{y}_{ij})|$$

The variable selection Strategy

$(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$:

1. If $(\bar{x}, \bar{y})$ is feasible for (P^*), the branch is pruned.
2. Else, if $y \neq xx^T$, take i^* such that :

$$(i^*, j^*) = \underset{\bar{x}_i \bar{x}_j \neq \bar{y}_{ij}}{\operatorname{argmax}} |s^*_{0ij}(\bar{x}_i \bar{x}_j - \bar{y}_{ij})|$$

3. Else, if $\bar{x}_{i^*} \notin \mathbb{N}$, take $i^* \in J$, such that $\bar{x}_{i^*} \notin \mathbb{N}$

The branching rules

$(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$ and i^* the selected index :

1. If x_{i^*} is an integer variable :
   Branch 1 : $x_i \leq \lfloor \bar{x}_{i^*} \rfloor$.
   Branch 2 : $x_i \geq \lceil \bar{x}_{i^*} \rceil$.

The branching rules

$(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$ and i^* the selected index :

1. If x_{i^*} is an integer variable :
   Branch 1 : $x_i \leq \lfloor \bar{x}_{i^*} \rfloor$.
   Branch 2 : $x_i \geq \lceil \bar{x}_{i^*} \rceil$.

2. If x_{i^*} is a continuous variable, let γ be a parameter in $[0, 1]$, and let $v_{i^*} = (1 - \gamma)\frac{u_{i^*} + l_{i^*}}{2} + \gamma \bar{x}_{i^*}$:
   Branch 1 : $x_i \leq v_{i^*}$.
   Branch 2 : $x_i \geq v_{i^*}$.

The branching rules

$(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$ and i^* the selected index :

1. If x_{i^*} is an integer variable :
   Branch 1 : $x_i \leq \lfloor \bar{x}_{i^*} \rfloor$.
   Branch 2 : $x_i \geq \lceil \bar{x}_{i^*} \rceil$.

2. If x_{i^*} is a continuous variable, let γ be a parameter in $[0, 1]$, and let $v_{i^*} = (1 - \gamma)\frac{u_{i^*} + l_{i^*}}{2} + \gamma \bar{x}_{i^*}$:
   Branch 1 : $x_i \leq v_{i^*}$.
   Branch 2 : $x_i \geq v_{i^*}$.

We use the depth-first search strategy for selecting the next subproblems.

The upper bounds

Two strategies :

1. $(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$, if $f_r(\bar{x}) \leq b_r$ and $\bar{x}_{i^*} \in \mathbb{N}$:
   $\Longrightarrow \bar{x}$ is a feasible solution to (P^*), and $f_0(\bar{x})$ is an upper bound.

The upper bounds

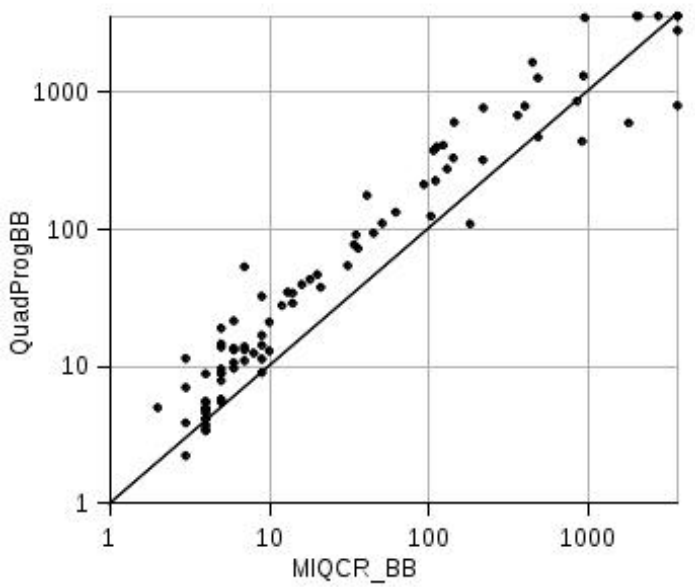
Two strategies :

1. $(\bar{x}, \bar{y})$ current solution of $(\overline{P}^*)$, if $f_r(\bar{x}) \leq b_r$ and $\bar{x}_{i^*} \in \mathbb{N}$:
   $\Longrightarrow \bar{x}$ is a feasible solution to (P^*), and $f_0(\bar{x})$ is an upper bound.

2. Every n nodes, we use the local search of `Cplex 12.6` or `Bonmin` to compute feasible local solutions to (P^*).

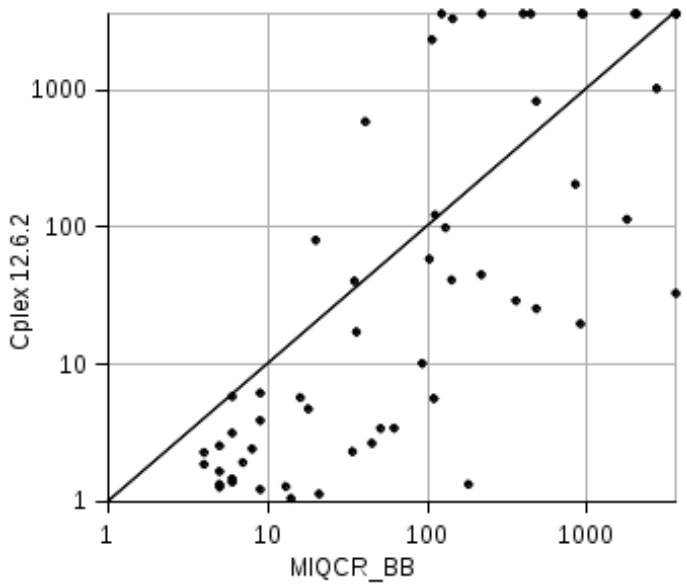
Some computational results

Box constrained *spar* instances with $n = 20$ to 100 ($r = 0, l_i = 0, u_i = 1, J = \emptyset$)



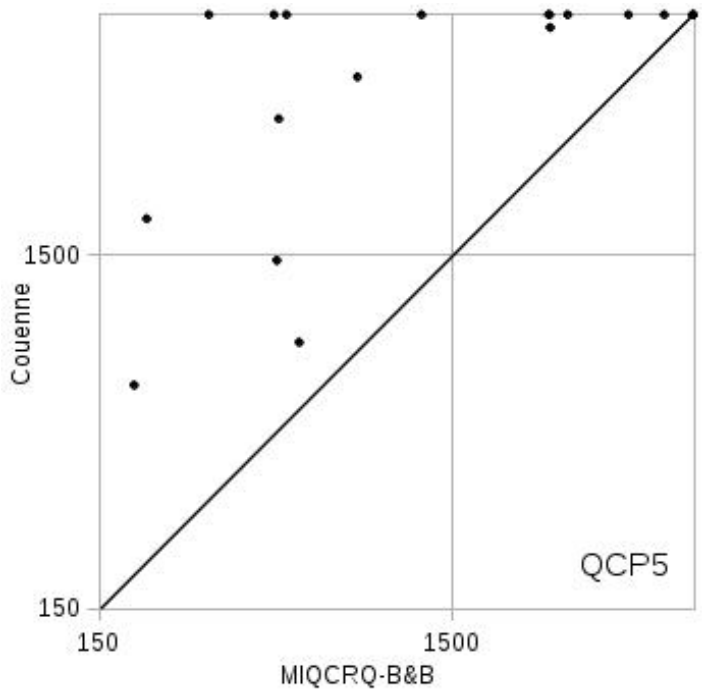
- Scale : 1 to 3600 seconds (time limit)
- `QuadprogBB` : 84 instances in 249 seconds
- `MIQCR-B&B` : 85 instances in 196 seconds

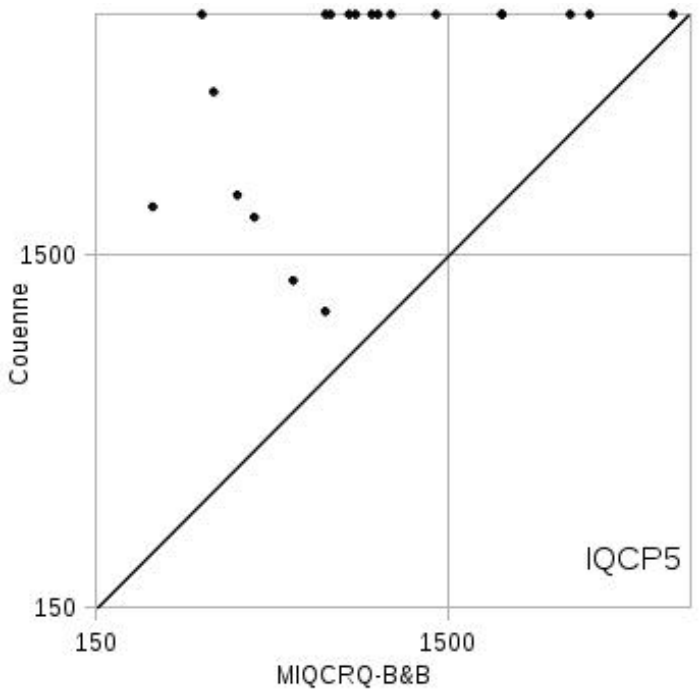
Box constrained *spar* instances with $n = 20$ to 100 ($r = 0, l_i = 0, u_i = 1, J = \emptyset$)



- Scale : 1 to 3600 seconds (time limit)
- `Cplex 12.6.2` 78 instances in 116 seconds (faster on smaller instances)
- `MIQCR-B&B` : 85 instances in 196 seconds

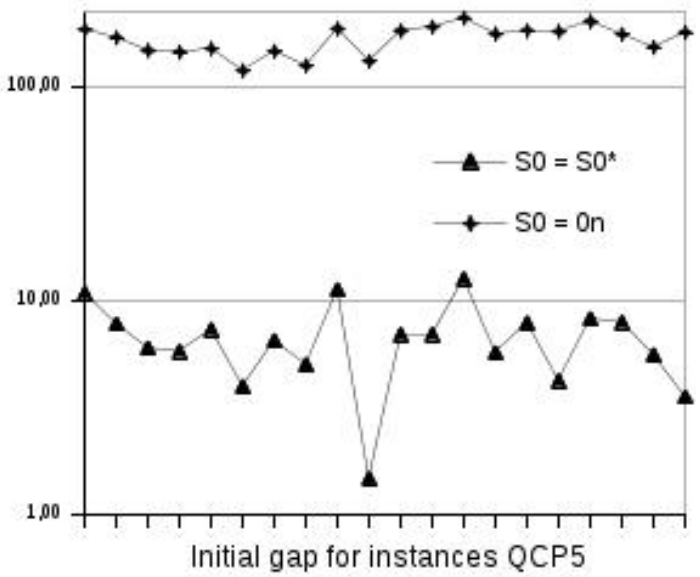
Quadratically constrained quadratic instances with $n = 30$ and 40 ($r = 5$, $l_i = 0$, $u_i = 20$, $J = \emptyset$ or $J = I$)





- Scale : 1 to 7200 seconds (time limit)
- Couenne with up to 30 variables.
- MIQCR-B&B 4 times faster.
- $n = 50$, MIQCR-B&B solves 6 instances of (QCP_5) and 4 of ($IQCP_5$) (among 10)

Quadratically constrained quadratic instances with $n = 30$ and 40 ($r = 5$, $l_i = 0$, $u_i = 20$, $J = \emptyset$ or $J = I$)



$$initial\ gap = |\frac{Opt - Bound}{Opt}| * 100$$

- Initial gap for $(\overline{P}^*)$ $(S_0 = S_0^*)$ 25 times smaller than with the complete linearization $(S_0 = \mathbf{0}_n)$.
- The sizes of the two relaxations are the same
- Price : solution of a SDP (about 10% of the total time on average).

Conclusion

Conclusion

- `MIQCR-CB-B&B` : A solution algorithm for Mixed-integer quadratically constrained problem in 2 phases

Conclusion

- `MIQCR-CB-B&B` : A solution algorithm for Mixed-integer quadratically constrained problem in 2 phases

- Our whole method can be viewed as an improvement of classic spatial branch-and-bound based on complete linearization ($S_r = \mathbf{0_n}$).

Conclusion

- `MIQCR-CB-B&B` : A solution algorithm for Mixed-integer quadratically constrained problem in 2 phases
- Our whole method can be viewed as an improvement of classic spatial branch-and-bound based on complete linearization ($S_r = \mathbf{0_n}$).
- Phase 1 : A dual heuristic for parameterized the tightness and the size of the reformulation

Conclusion

- `MIQCR-CB-B&B` : A solution algorithm for Mixed-integer quadratically constrained problem in 2 phases
- Our whole method can be viewed as an improvement of classic spatial branch-and-bound based on complete linearization ($S_r = \mathbf{0_n}$).
- Phase 1 : A dual heuristic for parameterized the tightness and the size of the reformulation
- Phase 2 : A B&B algorithm specialized for the class of problem (Integer or mixed)

Conclusion

- `MIQCR-CB-B&B` : A solution algorithm for Mixed-integer quadratically constrained problem in 2 phases
- Our whole method can be viewed as an improvement of classic spatial branch-and-bound based on complete linearization ($S_r = \mathbf{0_n}$).
- Phase 1 : A dual heuristic for parameterized the tightness and the size of the reformulation
- Phase 2 : A B&B algorithm specialized for the class of problem (Integer or mixed)
- Encouraging computational results.