Old and new algorithms for polynomial unconstrained optimization problems in binary variables

Yves Crama
HEC Management School, University of Liège, Belgium

Numa Research Day 2023, Ghent

Definitions

Pseudo-Boolean functions

A pseudo-Boolean function is a mapping $f : \{0,1\}^n \to \mathbb{R}$, that is, a real-valued function of $0-1$ variables.

Multilinear polynomials

Every pseudo-Boolean function can be represented – in a unique way – as a *multilinear polynomial* in its variables.

$$f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

(Note: $x^2 = x$.)

Example:

$f = 4 - 9x_1 - 5x_2 - 2x_3 + 13x_1x_2 + 13x_1x_3 + 6x_2x_3x_4 - 13x_1x_2x_3x_4$

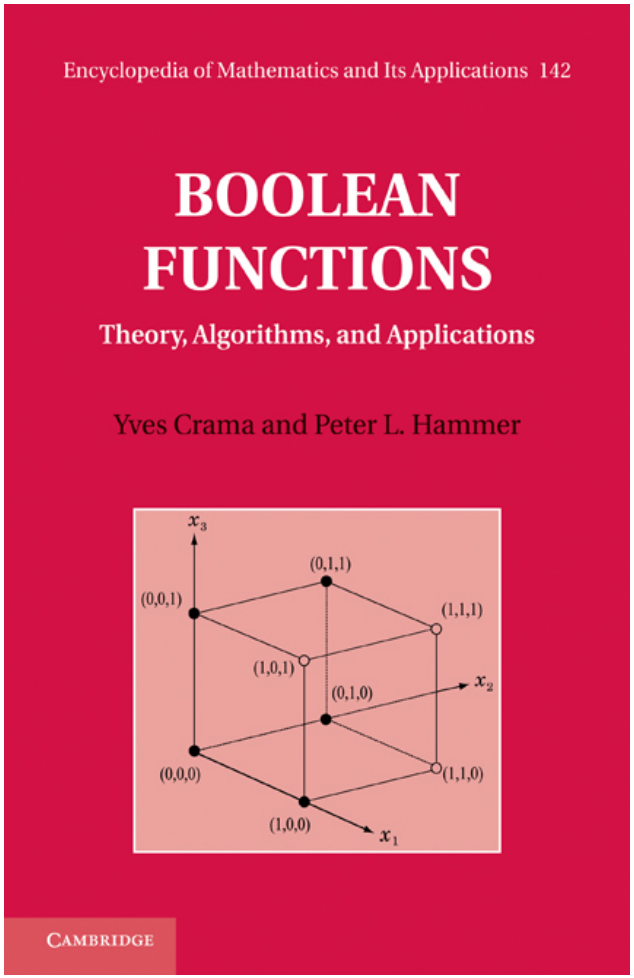
Some advertising...

Connections with Boolean functions:

BOOLEAN FUNCTIONS
Theory, Algorithms, and Applications

Yves CRAMA and Peter L. HAMMER
Cambridge University Press, 2011
710 pages

with contributions by C. Benzaken, E. Boros, N. Brauner, M.C. Golumbic, V. Gurvich, L. Hellerstein, T. Ibaraki, A. Kogan, K. Makino, B. Simeone



Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Numerous applications in various fields, e.g,

- Satisfiability and maximum satisfiability.
- Facility location.

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x\in\{0,1\}^n} f(x) = \sum_{S\in\mathcal{M}} a_S \prod_{k\in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

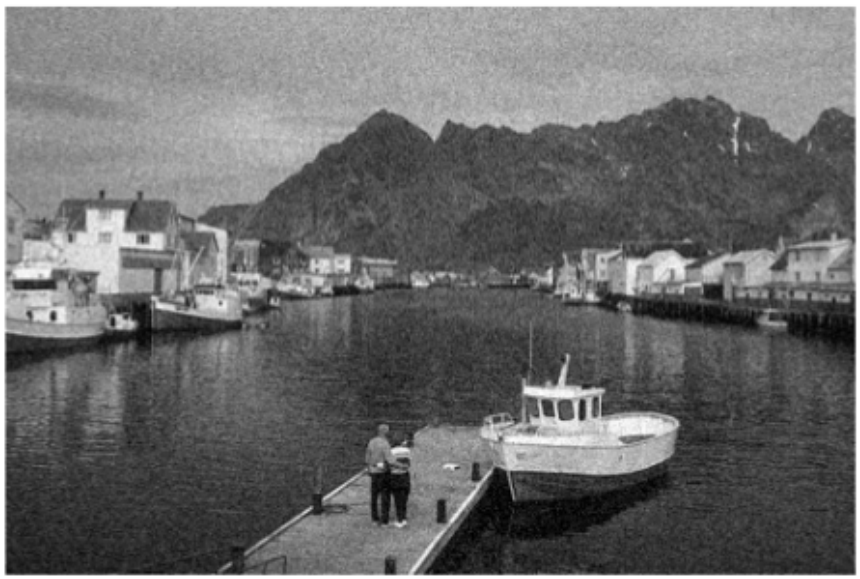
Numerous applications in various fields, e.g,

- Image restoration: given a blurred image, produce a "better" version of it.
- Modeled as *energy minimization* problem: pixels (B&W) are binary parameters p_j (initial image) or variables x_j (deblurred image).

$$\min E_w(x_1,\ldots,x_n) = \sum_{j\in P} (p_j - x_j)^2 + \sum_{w\in W} h_w(x_1,\ldots,x_n)$$

where $h_w(x_1,\ldots,x_n)$ measures the "deficiency" of the assignment $(x_1,\ldots,x_n)$ in a small window w of fixed size (based on smoothness, purity,...)

- If the window size is 16, then E_w is a polynomial of degree 16.



Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x\in\{0,1\}^n} f(x) = \sum_{S\in\mathcal{M}} a_S \prod_{k\in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Numerous applications in various fields, e.g,

- Telecommunications and statistical mechanics.
- Compute $\{-1,+1\}$ sequences $x = (x_0,\ldots,x_n)$ with low auto-correlations.
- Given $r \leq n$, find x to minimize

$$E_{n,r}(x) = \sum_{i=0}^{n-r}\sum_{d=1}^{r-1}\left(\sum_{j=i}^{i+r-1-d} x_j x_{j+d}\right)^2.$$

- $E_{n,r}$ is a polynomial of degree 4.
- Very hard as soon as $n, r \geq 40$.

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Complexity

PUB is NP-hard if f a multilinear polynomial f of degree 2 or more.

- Naturally models MAX CUT, MAX 2SAT, MAX STABLE SET, simple computer vision models, implementation of quantum computing...

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

- Some classical approaches:
  - *Linearization*.
  - *Quadratization*.
  - *Variable elimination*.
- Old ideas, new twists.
- Frequent connections with combinatorial structures.

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

- Some classical approaches:
  - *Linearization*.
  - *Quadratization*.
  - ***Variable elimination*.**

Basic algorithm

- A dynamic programming algorithm based on variable elimination (Hammer, Rosenberg and Rudeanu 1963)
- Revisited by Crama, Hansen and Jaumard (1990)
- Re-revisited in 2022!

J.V. Clausen, Y. Crama, R. Lusby, E. Rodriguez-Heck and S. Ropke, Solving unconstrained binary polynomial programs with limited reach, Working paper, 2023.

Basic algorithm

Central idea:

- Let $f^1(x_1, \ldots, x_n) := f(x_1, \ldots, x_n)$.
- Eliminate x_1, that is, produce an expression of the function

$$f^2(x_2, \ldots, x_n) \triangleq \min_{x_1} f^1(x_1, \ldots, x_n).$$

- How? Write $f^1(x_1, \ldots, x_n) = x_1 g(x_2, \ldots, x_n) + h(x_2, \ldots, x_n)$. (Straighforward if f is in polynomial form.)
- For any $(x_2, \ldots, x_n)$, minimize f^1 with respect to x_1:
  - if $g(x_2, \ldots, x_n) > 0$, then $x_1 = 0$;
  - if $g(x_2, \ldots, x_n) \leq 0$, then $x_1 = 1$.
- So: $f^2(x_2, \ldots, x_n) = \min\{0, g(x_2, \ldots, x_n)\} + h(x_2, \ldots, x_n)$.
- Repeat until all variables are eliminated.

Basic algorithm

Crucial step:

- $f^2(x_2, \ldots, x_n) = \min\{0, g(x_2, \ldots, x_n)\} + h(x_2, \ldots, x_n)$.
- Compute $\psi = \min\{0, g(x_2, \ldots, x_n)\}$
- Previous attempts: obtain a polynomial expression of ψ
- Hammer, Rosenberg and Rudeanu (1963) proceed by algebraic manipulations - never implemented.
- Crama, Hansen and Jaumard (1990) propose a branch-and-bound algorithm.
- Observe: if $g(x_2, \ldots, x_n)$ depends on a bounded number of variables (say, K variables), then an expression of ψ can be computed in time $O(2^K)$.
- This happens at all iterations of the basic algorithm if the co-occurrence graph of f has *bounded treewidth*.

Co-occurrence graph

Co-occurrence graph of a function $f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i$:

- vertices = variables
- $\{x_i, x_j\}$ is an edge if x_i and x_j appear in a same monomial S.

Example:

$f = 4 - 9x_1 - 5x_2 - 2x_3 + 13x_1x_2 + 13x_1x_3 + 6x_2x_3x_4$.
Edges: $\{x_1, x_2\}$, $\{x_1, x_3\}$,$\{x_2, x_3\}$, $\{x_2, x_4\}$, $\{x_3, x_4\}$.

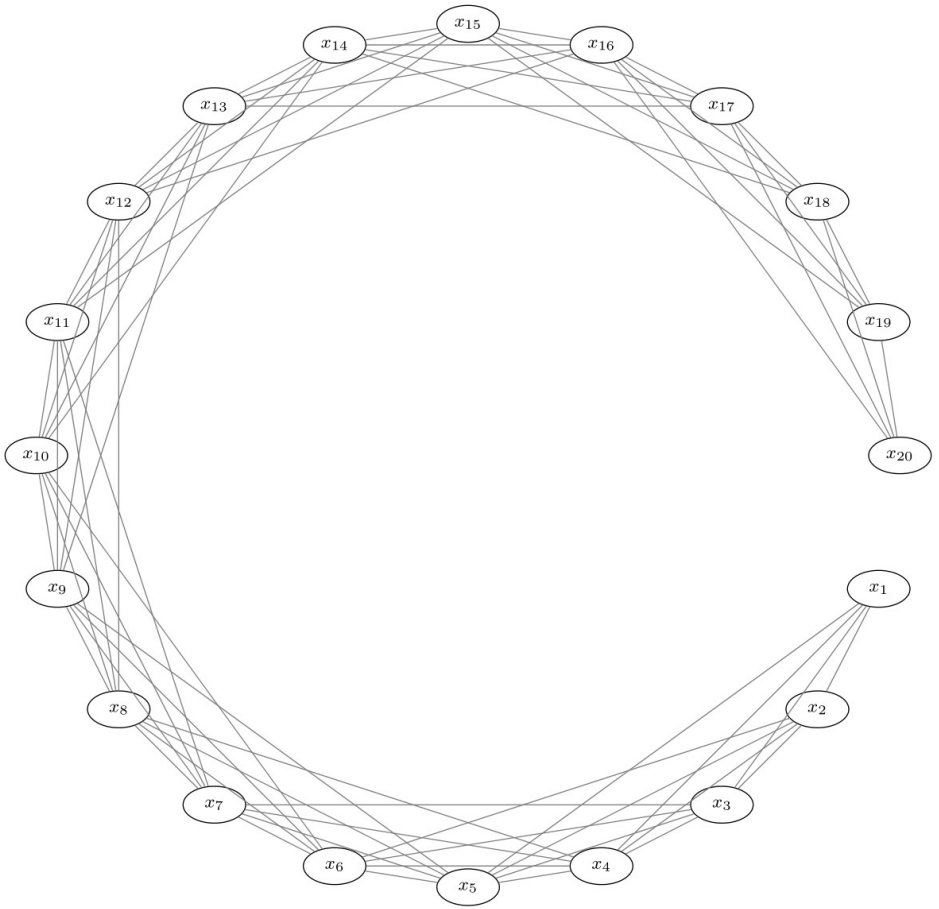
Basic algorithm with bounded treewidth

Crama, Hansen and Jaumard (1990)

When the co-occurrence graph of f has bounded treewidth K, the basic algorithm can be implemented to run in time $O(n2^K)$.

Low-autocorrelation sequence instances

- Low-autocorrelation sequence instances have a special structure.



Basic algorithm with bounded reach

- Low-autocorrelation sequence instances have bounded treewidth.
- They even have bounded *reach* r: if variables x_i and x_j appear together in a monomial, then $|j - i| \leq r$.
- Clausen et al. (2023) propose a variant of the basic algorithm which computes ψ in tabulated form, rather than polynomial form.
- This New BA is a polynomial algorithm for problems with bounded reach.

Computational results for New BA

Given $r \leq n$, find x to minimize

$$E_{n,r}(x) = \sum_{i=0}^{n-r} \sum_{d=1}^{r-1} \left(\sum_{j=i}^{i+r-1-d} x_j x_{j+d} \right)^2 .$$

Results with New BA:

- For the low-autocorrelation binary sequence problem, the New BA performs much better than linearization, quadratization (with PQCR), or previous versions of the basic algorithm (Old BA).
- 10 instances were solved to optimality for the first time. For example:
  - Instance 40.20: cannot be solved in 3 hours by linearization (gap > 100%) nor by PQCR (gap = 4%); solved in 460 sec by Old BA and in 9 sec by New BA;
  - Instance 50.25: cannot be solved in 3 hours by linearization (gap > 100%) nor by PQCR (does not obtain a lower bound) nor by Old BA (runs out of memory); solved in 468 sec by New BA.

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

- Some classical approaches:
  - ***Linearization***.
  - *Quadratization*.
  - *Variable elimination*.

Standard linearization (SL)

$$\text{(PUB)} \quad \min_{x\in\{0,1\}^n} \sum_{S\in\mathcal{M}} a_S \prod_{k\in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Fortet (1959), Glover and Woolsey (1973, 1974), McCormick (1976), etc.

1. Substitute monomials

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S = \prod_{k\in S} x_k, \qquad \forall S \in \mathcal{M}$$

$$y_S \in \{0,1\}, \qquad \forall S \in \mathcal{M}$$

$$x_k \in \{0,1\} \qquad \forall k = 1,\ldots,n$$

Standard linearization (SL)

$$\text{(PUB)} \quad \min_{x\in\{0,1\}^n} \sum_{S\in\mathcal{M}} a_S \prod_{k\in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Fortet (1959), Glover and Woolsey (1973, 1974), McCormick (1976), etc.

1. Substitute monomials

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S = \prod_{k\in S} x_k, \qquad \forall S \in \mathcal{M}$$

$$y_S \in \{0,1\}, \qquad \forall S \in \mathcal{M}$$

$$x_k \in \{0,1\} \qquad \forall k = 1,\ldots,n$$

2. Linearize constraints

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S \le x_k, \qquad \forall k \in S, \forall S \in \mathcal{M}$$

$$y_S \ge \sum_{k\in S} x_k - (|S|-1), \quad \forall S \in \mathcal{M}$$

$$y_S \in \{0,1\}, \qquad \forall S \in \mathcal{M}$$

$$x_k \in \{0,1\} \qquad \forall k = 1,\ldots,n$$

Standard linearization (SL)

$$\text{(PUB)} \quad \min_{x\in\{0,1\}^n} \sum_{S\in\mathcal{M}} a_S \prod_{k\in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Fortet (1959), Glover and Woolsey (1973, 1974), McCormick (1976), etc.

1. Substitute monomials

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S = \prod_{k\in S} x_k, \qquad \forall S \in \mathcal{M}$$

$$y_S \in \{0,1\}, \qquad \forall S \in \mathcal{M}$$

$$x_k \in \{0,1\} \qquad \forall k = 1,\ldots,n$$

3. Linear relaxation

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S \le x_k, \qquad \forall k \in S, \forall S \in \mathcal{M}$$

$$y_S \ge \sum_{k\in S} x_k - (|S|-1), \quad \forall S \in \mathcal{M}$$

$$0 \le y_S \le 1, \qquad \forall S \in \mathcal{M}$$

$$0 \le x_k \le 1 \qquad \forall k = 1,\ldots,n$$

Linear relaxation

Recall:

- For an optimization problem (P):

$$Z = \min f(x) \text{ s.t. } x \in X,$$

  a *relaxation* is another problem (R):

$$L = \min f(x) \text{ s.t. } x \in \tilde{X}$$

  such that $X \subseteq \tilde{X}$.
- Trivially, $L \leq Z$: the relaxation provides a *lower bound* on the optimal value of (P).
- We like (R) to be efficiently solvable (say, in polynomial time).
- We like L to be as close as possible to Z, i.e., (R) to be as *tight* as possible.

Standard linearization (SL)

$$\text{(PUB)} \quad \min_{x\in\{0,1\}^n} \sum_{S\in\mathcal{M}} a_S \prod_{k\in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

Fortet (1959), Glover and Woolsey (1973, 1974), McCormick (1976), etc.

1. Substitute monomials

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S = \prod_{k\in S} x_k, \qquad \forall S \in \mathcal{M}$$

$$y_S \in \{0,1\}, \qquad \forall S \in \mathcal{M}$$

$$x_k \in \{0,1\} \qquad \forall k = 1,\ldots,n$$

3. Linear relaxation

$$\min \sum_{S\in\mathcal{M}} a_S y_S + \sum_{i=1}^{n} a_i x_i$$

$$\text{s.t. } y_S \le x_k, \qquad \forall k \in S, \forall S \in \mathcal{M}$$

$$y_S \ge \sum_{k\in S} x_k - (|S|-1), \quad \forall S \in \mathcal{M}$$

$$0 \le y_S \le 1, \qquad \forall S \in \mathcal{M}$$

$$0 \le x_k \le 1 \qquad \forall k = 1,\ldots,n$$

Linear relaxation

Standard linearization polytope

$$P_L = \{(x, y) \in [0,1]^{n+|\mathcal{M}|} \mid y_S \leq x_k \ \ \forall k \in S, y_S \geq \sum_{k \in S} x_k - (|S| - 1) \ \ \forall S \in \mathcal{M}\}$$

Much studied in recent years.
A natural question: does P_L have fractional vertices?

- For a function containing a single nonlinear monomial: No.
- For two or more nonlinear terms, Yes! P_L is in general very weak!!!
- So, when is P_L integral?

Co-occurrence hypergraph

Co-occurrence hypergraph

For

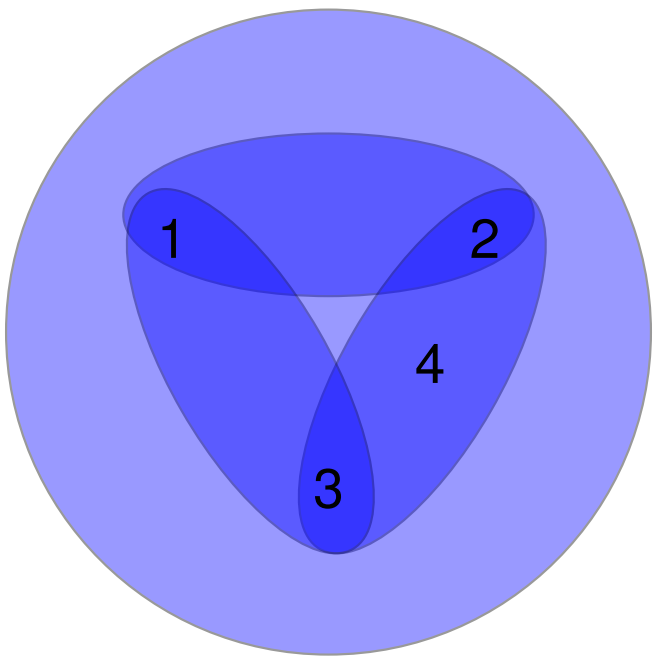
$$f(x_1, \ldots, x_n) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

$H_f = ([n], \mathcal{M})$ is the *co-occurrence hypergraph* associated with f.

Co-occurrence hypergraph

Example:

If $f = 4 - 9x_1 - 5x_2 - 2x_3 + 13x_1x_2 + 13x_1x_3 + 6x_2x_3x_4 - 13x_1x_2x_3x_4$, then $\mathcal{M} = \{12, 13, 234, 1234\}$.



Co-occurrence hypergraph

Co-occurrence hypergraph

For

$$f(x_1, \ldots, x_n) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

$H_f = ([n], \mathcal{M})$ is the *co-occurrence hypergraph* associated with f.

Definition: Berge cycles

For a hypergraph $H = (V, \mathcal{M})$, a **Berge cycle** of length p is a sequence

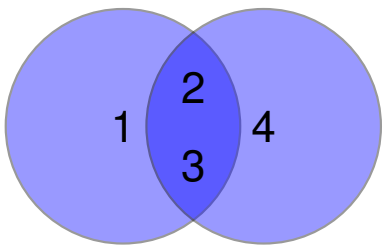
$$(i_1, S_1, i_2, S_2, \ldots, i_p, S_p, i_1),$$

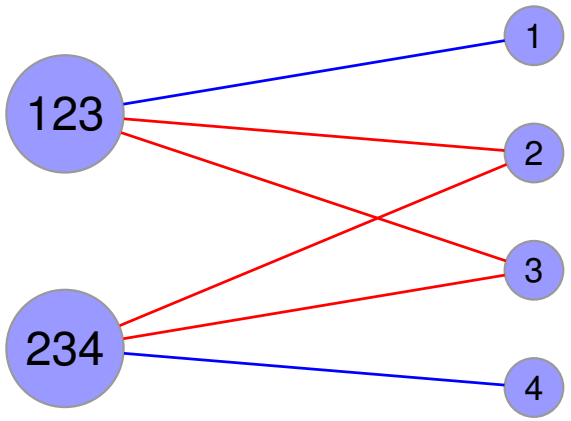
where

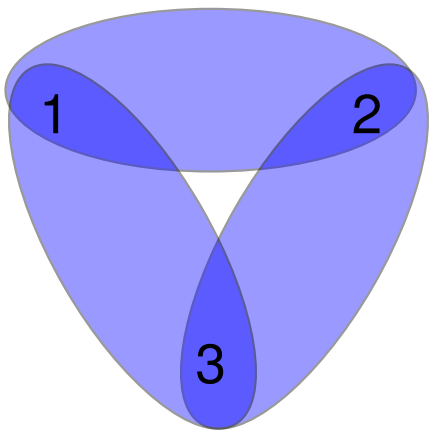
1. $i_1, i_2, \ldots, i_p$ are pairwise distinct vertices of V,
2. $S_1, S_2, \ldots, S_p$ are pairwise distinct edges of $\mathcal{M}$,
3. $i_j, i_{j+1} \in S_j$ for $j = 1, \ldots p-1$, and $i_1, i_p \in S_p$.

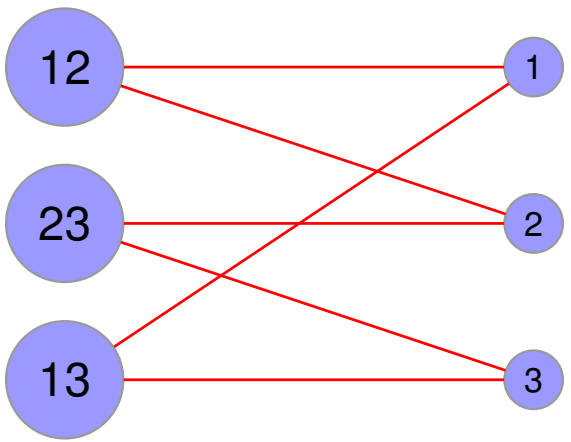
Berge cycles

Alternative definition using the ***bipartite*** *co-occurrence graph*.









Perfect standard linearization

(E. Rodríguez-Heck, Ch. Buchheim, Y. Crama, 2016)

All vertices of P_L are integral if and only if H_f has no Berge cycles.

- Generalizes a result of Padberg (1989) for quadratic functions.
- Closely related to a result of Crama (1988,1993) for an "irredundant" relaxation of P_L.
- Independently obtained by Del Pia and Khajavirad (2016).
- Can be checked efficiently.
- Recall: in general, P_L is extremely weak.

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

- Some classical approaches:
  - *Linearization*.
  - ***Quadratization***.
  - *Variable elimination*.

Polynomial unconstrained optimization in binary variables

$$\text{(PUB)} \quad \min_{x \in \{0,1\}^n} f(x) = \sum_{S \in \mathcal{M}} a_S \prod_{k \in S} x_k + \sum_{i=1}^{n} a_i x_i,$$

- ***Quadratization*.**
  - Idea: can we reduce PUB to the (constrained or unconstrained) quadratic case rather than to the (constrained) linear case?
  - Proposed by Rosenberg in 1975

Quadratization

Quadratization

The quadratic function $g(x, y)$, $(x, y) \in \{0, 1\}^{n+m}$ is a *quadratization* of the pseudo-Boolean function $f(x)$, $x \in \{0, 1\}^n$, if

$$f(x) = \min\{g(x, y) \mid y \in \{0, 1\}^m\} \quad \text{for all } x \in \{0, 1\}^n.$$

The y-variables are called *auxiliary* variables.

- $\min\{f(x) \mid x \in \{0, 1\}^n\} = \min\{g(x, y) \mid (x, y) \in \{0, 1\}^{n+m}\}$.
- Does every function f have a quadratization?

Existence

Existence of quadratizations (Rosenberg 1975)

Given the multilinear expression of a pseudo-Boolean function $f(x)$, one can construct in polynomial time a quadratization $g(x, y)$ of $f(x)$.

- Idea: in each term $\prod_{i \in A} x_i$ of f, with $\{1, 2\} \subseteq A$, replace the product $x_1 x_2$ by y:

  $$t(x, y) = \left(\prod_{i \in A \setminus \{1,2\}} x_i \right) y + M(x_1 x_2 - 2x_1 y - 2x_2 y + 3y),$$

  with M large enough.

Example

Example:

$\min_x f = 4 - 9x_1 - 5x_2 - 2x_3 + 13x_1x_2 + 13x_1x_3 + 6x_2x_3x_4 - 13x_1x_2x_3x_4$

Substitute x_3x_4:

$\min_{x,y} \quad 4 - 9x_1 - 5x_2 - 2x_3 + 13x_1x_2 + 13x_1x_3 + 6x_2y_{34} - 13x_1x_2y_{34} + M(x_3x_4 - 2x_3y_{34} - 2x_4y_{34} + 3y_{34})$

Substitute x_1x_2:

$\min \quad 4 - 9x_1 - 5x_2 - 2x_3 + 13y_{12} + 13x_1x_3 + 6x_2y_{34} - 13y_{12}y_{34} + M(x_3x_4 - 2x_3y_{34} - 2x_4y_{34} + 3y_{34}) + M(x_1x_2 - 2x_1y_{12} - 2x_2y_{12} + 3y_{12})$

Existence

Existence of quadratizations (Rosenberg 1975)

Given the multilinear expression of a pseudo-Boolean function $f(x)$, one can construct in polynomial time a quadratization $g(x, y)$ of $f(x)$.

- Idea: in each term $\prod_{i \in A} x_i$ of f, with $\{1, 2\} \subseteq A$, replace the product $x_1 x_2$ by y:

  $$t(x, y) = \left(\prod_{i \in A \setminus \{1,2\}} x_i \right) y + M(x_1 x_2 - 2x_1 y - 2x_2 y + 3y).$$

- In every minimizer of $t(x, y)$, $x_1 x_2 - 2x_1 y - 2x_2 y + 3y = 0$, $y = x_1 x_2$, and $t(x, y) = \prod_{i \in A} x_i$.
- Potential drawback: introduces many auxiliary variables.

Questions arising...

- Many quadratic reformulation procedures proposed in recent years.
- How many auxiliary variables are required?
- Which reformulations are “best”?
- etc.

Bounds

Bounds:

- lower and upper bounds on size of quadratizations

M. Anthony, E. Boros, Y. Crama and M. Gruber, Quadratization of symmetric pseudo-Boolean functions, *Discrete Applied Mathematics* 203 (2016) 1–12.

M. Anthony, E. Boros, Y. Crama and M. Gruber, Quadratic reformulations of nonlinear binary optimization problems, *Mathematical Programming* 162 (2017) 115-144.

E. Boros, Y. Crama and E. Rodrìguez-Heck, Compact quadratizations for pseudo-Boolean functions. *Journal of Combinatorial Optimization* 39 (2020) 687-707.

Generic quadratic reformulation framework

Example $f(x) = a_1 x_1 x_2 x_3 x_4 + a_2 x_1 x_2 x_3 + a_3 x_2 x_3 x_4$

1. **quadratization scheme**:
   $f(x) = a_1 (x_1 x_2)(x_3 x_4) + a_2 (x_1 x_2) x_3 + a_3 x_2 (x_3 x_4)$
2. **reformulate** into a quadratic problem

$$QCQP \begin{cases} \min \; g(x,y) = a_1 y_{12} y_{34} + a_2 y_{12} x_3 + a_3 x_2 y_{34} \\ \quad \text{s. t.} \;\; x_i \in \{0,1\} \quad \forall i \in [n], \quad y_{12} = x_1 x_2, \quad y_{34} = x_3 x_4 \end{cases}$$

Convert into penalized unconstrained QUB (e.g., Rosenberg 75)

$$\begin{cases} \min \; a_1 y_{12} y_{34} + a_2 y_{12} x_3 + a_3 x_2 y_{34} + \\ \qquad (|a_1| + |a_2|)(3y_{12} - 2x_1 y_{12} - 2x_2 y_{12} + x_1 x_2) + \\ \qquad (|a_1| + |a_3|)(3y_{34} - 2x_3 y_{34} - 2x_4 y_{34} + x_3 x_4) \\ \text{s. t.} \quad x_i \in \{0,1\} \quad \forall i \in [n] \qquad y \in \{0,1\} \end{cases}$$

or linearize the quadratic constraints

$$\begin{cases} \min \; a_1 y_{12} y_{34} + a_2 y_{12} x_3 + a_3 x_2 y_{34} \\ \text{s. t.} \quad x_i \in \{0,1\} \quad \forall i \in [n] \\ \quad y_{12} \le x_1, \; y_{12} \le x_2, \\ \quad y_{12} \ge x_1 + x_2 - 1, \; y_{12} \ge 0, \\ \quad y_{34} \le x_3, \; y_{34} \le x_4, \\ \quad y_{34} \ge x_3 + x_4 - 1, \; y_{34} \ge 0 \end{cases}$$

3. **solve** by any method for unconstrained or linearly constrained binary quadratic problems

Generic quadratic reformulation framework

- Different choices in Phase 1, Phase 2 or Phase 3 give rise to different exact solution methods.
- Different authors use similar methods under different names
  - (Rosenberg 1975) Iteratively, substitute the product of two variables by a new one until degree 2 is reached for all monomials. Enforce the equivalence by a penalty function.
  - (Buchheim & Rinaldi 2007) Add enough submonomials to obtain a self-pairwise cover. Build a linear reformulation together with a family of valid inequalities.
  - (Anthony et al. 2017) Represent f by a hypergraph and find a pairwise cover. Enforce the equivalence by a penalty function.
  - (Lazare 2020) Iteratively partition each monomial (degree >2) into two (new)submonomials. Build an equivalent binary quadratic problem with linear constraints.

Our work: Try to unify the three-phase framework and understand the impact of each choice.
Y. Crama, S. Elloumi, A. Lambert, E. Rodríguez-Heck, Quadratization and convexification in polynomial binary optimization, Working paper, 2022.

- Phase 1: Quadratization scheme
- Phase 2: Reformulation into a quadratic problem
- Phase 3: Solution of the reformulated problem

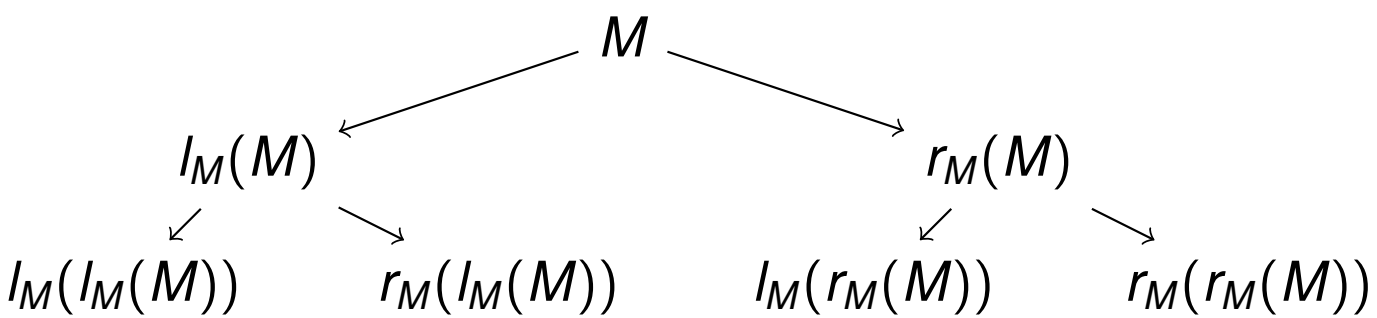
Definition (**Quadratization scheme of a monomial** M)

A quadratization scheme for a monomial M is a rooted directed acyclic graph $G_M = (V_M, A_M)$ with the following properties:

i) vertices in V_M are subsets of M, and M is the root ;

ii) the *leaves* are the singletons $\{i\}$, $i \in M$;

iii) when a vertex $E \in V$ is not a leaf, it has two children $l_M(E)$ and $r_M(E)$: $0 < |l_M(E)| < |E|$, $0 < |r_M(E)| < |E|$, and $E = l_M(E) \cup r_M(E)$.

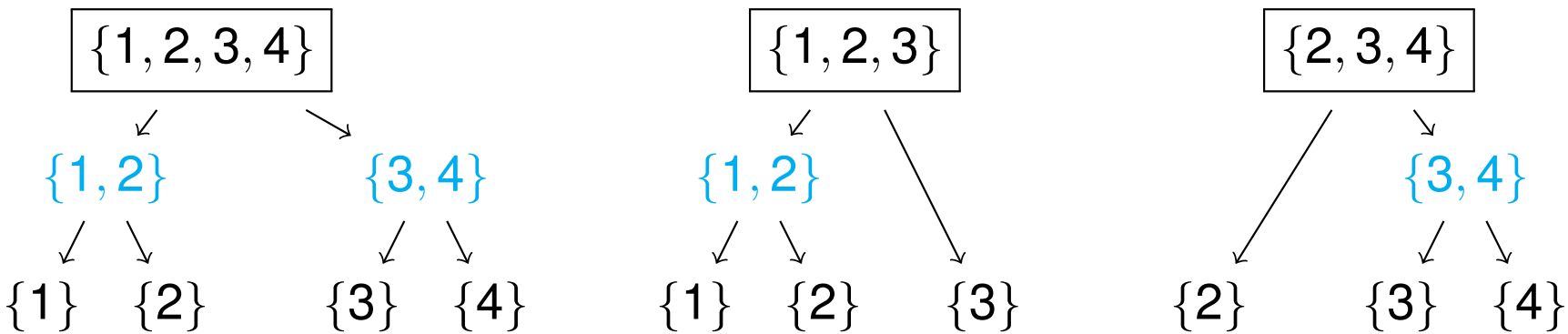


Possible non-disjoint scheme

Definition (**Quadratization scheme for a set of monomials**)

A *quadratization scheme* for a polynomial $\mathcal{M} \subseteq 2^{[n]}$ is a collection of quadratization schemes $\left\{G_M = (V_M, A_M) : M \in \mathcal{M}\right\}$, where each G_M is a quadratization scheme for the corresponding monomial $M \in \mathcal{M}$.

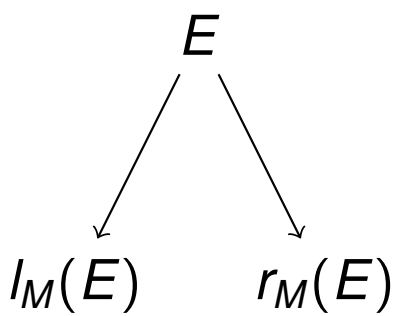
Example $f(x) = a_1 x_1 x_2 x_3 x_4 + a_2 x_1 x_2 x_3 + a_3 x_2 x_3 x_4$



Straightforward quadratic reformulation

- Rename the initial variables $x_1 \dots x_n$ into $z_1 \dots z_n$
- Add a new variable z_E per new monomial E in the quadratization scheme $\rightarrow$ get a total of N binary variables

$$(QCQP)\begin{cases} \min\limits_{z\in\{0,1\}^N} & g(z) = \sum\limits_{M\in\mathcal{M}} a_M z_{l_M(M)} z_{r_M(M)} \\ \text{s.t.} & \\ & z_E = z_{l_M(E)} z_{r_M(E)} \quad \forall E \in \mathcal{E}, \forall M \in \mathcal{M} : E \in \mathcal{E}_M \end{cases}$$

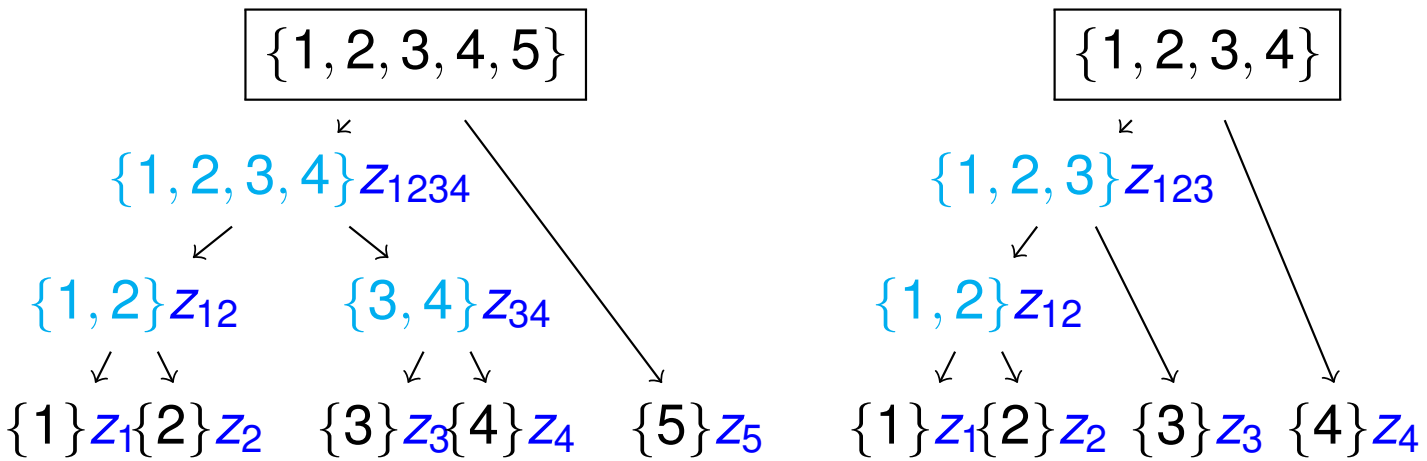


- Proof of equivalence: Iteratively check

$$z_{l_M(M)} z_{r_M(M)} = \prod_{i\in M} z_i = \prod_{i\in M} x_i$$

Straightforward quadratic reformulation

Example: $\min f = a_1 x_1 x_2 x_3 x_4 x_5 + a_2 x_1 x_2 x_3 x_4$



$$\begin{cases} \min\limits_{z \in \{0,1\}^{10}} & a_1 z_{1234} z_5 + a_2 z_{123} z_4 \\ \text{s.t.} & \\ & z_{1234} = z_{12} z_{34} \quad z_{12} = z_1 z_2 \quad z_{34} = z_3 z_4 \\ & z_{123} = z_{12} z_3 \quad z_{12} = z_1 z_2 \end{cases}$$

Handling the quadratic constraints

Goal: enforce the constraints $z_E = z_{l_M(E)} z_{r_M(E)}$ in every optimal solution as illustrated earlier.

- Unconstrained penalized binary quadratic problem (Rosenberg-like)

$$UPBQ \left\{ \min_{z \in \{0,1\}^N} \quad \sum_{M \in \mathcal{M}} a_M z_{l_M(M)} z_{r_M(M)} + \sum_{E \in \mathcal{E}} \Big(\sum_{\substack{M \in \mathcal{M}: \\ E \in \mathcal{E}_M}} |a_M| (3 z_E - 2 z_E z_{l_M(E)} - 2 z_E z_{r_M(E)} + z_{l_M(E)} z_{r_M(E)}) \right.$$

- Linearly constrained binary quadratic problem (standard inequalities)

$$LCBQ \left\{ \begin{array}{ll} \min\limits_{z \in \{0,1\}^N} & \sum\limits_{M \in \mathcal{M}} a_M z_{l_M(M)} z_{r_M(M)} \\ & z_E \leq z_{l_M(E)},\ z_E \leq z_{r_M(E)},\ z_E \geq z_{l_M(E)} + z_{r_M(E)} - 1,\ z_E \geq 0 \quad \forall E \in \mathcal{E}, \forall M \in \mathcal{M} : E \in \mathcal{E}_M \end{array} \right.$$

Phase 2- Solution of the reformulated problem

How can we solve *UPBQ* or *LCBQ*?
(Note: quadratic terms in the objective function.)

1. Linearize again: add a new variable and standard linear inequalities per quadratic term $\rightarrow$ get an ILP
2. Use Quadratic Convex Reformulation methods $\rightarrow$ get a MIQP with a convex continuous relaxation
   1. smallest eigenvalue method
   2. "basic" `QCR`
   3. `PQCR`: `QCR` improved by valid quadratic equalities

Quadratic Convex Reformulation methods

Smallest Eigenvalue Method (Hammer and Rubin 1970)

$$(\text{BQP}) \begin{cases} \min \ q(x) = x^T Q x \\ \text{s. t.} \\ Ax \leq b \\ x \in \{0,1\}^n \end{cases} \iff \begin{cases} \min \ x^T Q x + |\lambda_1(Q)| \sum_{i=1}^{n} (x_i^2 - x_i) \\ \text{s. t.} \\ Ax \leq b \\ x \in \{0,1\}^n \text{ convex continuous relaxation} \end{cases}$$

Quadratic Convex Reformulation QCR (Billionnet and Elloumi 2007)

$$\begin{cases} \min \ x^T Q x + \sum_{i=1}^{n} u_i (x_i^2 - x_i) \\ \text{s. t.} \\ Ax \leq b \\ x \in \{0,1\}^n \text{ convex continuous relaxation} \\ \qquad \text{for appropriate choices of } u_i \end{cases}$$

- Best u: equivalent to a semidefinite programming problem

$$(SDP) \begin{cases} \min \ < Q, X > \\ \text{s. t.} \\ X_{ii} = x_i \\ Ax \leq b \\ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \succeq 0 \\ X \in \mathbf{R}^{n \times n} \end{cases} .$$

Quadratic Convex Reformulation methods

- PQCR (Lazare 2020; Elloumi, Lambert, Lazare 2021) introduce valid quadratic equalities associated with the quadratization scheme to strengthen the SDP formulation.
  - $z_E = z_{l_M(E)} z_{r_M(E)}$
  - $z_E z_F = z_E$ if $F \subset E$
  - $z_E z_{E'} = z_F z_{F'}$ if $E \cup E' = F \cup F'$ (numerous !)

Summary

Given a quadratization scheme, we can derive two classes of reformulations and several solution methods

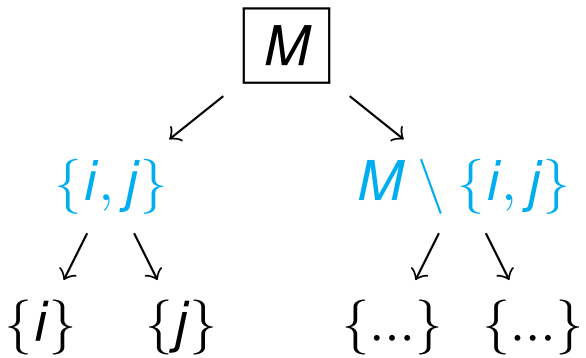
	UPBQ	LCBQ
Lin	UPBQ + Lin	LCBQ + Lin
QCR	UPBQ + QCR	LCBQ + QCR
PQCR	PQCR	

How does the quadratization scheme impact the solution methods ?
Some computational observations.

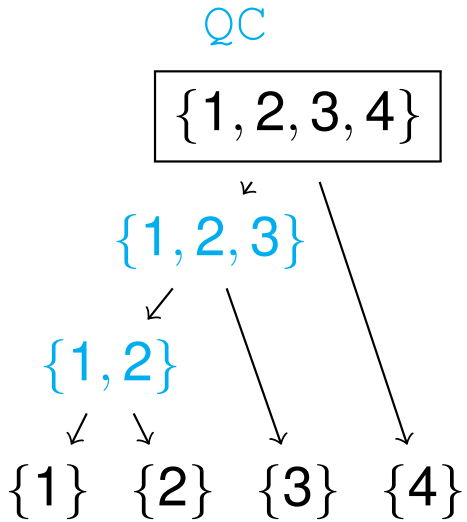
Four quadratization schemes

- QA Sort the monomial set in lexicographical order. In this order, iteratively *(i)* Select the first product of variables $x_i x_j$ that appears in a monomial of degree at least 3. *(ii)* For any monomial M containing i and j, set $l_M(M)$ to $\{i, j\}$ and $r_M(M)$ to $M \setminus \{i, j\}$. *(iii)* Add $l_M(M)$ and $r_M(M)$ to the sorted monomial set
- QB is similar to QA. The most frequent product $x_i x_j$ is selected
- QC Split $M = \{1, \ldots, d\}$ into $l_M(M) = \{1, \ldots, d-1\}$ and $r_M(M) = \{d\}$
- QD Split $M = \{1, \ldots, d\}$ into $l_M(M) = \{1, \ldots, d-1\}$ and $r_M(M) = \{2, \ldots, d\}$. Non-disjoint subsets l_M and r_M.

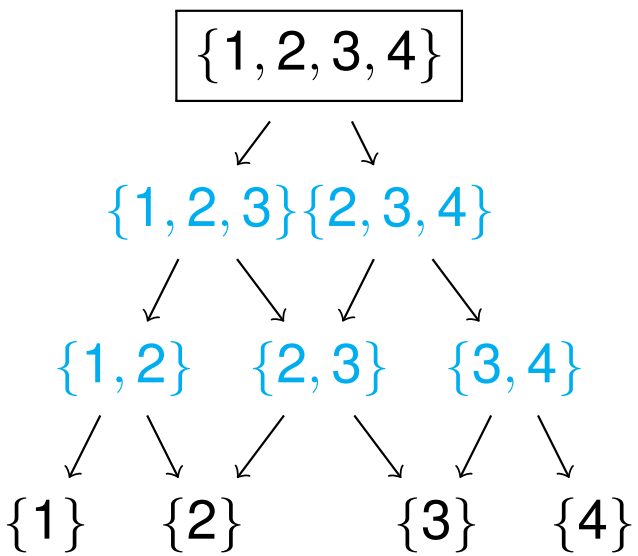
QA and QB



QC



QD

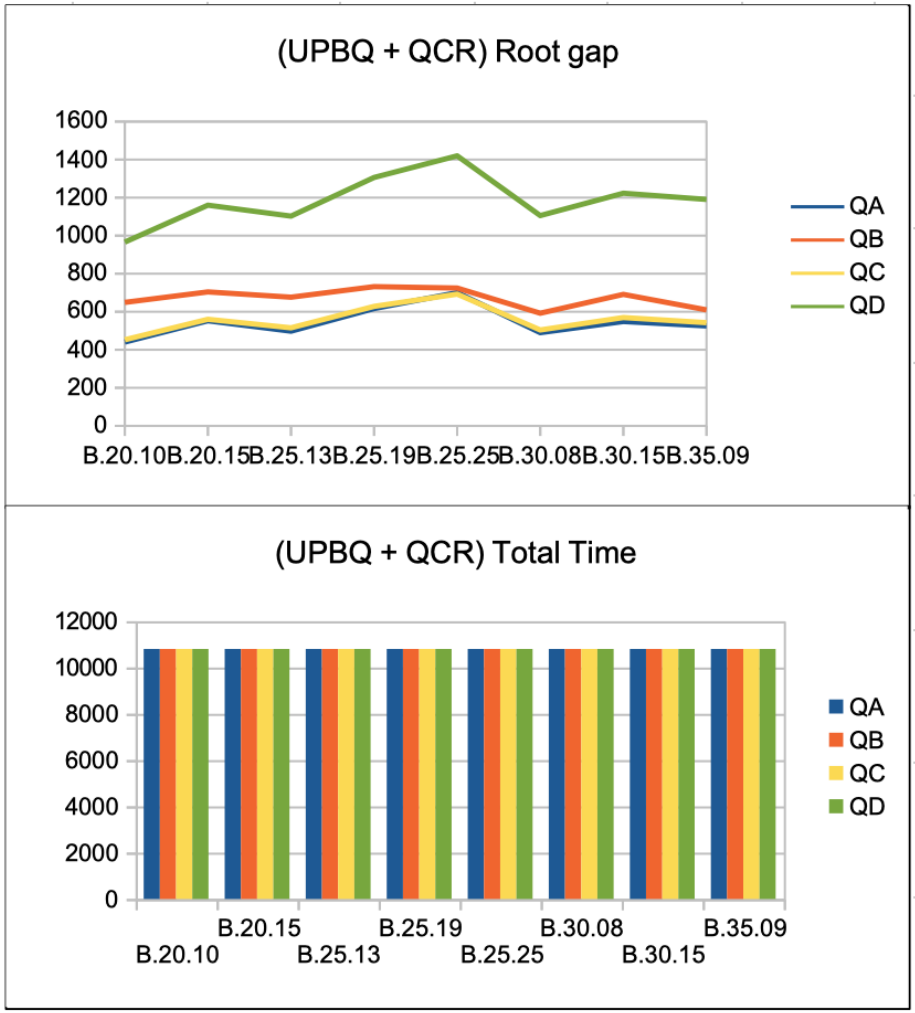


- In our test instances, the number of variables is on average $N = 217, 234, 656$, and 885 for QA, QB, QC, and QD , respectively.

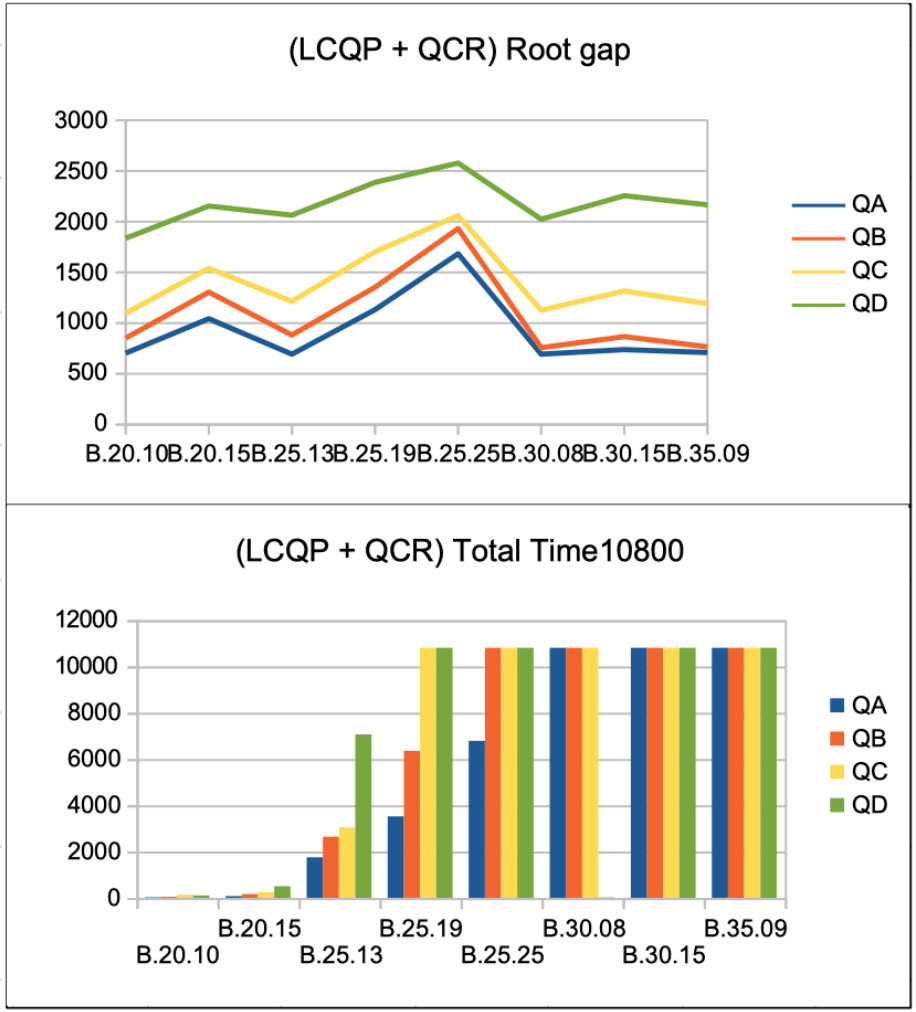
Settings

- Illustration: 8 instances of the Low Autocorrelation Binary Sequence problem (polynomials of degree 4, up to 35 initial variables)
- We use Gurobi as a solver
- Time limit 3 hours

QCR reformulations

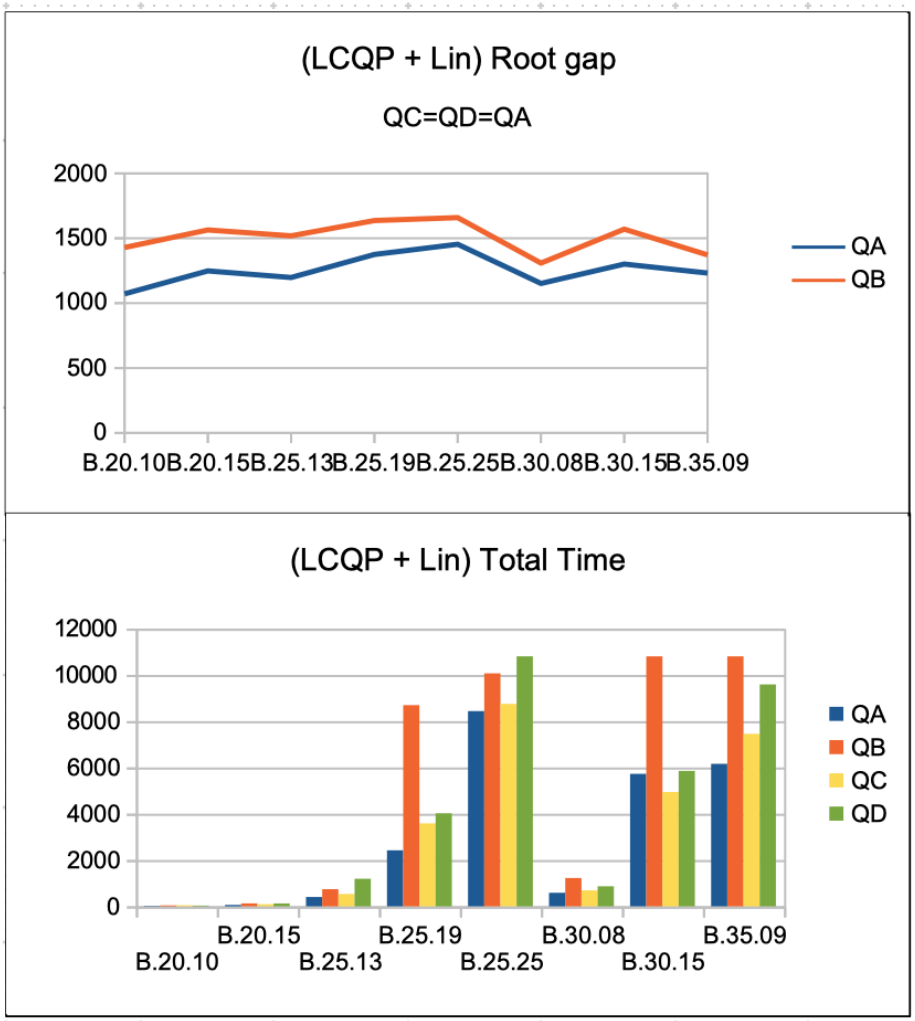


gap: QA and QC are best
time: all fail in 3h



gap: QA is best
time: QA is best

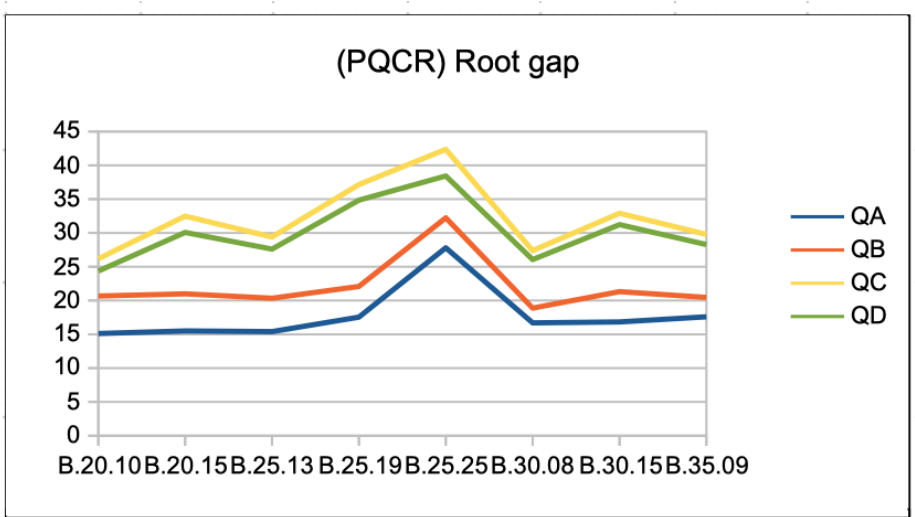
Linear reformulations

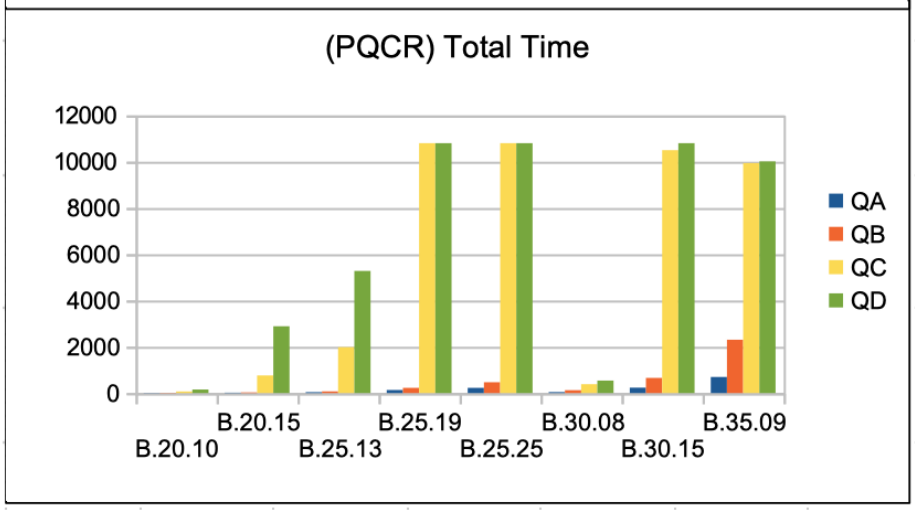


gap: QA, QC, and QD are best
time: QA and QC are best

- QA and QC perform better than QB and QD
- The initial gaps are huge, but the running time is improved with respect to QCR.

PQCR reformulations





gap: QA is best
time: QA is best

- In PQCR, QA and QB perform tremendously better than QC and QD
- The initial gaps are rather small
- PQCR is the best method. When coupled with QA, the 8 instances are solved within 3 minutes on av.
- Over the four quadratization schemes, QA is globally best and QD is worst. Recall $N = 217, 234, 656$, and 885 variables by QA, QB, QC, and QD

Comments and interpretation

Lessons learned:

- Three-phase framework for quadratic reformulations.
- Unifying view of "quadratization schemes".
- Several reformulations and solution methods can be applied with a given quadratization scheme.
- Computational results suggest that quadratization schemes with fewer variables have best performance.
- But the best performing method depends on the type of instances: `PQCR` (SDP-based) is best for low-autocorrelation problem (hard), while successive linearization (ILP-based) is best for image restoration instances (easy).
- In particular: good initial lower bounds (small duality gaps) do not guarantee small total running times.

Conclusions

- Polynomial unconstrained binary optimization problems are very hard nuts!
- Old ideas are still fruitful: linearization (1959), quadratization (1975), variable elimination (1963).
- Algorithms must be tailored carefully, must sometimes be specifically adapted for the problem at hand.
- Still a lot of work to do, both theoretical and computational.