November 17, 2022

Insights: A Mixing Method based Branch-and-Bound Solver for QUBO Problems

Joint work with Valentin Durante

FWF Der Wissenschaftsfonds.

UNIVERSITÄT KLAGENFURT

Modeling - Analysis - Optimization DOC.FUNDS DOCTORAL SCHOOL

Quadratic Unconstrained Binary Optimization (QUBO)

- internally solves problems of the following type:

QUBO in $\{-1, 1\}$-variables

Given $C \in \mathbb{R}^{n \times n}$, solve

$$\begin{aligned} \max \quad & x^\top C x \\ \text{s.t.} \quad & x \in \{-1, 1\}^n. \end{aligned} \tag{QUBO}$$

Quadratic Unconstrained Binary Optimization (QUBO)

- internally solves problems of the following type:

QUBO in $\{-1, 1\}$-variables

Given $C \in \mathbb{R}^{n \times n}$, solve

$$\begin{aligned} \max \quad & x^\top C x \\ \text{s.t.} \quad & x \in \{-1, 1\}^n. \end{aligned} \tag{QUBO}$$

- $\mathcal{NP}$-hard
- LP approaches exist only for sparse C
- solver is mainly developed for dense C

Quadratic Unconstrained Binary Optimization (QUBO)

- internally solves problems of the following type:

QUBO in $\{-1, 1\}$-variables

Given $C \in \mathbb{R}^{n \times n}$, solve

$$\begin{aligned} \max \quad & x^\top C x \\ \text{s.t.} \quad & x \in \{-1, 1\}^n. \end{aligned} \tag{QUBO}$$

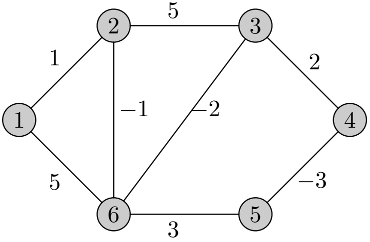
- $\mathcal{NP}$-hard
- LP approaches exist only for sparse C
- solver is mainly developed for dense C

Example:

Max-Cut Problem: $C = \frac{1}{4} L(G)$, where $L(G)$ Laplacian matrix

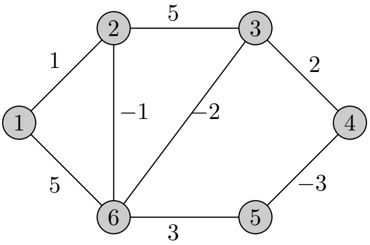
The (Weighted) Max-Cut Problem

Given: undirected graph $G = (V, E)$ with edge weights $w \in \mathbb{R}^E$



The (Weighted) Max-Cut Problem

Given: undirected graph $G = (V, E)$ with edge weights $w \in \mathbb{R}^E$



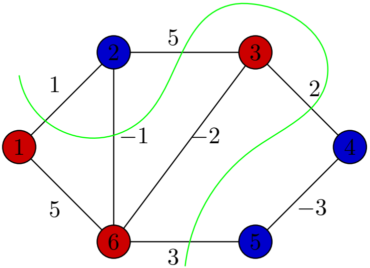
Max-Cut

Find a maximum cut in G, i.e., an optimal solution of

$$\max_{S \subseteq V} \sum_{i \in S,\ j \in V \setminus S} w_{ij}. \tag{MC}$$

The (Weighted) Max-Cut Problem

Given: undirected graph $G = (V, E)$ with edge weights $w \in \mathbb{R}^E$



Max-Cut

Find a maximum cut in G, i.e., an optimal solution of

$$\max_{S \subseteq V} \sum_{i \in S,\ j \in V \setminus S} w_{ij}. \tag{MC}$$

Examples I

QUBO in $\{0,1\}$-variables

$$\max_{x \in \{0,1\}^n} \left\{ x^\top Q x + q^\top x \right\}$$

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

Examples I

QUBO in $\{0,1\}$-variables

$$\max_{x \in \{0,1\}^n} \left\{ x^\top Q x + q^\top x \right\}$$

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

$$\Leftrightarrow$$

Reformulation in $\{-1,1\}$-variables

$$\max_{x \in \{-1,1\}^{n+1}} x^\top C x$$

where

$$C := \frac{1}{4} \begin{bmatrix} e^\top Q e + 2q^\top e & e^\top Q + q^\top \\ Qe + q & Q \end{bmatrix}.$$

Examples II

Linearly constrained binary quadratic problem (BQP)

$$\begin{aligned} \min \quad & x^\top Q x + q^\top x \\ \text{s.t.} \quad & Ax = b \\ & x \in \{0,1\}^n \end{aligned} \tag{BQP}$$

where $Q \in \mathbb{R}^{n \times n}$, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.

Examples II

Linearly constrained binary quadratic problem (BQP)

$$\begin{aligned} \min \quad & x^\top Q x + q^\top x \\ \text{s. t.} \quad & Ax = b \\ & x \in \{0,1\}^n \end{aligned} \tag{BQP}$$

where $Q \in \mathbb{R}^{n \times n}$, $q \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.

For some $C \in \mathbb{R}^{(n+1)\times(n+1)}$, (BQP) is equivalent to

Reformulation (used in *BiqBin* solver)

$$\begin{aligned} \min \quad & x^\top C x \\ \text{s. t.} \quad & x \in \{-1,1\}^{n+1} \\ & x_0 = 1. \end{aligned}$$

Examples III

Maximum Stable Set Problem

$$\begin{aligned} \max \quad & e^\top x \\ \text{s.t.} \quad & x_i x_j = 0, \quad \forall ij \in E \\ & x \in \{0,1\}^n \end{aligned} \tag{MSSP}$$

Examples III

Maximum Stable Set Problem

$$\begin{aligned} \max \quad & e^\top x \\ \text{s.t.} \quad & x_i x_j = 0, \quad \forall ij \in E \\ & x \in \{0,1\}^n \end{aligned} \tag{MSSP}$$

$$\Leftrightarrow$$

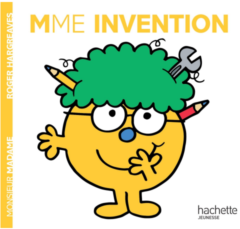
Reformulation of (MSSP)

$$\begin{aligned} \max \quad & \left\{ \frac{n}{2} + \frac{1}{2} e^\top x - n \sum_{ij \in E} (x_i + 1)(x_j + 1) \right\} \\ \text{s.t.} \quad & x \in \{-1,1\}^n \end{aligned}$$

Live demonstration!

Solvers for dense C using Semidefinite Programming

Solvers for dense C using Semidefinite Programming

BiqMac (2010)	BiqCrunch (2016)
	
MADAM (2021)	BiqBin (2022)
	

Semidefinite Relaxation

We introduce $X := xx^\top$:

- $x^\top Cx = \langle C, xx^\top \rangle = \langle C, X \rangle$
- $\text{diag}(X) = e$
- $X \succeq 0$
- $\text{rank}(X) = 1$

Semidefinite Relaxation

We introduce $X := xx^\top$:

- $x^\top Cx = \langle C, xx^\top \rangle = \langle C, X \rangle$
- $\operatorname{diag}(X) = e$
- $X \succeq 0$
- $\operatorname{rank}(X) = 1$

Equivalent formulations

$$\begin{aligned} \max \quad & x^\top Cx \\ \text{s.t.} \quad & x \in \{-1,1\}^n \end{aligned} \quad \Leftrightarrow \quad \begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \operatorname{diag}(X) = e \\ & X \succeq 0 \\ & \operatorname{rank}(X) = 1 \end{aligned}$$

Semidefinite Relaxation

We introduce $X := xx^\top$:

- $x^\top Cx = \langle C, xx^\top \rangle = \langle C, X \rangle$
- $\operatorname{diag}(X) = e$
- $X \succeq 0$
- $\operatorname{rank}(X) = 1$

Semidefinite relaxation

$$\begin{aligned} \max \quad & x^\top Cx \\ \text{s. t.} \quad & x \in \{-1, 1\}^n \end{aligned} \quad \leq \quad \begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \operatorname{diag}(X) = e \\ & X \succeq 0 \\ & \sout{\operatorname{rank}(X) = 1} \end{aligned}$$

Semidefinite Relaxation

We introduce $X := xx^\top$:

- $x^\top Cx = \langle C, xx^\top \rangle = \langle C, X \rangle$
- $\operatorname{diag}(X) = e$
- $X \succeq 0$
- $\operatorname{rank}(X) = 1$

Semidefinite relaxation

$$\begin{aligned} \max \quad & x^\top Cx \\ \text{s.t.} \quad & x \in \{-1, 1\}^n \end{aligned} \quad \leq \quad \begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \operatorname{diag}(X) = e \\ & X \succeq 0 \\ & \cancel{\operatorname{rank}(X) = 1} \end{aligned}$$

- all mentioned solvers: additional ‘clique’ inequalities
- competitive implementations possible without inequalities?!

Low-rank Factorization $X = V^\top V$

Factorization of $X \succeq 0$

$$X = V^\top V$$

for some $V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$ with $k \leq n$.

Low-rank Factorization $X = V^\top V$

Factorization of $X \succeq 0$

$$X = V^\top V$$

for some $V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$ with $k \leq n$.

- $X_{ij} = v_i^\top v_j \quad \Rightarrow \quad \langle C, X \rangle = \sum_{i,j=1}^n C_{ij} X_{ij} = \sum_{i,j=1}^n C_{ij} v_i^\top v_j$
- $\mathrm{diag}(X) = e \quad \Leftrightarrow \quad \|v_i\| = 1, \; i = 1, \dots, n$

Low-rank Factorization $X = V^\top V$

Factorization of $X \succeq 0$

$$X = V^\top V$$

for some $V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$ with $k \leq n$.

- $X_{ij} = v_i^\top v_j \quad \Rightarrow \quad \langle C, X \rangle = \sum_{i,j=1}^n C_{ij} X_{ij} = \sum_{i,j=1}^n C_{ij} v_i^\top v_j$
- $\operatorname{diag}(X) = e \quad \Leftrightarrow \quad \|v_i\| = 1, \; i = 1, \dots, n$

Optimization Problem (SDP-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^n C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \; i = 1, \dots, n \end{aligned} \qquad \text{(SDP-vec)}$$

Low-rank Factorization $X = V^\top V$

Factorization of $X \succeq 0$

$$X = V^\top V$$

for some $V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$ with $k \leq n$.

- $X_{ij} = v_i^\top v_j \quad \Rightarrow \quad \langle C, X \rangle = \sum_{i,j=1}^n C_{ij} X_{ij} = \sum_{i,j=1}^n C_{ij} v_i^\top v_j$
- $\operatorname{diag}(X) = e \quad \Leftrightarrow \quad \|v_i\| = 1, \; i = 1, \dots, n$

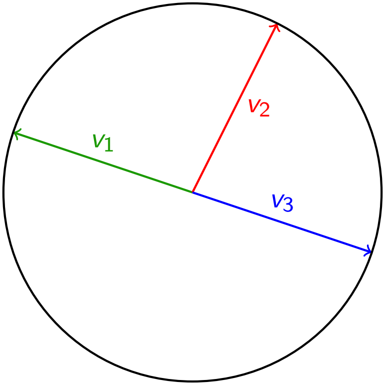
Optimization Problem (SDP-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \; i = 1, \dots, n \end{aligned} \qquad \text{(SDP-vec)}$$

- (SDP) $\Leftrightarrow$ (SDP-vec) for $k > \sqrt{2n}$ [cf. Pataki, 1998]

Geometric Interpretation

$$
\begin{aligned}
v_i^\top v_j &= \|v_i\| \cdot \|v_j\| \cdot \cos \measuredangle(v_i, v_j) \\
&= \cos \measuredangle(v_i, v_j)
\end{aligned}
$$



Coordinate Ascent Method

Optimization Problem (SDP-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \ i = 1, \dots, n \end{aligned} \tag{SDP-vec}$$

Coordinate Ascent Method

Optimization Problem (SDP-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s.t.} \quad & \|v_i\| = 1, \; i = 1, \dots, n \end{aligned} \qquad \text{(SDP-vec)}$$

Coordinate Ascent

We fix all but one vector v_i. (SDP-vec) reduces to

$$\begin{aligned} \max \quad & g^\top v_i = \|g\| \cdot \|v_i\| \cdot \cos \measuredangle(g, v_i) \\ \text{s.t.} \quad & \|v_i\| = 1, \; v_i \in \mathbb{R}^k \end{aligned}$$

where $g = \sum_j^n c_{ij} v_j = V \cdot c_i$

Coordinate Ascent Method

Optimization Problem (SDP-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \; i = 1, \dots, n \end{aligned} \qquad \text{(SDP-vec)}$$

Coordinate Ascent

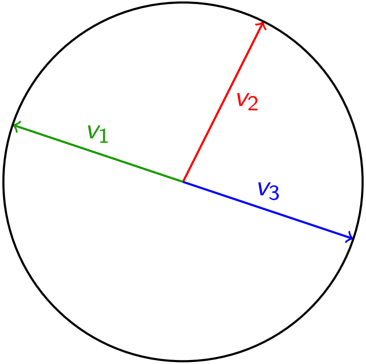
We fix all but one vector v_i. (SDP-vec) reduces to

$$\begin{aligned} \max \quad & g^\top v_i = \|g\| \cdot \|v_i\| \cdot \cos \measuredangle(g, v_i) \\ \text{s. t.} \quad & \|v_i\| = 1, \; v_i \in \mathbb{R}^k \end{aligned}$$

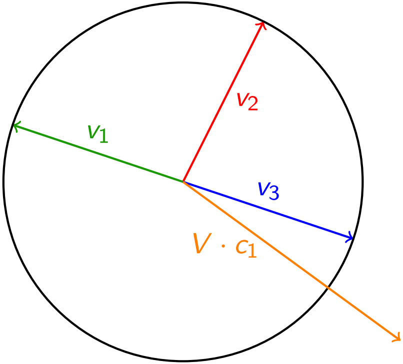
where $g = \sum_j^n c_{ij} v_j = V \cdot c_i$

- closed-form solution: $v_i = \frac{g}{\|g\|}$ for $g \neq 0$

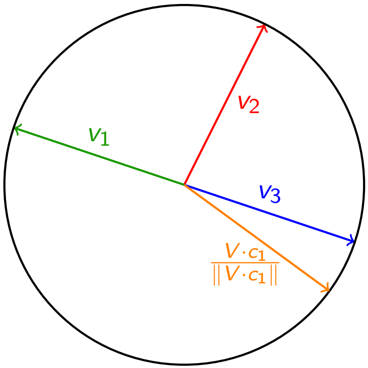
Mixing Method (Wang el al., 2018)



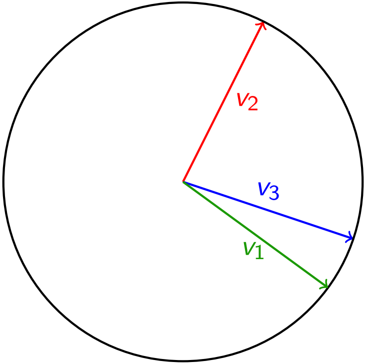
Mixing Method (Wang el al., 2018)



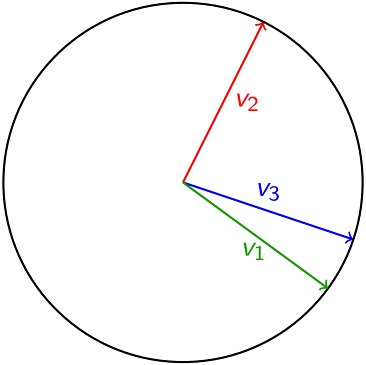
Mixing Method (Wang el al., 2018)



Mixing Method (Wang el al., 2018)



Mixing Method (Wang el al., 2018)



Mixing Method

- repeat for $v_1, v_2, \ldots, v_n$ again and again
- initialize $v_1, \ldots, v_n$ randomly on the unit sphere

Algorithm: Mixing Method

Algorithm 1: Mixing Method (Wang et al., 2018)

Input: $C = (c_1|\dots|c_n) \in \mathbb{R}^{n\times n}$ with $\operatorname{diag}(C) = 0$, $k \in \mathbb{N}_{\geq 1}$
Output: approximate solution $V = (v_1|\dots|v_n) \in \mathbb{R}^{k\times n}$ of (SDP-vec)

for $i \leftarrow 1$ **to** n **do**
  $v_i \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;

while *not yet converged* **do**
  for $i \leftarrow 1$ **to** n **do**
    $v_i \leftarrow \frac{V\cdot c_i}{\|V\cdot c_i\|}$;

Algorithm: Mixing Method

Algorithm 1: Mixing Method (Wang et al., 2018)

Input: $C = (c_1|\dots|c_n) \in \mathbb{R}^{n\times n}$ with $\operatorname{diag}(C) = 0,\ k \in \mathbb{N}_{\geq 1}$
Output: approximate solution $V = (v_1|\dots|v_n) \in \mathbb{R}^{k\times n}$ of (SDP-vec)

for $i \leftarrow 1$ **to** n **do**
  $v_i \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;

while *not yet converged* **do**
  for $i \leftarrow 1$ **to** n **do**
    $v_i \leftarrow \frac{V\cdot c_i}{\|V\cdot c_i\|}$;

Theorem (Wang et al., 2018)

The Mixing Method converges linearly to the global optimum under a non-degeneracy assumption.

Algorithm: Mixing Method

Algorithm 1: Mixing Method (Wang et al., 2018)

Input: $C = (c_1|\dots|c_n) \in \mathbb{R}^{n\times n}$ with $\operatorname{diag}(C) = 0$, $k \in \mathbb{N}_{\geq 1}$
Output: approximate solution $V = (v_1|\dots|v_n) \in \mathbb{R}^{k\times n}$ of (SDP-vec)

for $i \leftarrow 1$ **to** n **do**
  $v_i \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;

while *not yet converged* **do**
  for $i \leftarrow 1$ **to** n **do**
    $v_i \leftarrow \frac{V\cdot c_i}{\|V\cdot c_i\|}$;

Theorem (Wang et al., 2018)

The Mixing Method converges linearly to the global optimum under a non-degeneracy assumption.

- objective value is strictly increasing
- value increases by $2(\|g\| - v_i^\top g)$ for each update $g = V \cdot c_i$

Stopping criteria

δ

Let $\delta \in \mathbb{R}_+$ denote the accumulated improvement of the objective during the last execution of the **while** loop.

Stopping criteria

δ

Let $\delta \in \mathbb{R}_+$ denote the accumulated improvement of the objective during the last execution of the **while** loop.

Function tolerance

- stop **if** $\delta < \texttt{tol\_delta\_abs}$
- stop **if** $\delta < \texttt{tol\_delta\_rel} \cdot \left(1 + \left|\langle C, V^\top V \rangle\right|\right)$

Stopping criteria

δ

Let $\delta \in \mathbb{R}_+$ denote the accumulated improvement of the objective during the last execution of the **while** loop.

Function tolerance

- stop **if** $\delta < \texttt{tol\_delta\_abs}$
- stop **if** $\delta < \texttt{tol\_delta\_rel} \cdot \left(1 + \left|\langle C, V^\top V\rangle\right|\right)$

Step tolerance

- stop **if** $\|V_{\text{old}} - V_{\text{new}}\|_F < \texttt{tol\_V\_abs}$
- stop **if** $\|V_{\text{old}} - V_{\text{new}}\|_F < \texttt{tol\_V\_rel} \cdot (1 + \|V_{\text{old}}\|_F)$

Stopping criteria

δ

Let $\delta \in \mathbb{R}_+$ denote the accumulated improvement of the objective during the last execution of the **while** loop.

Function tolerance

- stop **if** $\delta < \texttt{tol\_delta\_abs}$
- stop **if** $\delta < \texttt{tol\_delta\_rel} \cdot \left(1 + \left|\langle C, V^\top V\rangle\right|\right)$

Step tolerance

- stop **if** $\|V_{\text{old}} - V_{\text{new}}\|_F < \texttt{tol\_V\_abs}$
- stop **if** $\|V_{\text{old}} - V_{\text{new}}\|_F < \texttt{tol\_V\_rel} \cdot (1 + \|V_{\text{old}}\|_F)$

We use

- $\texttt{tol\_delta\_abs} = \texttt{tol\_delta\_rel} = \texttt{tol\_V\_abs} = 0$
- $\texttt{tol\_V\_rel} = 0.013$

Upper bounds via weak duality

Duality

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \operatorname{diag}(X) = e \\ & X \succeq 0 \end{aligned} \tag{SDP}$$

$$\begin{aligned} \min \quad & e^\top y \\ \text{s. t.} \quad & \operatorname{Diag}(y) - C \succeq 0 \\ & y \in \mathbb{R}^n \end{aligned} \tag{DSDP}$$

Upper bounds via weak duality

Duality

$$\begin{array}{rl} \max & \langle C, X \rangle \\ \text{s.t.} & \operatorname{diag}(X) = e \\ & X \succeq 0 \end{array} \qquad \text{(SDP)}$$

$$\begin{array}{rl} \min & e^\top y \\ \text{s.t.} & \operatorname{Diag}(y) - C \succeq 0 \\ & y \in \mathbb{R}^n \end{array} \qquad \text{(DSDP)}$$

Proposition [Wang et al., 2018]

Assume that $\operatorname{diag}(C) = 0$. If V^* is optimal for (SDP-vec), then the vector $y^* \in \mathbb{R}^n$ with entries $y_i^* = \|V \cdot c_i\|_2$ is optimal for (DSDP).

Upper bounds via weak duality

Duality

$$\begin{array}{ll} \max & \langle C, X\rangle \\ \text{s.t.} & \operatorname{diag}(X) = e \\ & X \succeq 0 \end{array} \qquad \text{(SDP)} \qquad\qquad \begin{array}{ll} \min & e^\top y \\ \text{s.t.} & \operatorname{Diag}(y) - C \succeq 0 \\ & y \in \mathbb{R}^n \end{array} \qquad \text{(DSDP)}$$

Proposition [Wang et al., 2018]

Assume that $\operatorname{diag}(C) = 0$. If V^* is optimal for (SDP-vec), then the vector $y^* \in \mathbb{R}^n$ with entries $y_i^* = \|V \cdot c_i\|_2$ is optimal for (DSDP).

After stopping the Mixing Method with approximate $\tilde{V}$:

- approximate but non-feasible dual variables: $\tilde{y}_i = \|\tilde{V} \cdot c_i\|_2$

Upper bounds via weak duality

Duality

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \operatorname{diag}(X) = e \\ & X \succeq 0 \end{aligned} \qquad \text{(SDP)} \qquad\qquad \begin{aligned} \min \quad & e^\top y \\ \text{s. t.} \quad & \operatorname{Diag}(y) - C \succeq 0 \\ & y \in \mathbb{R}^n \end{aligned} \qquad \text{(DSDP)}$$

Proposition [Wang et al., 2018]

Assume that $\operatorname{diag}(C) = 0$. If V^* is optimal for (SDP-vec), then the vector $y^* \in \mathbb{R}^n$ with entries $y_i^* = \|V \cdot c_i\|_2$ is optimal for (DSDP).

After stopping the Mixing Method with approximate $\tilde{V}$:

- approximate but non-feasible dual variables: $\tilde{y}_i = \|\tilde{V} \cdot c_i\|_2$
- feasible dual variables: $y = \tilde{y} - \lambda_{\min}(\operatorname{Diag}(\tilde{y}) - C)\, e$

Other possibility

We use the dual bound

$$e^\top \tilde{y} - n\lambda_{\min}\left(\mathrm{Diag}(\tilde{y}) - C\right).$$

Other possibility

We use the dual bound

$$e^\top \tilde{y} - n\lambda_{\min}\left(\mathrm{Diag}(\tilde{y}) - C\right).$$

Better upper bound [Jansson et al., 2007]

Let $\tilde{y} \in \mathbb{R}^n$ and $\bar{x}$ such that $\lambda_{\max}(X) \leq \bar{x}$ for some optimal X of (SDP). Then

$$e^\top \tilde{y} - \sum_{\lambda_k(\mathrm{Diag}(\tilde{y}) - C) < 0} \lambda_k \bar{x}$$

is an upper bound on (SDP).

Other possibility

We use the dual bound

$$e^\top \tilde{y} - n\lambda_{\min}\left(\mathrm{Diag}(\tilde{y}) - C\right).$$

Better upper bound [Jansson et al., 2007]

Let $\tilde{y} \in \mathbb{R}^n$ and $\bar{x}$ such that $\lambda_{\max}(X) \leq \bar{x}$ for some optimal X of (SDP). Then

$$e^\top \tilde{y} - \sum_{\lambda_k(\mathrm{Diag}(\tilde{y})-C)<0} \lambda_k \bar{x}$$

is an upper bound on (SDP).

- slightly better bounds
- but: computing $\bar{x}$ requires another eigenvalue computation

Branch-and-Bound

Branching:

- we branch on products X_{ij} (like in *BiqMac*)
- branching on $X_{n-1,n}$ results in $C' \in \mathbb{R}^{(n-1)\times(n-1)}$ with entries

$$c'_{ij} = \begin{cases} c_{ij} & 1 \leq i,j \leq n-1 \\ c_{i,n-1} \pm c_{in} & 1 \leq i < n-1,\ j = n-1 \\ c_{n-1,j} \pm c_{n,j} & i = n-1,\ 1 \leq j < n-1 \\ c_{n-1,n-1} \pm 2c_{n-1,n} + c_{n,n} & i = j = n-1 \end{cases}$$

Branch-and-Bound

Branching:

- we branch on products X_{ij} (like in *BiqMac*)
- branching on $X_{n-1,n}$ results in $C' \in \mathbb{R}^{(n-1)\times(n-1)}$ with entries

$$c'_{ij} = \begin{cases} c_{ij} & 1 \le i,j \le n-1 \\ c_{i,n-1} \pm c_{in} & 1 \le i < n-1,\ j = n-1 \\ c_{n-1,j} \pm c_{n,j} & i = n-1,\ 1 \le j < n-1 \\ c_{n-1,n-1} \pm 2c_{n-1,n} + c_{n,n} & i = j = n-1 \end{cases}$$

- best-first search (largest upper bound)

Branch-and-Bound

Branching:

- we branch on products X_{ij} (like in *BiqMac*)
- branching on $X_{n-1,n}$ results in $C' \in \mathbb{R}^{(n-1)\times(n-1)}$ with entries

$$c'_{ij} = \begin{cases} c_{ij} & 1 \leq i, j \leq n-1 \\ c_{i,n-1} \pm c_{in} & 1 \leq i < n-1,\ j = n-1 \\ c_{n-1,j} \pm c_{n,j} & i = n-1,\ 1 \leq j < n-1 \\ c_{n-1,n-1} \pm 2c_{n-1,n} + c_{n,n} & i = j = n-1 \end{cases}$$

- best-first search (largest upper bound)

Bounding:

- primal (lower) bounds via heuristics
- dual (upper) bounds like discussed before

Branching Example

$$C = \begin{pmatrix} 2 & -1 & 3 & -2 \\ -1 & -1 & 1 & 2 \\ 3 & 1 & 1 & -1 \\ -2 & 2 & -1 & 1 \end{pmatrix}$$

Branching on $(2,3)$ with $X_{23} = x_2 \cdot x_3 = 1$:

$$\begin{pmatrix} 2 & -1+3 & 3 & -2 \\ -1+3 & -1+1+2\cdot 1 & 1 & 2-1 \\ 3 & 1 & 1 & -1 \\ -2 & 2-1 & -1 & 1 \end{pmatrix} \xrightarrow[\text{row/column 3}]{\text{remove}} C' = \begin{pmatrix} 2 & 2 & -2 \\ 2 & 2 & 1 \\ -2 & 1 & 1 \end{pmatrix}$$

Branching on $(2,3)$ with $X_{23} = x_2 \cdot x_3 = -1$:

$$\begin{pmatrix} 2 & -1-3 & 3 & -2 \\ -1-3 & -1+1-2\cdot 1 & 1 & 2+1 \\ 3 & 1 & 1 & -1 \\ -2 & 2+1 & -1 & 1 \end{pmatrix} \xrightarrow[\text{row/column 3}]{\text{remove}} C' = \begin{pmatrix} 2 & -4 & -2 \\ -4 & -2 & 3 \\ -2 & 3 & 1 \end{pmatrix}$$

Branching decision

- SDP approaches in literature only use X for branching decision
  - often: branching on most fractional variable
  - some solvers branch in first row/column only

Branching decision

- SDP approaches in literature only use X for branching decision
  - often: branching on most fractional variable
  - some solvers branch in first row/column only

Branching decision based on dual variables

We determine the branching decision (i, j) in $\mathcal{O}(n)$:

Branching decision

- SDP approaches in literature only use X for branching decision
  - often: branching on most fractional variable
  - some solvers branch in first row/column only

Branching decision based on dual variables

We determine the branching decision (i, j) in $\mathcal{O}(n)$:

1. Find $i = \operatorname{argmax}_k \{y_k\}$.

Branching decision

- SDP approaches in literature only use X for branching decision
  - often: branching on most fractional variable
  - some solvers branch in first row/column only

Branching decision based on dual variables

We determine the branching decision (i, j) in $\mathcal{O}(n)$:

1. Find $i = \operatorname{argmax}_k \{y_k\}$.
2. Find $j = \operatorname{argmax}_k \{(y_i + y_k) \cdot f(X_{ik}) \colon |X_{ik}| \leq 0.875\}$.

- where $f\colon \{-1, 1\} \to [0, 1]$ decreasing in $|X_{ik}|$

Feature: Early branching

Assumption

Finding an optimal solution with heuristics is easy.

Observation

The Mixing Method produces primal feasible iterates for (SDP).

Feature: Early branching

Assumption

Finding an optimal solution with heuristics is easy.

Observation

The Mixing Method produces primal feasible iterates for (SDP).

Stopping criteria have an impact on:

- solutions found by heuristics (important for pruning)

Feature: Early branching

Assumption

Finding an optimal solution with heuristics is easy.

Observation

The Mixing Method produces primal feasible iterates for (SDP).

Stopping criteria have an impact on:

- solutions found by heuristics (important for pruning)
- branching decision (important for overall efficiency)

Feature: Early branching

Assumption

Finding an optimal solution with heuristics is easy.

Observation

The Mixing Method produces primal feasible iterates for (SDP).

Stopping criteria have an impact on:

- solutions found by heuristics (important for pruning)
- branching decision (important for overall efficiency)
- upper bound (important for pruning and best-first search)

Feature: Early branching

Assumption

Finding an optimal solution with heuristics is easy.

Observation

The Mixing Method produces primal feasible iterates for (SDP).

Stopping criteria have an impact on:

- solutions found by heuristics (important for pruning)
- branching decision (important for overall efficiency)
- upper bound (important for pruning and best-first search)

Early branching

Feature: Early branching

Assumption

Finding an optimal solution with heuristics is easy.

Observation

The Mixing Method produces primal feasible iterates for (SDP).

Stopping criteria have an impact on:

- solutions found by heuristics (important for pruning)
- branching decision (important for overall efficiency)
- upper bound (important for pruning and best-first search)

Early branching

Immediately branch if we have done at least 4 iterations of the **while** loop and we know that the optimal value of (SDP) will be larger than the best known lower bound found by heuristics.

Feature: Variable fixing

Given: Dual feasible solution $\mathrm{Diag}(y) - C \succeq 0$ for $C \in \mathbb{R}^{n \times n}$.

Notation

- $C_{/j}$ denotes matrix C without row j and column j.
- $y_{/j}$ denotes vector y without entry j.

Feature: Variable fixing

Given: Dual feasible solution $\mathrm{Diag}(y) - C \succeq 0$ for $C \in \mathbb{R}^{n \times n}$.

Notation

- $C_{/j}$ denotes matrix C without row j and column j.
- $y_{/j}$ denotes vector y without entry j.

Branching on $(1, j)$ would yield cost matrix $\tilde{C} \in \mathbb{R}^{(n-1)\times(n-1)}$ with $C_{/j} - \tilde{C} = \begin{pmatrix} 0 & \delta^\top \\ \delta & 0 \end{pmatrix}$ for some $\delta \in \mathbb{R}^{n-2}$.

Feature: Variable fixing

Given: Dual feasible solution $\mathrm{Diag}(y) - C \succeq 0$ for $C \in \mathbb{R}^{n \times n}$.

Notation

- $C_{/j}$ denotes matrix C without row j and column j.
- $y_{/j}$ denotes vector y without entry j.

Branching on $(1, j)$ would yield cost matrix $\tilde{C} \in \mathbb{R}^{(n-1)\times(n-1)}$ with $C_{/j} - \tilde{C} = \begin{pmatrix} 0 & \delta^\top \\ \delta & 0 \end{pmatrix}$ for some $\delta \in \mathbb{R}^{n-2}$.

Lemma

$$\tilde{y} := y_{/j} + \begin{pmatrix} \|\delta\|_1 \\ |\delta_1| \\ \vdots \\ |\delta_{n-2}| \end{pmatrix}$$ is dual feasible, i.e., $\mathrm{Diag}(\tilde{y}) - \tilde{C} \succeq 0$.

Proof.

$$\operatorname{Diag}(\tilde{y}) - \tilde{C} = \operatorname{Diag}\left(y_{/j} + \begin{pmatrix} \|\delta\|_1 \\ |\delta_1| \\ \vdots \\ |\delta_{n-2}| \end{pmatrix} \right) - \left(C_{/j} - \begin{pmatrix} 0 & \delta^\top \\ \delta & 0 \end{pmatrix} \right)$$

$$= \operatorname{Diag}\left(y_{/j}\right) + \operatorname{Diag}\left(\begin{pmatrix} \|\delta\|_1 \\ |\delta_1| \\ \vdots \\ |\delta_{n-2}| \end{pmatrix} \right) - C_{/j} + \begin{pmatrix} 0 & \delta^\top \\ \delta & 0 \end{pmatrix}$$

$$= \underbrace{\operatorname{Diag}\left(y_{/j}\right) - C_{/j}}_{\succeq 0} + \underbrace{\operatorname{Diag}\left(\begin{pmatrix} \|\delta\|_1 \\ |\delta_1| \\ \vdots \\ |\delta_{n-2}| \end{pmatrix} \right) + \begin{pmatrix} 0 & \delta^\top \\ \delta & 0 \end{pmatrix}}_{\succeq 0} \succeq 0$$

□

Variable fixing

- bound at current node: $e^\top y$

'Free' dual bound if we would branch

Dual bound after branching on (i, j): $e^\top \tilde{y} + 2\|\delta\|_1 \pm 2c_{ij}$.

- difference of bounds: $-y_j + 2\sum_{k\neq i,j} |c_{jk}| \pm 2c_{ij}$

Variable fixing

- bound at current node: $e^\top y$

'Free' dual bound if we would branch

Dual bound after branching on (i,j): $e^\top \tilde{y} + 2\|\delta\|_1 \pm 2c_{ij}$.

- difference of bounds: $-y_j + 2\sum_{k\neq i,j} |c_{jk}| \pm 2c_{ij}$
- best scenario: 'free' dual bound worse than best known primal bound

Variable fixing

- bound at current node: $e^\top y$

'Free' dual bound if we would branch

Dual bound after branching on (i, j): $e^\top \tilde{y} + 2\|\delta\|_1 \pm 2c_{ij}$.

- difference of bounds: $-y_j + 2\sum_{k \neq i,j} |c_{jk}| \pm 2c_{ij}$
- best scenario: 'free' dual bound worse than best known primal bound

How we use it

- check all $\mathcal{O}(n^2)$ candidates in $\mathcal{O}(n^2)$ time
- do usual branching step + additional fixation(s)

Variable fixing

- bound at current node: $e^\top y$

'Free' dual bound if we would branch

Dual bound after branching on (i, j): $e^\top \tilde{y} + 2\|\delta\|_1 \pm 2c_{ij}$.

- difference of bounds: $-y_j + 2\sum_{k \neq i,j} |c_{jk}| \pm 2c_{ij}$
- best scenario: 'free' dual bound worse than best known primal bound

How we use it

- check all $\mathcal{O}(n^2)$ candidates in $\mathcal{O}(n^2)$ time
- do usual branching step + additional fixation(s)

Issue

Conflict with early branching (no dual feasible solution)!

Primal heuristic

Algorithm 2: Goemans-Williamson hyperplane rounding

Input: $V = (v_1, \dots, v_n) \in \mathbb{R}^{k \times n}$ (such that $V^\top V = X$)
Output: $x \in \{-1, 1\}^n$ (feasible solution for QUBO/Max-Cut)

$h \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;
for $i \leftarrow 1$ **to** n **do**

$$x_i \leftarrow \begin{cases} +1, & \text{if } h^\top v_i \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

return x;

Primal heuristic

Algorithm 2: Goemans-Williamson hyperplane rounding

Input: $V = (v_1, \dots, v_n) \in \mathbb{R}^{k \times n}$ (such that $V^\top V = X$)
Output: $x \in \{-1, 1\}^n$ (feasible solution for QUBO/Max-Cut)

$h \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;
for $i \leftarrow 1$ **to** n **do**

$$x_i \leftarrow \begin{cases} +1, & \text{if } h^\top v_i \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

return x;

- local search to improve the solution (one-opt and two-opt)

Primal heuristic

Algorithm 2: Goemans-Williamson hyperplane rounding

Input: $V = (v_1, \ldots, v_n) \in \mathbb{R}^{k \times n}$ (such that $V^\top V = X$)
Output: $x \in \{-1, 1\}^n$ (feasible solution for QUBO/Max-Cut)

$h \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;
for $i \leftarrow 1$ **to** n **do**

$$x_i \leftarrow \begin{cases} +1, & \text{if } h^\top v_i \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

return x;

- local search to improve the solution (one-opt and two-opt)
- reasonable candidates for local search

Primal heuristic

Algorithm 2: Goemans-Williamson hyperplane rounding

Input: $V = (v_1, \dots, v_n) \in \mathbb{R}^{k \times n}$ (such that $V^\top V = X$)
Output: $x \in \{-1, 1\}^n$ (feasible solution for QUBO/Max-Cut)

$h \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$;
for $i \leftarrow 1$ **to** n **do**

$$x_i \leftarrow \begin{cases} +1, & \text{if } h^\top v_i \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

return x;

- local search to improve the solution (one-opt and two-opt)
- reasonable candidates for local search
- 'good' hyperplane idea